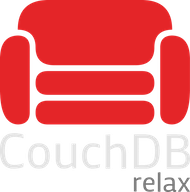
****

**Nhóm thực hiện: 13**

|  |  |
| --- | --- |
| **Đỗ Phúc Hậu** | **42.01.104.043** |
| **Cao Đức Minh** | **42.01.104.082** |
| **Nguyễn Thanh Sang** | **42.01.104.132** |
| **Nguyễn Hoàng Anh Tú** | **42.01.104.148** |
| **Trần Thuận Thiên** | **42.01.104.169** |

**Giảng viên hướng dẫn:**

***Th.s Lương Trần Hy Hiến***

**TÌM HIỂU VỀ COUCHDB**

**MÔN HỌC: CƠ SỞ DỮ LIỆU NÂNG CAO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙢🙠🕮🙢🙠**

**Năm học 2018 - 2019**

**Mục lục**

[**CouchDB là gì?** 3](#_Toc531133167)

[**Mô hình dữ liệu** 3](#_Toc531133168)

[Lưu trữ dạng tài liệu 3](#_Toc531133169)

[**Đặc điểm của CouchDB** 3](#_Toc531133170)

[Thuộc tính ACID 3](#_Toc531133171)

[Khả năng nén 4](#_Toc531133172)

[Views 4](#_Toc531133173)

[Mô hình View 4](#_Toc531133174)

[Phân tán cập nhật và nhân rộng 5](#_Toc531133175)

[Xung đột 5](#_Toc531133176)

[**Tại sao nên sử dụng CouchDB?** 5](#_Toc531133177)

[Thư giãn 5](#_Toc531133178)

[Cách thức mô hình hóa dữ liệu khác biệt 6](#_Toc531133179)

[**Tính nhất quán cuối cùng** 6](#_Toc531133180)

[Không khóa 6](#_Toc531133181)

[Tính xác thực 7](#_Toc531133182)

[Mô hình nhân rộng 7](#_Toc531133183)

[**Điểm mạnh của CouchDB** 8](#_Toc531133184)

[**Mặt hạn chế của CouchDB** 8](#_Toc531133185)

[**Sự khác nhau giữa CouchDB và Microsoft SQL Server** 9](#_Toc531133186)

[**Các màn hình nghiệp vụ** 11](#_Toc531133187)

[Phần mềm quản lý siêu thị sử dụng CouchDB 11](#_Toc531133188)

[**Các nguồn tham khảo:** 15](#_Toc531133189)

# **CouchDB là gì?**

Apache CouchDB là phần mềm cơ sở dữ liệu nguồn mở tập trung vào tính dễ sử dụng và có kiến ​​trúc có thể mở rộng. Nó có kiến ​​trúc cơ sở dữ liệu NoSQL theo hướng tài liệu (document), nó sử dụng JSON để lưu trữ dữ liệu. Truy cập tài liệu bằng trình duyệt web, qua HTTP, truy vấn, kết hợp và chuyển đổi tài liệu bằng JavaScript. CouchDB hoạt động tốt với các ứng dụng web và di động hiện đại.

# **Mô hình dữ liệu**

## Lưu trữ dạng tài liệu

Kho lưu trữ tài liệu CouchDB lưu trữ dưới dạng tài liệu. Mỗi tài liệu được đặt tên duy nhất trong cơ sở dữ liệu và CouchDB cung cấp một API RESTful HTTP để đọc và cập nhật (thêm, chỉnh sửa, xóa) các tài liệu cơ sở dữ liệu.

Tài liệu là đơn vị dữ liệu chính trong CouchDB bao gồm bất kỳ số trường và tệp đính kèm nào. Tài liệu cũng bao gồm siêu dữ liệu được duy trì bởi hệ thống cơ sở dữ liệu. Trường tài liệu được đặt tên duy nhất và chứa các giá trị của các loại khác nhau (văn bản, số, boolean, danh sách, v.v.) và không có giới hạn được đặt cho kích thước văn bản hoặc số phần tử.

Mô hình cập nhật tài liệu CouchDB là không có khóa và mang hướng tích cực. Chỉnh sửa tài liệu được thực hiện bởi các ứng dụng khách, tải tài liệu, áp dụng thay đổi và lưu chúng trở lại cơ sở dữ liệu. Nếu một khách hàng khác chỉnh sửa cùng một tài liệu lưu thay đổi của họ trước, khách hàng sẽ nhận được lỗi xung đột chỉnh sửa khi lưu. Để giải quyết xung đột cập nhật, phiên bản tài liệu mới nhất có thể được mở, các chỉnh sửa được áp dụng lại và cập nhật sẽ được thử lại. Các cập nhật tài liệu (thêm, chỉnh sửa, xóa) một là được lưu lại hai là không được lưu lại, hoặc là hoàn toàn thành công hoặc thất bại hoàn toàn. Cơ sở dữ liệu không bao giờ chứa một phần tài liệu được lưu hoặc chỉnh sửa.

# **Đặc điểm của CouchDB**

## Thuộc tính ACID

Bố cục tập tin của CouchDB tuân theo tất cả các thuộc tính của ACID. Trên đĩa, CouchDB không bao giờ ghi đè dữ liệu đã cam kết hoặc các cấu trúc liên quan, đảm bảo tệp cơ sở dữ liệu luôn ở trạng thái nhất quán.

Các cập nhật tài liệu (thêm, chỉnh sửa, xóa) được tuần tự hóa. Trình đọc cơ sở dữ liệu không bao giờ bị khóa và không bao giờ phải chờ đợi những người ghi hoặc người đọc khác. Bất kỳ số lượng khách hàng nào cũng có thể đọc tài liệu mà không bị khóa hoặc gián đoạn bởi các bản cập nhật đồng thời, ngay cả trên cùng một tài liệu. Các hoạt động đọc CouchDB sử dụng mô hình Kiểm soát tương tranh đa phiên bản (MVCC) trong đó mỗi khách hàng nhìn thấy một ảnh chụp nhất quán của cơ sở dữ liệu từ đầu đến cuối hoạt động đọc. Tài liệu được lập chỉ mục trong B-tree theo tên của chúng (DocID) và ID chuỗi. Mỗi bản cập nhật cho một cá thể cơ sở dữ liệu tạo ra một số thứ tự mới. ID chuỗi được sử dụng sau này để tìm kiếm các thay đổi trong cơ sở dữ liệu một cách từng bước. Các chỉ mục B-tree này được cập nhật đồng thời khi các tài liệu được lưu hoặc xóa. Các cập nhật chỉ mục luôn xảy ra ở cuối tệp (các bản cập nhật chỉ thêm vào).

## Khả năng nén

Không gian bị lãng phí được phục hồi bằng cách nén thường xuyên. Theo lịch biểu, hoặc khi tệp cơ sở dữ liệu vượt quá một lượng không gian lãng phí nhất định, quá trình nén sẽ nhân bản tất cả dữ liệu đang hoạt động vào một tệp mới và sau đó loại bỏ tệp cũ. Cơ sở dữ liệu vẫn hoàn toàn trực tuyến toàn bộ thời gian và tất cả các bản cập nhật và lần đọc được phép hoàn tất thành công. Tệp cơ sở dữ liệu cũ chỉ bị xóa khi tất cả dữ liệu đã được sao chép và tất cả người dùng đã chuyển sang tệp mới.

## Views

Thuộc tính ACID chỉ xử lý lưu trữ và cập nhật, CouchDB cũng cần khả năng hiển thị dữ liệu. Không giống như các cơ sở dữ liệu SQL, nơi dữ liệu phải được phân tích một cách cẩn thận thành các bảng, dữ liệu trong CouchDB được lưu trữ trong các tài liệu bán cấu trúc. Các tài liệu CouchDB rất linh hoạt và mỗi tài liệu có cấu trúc ngầm riêng của nó, làm giảm bớt các vấn đề khó khăn nhất và các bẫy của lược đồ bảng nhân bản hai chiều và dữ liệu chứa của chúng.

Nhưng ngoài việc hoạt động như một máy chủ kiểu tập tin, một mô hình tài liệu đơn giản để lưu trữ và chia sẻ dữ liệu quá đơn giản để xây dựng các ứng dụng thực - đơn giản là không đủ những thứ chúng ta mong muốn.

CouchDB cho phép xem dữ liệu theo nhiều cách khác nhau. Điều cần thiết là một cách để lọc, sắp xếp và báo cáo dữ liệu không bị phân tách thành các bảng.

### Mô hình View

Để giải quyết vấn đề khi đưa kiểu cấu trúc trở lại dữ liệu phi cấu trúc và bán cấu trúc, CouchDB tích hợp một mô hình View. Views là phương pháp tổng hợp và báo cáo về các tài liệu trong cơ sở dữ liệu và được xây dựng theo yêu cầu để tổng hợp, tham gia và báo cáo về tài liệu cơ sở dữ liệu. Vì chế độ xem được tạo động và không ảnh hưởng đến tài liệu cơ bản,chúng ta có thể có nhiều lần biểu diễn Views khác nhau của cùng một dữ liệu tùy thích.

Views định nghĩa hoàn toàn ảo và chỉ hiển thị các tài liệu từ cá thể cơ sở dữ liệu hiện tại, làm cho chúng tách biệt với dữ liệu mà chúng hiển thị và tương thích với bản sao. Các View của CouchDB được định nghĩa bên trong các document thiết kế đặc biệt và có thể tái tạo trên các cá thể cơ sở dữ liệu như các tài liệu thông thường, sao cho không chỉ sao chép dữ liệu trong CouchDB mà còn sao chép toàn bộ các thiết kế ứng dụng.

## Phân tán cập nhật và nhân rộng

CouchDB là một hệ thống cơ sở dữ liệu phân tán ngang hàng. Nó cho phép người dùng và máy chủ truy cập và cập nhật cùng một dữ liệu được chia sẻ trong khi ngắt kết nối. Những thay đổi đó sau đó có thể được nhân rộng hai chiều sau đó.

Các mô hình lưu trữ, xem và bảo mật tài liệu CouchDB được thiết kế để làm việc cùng nhau để tạo ra sự sao chép hai chiều thực sự hiệu quả và đáng tin cậy. Cả hai tài liệu và thiết kế đều có thể sao chép, cho phép các ứng dụng cơ sở dữ liệu đầy đủ (bao gồm thiết kế ứng dụng, logic và dữ liệu) được sao chép sang máy tính xách tay để sử dụng ngoại tuyến hoặc được sao chép tới máy chủ ở các văn phòng ở xa nơi kết nối chậm hoặc không đáng tin cậy.

## Xung đột

Phát hiện và quản lý xung đột là các vấn đề chính đối với bất kỳ hệ thống chỉnh sửa được phân phối nào. Hệ thống lưu trữ CouchDB xử lý các xung đột sửa đổi như một trạng thái chung, không phải là một ngoại lệ đặc biệt. Mô hình xử lý xung đột rất đơn giản và “không phá hoại” trong khi vẫn duy trì ngữ nghĩa document đơn lẻ và cho phép giải quyết xung đột phi tập trung.

CouchDB cho phép bất kỳ số lượng các document xung đột nào tồn tại đồng thời trong cơ sở dữ liệu, với mỗi cá thể cơ sở dữ liệu quyết định document nào là “người chiến thắng” và đó là xung đột. Chỉ có document chiến thắng mới có thể xuất hiện trong các View, trong khi các xung đột "mất" vẫn có thể truy cập được và vẫn còn trong cơ sở dữ liệu cho đến khi bị xóa hoặc bị xóa trong quá trình nén cơ sở dữ liệu. Bởi vì tài liệu xung đột vẫn là tài liệu thường xuyên, chúng sao chép giống như các tài liệu thông thường và tuân theo các quy tắc bảo mật và xác thực giống nhau.

Khi xảy ra các chỉnh sửa xung đột, mọi bản sao cơ sở dữ liệu đều thấy cùng một phiên bản chiến thắng và mỗi bản có cơ hội giải quyết xung đột. Giải quyết xung đột có thể được thực hiện thủ công hoặc, tùy thuộc vào bản chất của dữ liệu và xung đột, bởi các tác nhân tự động. Hệ thống đưa ra các giải pháp xung đột có thể xử lý được trong khi vẫn duy trì ngữ nghĩa cơ sở dữ liệu document duy nhất.

Quản lý xung đột tiếp tục hoạt động ngay cả khi nhiều người dùng bị ngắt kết nối cố gắng giải quyết các xung đột tương tự. Nếu các xung đột được giải quyết dẫn đến nhiều xung đột hơn, hệ thống sẽ điều chỉnh chúng theo cùng một cách, xác định cùng một” người chiến thắng” trên mỗi máy và duy trì ngữ nghĩa document đơn lẻ.

# **Tại sao nên sử dụng CouchDB?**

## Thư giãn

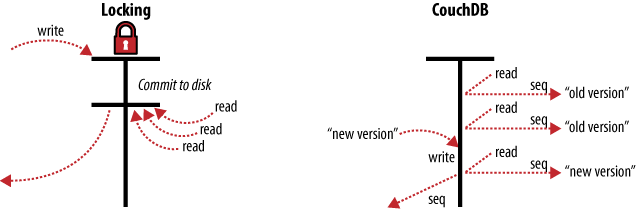
Tại sao thư giãn lại quan trọng? Năng suất của nhà phát triển tăng gần gấp đôi trong 5 năm qua. Lý do chính cho việc tăng cường là các công cụ mạnh mẽ hơn dễ sử dụng hơn. CouchDB tập trung thiết kế cốt lõi vào tính dễ sử dụng. Đây là một trong những lý do tại sao CouchDB thư giãn: học CouchDB và hiểu các khái niệm cốt lõi của nó nên cảm thấy tự nhiên đối với hầu hết mọi người đã làm bất kỳ công việc nào trên Web. Và nó vẫn còn khá dễ dàng để giải thích cho người phi kỹ thuật. Kiến trúc bên trong của nó là khả năng chịu lỗi và các lỗi xảy ra trong môi trường được kiểm soát và được xử lý một cách nhẹ nhàng. Các vấn đề đơn lẻ không xếp tầng thông qua toàn bộ hệ thống máy chủ nhưng vẫn bị cô lập trong các yêu cầu duy nhất. Khái niệm cốt lõi của CouchDB rất đơn giản (nhưng mạnh mẽ) và được hiểu rõ. Các nhóm hoạt động không phải lo sợ hành vi ngẫu nhiên và các lỗi không thể truy cập được. Nếu có điều gì sai, bạn có thể dễ dàng tìm ra vấn đề là gì, nhưng những tình huống này rất hiếm. CouchDB cũng được thiết kế để xử lý lưu lượng truy cập khác nhau một cách dễ dàng. Ví dụ, nếu một trang web đang trải qua một đột biến đột ngột trong lưu lượng truy cập, CouchDB nói chung sẽ hấp thụ rất nhiều yêu cầu đồng thời mà không bị sập. Có thể mất thêm một chút thời gian cho mỗi yêu cầu, nhưng tất cả đều được phản hồi.

## Cách thức mô hình hóa dữ liệu khác biệt

# **Tính nhất quán cuối cùng**

## Không khóa

Một bảng trong cơ sở dữ liệu quan hệ là một cấu trúc dữ liệu duy nhất. Nếu bạn muốn sửa đổi một bảng, cập nhật một hàng - hệ thống cơ sở dữ liệu phải đảm bảo rằng không ai khác đang cố gắng cập nhật hàng đó và không ai có thể đọc từ hàng đó trong khi nó đang được cập nhật. Cách phổ biến để xử lý việc này là sử dụng cái-được-gọi-là khóa. Nếu nhiều khách hàng muốn truy cập vào một bảng, khách hàng đầu tiên sẽ nhận được khóa, khiến mọi người khác phải chờ đợi. Khi yêu cầu của khách hàng đầu tiên được xử lý, khách hàng tiếp theo được cấp quyền truy cập trong khi mọi người khác chờ đợi, v.v. Việc thực thi nối tiếp các yêu cầu này, ngay cả khi chúng đến song song, lãng phí một lượng đáng kể sức mạnh xử lý của máy chủ . Dưới tải trọng cao, một cơ sở dữ liệu quan hệ có thể dành nhiều thời gian hơn để tìm ra ai được phép làm gì, và theo thứ tự nào, hơn là thực hiện bất kỳ công việc thực tế nào.Thay vì khóa, CouchDB sử dụng Multi-Version Concurrency Control (MVCC) để quản lý truy cập đồng thời vào cơ sở dữ liệu.



Các document trong CouchDB được phiên bản hóa. Nếu muốn thay đổi một giá trị trong một tài liệu, bạn tạo một phiên bản hoàn toàn mới của document đó và lưu nó vào document cũ. Sau khi thực hiện việc này, bạn kết thúc với hai phiên bản của cùng một document, một phiên bản cũ và một document mới. Làm thế nào điều này lại cung cấp một cải tiến trên khóa? Xem xét một bộ yêu cầu muốn truy cập document. Yêu cầu đầu tiên đọc document. Trong khi điều này đang được xử lý, yêu cầu thứ hai sẽ thay đổi document. Vì yêu cầu thứ hai bao gồm một phiên bản hoàn toàn mới của document, CouchDB có thể chỉ cần gắn thêm nó vào cơ sở dữ liệu mà không phải đợi yêu cầu đọc để hoàn thành. Khi một yêu cầu thứ ba muốn đọc cùng một document, CouchDB sẽ trỏ nó đến phiên bản mới vừa được viết. Trong toàn bộ quá trình này, yêu cầu đầu tiên vẫn có thể đọc phiên bản gốc. Một yêu cầu đọc sẽ luôn nhìn thấy ảnh chụp nhanh gần đây nhất của cơ sở dữ liệu hiện tại thời điểm bắt đầu yêu cầu.

## Tính xác thực

CouchDB cung cấp một cách mạnh mẽ để thực hiện xác nhận từng document từ bên trong cơ sở dữ liệu. CouchDB có thể xác thực các document bằng cách sử dụng các hàm JavaScript tương tự như các hàm được sử dụng cho MapReduce. Mỗi khi cố gắng sửa đổi một document, CouchDB sẽ thông qua chức năng xác nhận một bản sao của document hiện có, một bản sao của document mới và một tập hợp các thông tin bổ sung, chẳng hạn như chi tiết xác thực người dùng. Chức năng xác thực bây giờ có cơ hội để phê duyệt hoặc từ chối cập nhật. Bằng cách làm việc với hạt và cho phép CouchDB thực hiện điều này cho chúng ta, chúng ta tiết kiệm một số lượng lớn các chu trình CPU mà có thể đã được sử dụng để tuần tự hóa các đồ thị đối tượng từ SQL, chuyển chúng thành các đối tượng miền và sử dụng các đối tượng đó để thực hiện xác nhận mức ứng dụng .

## Mô hình nhân rộng

Các hoạt động của CouchDB diễn ra trong bối cảnh của một document. Khi CouchDB đạt được sự nhất quán cuối cùng giữa nhiều cơ sở dữ liệu bằng cách sử dụng nhân rộng gia tăng, bạn không còn phải lo lắng về các máy chủ cơ sở dữ liệu có thể giữ liên lạc thường xuyên. Gia tăng nhân rộng là một quá trình mà các thay đổi tài liệu được sao chép theo định kỳ giữa các máy chủ. Chúng tôi có thể xây dựng cái được gọi là cụm cơ sở dữ liệu không có gì chia sẻ, trong đó mỗi nút độc lập và tự cung tự cấp, không để lại một điểm tranh chấp nào trên toàn hệ thống. Bạn cần mở rộng cụm cơ sở dữ liệu CouchDB của mình? Chỉ cần ném vào một máy chủ khác. Như được minh họa trong hình. Nhân rộng gia tăng giữa các nút CouchDB, với bản sao gia tăng của CouchDB, bạn có thể đồng bộ hóa dữ liệu giữa hai cơ sở dữ liệu bất kỳ mà bạn thích và bất cứ khi nào bạn muốn. Sau khi nhân rộng, mỗi cơ sở dữ liệu có thể làm việc độc lập. Bạn có thể sử dụng tính năng này để đồng bộ hóa các máy chủ cơ sở dữ liệu trong một cụm hoặc giữa các trung tâm dữ liệu bằng cách sử dụng bộ lập lịch công việc như cron hoặc bạn có thể sử dụng nó để đồng bộ hóa dữ liệu với máy tính xách tay để làm việc ngoại tuyến. Mỗi cơ sở dữ liệu có thể được sử dụng theo cách thông thường, và những thay đổi giữa các cơ sở dữ liệu có thể được đồng bộ hóa sau này theo cả hai hướng. Nhân rộng gia tăng giữa các nút CouchDB Điều gì sẽ xảy ra khi bạn thay đổi cùng một tài liệu trong hai cơ sở dữ liệu khác nhau và muốn đồng bộ hóa chúng với nhau? Hệ thống sao chép của CouchDB đi kèm với việc phát hiện và giải quyết xung đột tự động. Khi CouchDB phát hiện rằng một tài liệu đã được thay đổi trong cả hai cơ sở dữ liệu, nó sẽ gắn cờ tài liệu này là xung đột, giống như chúng nằm trong một hệ thống kiểm soát phiên bản thông thường. Đây không phải là rắc rối vì nó có thể âm thanh đầu tiên. Khi hai phiên bản của tài liệu xung đột trong quá trình sao chép, phiên bản chiến thắng được lưu dưới dạng phiên bản mới nhất trong lịch sử của tài liệu. Thay vì ném phiên bản mất đi, như bạn có thể mong đợi, CouchDB lưu phiên bản này làm phiên bản trước đó trong lịch sử của tài liệu, để bạn có thể truy cập nó nếu cần. Điều này xảy ra tự động và nhất quán, vì vậy cả hai cơ sở dữ liệu sẽ thực hiện chính xác sự lựa chọn tương tự. Bạn có thể xử lý các xung đột theo cách có ý nghĩa với ứng dụng của mình. Bạn có thể để nguyên các phiên bản tài liệu đã chọn, hoàn nguyên về phiên bản cũ hơn hoặc cố gắng hợp nhất hai phiên bản và lưu kết quả.

# **Điểm mạnh của CouchDB**

* Lược đồ thiết kế linh hoạt
* Lưu trữ dưới dạng JSON
* Mã nguồn mở
* Có API dạng RESTful giúp việc giao tiếp CSDL được đơn giản
* Thủ tục Map/Reduce giúp truy vấn nhanh
* Dễ dàng nhân rộng
* Không có khóa đọc
* Khả năng nhân rộng đơn giản và hai chiều
* Khả năng nén giúp tối ưu tình trạng không chuẩn hóa

# **Mặt hạn chế của CouchDB**

* Lưu trữ dưới dạng JSON

Điều đó có nghĩa là tài liệu khá lớn (BigData, băng thông mạng, tốc độ) và có các tên khóa mô tả thực sự nặng, vì chúng thêm vào kích thước tài liệu.

* Không tích hợp Full Text Search
* Không hỗ trợ các giao tác
* **Kho chứa dữ liệu**

Vấn đề với điều này là các View tạm thời trong CouchDB trên tập hợp dữ liệu lớn là rất chậm. Sử dụng CouchDB và các View vĩnh viễn có thể hoạt động khá tốt. Tuy nhiên, trong hầu hết các trường hợp, một cơ sở dữ liệu định hướng cột của một số loại sẽ tốt hơn cho công việc chứa dữ liệu.

* Không thiết kế cho việc cập nhật thường xuyên

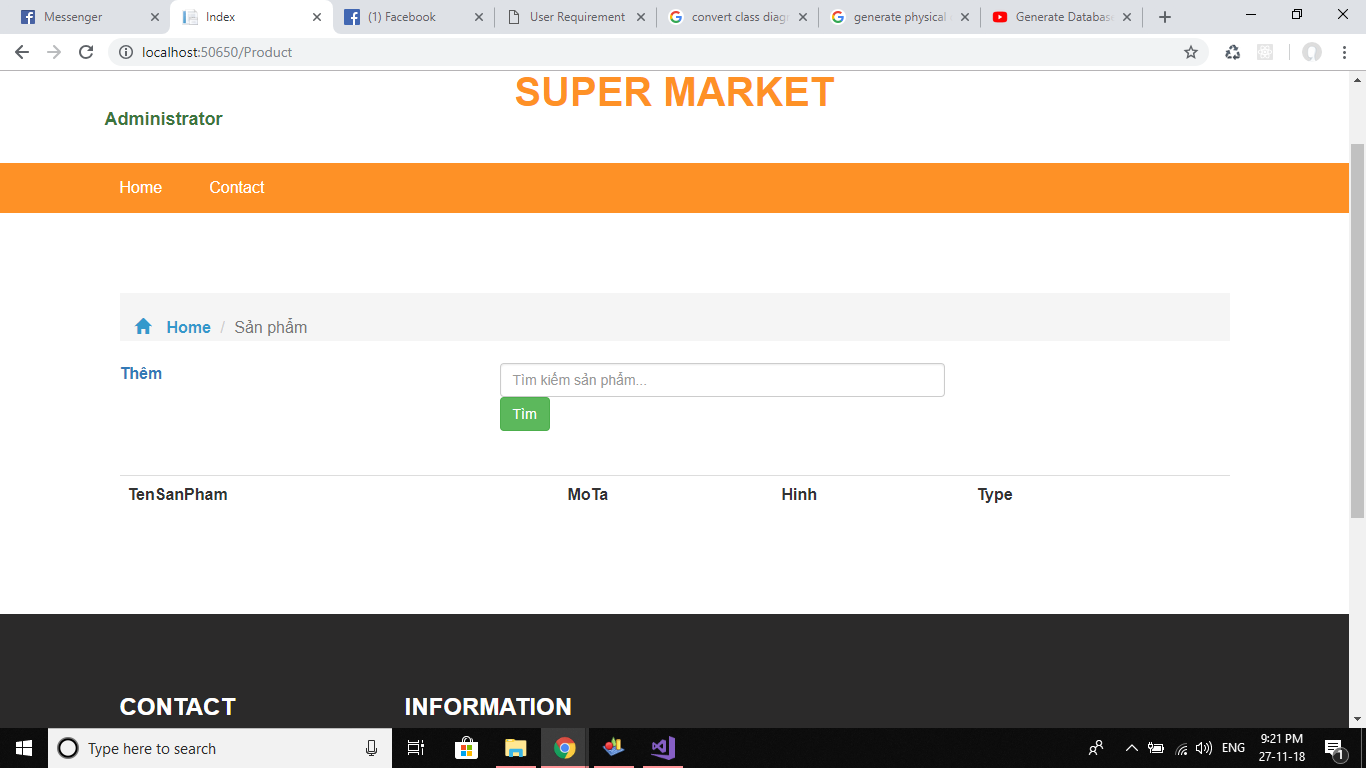
Truy vấn tùy ý là tốn kém. Để thực hiện truy vấn nếu chưa tạo chế độ xem cần tạo chế độ xem tạm thời. Điều này có thể được giải quyết ở một mức độ nào đó bằng cách sử dụng Lucene.

# **Sự khác nhau giữa CouchDB và Microsoft SQL Server**

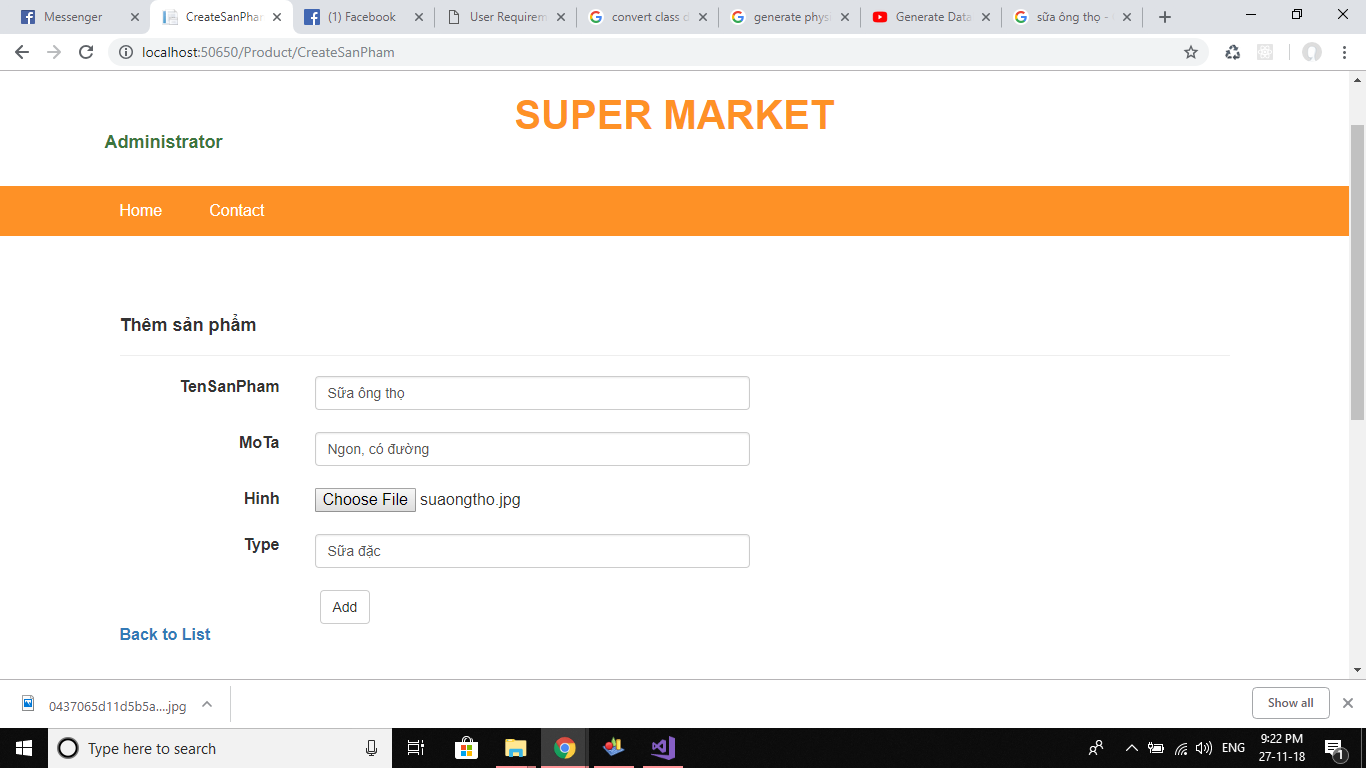
|  |  |  |
| --- | --- | --- |
| **Tên** | **CouchDB** | **Microsoft SQL Server** |
| Mô tả | Dạng JSON – lưu trữ tài liệu được lấy cảm hứng từ Lotus Notes, có thể mở rộng từ các cụm máy chủ được phân phối trên toàn cầu đến điện thoại di động. | Microsofts hệ quản trị cơ sở dữ liệu hướng quan hệ |
| Mô hình cơ sở dữ liệu | Lưu trữ tài liệu | DBMS kiểu quan hệ |
| Nhà phát triển | Apache Software Foundation | Microsoft |
| Chính thức phát hành | 2005 | 1989 |
| Phát hành hiện tại | 2.2.0, August 2018 | SQL Server 2017, October 2017 |
| Giấy phép | Mã nguồn mở | Thương mại |
| Lược đồ dữ liệu | Lược đồ tự do | yes |
| Định nghĩa trước kiểu dữ liệu | no | yes |
| Các API và phương thức truy cập khác | RESTful HTTP/JSON API | OLE DB Tabular Data Stream (TDS) ADO.NET JDBC ODBC |
| Store procedure | Xem trong Javascript | Transact SQL, .NET languages, R, Python and (with SQL Server 2019) Java |
| Triggers | yes | yes |
| Phương thức nhân rộng | Master-master replication Master-slave replication | Tùy thuộc vào phiên bản SQL |
| MapReduce | yes | no |
| Khái niệm nhất quán | Eventual Consistency | Immediate Consistency |
| Khóa ngoại | no | yes |
| Hỗ trợ transaction | no | ACID |
| Điều khiển truy cập | Quyền truy cập cho người dùng có thể được định nghĩa trong mỗi cơ sở dữ liệu | Quyền truy cập hạt mịn theo tiêu chuẩn SQL |

# **Các màn hình nghiệp vụ**

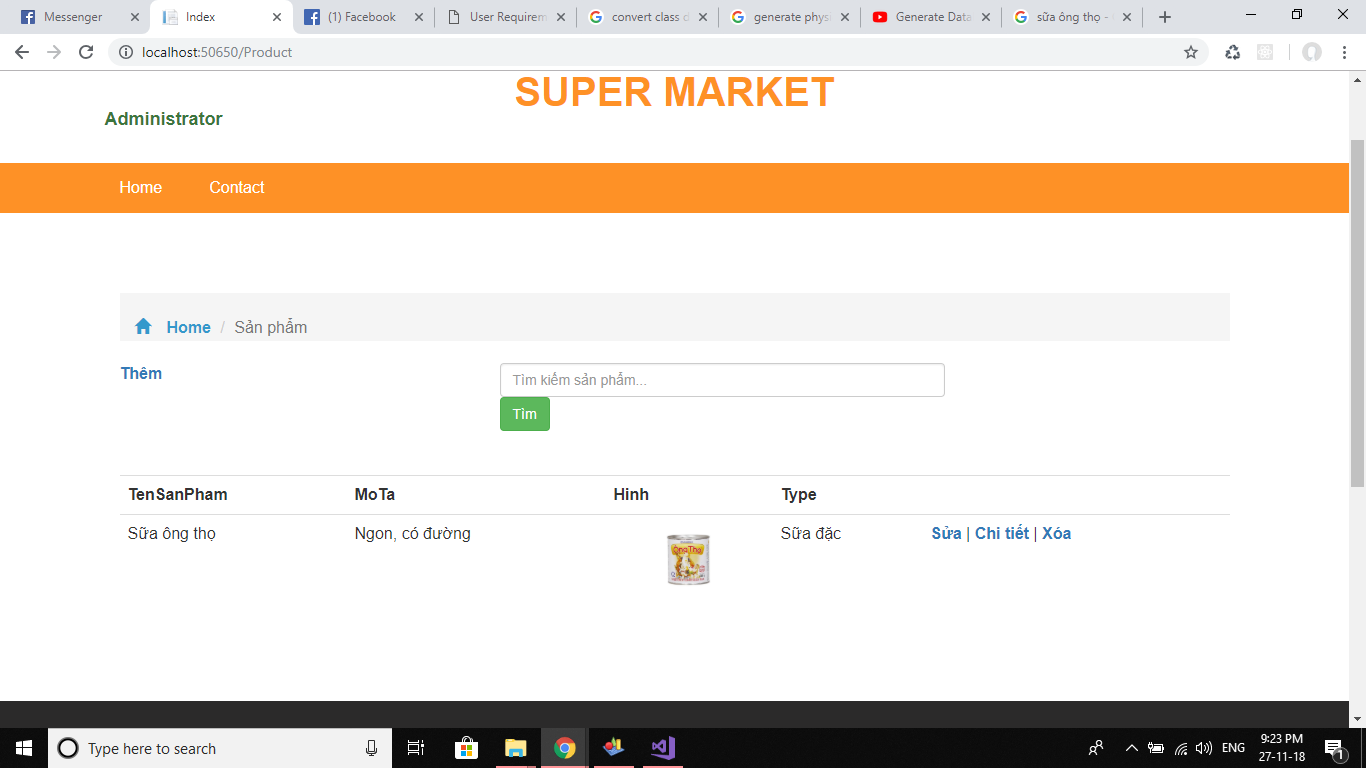
## Phần mềm quản lý siêu thị sử dụng CouchDB



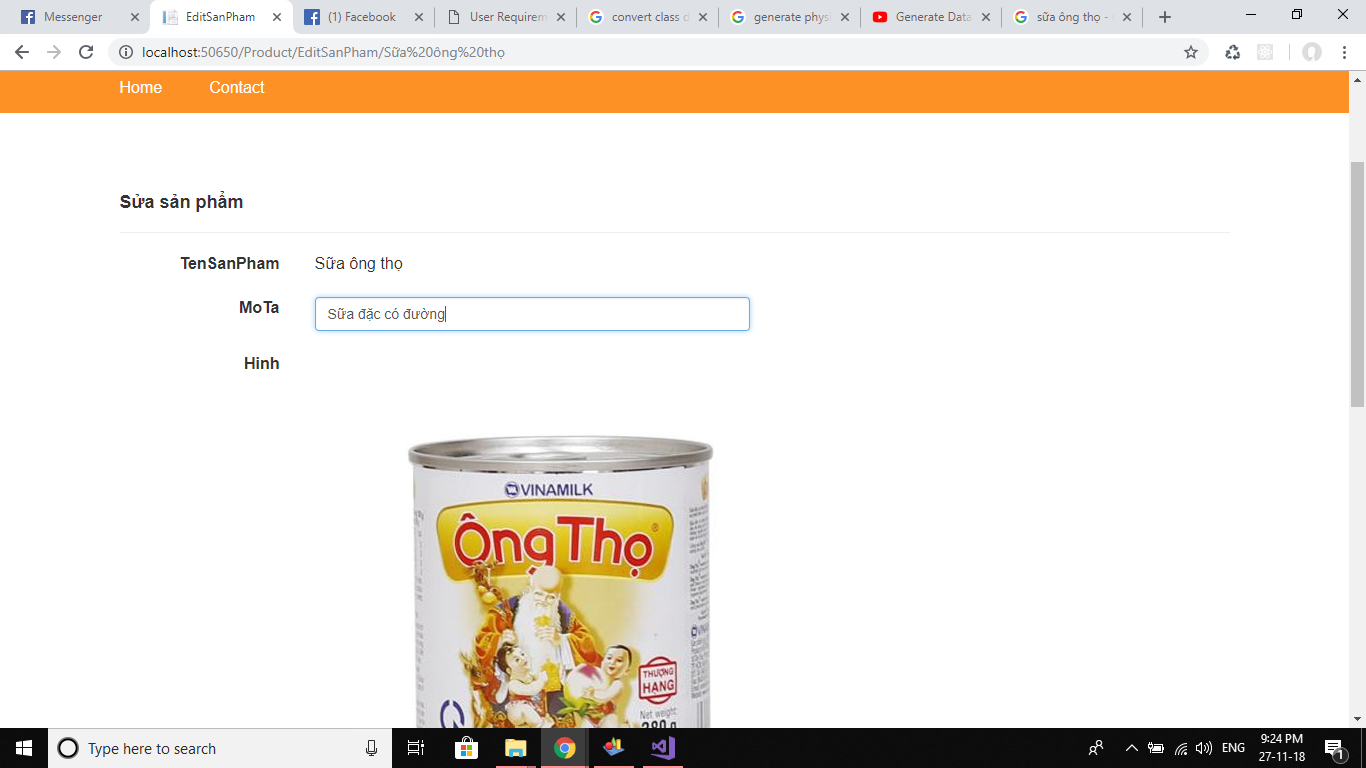
Màn hình chính



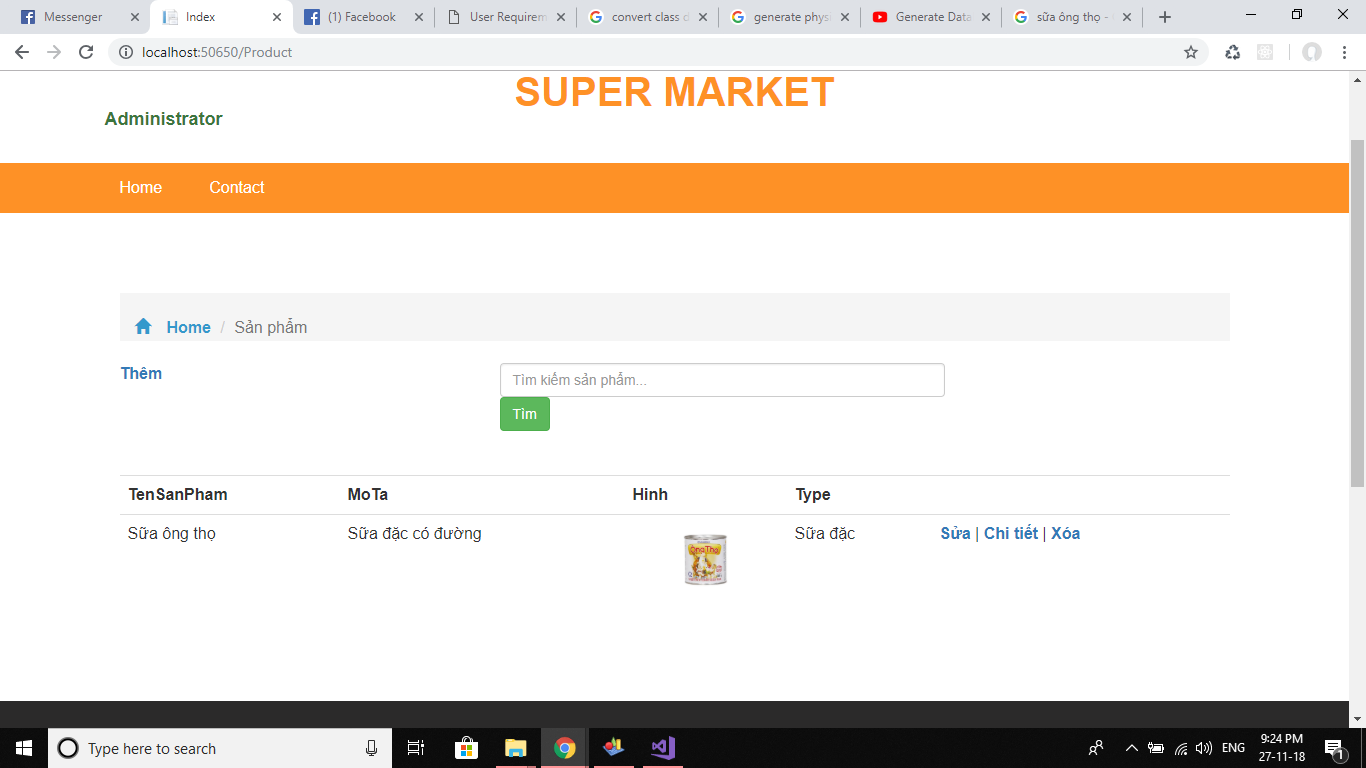
Màn hình thêm sản phẩm



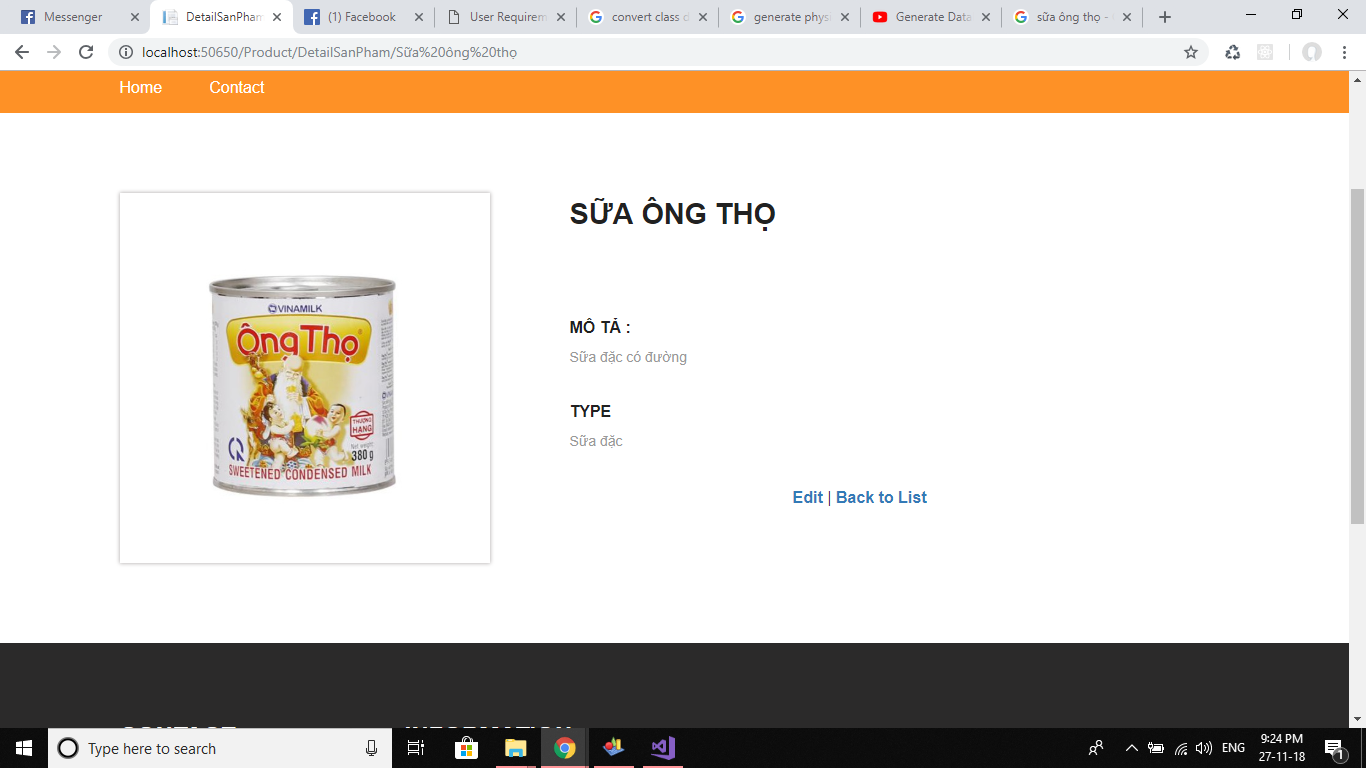
Sản phẩm sau khi được thêm vào



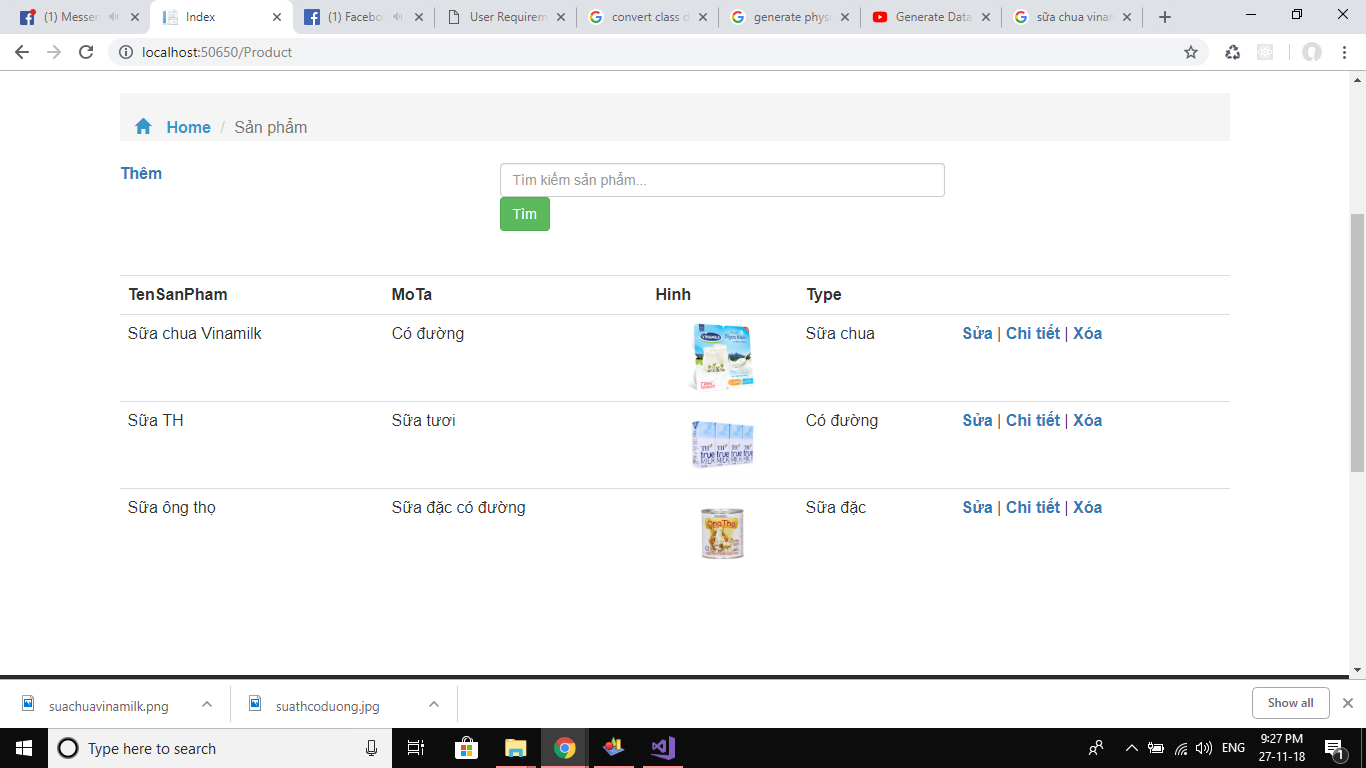
Màn hình chỉnh sửa sản phẩm



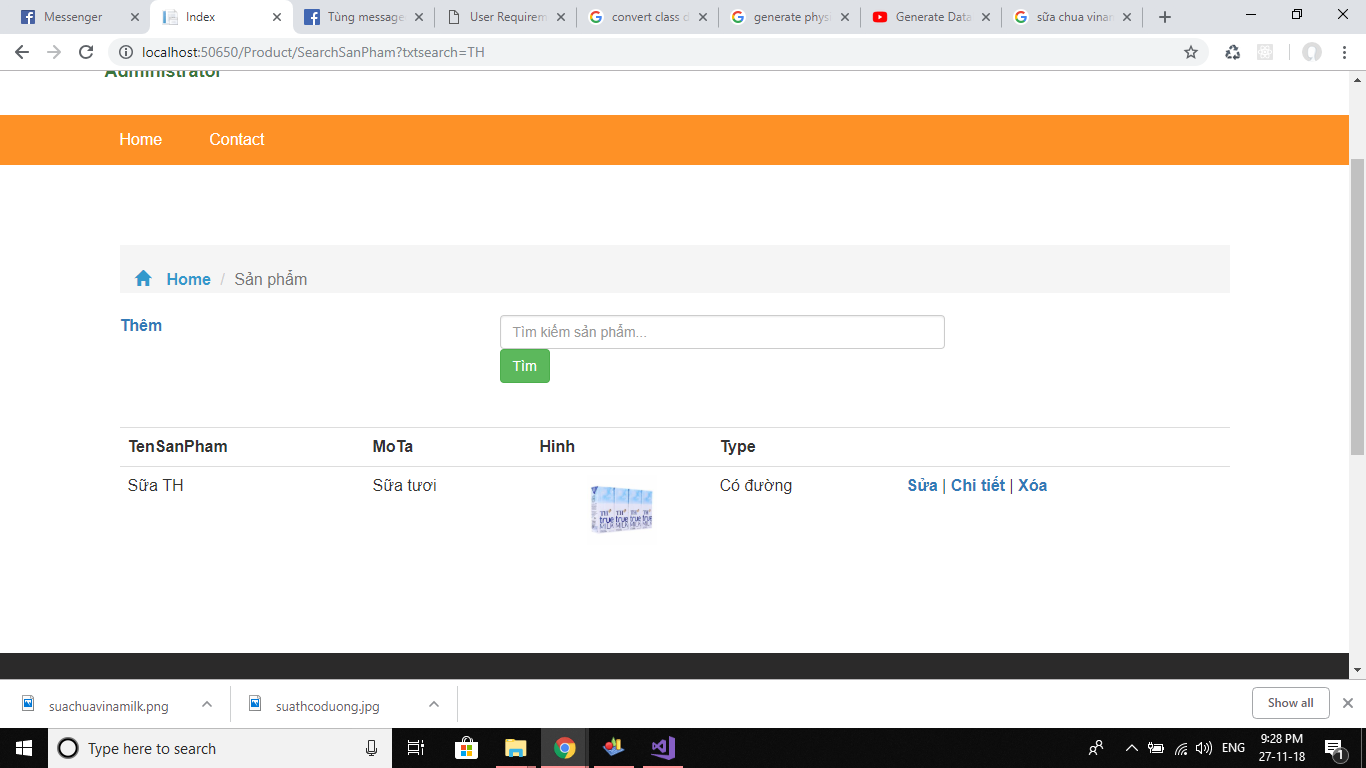
Sản phẩm sau khi được chỉnh sửa



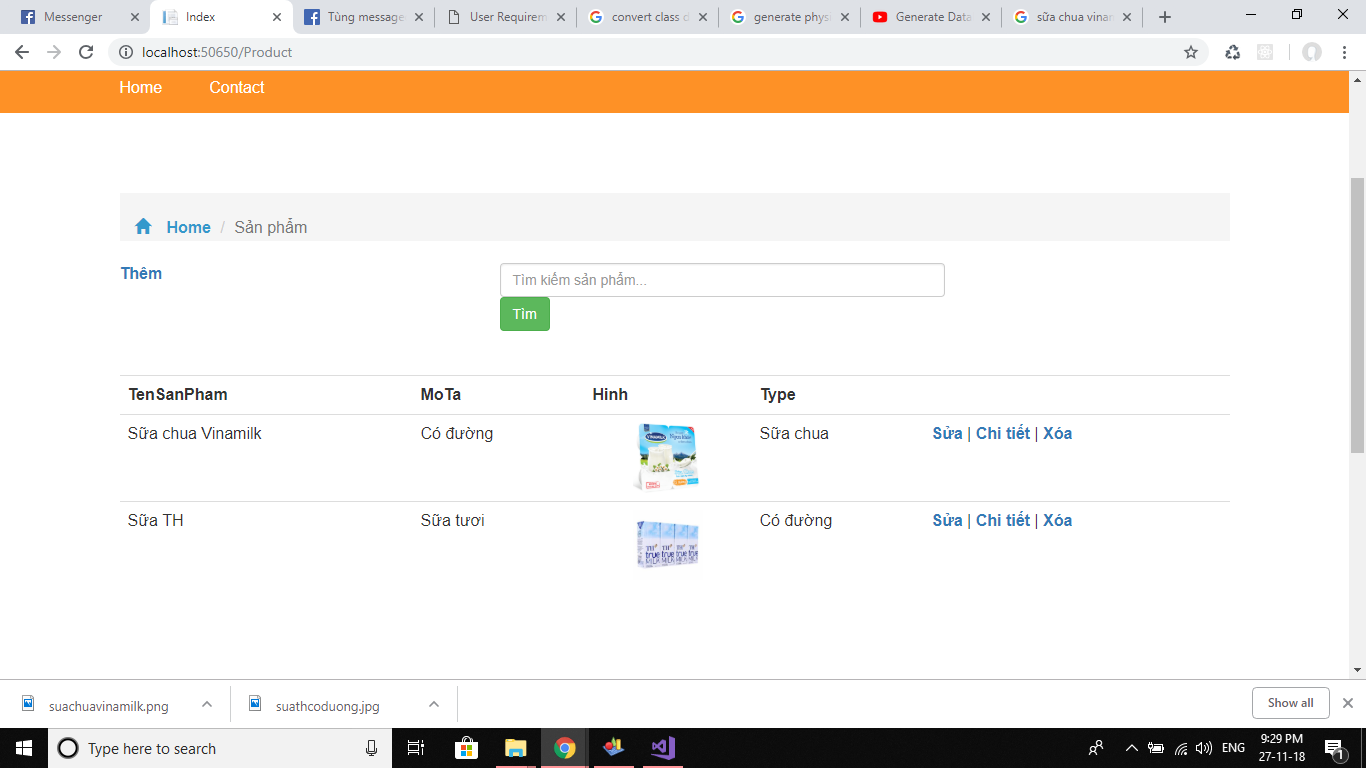
Màn hình xem chi tiết sản phẩm



Một số sản phẩm sau khi thêm



Màn hình tìm kiếm sản phẩm với từ khóa “TH”



Sản phẩm “Sữa ông thọ” sau khi bị xóa

# **Các nguồn tham khảo:**

<http://docs.couchdb.org/en/stable/intro/index.html>

<https://www.quora.com/What-are-the-pros-and-cons-of-CouchDB>

<https://stackoverflow.com/questions/7858699/disadvantages-of-couchdb>

https://db-engines.com/en/system/CouchDB;MongoDB