TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

\_\_🕮\_\_

**ĐỒ ÁN MÔN CƠ SỞ DỮ LIỆU NÂNG CAO**

**ĐỀ TÀI: TÌM HIỂU VỀ MONGODB**

**Giảng viên hướng dẫn:**Ths. Lương Trần Hy Hiến

**Sinh viên thực hiện:**Nhóm 16

Đặng Minh Cường – 42.01.104.210

Trần Lương Phán – 41.01.104.091

Phạm Thị Thanh My – 42.01.104.085

Vũ Tú Nhi – 42.01.104.100

*Thành phố Hồ Chí Minh – năm 2018*

🙣 Mục lục 🙡

[**I.Giới thiệu về MongoDB:** 3](#_Toc528835432)

[1.NoSQL là gì? 3](#_Toc528835433)

[2.MogoDB là gì? 3](#_Toc528835434)

[3.Lịch sử phát triển MongoDB. 4](#_Toc528835435)

[**II.Các thuật ngữ hay sử dụng trong MongoDB:** 5](#_Toc528835436)

[**III. Cài đặt và sử dụng** 6](#_Toc528835437)

[1. Tải MongoDB Community Edtion 6](#_Toc528835438)

[2. Thêm MongoDB vào System Path của máy tính 9](#_Toc528835439)

[3. Tạo thư mục database và log 11](#_Toc528835440)

[4. Kết nối tới MongoDB 11](#_Toc528835441)

[**IV. Ưu điểm và nhược điểm** 12](#_Toc528835442)

[1. So sánh MySQL với MongoDB 12](#_Toc528835443)

[2. Ưu điểm MongoDB 13](#_Toc528835444)

[3. Nhược điểm MongoDB 13](#_Toc528835445)

[4. Khi nào nên sử dụng MongoDB 13](#_Toc528835446)

[**V. Truy vấn dữ liệu trên MongoDB** 14](#_Toc528835447)

[1. Lệnh dùng trong Database: 14](#_Toc528835448)

[2. Lệnh dùng trong collection: 16](#_Toc528835449)

[**VI. Truy vấn dữ liệu MongoDB trên ngôn ngữ C#/.NET:** 21](#_Toc528835450)

[1. Cài đặt các gói NuGet: 21](#_Toc528835451)

[2. Tạo Collection: 22](#_Toc528835452)

[3. Tạo / Chèn dữ liệu: 24](#_Toc528835453)

[4. Phương thức FIND: 25](#_Toc528835454)

[5. Biểu thức LINQ: 26](#_Toc528835455)

[6. Phương thức SKIP: 27](#_Toc528835456)

[7. Phương thức SORT: 28](#_Toc528835457)

[**VII. Phần Demo sản phẩm** 29](#_Toc528835458)

[1. Tạo môi trường 30](#_Toc528835459)

[2. Giới thiệu sản phẩm Demo 31](#_Toc528835460)

[2.1. Tạo giao diện chính cho trang *“quản lý món ăn”* 31](#_Toc528835461)

[2.2. Tạo chức năng thêm sản phẩm 35](#_Toc528835462)

[2.3. Tạo chức năng xem chi tiết sản phẩm 38](#_Toc528835463)

[2.4. Tạo chức năng sửa sản phẩm 41](#_Toc528835464)

[2.5. Tạo chức năng xóa sản phẩm 45](#_Toc528835465)

[2.6. Tạo chức năng tìm kiếm sản phẩm 47](#_Toc528835466)

**VIII.** **Tổng kết và đánh giá** 50

[1.Tổng kết](#_Toc528835466) 50

[2.Đánh giá](#_Toc528835466) 50

[**TÀI LIỆU KHAM KHẢO** 51](#_Toc528835467)

I.Giới thiệu về MongoDB:

Trong những năm gần đây, với sự ra đời và phát triển mạnh mẽ của NoSQL thì MongoDB cũng đang nhận được nhiều sự chú ý trong cộng đồng công nghệ. Điểm mạnh của NoSQL nói chung và MongoDB nói riêng đó là tính linh hoạt trong việc cấu trúc dữ liệu do đó nó giúp đáp ứng tốt với những thay đổi hay việc mở rộng cơ sở dữ liệu. Nếu bạn là người mới nghiên cứu về NoSQL thì đây là một cơ sở dữ liệu đầu tiên nên tiếp cận bởi nó dễ dàng cài đặt và sử dụng để tiếp cận với những ý tưởng lưu trữ mới.

1.NoSQL là gì?

NoSQL là 1 dạng CSDL mã nguồn mở không sử dụng Transact-SQL để truy vấn thông tin. NoSQL viết tắt bởi: None-Relational SQL, hay có nơi thường gọi là Not-Only SQL. CSDL này được phát triển trên Javascript Framework với kiểu dữ liệu JSON. (Cú pháp của JSON là “key:value”) NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS về tốc độ, tính năng, khả năng mở rộng, memory cache,...



*Hình 1.1.1: Carlo Strozzi*

*NoSQL ra đời năm 1998 bởi Carlo Strozzi khi ông lập mới một hệ cơ sở dữ liệu quan hệ mã nguồn mở nhanh và nhẹ không liên quan đến SQL Vào năm 2009, Eric Evans, nhân viên của Rackspace giới thiệu lại thuật ngữ NoSQL khi Johan Oskarsson của Last.fm muốn tổ chức một hội thảo về cơ sở dữ liệu nguồn mở phân tán. Thuật ngữ NoSQL đánh dấu bước phát triển của thế hệ CSDL mới: phân tán (distributed) + không ràng buộc (non-relational).*

2.MongoDB là gì?

MongoDB (bắt nguồn từ “humongous”) là một hệ cơ sở dữ liệu NoSQL mã nguồn mở được thiết kế theo kiểu hướng đối tượng trong đó các bảng được cấu trúc một cách linh hoạt cho phép các dữ liệu lưu trên bảng không cần phải tuân theo một dạng cấu trúc nhất định nào. Chính do cấu trúc linh hoạt này nên MongoDB có thể đượng dùng để lưu trữ các dữ liệu có cấu trúc phức tạp và đa dạng và không cố định (hay còn gọi là Big Data)

[](https://bigsonata.files.wordpress.com/2014/06/logo.png).

Hình 1.2.1: Logo của MongoDB

**MongoDB** đứng ở tốp đầu trong danh sách các Hệ quản trị cơ sở dữ liệu NoSQL. Thay cho việc lưu trữ dữ liệu vào các bảng có quan hệ với nhau như truyền thống, MongoDB lưu các dữ liệu cấu trúc dưới dạng giống với JSON(JavaScript Object Notation) và gọi tên là BSON

MongoDB có hai phiên bản Community (miễn phí) và Enterprise (trả phí). Mã nguồn, quản lý phiên bản đặt tại Github [1], phát hành theo giấy phép phần mềm mã nguồn mở GNU Affero General Public License (AGPL) hoặc Apache License tùy theo phiên bản. MongoDB được viết bằng ngôn ngữ lập trình C++ (80,1%), JavaScript (16,5%), Python (2,8%) và các ngôn ngữ lập trình khác (0,6%).  MongoDB là hệ quản trị cơ sở dữ liệu chạy trên nhiều hệ điều hành (Linux, Windows, MacOS), trên kiến trúc máy tính 32bit hoặc 64 bit (tuy nhiên không khuyến khích chạy trên máy tính 32bit do kích thước tập tin sẽ bị giới hạn bởi khả năng đánh địa chỉ ô nhớ của hệ điều hành).

3.Lịch sử phát triển mongoDB.

Dự án MongoDB được bắt đầu triển khai vào tháng 10 năm 2007 bởi công ty phần mềm **10gen** trong khi công ty này đang xây dựng một nền tảng như là dịch vụ (Platform as a Service) giống như Google App Engine. Năm 2009, công ty chuyển sang mô hình phát triển nguồn mở, với công ty cung cấp hỗ trợ thương mại và các dịch vụ khác. Năm 2013, 10gen đổi tên thành MongoDB Inc. Nó được một số tổ chức sử dụng trong thực tế như:

* Caigslist : Công ty làm việc trong lịch vực môi giới quảng cáo trên các website khác (giống adMicro của Việt Nam). MongoDB giúp cho công ty này quản lý hàng tỉ các bản ghi quảng cáo thuận tiện và nhanh chóng.
* Foursquare là một mạng xã hội gắn các thông tin địa lý. Công ty này cần lưu dữ liệu của rất rất nhiều vị trí của các địa điểm như quán cafe, nhà hàng, điểm giải trí, lịch sử, … và ghi lại những nơi mà người sử dụng đã đi qua.
* CERN : Trung tâm nghiên cứu năng lượng nguyên tử của Châu Âu, sử dụng MongoDB để lưu trữ lại các kết quả, dữ liệu thí nghiệm của mình. Đây là một lượng dữ liệu khổng lồ sẽ dùng để sử dụng trong tương lai.

II.Các thuật ngữ hay sử dụng trong MongoDB:

**\_id**: Là trường bắt buộc có trong mỗi document. Trường \_id đại diện cho một giá trị duy nhất trong document MongoDB. Trường \_id cũng có thể được hiểu là khóa chính trong document. Nếu bạn thêm mới một document thì MongoDB sẽ tự động sinh ra một \_id đại diện cho document đó và là duy nhất trong cơ sở dữ liệu MongoDB.

**Collection**: Là nhóm của nhiều document trong MongoDB. Collection có thể được hiểu là một bảng tương ứng trong cơ sở dữ liệu RDBMS (Relational Database Management System). Collection nằm trong một cơ sở dữ liệu duy nhất. Các collection không phải định nghĩa các cột, các hàng hay kiểu dữ liệu trước.

**Cursor**: Đây là một con trỏ đến tập kết quả của một truy vấn. Máy khách có thể lặp qua một con trỏ để lấy kết quả.

**Database**: Nơi chứa các Collection, giống với cơ sở dữ liệu RDMS chúng chứa các bảng. Mỗi Database có một tập tin riêng lưu trữ trên bộ nhớ vật lý. Một mấy chủ MongoDB có thể chứa nhiều Database.

**Document**: Một bản ghi thuộc một Collection thì được gọi là một Document. Các Document lần lượt bao gồm các trường tên và giá trị.

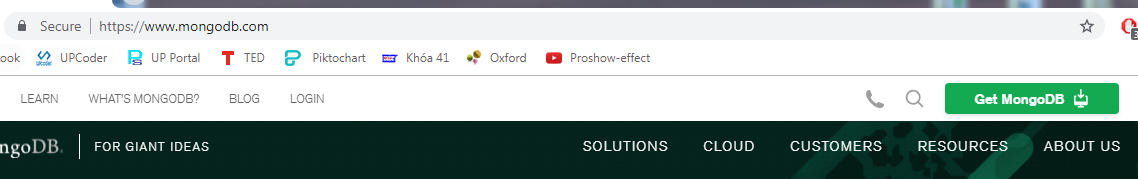
**Field**: Là một cặp name – value trong một document. Một document có thể có không hoặc nhiều trường. Các trường giống các cột ở cơ sở dữ liệu quan hệ.

**JSON**: Viết tắt của JavaScript Object Notation. Con người có thể đọc được ở định dạng văn bản đơn giản thể hiện cho các dữ liệu có cấu trúc. Hiện tại JSON đang hỗ trợ rất nhiều ngôn ngữ lập trình.

**Index**: Là những cấu trúc dữ liệu đặc biệt, dùng để chứa một phần nhỏ của các tập dữ liệu một cách dễ dàng để quét. Chỉ số lưu trữ giá trị của một fields cụ thể hoặc thiết lập các fields, sắp xếp theo giá trị của các fields này. Index hỗ trợ độ phân tích một cách hiệu quả các truy vấn. Nếu không có chỉ mục, MongoDB sẽ phải quét tất cả các documents của collection để chọn ra những document phù hợp với câu truy vấn. Quá trình quét này là không hiệu quả và yêu cầu MongoDB để xử lý một khối lượng lớn dữ liệu.

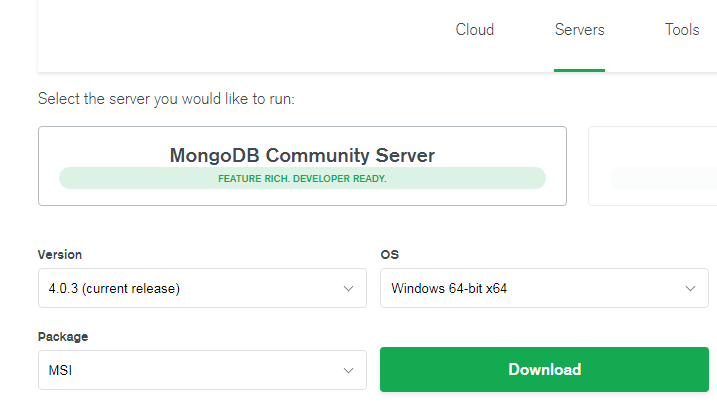
III. Cài đặt và sử dụng:

1. Tải MongoDB Community Edtion

1. Truy cập vào [*www.mongodb.com*](http://www.mongodb.com) và nhấn vào nút xanh lá trên cùng bên phải Get MongoDB

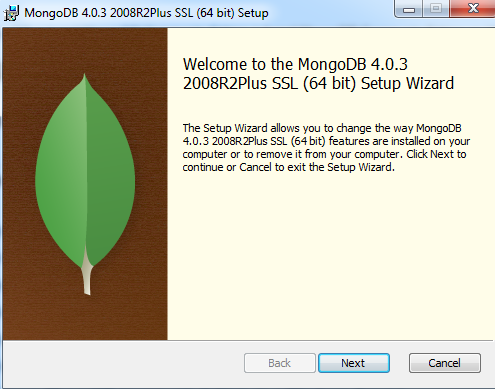
*Hình 3.1.1:* *Truy cập vào* [*www.mongodb.com*](http://www.mongodb.com)

1. Bấm vào Tab Servers , chọn MongoDB Community Server. Hộp Version – phiên bản và OS- hệ điều hành để hệ thống Web tự kiểm tra và lựa chọn phù hợp. Hộp Package chọn MSI và nhấn nút Download

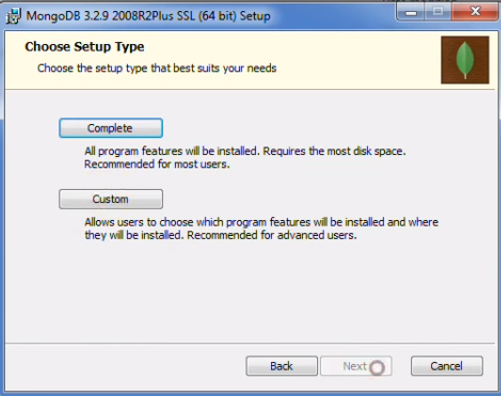


*Hình 3.1.2: Download mongoDB*

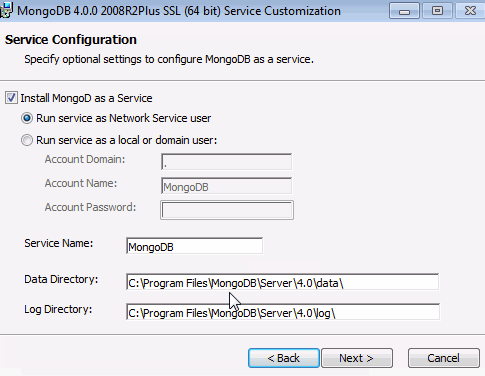
1. Bật file **.msi** sau khi đã tải về và cài đặt theo từng bước , nhấn nút **Next** để qua các bước tiếp theo



*Hình 3.1.3:Nhấn nút Next để cài đặt mongoDB*

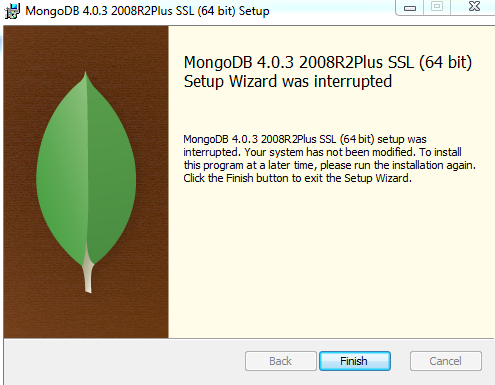
1. ****Chọn **Complete** setup để cài đặt đầy đủ chức năng và đường dẫn cài đặt theo mặc định là **C:\Program Files\MongoDB\Server\4.0**. (Có thể chọn **Custom** setup nếu muốn tùy chọn các chức năng muốn cài đặt và đường dẫn chứa MongoDB.)

*Hình 3.1.4: Chọn Complete để cài đặt đầy đủ chức năng*

1. Từ MongoDB phiên bản 4.0, có thể cấu hình và khởi động MongoDB như một service trong khi cài đặt, và service MongoDB được khởi động khi cài đặt thành công.

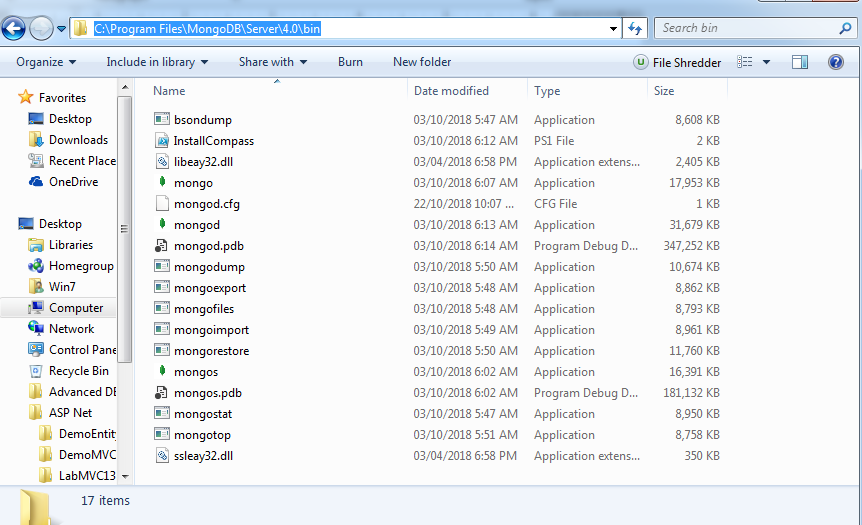
*Hình 3.1.5*

1. Kết thúc cài đặt, nhấp nút **Finish**.

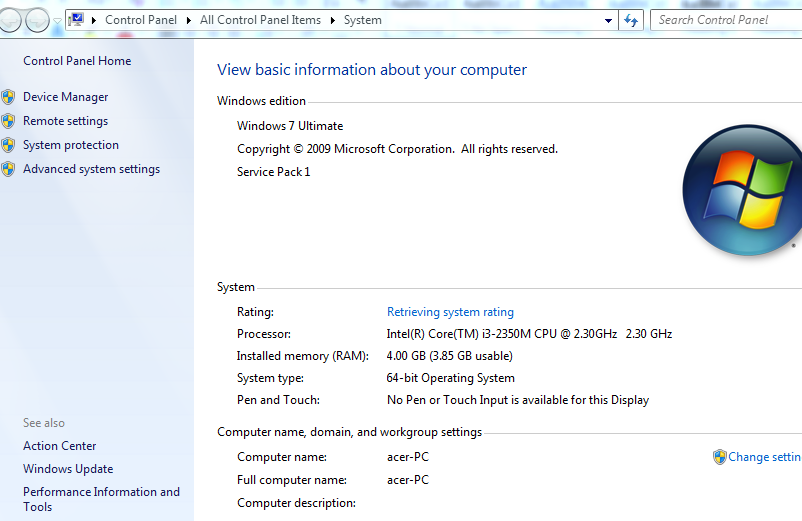


*Hình 3.1.6: Kết thúc cài đặt, nhấp nút Finish.*

2. Thêm MongoDB vào System Path của máy tính

1. Tìm thư đường dẫn chứa thư mục **bin** chương trình, mặc định ở **C:\Program Files\MongoDB\Server\4.0\bin** và copy đường dẫn.

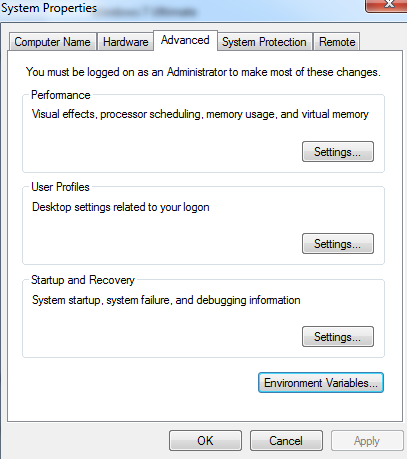
*Hình 3.2.1:* *Tìm thư đường dẫn chứa thư mục bin*

1. Chuột phải vào **My Computer** chọn **Properties** hoặc paste đường dẫn này **Control Panel\All Control Panel Items\System**. Chọn **Advanced System Settings**.

Hii

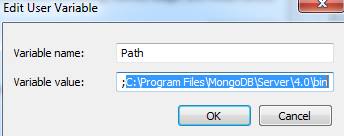
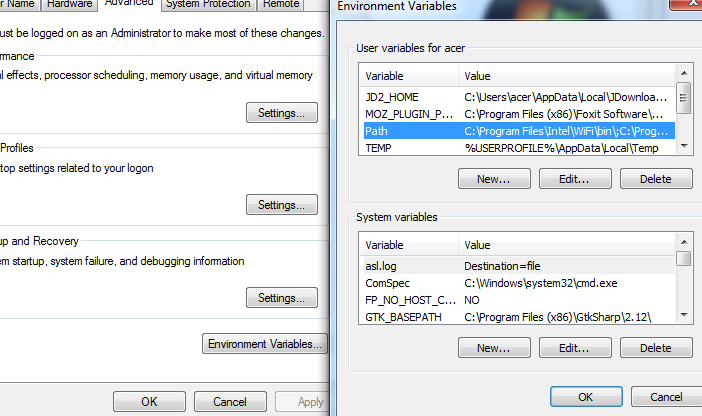
*Hình 3.2.2: Vào My Computer chọn Properties*

1. Ở tab **Advance**d, nhấn **Environment Variables…**



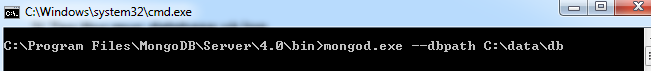
*Hình 3.2.3: Chọn Environment Variables*

1. Chọn **Path**, nhấn **Edit** và paste đường dẫn trên vào rồi nhấn **OK**

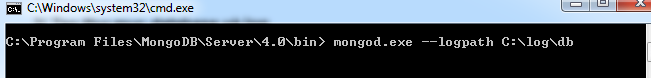


*Hình 3.2.4: Chọn Path và nhấn Edit và paste đường dẫn vào*

3. Tạo thư mục database và log

1. Tạo một đường dẫn thư mục **data\db** để chứa các dữ liệu của MongoDB , tạo ở một ổ bất kì bằng. Ví dụ **C:\data\db**. Sau đó gõ lệnh sau trong **cmd** : **C:\Program Files\ MongoDB\Server\4.0\bin> mongod.exe --dbpath C:\data\db**

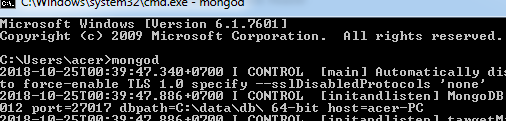
*Hình 3.3.1: Tạo một đường dẫn thư mục data\db*

1. Tạo một đường dẫn thư mục **log\db**, tạo ở một ổ bất kì bằng. Ví dụ **C:\data\db**. Sau đó gõ lệnh sau trong **cmd** : **C:\Program Files\ MongoDB\Server\4.0\bin> mongod.exe --logpath C:\log\db**

*Hình 3.3.2:* *Tạo một đường dẫn thư mục log\db*

4. Kết nối tới MongoDB

1. Bật cmd và gõ lệnh **mongod**



*Hình 3.4.1: thực hiện lệnh mongod*

1. Bật một cửa sổ cmd khác và gõ lệnh **mongo**

*Hình 3.4.2:* *Bật một cửa sổ cmd khác và gõ lệnh mongo*

1. Sau đó có thể gõ lệnh **show dbs**



*Hình 3.4.3: Gõ lệnh show dbs*

IV. Truy vấn dữ liệu trên MongoDB:

1. Lệnh dùng trong Database:

- Lệnh chọn Database để thao tác:

*use <Database\_name>*

Với Database\_name: tên Database bạn muốn lấy để thao tác.

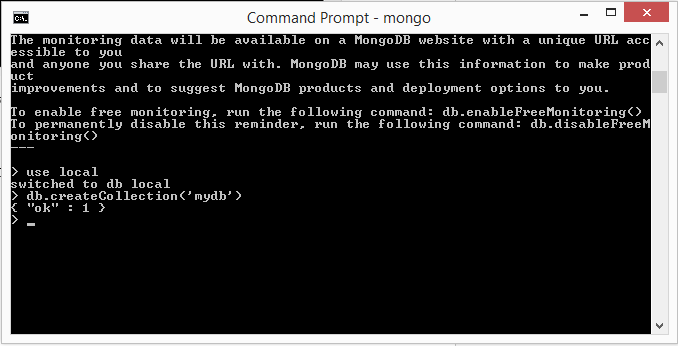


*Hình 4.1.1: Chạy lệnh use <database> trên cmd*

- Lệnh tạo collection: tạo 1 collection hoặc view trên database đã chọn

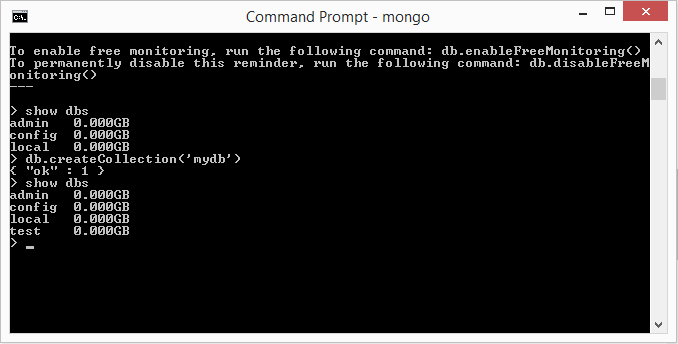
*db.createCollection(Collection\_name)*

Với Collection\_name là tên collection bạn muốn tạo.



*Hình 4.1.2: Chạy lệnh tạo collection trên cmd*

Nếu chưa chọn database, Mongo sẽ tự tạo 1 database “test” để chứa collection

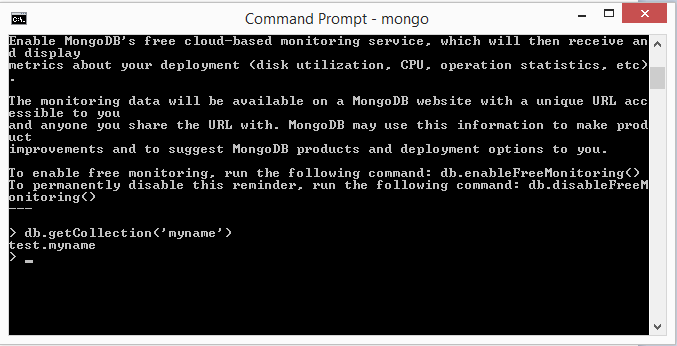


*Hình 4.1.3: Mongo tự tạo database chứa collection*

- Lệnh truy cập collection: Trả về một collection hoặc đối tượng View. Được sử dụng để truy cập các collection có tên hợp lệ trong Mongo Shell, nếu collection không tồn tại, Mongo sẽ tự tạo mới collection có tên tương ứng.

*db.getCollection(name)*

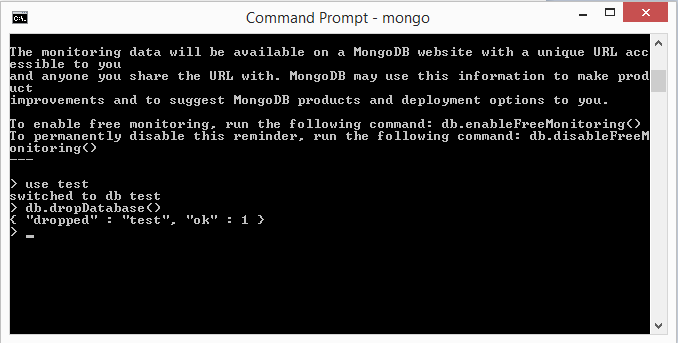
Với name: tên collection bạn muốn lấy.



*Hình 4.1.4: Chạy lệnh lấy collection trên cmd*

- Lệnh xóa Database: Xóa Database đang thao tác.

*db.dropDatabase()*

**

*Hình 4.1.5: Chạy lệnh xóa Database*

2. Lệnh dùng trong collection:

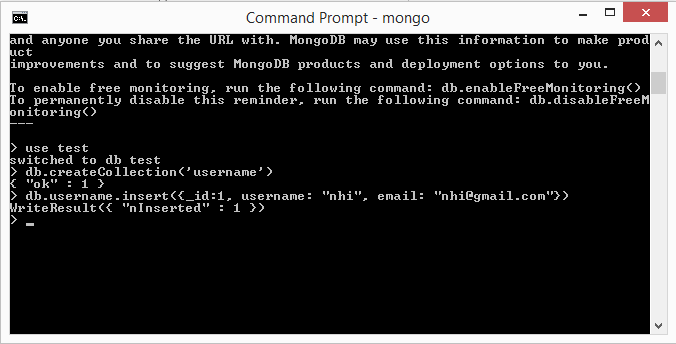
- Lệnh thêm document vào collection: Có thể thêm 1 hoặc nhiều document vào collection, nếu collection không tồn tại thì lệnh sẽ tự tạo collection cùng tên.

*db.collection.insert({document})*

Với:

+ collection: tên collection muốn thêm document.

+ document: một document hoặc mảng document cần thêm.



*Hình 4.2.1: Chạy lệnh thêm document trên cmd*

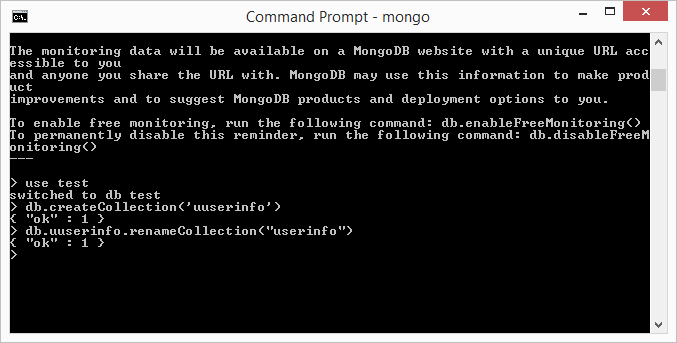
- Lệnh đổi tên collection:

*db.collection.renameCollection(new\_name)*

Với:

+ collection: tên collection muốn đổi tên.

+ new\_name: tên collection mới.



*Hình 4.2.2: Đổi tên collection từ uuserinfo thành userinfo*

- Lệnh update collection: cập nhật một hay nhiều document của một collection.

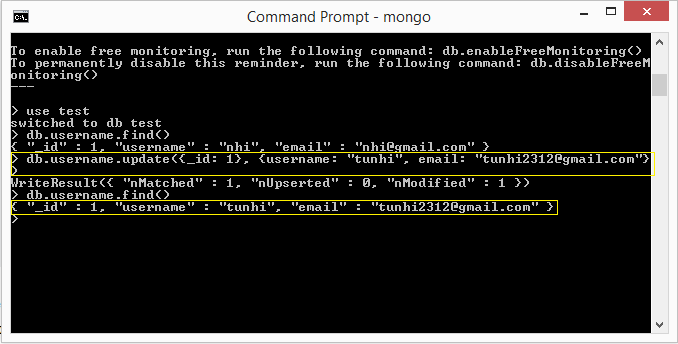
*db.collection.update(query, update, options)*

Với:

+ query: tiêu chí để update.

+ update: document muốn update.

+ options: các document khác không bắt buộc.

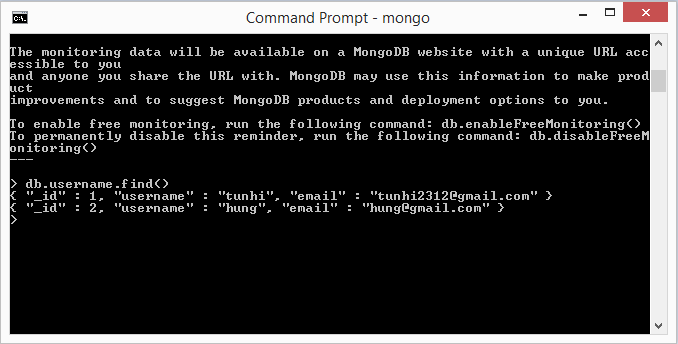


*Hình 4.2.3: Update thông tin user có “\_id = 1”*

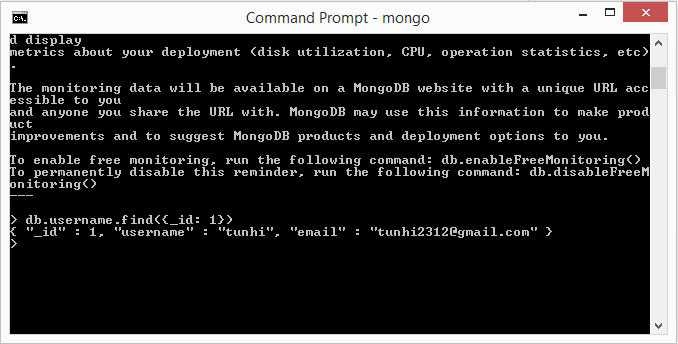
- Lệnh truy xuất document: truy xuất document theo tiêu chí được chỉ định.

*db.collection.find(query)*

Với query: tiêu chí truy xuất document, nếu không có tiêu chí, Mongo sẽ trả về tất cả document có trong collection.



*Hình 4.2.4: Lệnh find mặc định trả về tất cả document đang có trong collection*

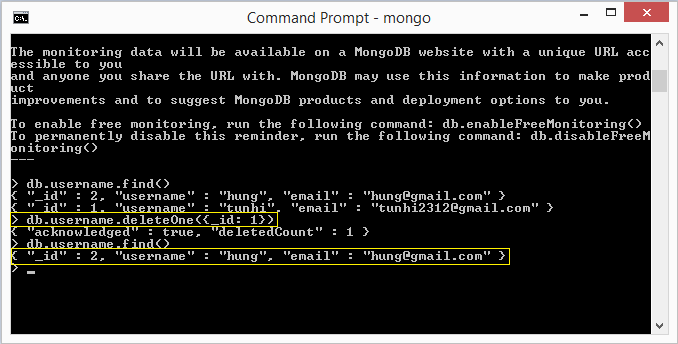


*Hình 4.2.5: Truy xuất thông tin của thành viên có “\_id =1”*

- Lệnh xóa một document: xóa một document theo tiêu chí được chỉ định.

*db.collection.deleteOne(query)*

Với query: tiêu chí truy xuất document bằng toán tử truy vấn.

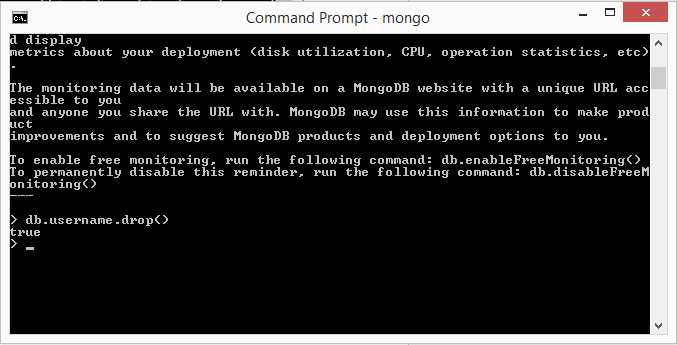


*Hình 4.2.6: Xóa thông tin user có “\_id = 1”*

- Lệnh xóa collection:

*db.collection.drop()*

Với collection: tên collection muốn xóa.



*Hình 4.2.7: Xóa collection tên username*

V. Truy vấn dữ liệu MongoDB trên ngôn ngữ C#/.NET:

1. Cài đặt các gói NuGet:

*Để cài đặt trình điều khiển, chúng ta cần cài đặt các gói NuGet cần thiết:*

*MongoDB.Bson* : Thư viện BSON độc lập xử lý việc chuyển đổi các POCO thành các kiểu BSON (là định dạng tệp cho MongoDB) và ngược lại.

*MongoDB.Driver.Core*: Đây là một driver của chính nó và có các thành phần cốt lõi của driver (như cách kết nối với mongoDB, kết nối tổng hợp, và các kết nối khác) để giao tiếp từ .Net đến MongoDB và ngược lại, có sự phụ thuộc vào MongoDB.Bson.

*MongoDB.Driver* : Phụ thuộc vào Driver.Core, thư viện mà phụ thuộc vào MongoDB.Bson. Nó dễ sử dụng API hơn các thành phần của driver, có các phương thức async và hỗ trợ truy vấn với LINQ.

Chạy lệnh sau để có được tất cả ba gói được cài đặt cùng một lúc:

***Install-Package******MongoDB.Driver***

Để kết nối với một cơ sở dữ liệu, chúng ta sử dụng lớp MongoClient để truy cập vào mongoDB và thông qua nó, chọn cơ sở dữ liệu chúng ta muốn sử dụng. Lớp này có bốn hàm tạo.

- Một contructor không tham số theo mặc định sẽ kết nối trên cổng 27017:

***var*** *client =* ***new*** *MongoClient();*

- Truyền tham số chuỗi kết nối:

***var*** *connectionString = "mongodb://localhost:27017";*

***var*** *client =* ***new*** *MongoClient(connectionString);*

- Sử dụng MongoUrl tương tự như sử dụng hàm tạo connectionstring có tham số. Bạn có thể tạo hàm này bằng cách sử dụng hàm tạo mặc định:

***var*** *client =* ***new*** *MongoClient(****new*** *MongoUrl("mongodb://localhost:27017"));*

-Sử dụng phương thức Create (tĩnh) từ lớp Client:

***var*** *client =* ***new*** *MongoClient(MongoUrl.Create("mongodb://localhost:27017"));*

Với MongoClient, có nhiều thứ chúng ta có thể thao tác như xóa một cơ sở dữ liệu, lấy một cơ sở dữ liệu, hoặc lấy tên một trong các cơ sở dữ liệu trên máy chủ. Không khó khăn để tạo một cơ sở dữ liệu vì một khi bạn chọn một cơ sở dữ liệu và chèn dữ liệu vào nó, nó sẽ tự động tạo ra cơ sở dữ liệu.

Phương pháp GetDatabase sẽ tự động tạo cơ sở dữ liệu theo cú pháp:

*IMongoDatabase db = client.GetDatabase("Database\_name");*

Với:Database\_name là tên cơ sở dữ liệu bạn muốn.

Các phương thức GetDatabase sẽ trả về một đối tượng là một đại diện của một cơ sở dữ liệu, từ đó chúng ta có thể truy cập vào các Collection khác nhau và thao tác với cơ sở dữ liệu. Với đối tượng trên cơ sở dữ liệu, bạn có thể tạo, đổi tên, truy xuất hoặc nhận danh sách các Collection từ cơ sở dữ liệu. Tài liệu được lưu trữ trong Collection, vì vậy bạn có thể nghĩ về một Collection dưới dạng bảng và tài liệu dưới dạng bản ghi trong bảng, nếu bạn quen sử dụng SQL.

2. Tạo Collection:

Để tạo một Collection, chúng ta sử dụng phương thức CreateCollection hoặc phương thức CreateCollectionAsync của đối tượng ImongoDatabase. Phương thức này có ba tham số (trong đó hai tham số cuối là không bắt buộc): tên của Collection, khởi tạo Collection options, CancellationToken. Cú pháp như sau:

***void******CreateCollection****(*

***string*** *name,*

*CreateCollectionOptions options = null,*

*CancellationToken cancellationToken = null*

*)*

*Task* ***CreateCollectionAsync****(*

***string*** *name,*

*CreateCollectionOptions options = null,*

*CancellationToken cancellationToken = null*

*)*

Bạn có thể tạo một Collection theo mặc định, bằng cách chỉ thiết lập tên của Collection:

***await*** *db.CreateCollectionAsync("Collection\_name");*

Với: Collection\_name là tên của Collection.

CreateCollectionOptions chỉ định các cài đặt cho một Collection, ví dụ như số lượng tối đa của dữ liệu nó chứa. Đây là một ví dụ:

***await*** *db.CreateCollectionAsync("students",* ***new*** *CreateCollectionOptions*

*{*

*AutoIndexId = false,*

*MaxDocuments = 25,*

*Capped = true*

*});*

GetCollection:

Sau khi tạo một Collection, nó sẽ yêu cầu một bước kiểm tra xem Collection có tồn tại hay không, và sau đó đổ dữ liệu vào Collection. GetCollection tự động tạo ra một Collection nếu không tồn tại và bổ sung dữ liệu vào Collection đó. Cách gọi phương thức GetCollection:

***static******async*** *Task* ***MainAsync****()*

*{*

*......*

*IMongoCollection<BsonDocument> collection = db.GetCollection<BsonDocument>("Collection\_name");*

*}*

Với Collection\_name là tên của Collection.

BsonDocument là một dạng của MongoDB.Bson package, thư viện đại diện cho một tài liệu BSON và với dạng này, chúng ta có thể làm việc với bất kỳ dạng dữ liệu nào. Gói này chứa tất cả các dạng BSON cơ bản và một vài thứ khác để làm việc với BSON.

Trong gói này, chúng ta có các lớp đại diện cho các kiểu BSON và cách ánh xạ giữa các kiểu .NET và BsonValues. Một vài trong số đó là:

- BsonDocument như trong hàm gọi GetCollection trên.

- BsonElement đại diện cho một phần tử BSON.

- BsonValue là một lớp cơ sở trừu tượng được sử dụng bởi các lớp con khác nhau như BsonString, BsonInt32 và nhiều thứ khác.

BsonDocument là từ điển của chuỗi giá trị BSON, vì vậy chúng ta có thể khởi tạo như sau:

***var*** *document =* ***new*** *BsonDocument*

*{*

*{"firstname", BsonValue.Create("Peter")},*

*{"lastname",* ***new*** *BsonString("Mbanugo")},*

*{ "subjects",* ***new*** *BsonArray(****new****[] {"English", "Mathematics", "Physics"}) },*

*{ "class", "JSS 3" },*

*{ "age",* ***int****.MaxValue }*

*};*

hoặc sử dụng phương pháp Add có một số tình trạng quá tải:

***var*** *document =* ***new*** *BsonDocument();*

*document.Add("name", "Steven Johnson");*

*document.Add("age", 23);*

*document.Add("subjects",* ***new*** *BsonArray() {"English", "Mathematics", "Physics"});*

*hoặc sử dụng indexer:*

*document["class"] = "JSS 3";*

3. Tạo / Chèn dữ liệu:

Các dữ liệu được lưu trữ trong một Collection. Mongo Collection cung cấp các phương thức để chèn một dữ liệu duy nhất tại một thời điểm hoặc nhiều dữ liệu cùng một lúc. Để làm điều này, chúng ta phải:

Tạo một đối tượng kiểu Imongocollection để đại diện cho Collection mà chúng ta muốn thao tác:

***var*** *collection = db.GetCollection<BsonDocument>("Collection\_name");*

Với Collection\_name là tên của Collection.

Tiếp theo tạo dữ liệu mà bạn muốn chèn.

Ví dụ:

***var*** *document =* ***new*** *BsonDocument*

*{*

*{"firstname", BsonValue.Create("Peter")},*

*{"lastname",* ***new*** *BsonString("Mbanugo")},*

*{ "subjects",* ***new*** *BsonArray(****new****[] {"English", "Mathematics", "Physics"}) },*

*{ "class", "JSS 3" },*

*{ "age", 45}*

*};*

Và cuối cùng chèn dữ liệu:

Sử dụng ***await*** *collection.InsertOneAsync(document);*

Hoặc *collection.InsertOne(document);*

Bạn cũng có thể chèn nhiều dữ liệu cùng một lúc bằng cách sử dụng phương thức InsertMany hoặc phương thức InsertManyAsync. Giả sử chúng ta có ba học sinh mới trong trường, chúng ta có thể chèn tất cả cùng một lúc bằng cách sử dụng phương thức này, chúng sẽ được chèn vào cùng một lúc.

***await*** *collection.InsertManyAsync(threestudent);*

4. Phương thức FIND:

Phương thức này trả về một giao diện IFindFluent. Đây là một giao diện cung cấp cho chúng ta cú pháp đơn giản cho những thao tác như như Count, Skip, Sort, và Limit. Từ IfindFluent chúng ta cũng có thể trả về một con trỏ (bằng cách gọi ToCursor hoặc ToCursorAsync) hoặc một List (bằng cách gọi ToList hoặc ToListAsync). Chúng ta có thể in tất cả các dữ liệu bằng cách sử dụng phương thức Find, Ví dụ:

*await collection.Find(FilterDefinition<BsonDocument>.Empty)*

*.ForEachAsync(doc => Console.WriteLine(doc));*

Tìm dữ liệu cụ thể:

Thay vì truy xuất tất cả dữ liệu, chúng ta có thể chỉ định truy vấn một dữ liệu phù hợp với Find. Có thể truy vấn một dữ liệu cụ thể bằng các cách sau:

- Sử dụng BsonDocument hoặc chuỗi

Chúng ta có thể định nghĩa một BsonDocument như một bộ lọc hay truy vấn, phương thức này sẽ tìm các dữ liệu khớp với các trường được định nghĩa trong chương trình. Ví dụ, tìm học sinh có tên “Peter”:

+ Sử dụng BsonDocument:

*var filter = new BsonDocument("FirstName", "Peter");*

*await collection.Find(filter)*

*.ForEachAsync(document => Console.WriteLine(document));*

+ Sử dụng chuỗi:

*var filter = "{ FirstName: 'Peter'}";*

*await collection.Find(filter)*

*.ForEachAsync(document => Console.WriteLine(document));*

- Sử dụng FilterDefinitionBuilder:

Bạn có thể sử dụng FilterDefinitionBuilder, một Builder cho FilterDefinition. Nó cung cấp một bộ các phương thức để xây dựng các truy vấn, và Lt, là một trong số các phương thức. Chúng ta có thể truy vấn bằng cách sử dụng FilterDefinitionBuilder như sau:

***var*** *filter =* ***new*** *FilterDefinitionBuilder<BsonDocument>().Lt("age", 25);*

Hoặc sử dụng phương thức overload nhận một biểu thức LINQ:

***var*** *filter =* ***new*** *FilterDefinitionBuilder<Student>().Lt( student => student.Age, 25);*

Ngoài ra, bạn có thể sử dụng một Builders lớp tĩnh để xây dựng một định nghĩa bộ lọc và lớp này cũng có các phương thức trợ giúp tĩnh để xây dựng các thứ khác, như định nghĩa tham chiếu, định nghĩa sắp xếp và một số thứ khác.

*var filter = Builders<BsonDocument>.Filter.Lt("age", 25);*

*var filter = Builders<Student>.Filter.Lt(student => student.Age, 25);*

Driver cũng quá tải 3 toán tử cho định nghĩa bộ lọc. Gồm các toán tử and (&), or (|) và not (!). Ví dụ, chúng tôi muốn học sinh có độ tuổi dưới 25 và tên là Peter, chúng ta có thể xây dựng truy vấn bằng cách sử dụng builder helper và quá tải toán tử & như sau:

var builder = Builders<BsonDocument>.Filter;

var filter = builder.Lt("Age", 40) & builder.Eq("FirstName", "Peter");

5. Biểu thức LINQ:

Các quá tải toán tử trong các phương thức trên có biểu thức LINQ và khi chúng ta có một đối tượng, chúng ta có thể xây dựng một truy vấn bằng cách sử dụng biểu thức LINQ. Ví dụ, chúng ta muốn truy xuất những sinh viên có độ tuổi dưới 25 và tên đầu tiên không phải là Peter và in ra họ và tên của họ:

***var*** *collection = db.GetCollection<Student>("students");*

***await*** *collection.Find(student => student.Age < 25 && student.FirstName != "Peter")*

*.ForEachAsync(student => Console.WriteLine(student.FirstName + " " + student.LastName));*

Với MongoDB, bạn có thể giới hạn số lượng dữ liệu bằng cách gọi Limit phương thức của IfindFluent trả về từ việc gọi Find. Ví dụ, truy sinh viên ở độ tuổi dưới 40, giới hạn kết quả của dữ liệu tối đa là hai sinh viên, chúng ta gọi hàm Limit() với giá trị là 2:

***int*** *count = 1;*

***await*** *collection.Find(x => x.Age < 40)*

*.Limit(2)*

*.ForEachAsync(*

*student =>*

*{*

*Console.WriteLine($"S/N: {count} \t Id: {student.Id}, FirstName: {student.FirstName}, LastName: {student.LastName}");*

*count++;*

*});*

6. Phương thức SKIP:

Nếu chúng ta muốn cơ sở dữ liệu bỏ qua một số tài liệu, chúng ta sử dụng phương thức Skip.Ví dụ, bỏ qua tất cả sinh viên với độ tuổi dưới 40:

***int*** *count = 1;*

***await*** *collection.Find(x => x.Age < 40)*

*.Skip(1)*

*.ForEachAsync(*

*student =>*

*{*

*Console.WriteLine($"S/N: {count} \t Id: {student.Id}, FirstName: {student.FirstName}, LastName: {student.LastName}");*

*count++;*

*});*

7. Phương thức SORT:

Các phương thức Sort của giao diện cần SortDefinition, mặc nhiên sẽ chuyển đổi từ chuỗi hoặc BsonDocument giống như FilterDefinition. Vì vậy, nếu chúng ta muốn sắp xếp LastName theo một thứ tự tăng dần bằng cách sử dụng một chuỗi như định nghĩa:

***await*** *collection.Find(FilterDefinition<Student>.Empty)*

*.Skip((currentPage - 1) \* pageSize)*

*.Limit(pageSize)*

*.Sort("{LastName: 1}")*

*.ForEachAsync(*

*student =>*

*{*

*Console.WriteLine($"S/N: {count}, \t Id: {student.Id}, FirstName: {student.FirstName}, LastName: {student.LastName}, Age: {student.Age}");*

*count++;*

*});*

Và nếu chúng ta muốn sắp xếp LastName theo thứ tự giảm dần bằng cách sử dụng một BsonDocument:

***await*** *collection.Find(FilterDefinition<Student>.Empty)*

*.Skip((currentPage - 1) \* pageSize)*

*.Limit(pageSize)*

*.Sort(****new*** *BsonDocument("LastName", -1))*

*.ForEachAsync(*

*student =>*

*{*

*Console.WriteLine($"S/N: {count}, \t Id: {student.Id}, FirstName: {student.FirstName}, LastName: {student.LastName}, Age: {student.Age}");*

*count++;*

*});*

Chúng ta cũng có thể sử dụng SortDefinitionBuilder. Vì vậy, chúng tôi có thể sử dụng builder helper để tạo hàm sắp xếp như sau:

***await*** *collection.Find(FilterDefinition<Student>.Empty)*

*.Skip((currentPage - 1) \* pageSize)*

*.Limit(pageSize)*

*.Sort(Builders<Student>.Sort.Descending("LastName"))*

*.ForEachAsync(*

*student =>*

*{*

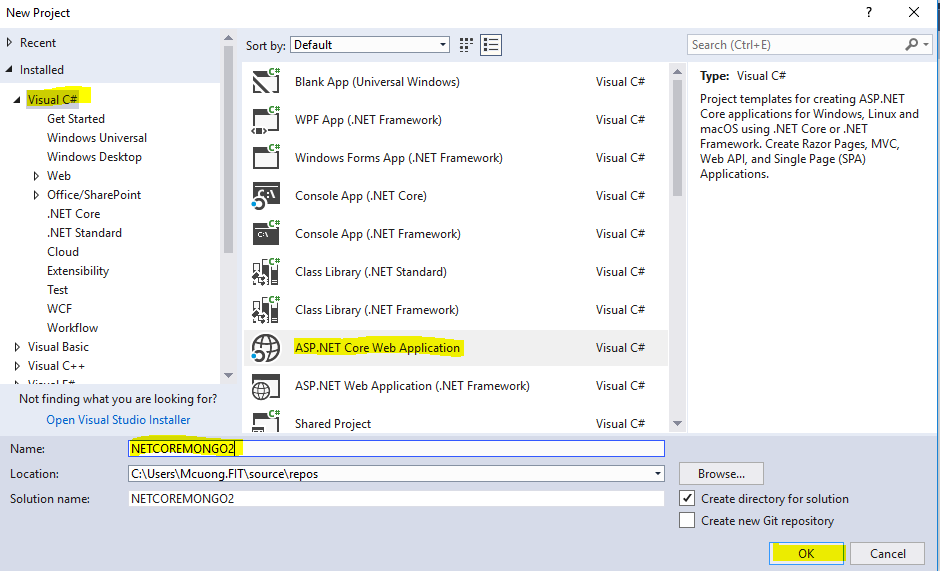
*Console.WriteLine($"S/N: {count}, \t Id: {student.Id}, FirstName: {student.FirstName}, LastName: {student.LastName}, Age: {student.Age}");*

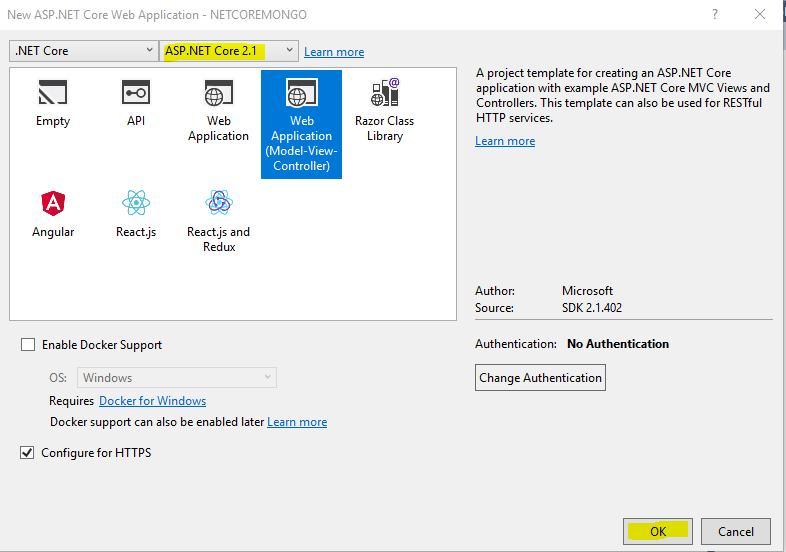
*count++;*

*});*

VI. Phần Demo sản phẩm:

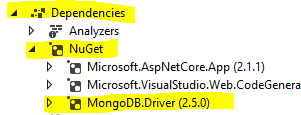
*Sản phẩm demo của nhóm sử dụng NoSQL mongoDB 4.0 và**ASP.NET CORE MVC2.1*

1. Tạo môi trường

*Hình 7.1.1: Chọn ASP.NET Core Web Application và đặt tên cho folder chứa sản phẩm*

*Hình 7.1.2: Chọn Web Application(Model-View-Controller)*

3*Hình 7.1.3: Hai thư viện được cài trong thư mục NuGet*

* Sau khi cài Các thư viện trên kiểm tra trong thư mục “NuGet” :

*Hình 7.1.4: Thư mục NuGet sau khi cài 2 thư viện trên*

1. Giới thiệu sản phẩm Demo

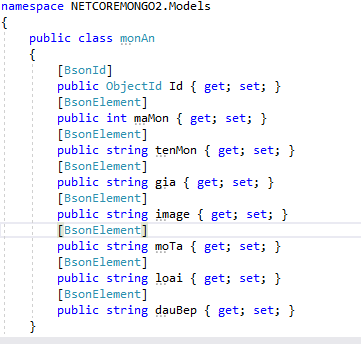
*Sau đây nhóm sẽ trình bày demo 1 chức năng “****quản lý món ăn****” của 1 nhà hàng tên là”OhanaFood”,web cho phép khách tự Order món tại bàn ăn thông qua màn hình tại mỗi bàn.*

* 1. Tạo giao diện chính cho trang *“quản lý món ăn”*

Đầu tiên: tạo Model tên là “monAn”:

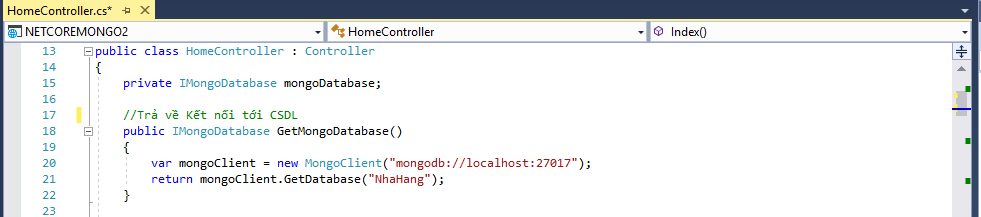
4

*Hình 7.2.1: Tạo Model tên là “monAn”*

Thêm các thuộc tính cần quản lý:

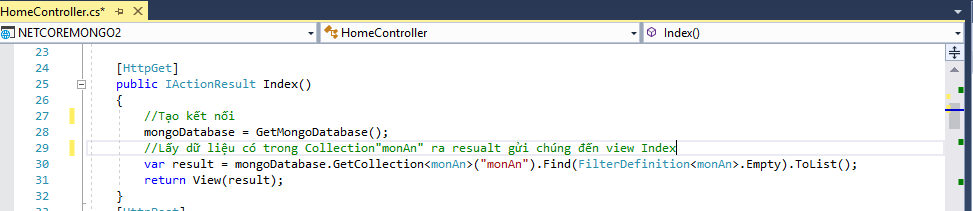
*Hình 7.2.2: Thêm các thuộc tính cần quản lý*

Tại HomeController tạo đối tượng “mongoDatabase” đại diện csdl Mongodb.

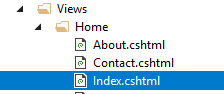
Tạo hàm GetMongoDatabase() trả về kết nối tới Client . CSDL trong bài tên là”NhaHang” nếu chưa có CSDL, sẽ tự động sinh ra.

*Hình 7.2.3: Tạo đối tượng “mongoDatabase” đại diện csdl Mongodb*

Tại Action “Index” tạo kết nối tới CSDL, lấy tất cả các Document trong Collection”monAn” ra biến result rồi gửi chúng đến view “Index.cshtml”

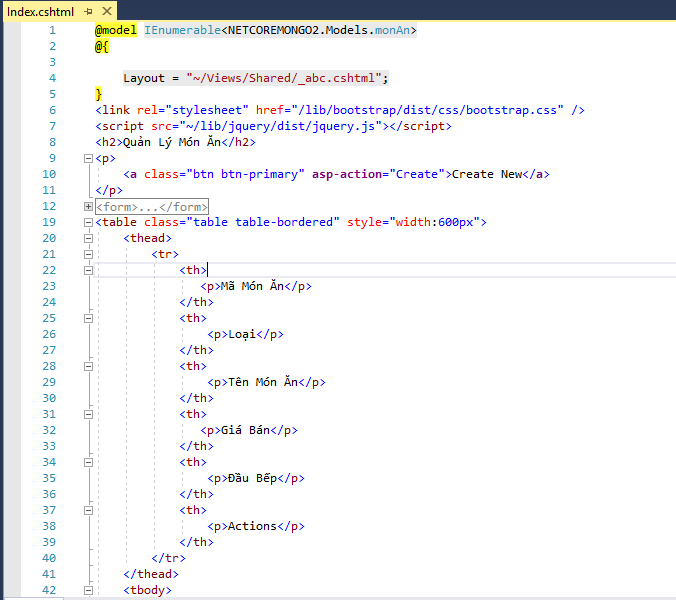
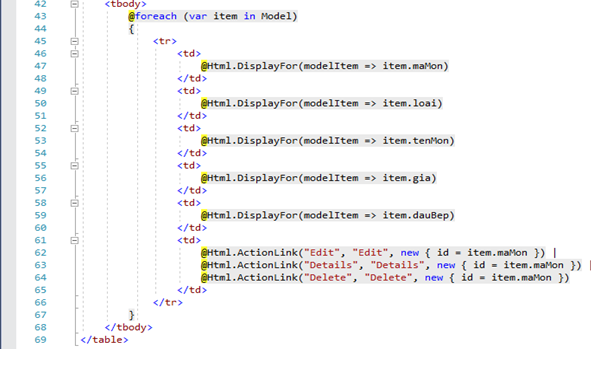


*Hình 7.2.4: Tại Action “Index” tạo kết nối tới CSDL*

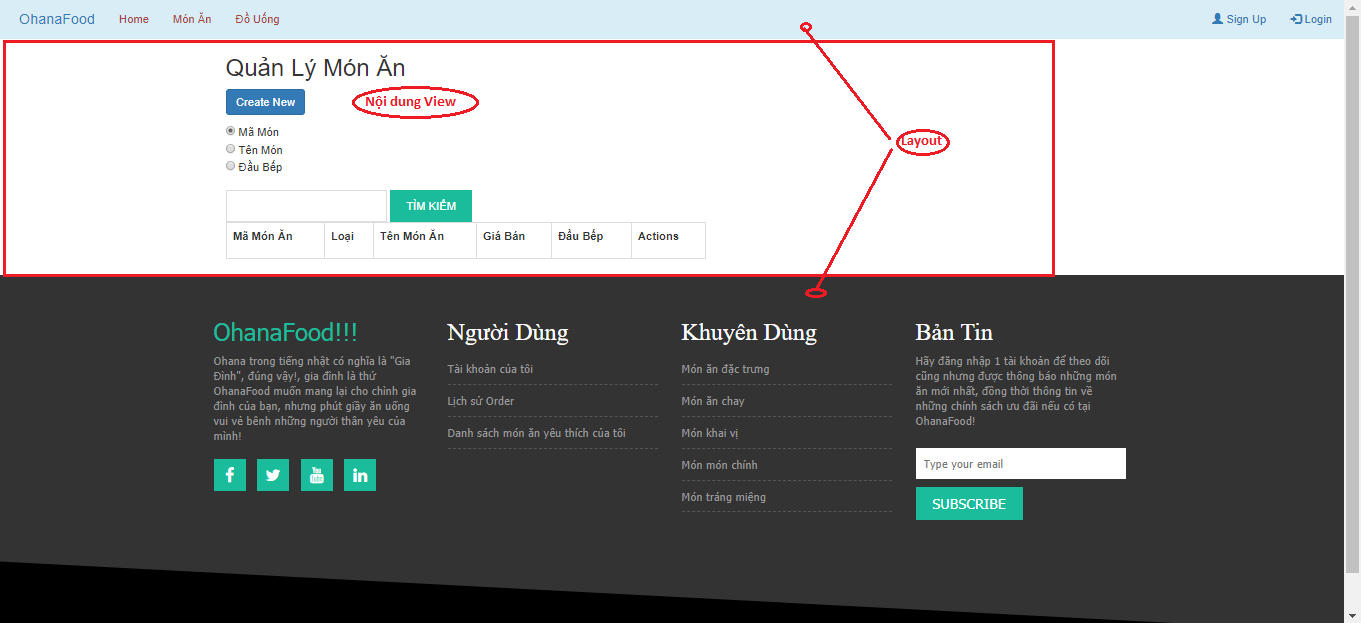
Tại thư mục **Views** tạo view tên là Index.cshtml

*Hình 7.2.5: Tạo view tên là Index.cshtml*

Trong view “Index.cshtml” xây dựng 1 bảng chứa dữ liệu từ Controller gửi qua , trong trường hợp này là “result” đại diện của nó trong view là “Model”.Sau đó đổ dữ liệu ra bảng thông qua vòng foreach như hình bên dưới.

*Hình 7.2.6: Nội dung file “Index.cshtml”*

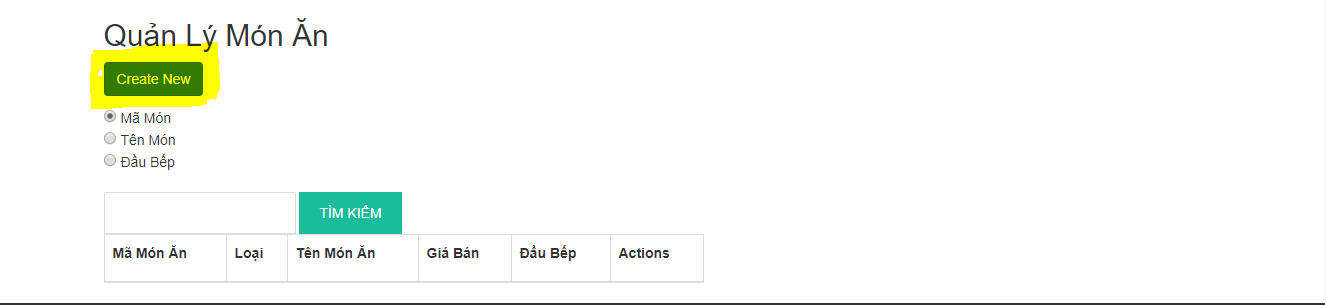
Đây là giao diện sau khi ta chạy bảng chưa có dữ liệu.



*Hình 7.2.7: Giao diện khi chạy chưa có dữ liệu*

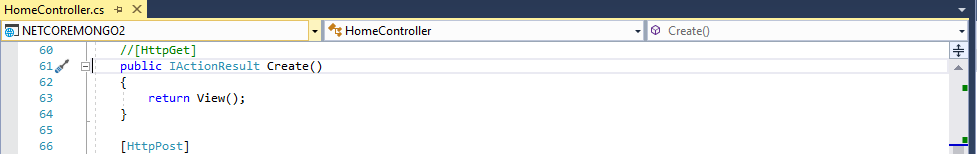
* 1. Tạo chức năng thêm sản phẩm

Bây giờ chúng ta sẽ tạo sự kiện cho nút “Create New” để tạo 1 Document mới trong CSDL.

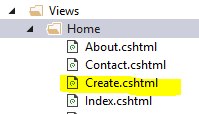


*Hình 7.2.2.1: Tạo sự kiện cho nút “Create New”*

Trở về HomeController tạo Action “Create” là sự kiện khi nhấn nút “Create New” trong “Index”



*Hình 7.2.2.2: Tạo Action “Create” trong HomeController*

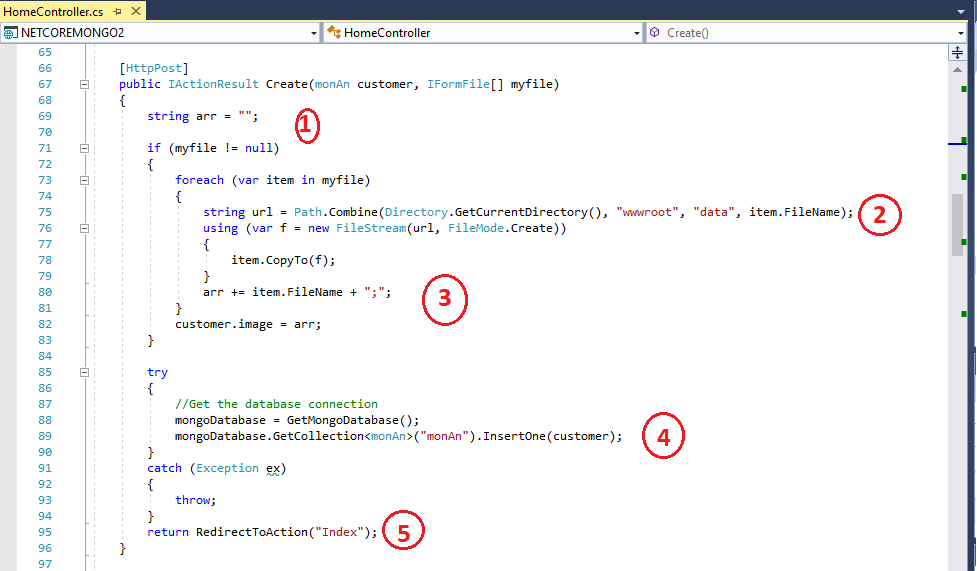
Tạo view “Create.cshtml” trong thư mục **Views** là view được Action “Create” goi ở Controller



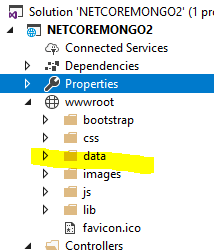
*Hình 7.2.2.3: Tạo “Create.cshtml” trong thư mục* ***Views***

Tại HomeController tạo Action “Create” nhận dữ liệu từ “Create.cshtml” thông qua[HttpPost]

Hàm sẽ nhận vào 1 đối tượng kiểu “monAn”, đầu tiên tạo chuỗi “arr” mục đích lưu lại tên của mỗi tấm hình, sau đó tạo vòng foreach lưu các hình ảnh đó vào thư mục data trong thư mục **wwwroot** ,trong mỗi vong for trước khi kết thúc ta lưu tên mỗi bức hình đó vào biến “arr”cách nhau bởi dấu “;” mục đích để phân biện tên mỗi bức hình để sau dễ bóc tách truy xuất, sau đó gắn thuộc tính “image” của đối tượng được gửi qua từ view, thuộc tính “image” chưa có dữ liệu cho biến “arr”,tiếp đến tạo kết nối tới CSDL và thêm vào Collection “monAn” thông qua hàm “InsertOne” cuối cùng ta gọi lại Action “Index” để cập nhận dữ liệu hiện thị chúng.

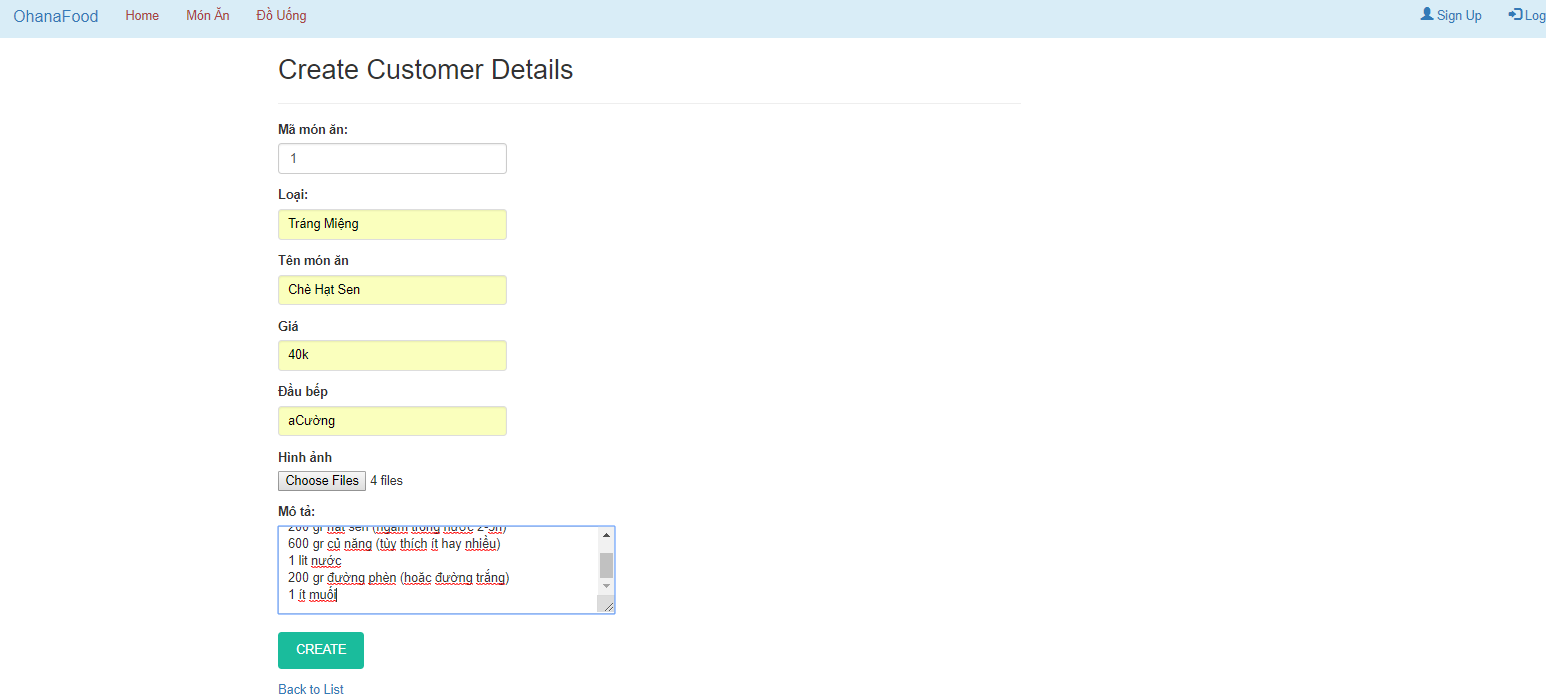


*Hình 7.2.2.4: Nhận dữ liệu từ “Create.cshtml” thông qua[HttpPost]*

Trong thư mục **wwwroot** tạo file”data” để chứa các hình ảnh gửi từ “Create.cshtml” thông qua biến “myfile”.

*Hình 7.2.2.5: Tạo file”data” trong thư mục wwwroot*

Đây là view “Create.cshtml”



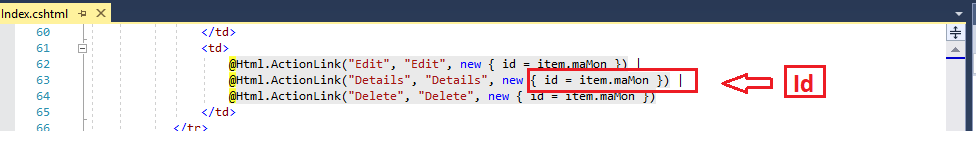
*Hình 7.2.2.6: file “Create.cshtml”*

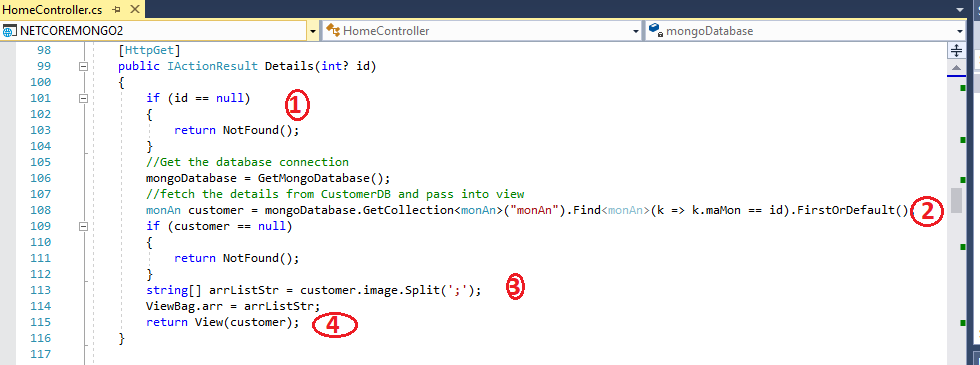
* 1. Tạo chức năng xem chi tiết sản phẩm

Bây giờ chung ta sẽ tạo sự liện cho “Details” hiển thị thông tin chi tiết của đối tượng

*Hình 7.2.3.1: Tạo sự kiện Details*

Để phân biệt các đối tượng với nhau ta sẽ lưu giữ Id của đối tượng đó mỗi khi đổ dữ liệu ra table trong view “Index.cshtml”



Trở lại HomeController tạo Action “Details” nhận vào 1 Id được gửi qua từ view” Index.cshtml”

*Hình 7.2.3.2: Tạo Action “Details” trong HomeController*

*Trong đó:*

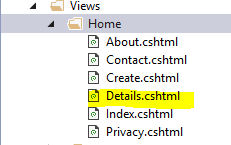
*1.Kiểm tra có nhận được Id hay không*

*2.Tạo kết nối tới CSDL tìm đối tượng thông qua Id bằng hàm “Find” truyền vào nó điều kiện lọc “maMon=Id”*

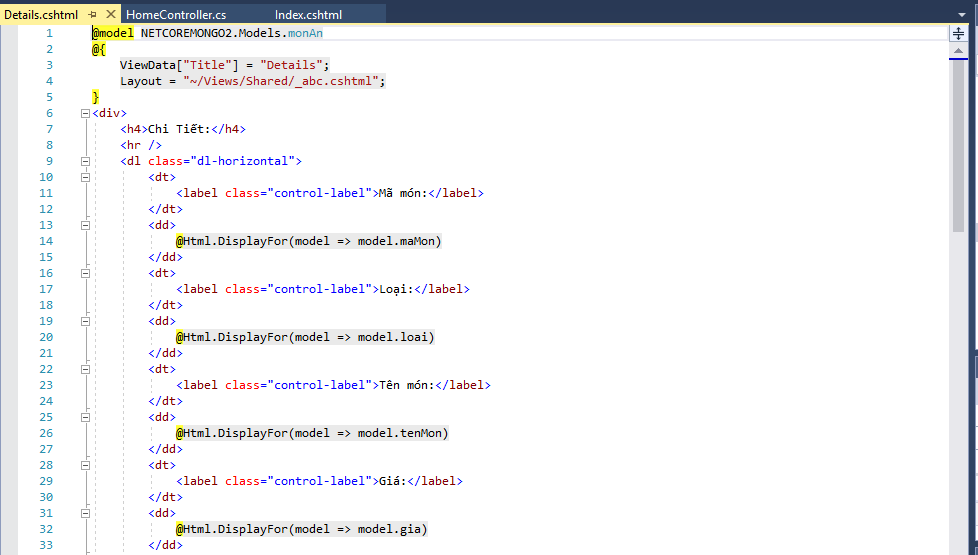
*3.Tạo 1 mảng “arrListStr” chứa tên mỗi bức hình bằng cách bóc tách Thuộc tính “image” của đối tương tìm thấy bằng hàm Split() lọc bằng dấu “;” đã được chèn giữa mỗi tên hình ở Action “Create” và gắn vào ViewBag.arr để gửi qua View “Details.cshtml” nhằm mục đích hiển thị hình ảnh*

*4.Gửi đối tượng và ViewBag.arr qua view “Details.cshtml”*

Tạo view “Details.cshtml”

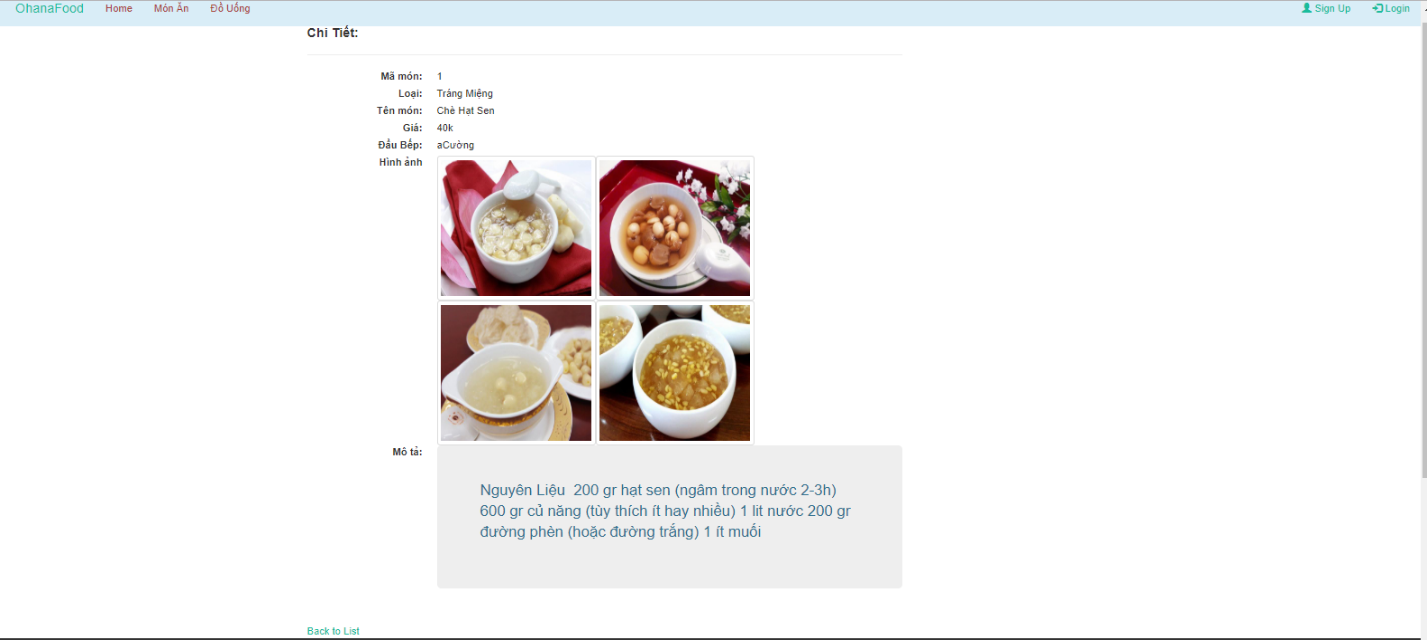


*Hình 7.2.3.3: file “Details.cshtml” trong thư mục view*

*Hình 7.2.3.4: Nội dung* *file “Details.cshtml”*

Sau khi nhấn “Details” trong view”Index.cshtml”

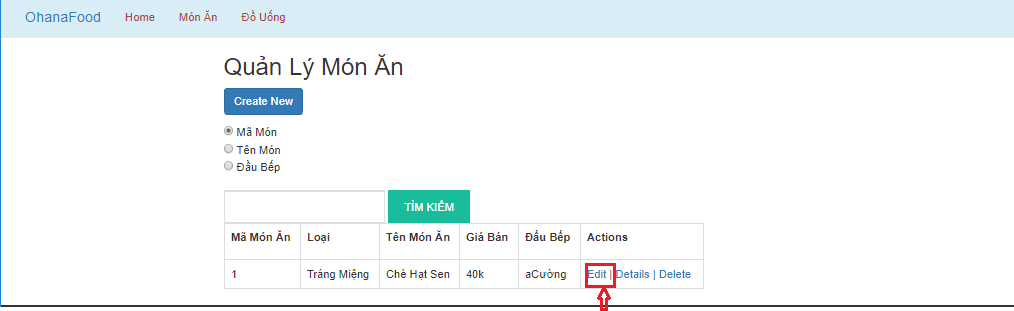




*Hình 7.2.3.5: Chi tiết sản phẩm sau khi nhấn vào Details*

* 1. Tạo chức năng sửa sản phẩm

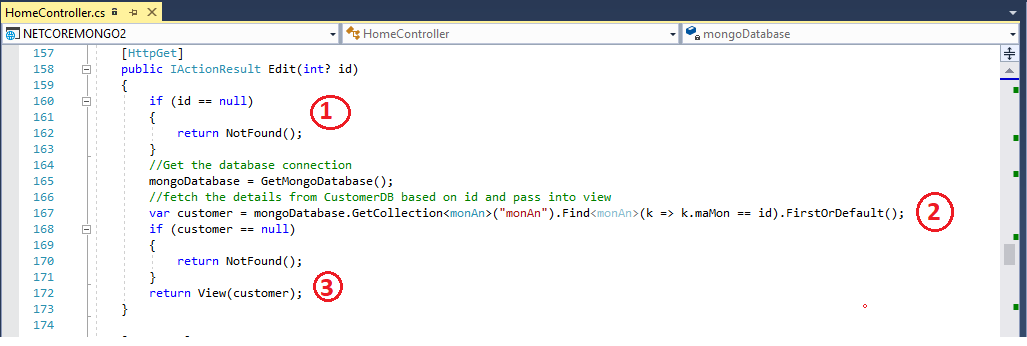
Bây giờ chúng ta tiếp tục tạo sự kiện cho “Edit”



*Hình 7.2.4.1: Tạo sự kiện Edit*

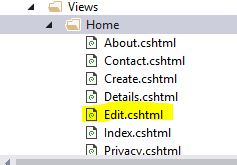
Tạo Action “Edit” trong HomeController nó cũng nhận vào “Id” giống Action “Details”

Sau khi lấy được đối tượng trong CSDL nó sẽ gửi qua view “Edit.cshtml” để chỉnh sửa



*Hình 7.2.4.2: Tạo Action “Edit” trong HomeController*

Tạo View “Edit.cshtml”



*Hình 7.2.4.3: file “Edit.cshtml” trong thư mục view*

*Hình 7.2.4.4: Nội dung file “Edit.cshtml*

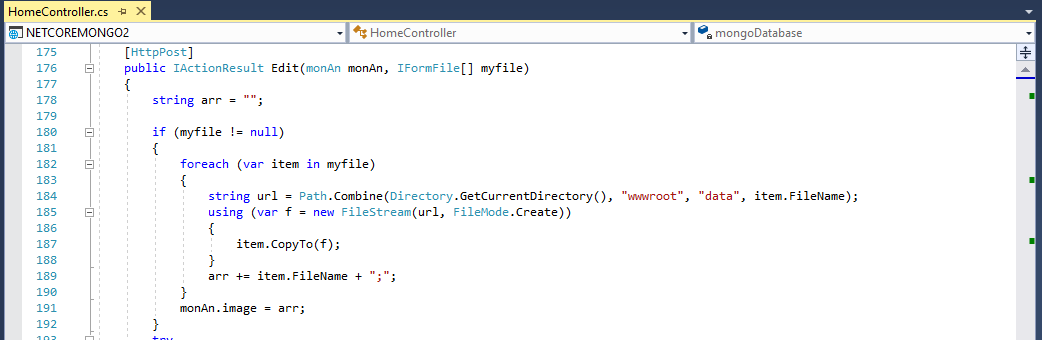
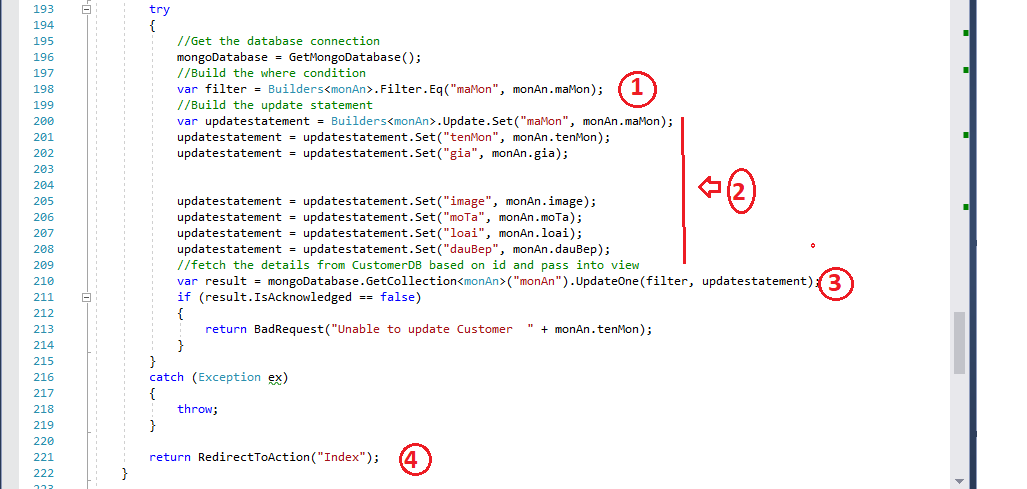
Trong HomeController tạo Action “Edit” nhận dữ liệu từ view “Edit.cshtml” gửi về, nó nhận từ được 1 đồi tượng monAn và 1 mãng chứa các hình ảnh mới sau khi chỉnh sửa của đối tượng

*1. tạo filter lọc các Document trong CSDL lấy ra đối tượng cần chỉnh sửa*

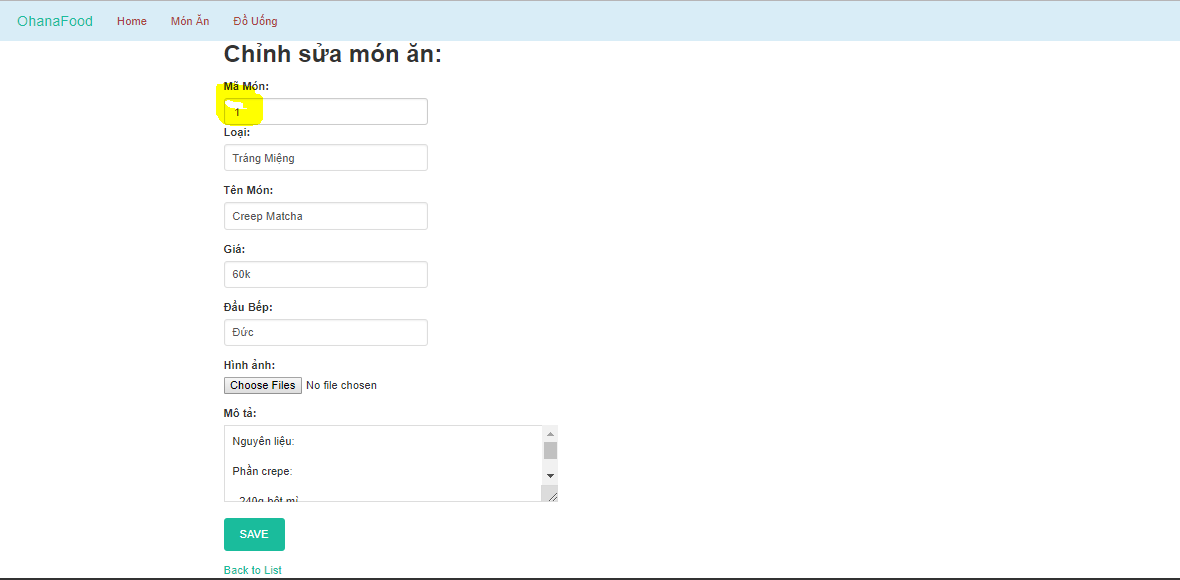
*2. tạo biến “updatestatement” gắn các thuộc tính của đối tượng đã được chỉnh sửa “monAn”*

*3. update đối tượng thông qua hàm UpdateOne() chuyền vào nó filter để chọn đối tượng update, và đối tượng updatestement.*

*4. gọi lại Action”Index” để chập nhận dữ liệu*

*Hình 7.2.4.5: Tạo Action “Edit” trong HomeController*

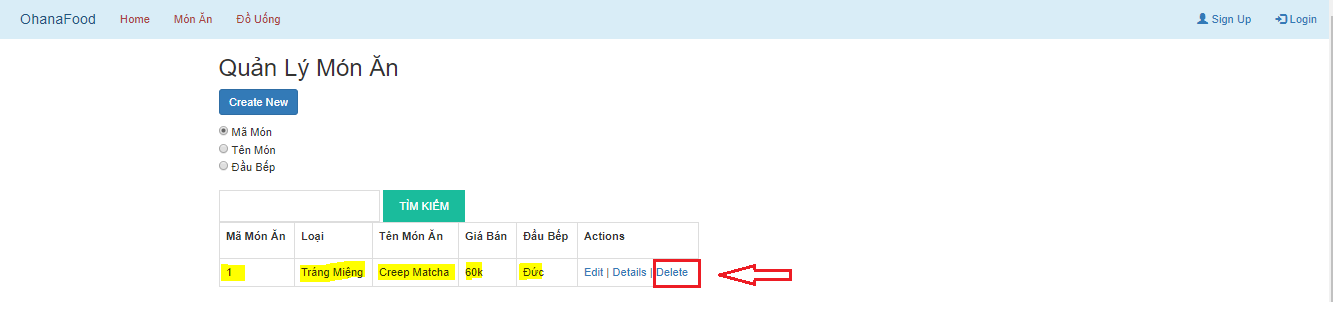
Đây là view “Edit.cshtml” sau khi chạy chương trình



*Hình 7.2.4.6: Màn hình sửa sản phẩm*

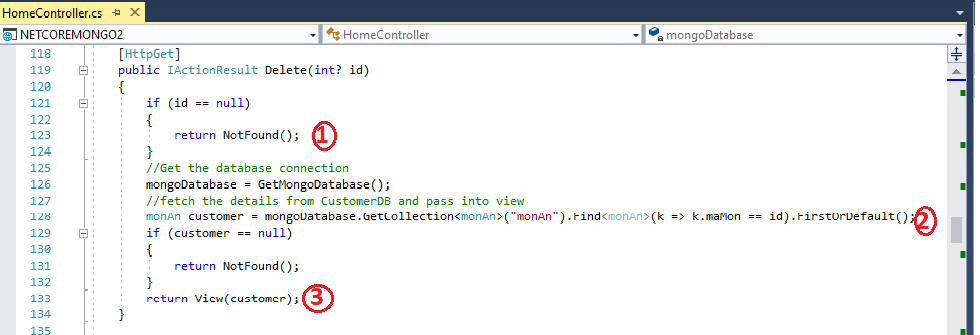
* 1. Tạo chức năng xóa sản phẩm

Tiếp theo chúng ta tiếp tục tạo sự kiện cho “Delete”



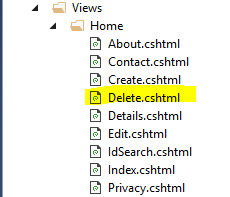
*Hình 7.2.5.1: Tạo sự kiện Delete*

Trở về HomeControllerr tạo Action “Delete” tương tự nó nhận vào 1 “Id” để tìm đối tượng trong CSDL và gửi nó qua view “Delete.cshtml”

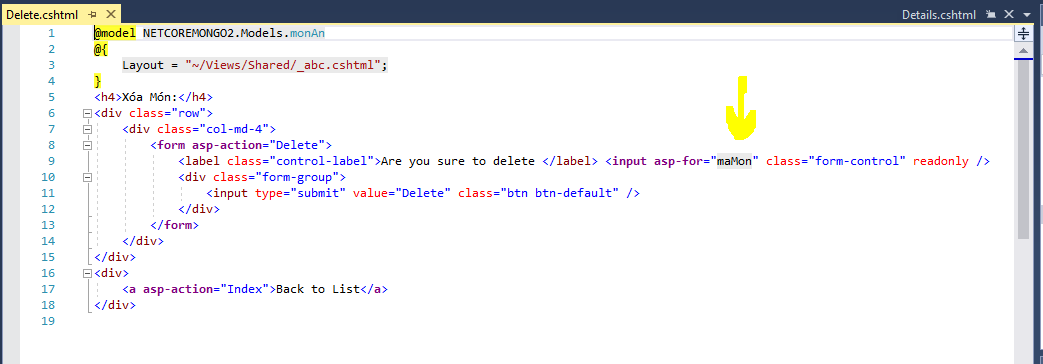


*Hình 7.2.5.2: Tạo Action “Delete” trong HomeController*

Tạo view”Delete.cshtml”

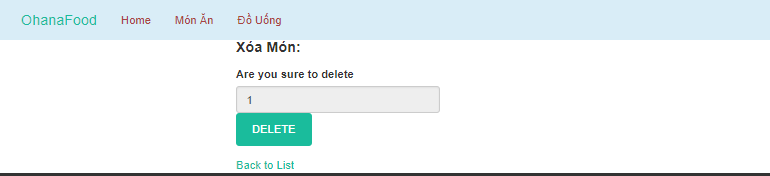


*Hình 7.2.5.3: file “Delete.cshtml” trong thư mục view*



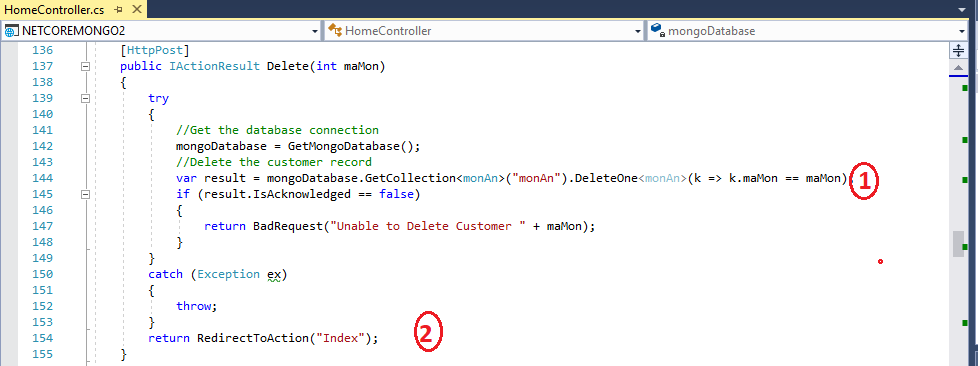
*Hình 7.2.5.4: Nội dung file “Delete.cshtml*

Đầy là nội dung view “Delete.cshtml” hỏi người dung thật sự muốn xóa món ăn có mã món =mammon của đối tượng được gửi từ Action “Delete” hay không.



*Hình 7.2.5.5: Màn hình xóa sản phẩm*

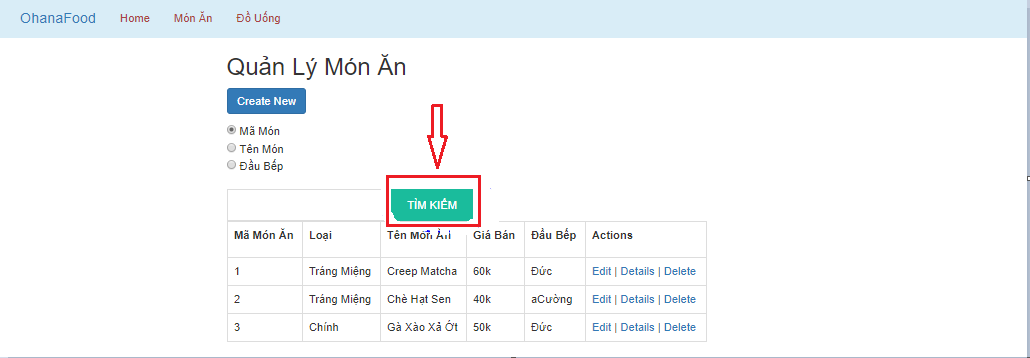
Trở về HomeController tạo Action”Delete” để nhận “maMon” thông qua [HttpPost] được gửi từ view “ Delete.cshtml” và xóa đối tượng đó trong CSDL thông qua hàm DeleteOne() truyền vào mammon để lọc lấy đối tượng cần xóa



*Hình 7.2.5.6*

* 1. Tạo chức năng tìm kiếm sản phẩm

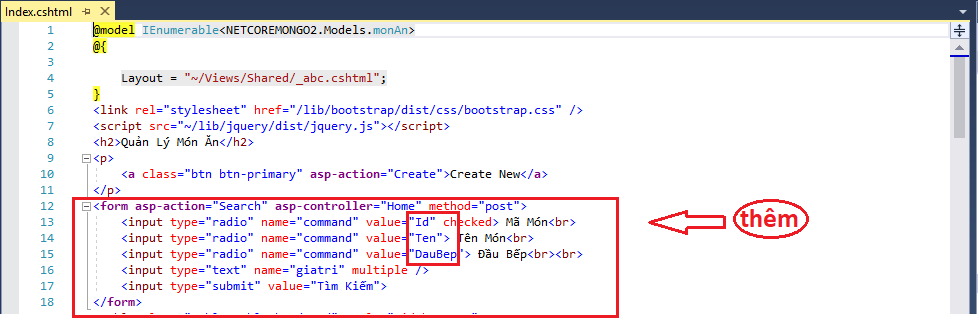
Cuối cùng là bày chức năng “Search” ý tưởng : có thể tìm kiếm theo các thuộc tính của đối tượng, ở đây chúng ta demo 3 thuộc tính “maMon, TenMon, DauBep”



*Hình 7.2.6.1: Tạo sự kiện nút “Tìm Kiếm”*

Trong view”Index.cshtml” thêm form gọi tới Action”Search” bên phía Controller để sử lý,

Nó gửi dữ liệu thông qua biến “command” , và “giatri”



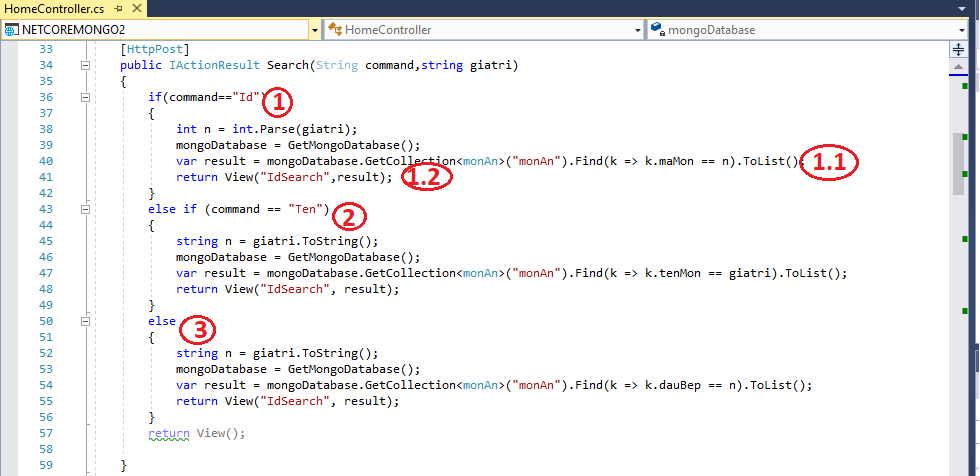
*Hình 7.2.6.2*

Trở về HomeController tạo Action “Search” nhận dữ liệu từ view “Index.cshtml” gửi về

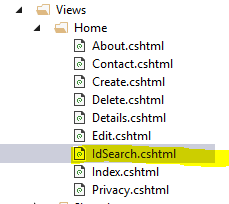
Command lưu giá trị(value) là “Id” , “Ten” hay “DauBep”, biến giatri chứa th6on tin người dung nhập

Tạo truy vấn if else, để kiểm tra người dung muốn tìm kiếm theo thuộc tính nào thông qua biến command ta có thể xác định được, sau đó ta tìm đối tượng đó trong CSDL lọc bằng biền “giatri”

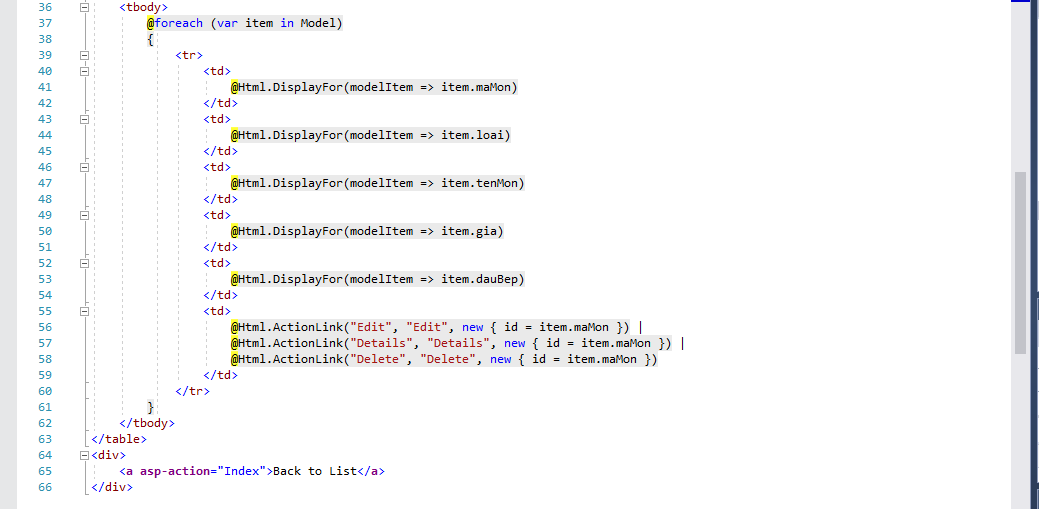
Cuối cùng gửi những đối tượng tím thấy được qua view “IdSearch.cshtml”



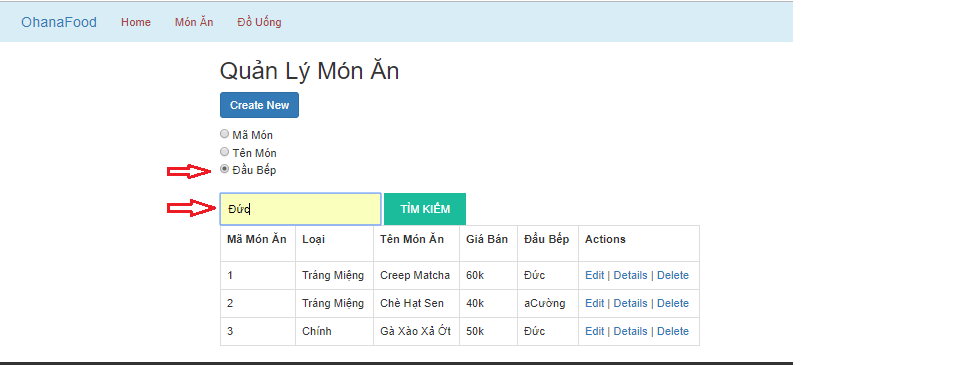
*Hình 7.2.6.3*

Tạo view “IdSearch.cshtml”

*Hình 7.2.6.4: file “IdSearch.cshtml” trong thư mục view*

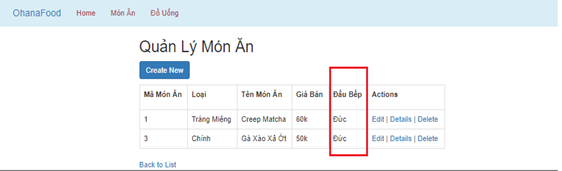


*Hình 7.2.6.5: Nội dung file “IdSearch.cshtml”*

Chúng ta hay thử tìm theo tên đầu bếp là “Đức”

*Hình 7.2.6.6: Màn hình tìm kiếm sản phẩm*

Kết quả có 2 đầu bếp là “Đức” trong CSDL



*Hình 7.2.6.7: Màn hình hiển thị danh sách sau khi tìm kiếm theo tên đầu bếp*

VII. Ưu điểm và nhược điểm:

1. So sánh MySQL với MongoDB
2. Về ngôn ngữ truy vấn

|  |  |
| --- | --- |
| \*My SQL:  **INSERT INTO** users (user\_id, age, status)  **VALUES** ('bcd001', 45, 'A') SELECT \* FROM users  **UPDATE** users **SET** status = 'C'  **WHERE** age > 25  db.start\_transaction()  cursor.execute(orderInsert, orderData)  cursor.execute(stockUpdate, stockData)  db.commit() | \*MongoDB:  db.users.insert({  user\_id:'bcd001',  age: 45,  status: 'A'  }) db.users.find() db.users.update(  { age: { $gt: 25 } },  { $set: { status: 'C' } },  { multi: true }  ) s.start\_transaction()  orders.insert\_one(order, session=s)  stock.update\_one(item, stockUpdate, session=s)  s.commit\_transaction() |

1. Về tính năng

|  |  |  |
| --- | --- | --- |
|  | **MySQL** | **MongoDB** |
| Open source | Yes | Yes |
| ACID Transactions (Atomicity-nguyên tử, Consistency-nhất quán, Isolation-độc lập và Durability-bền vững) | Yes | Yes |
| Flexible, rich data model | No | Yes |
| Schema governance | Yes | Yes |
| Expressive joins, faceted search, graphs queries, powerful aggregations | Yes | Yes |
| Idiomatic, native language drivers | No | Yes |
| Horizontal scale-out with data locality controls | No | Yes |
| Analytics and Business Intelligence ready | Yes | Yes |
| Enterprise grade security and mature management tools | Yes | Yes |
| Database as a service on all major clouds | Yes | Yes |

2. Ưu điểm MongoDB

* **Dữ liệu linh hoạt - MongoDB là document database, dữ liệu lưu dưới dạng JSON, không bị bó buộc về số lượng field, kiểu dữ liệu…**
* **Ngôn ngữ truy vấn phong phú -** MongoDB có sẵn các method để thực hiện create, read, update, delete dữ liệu(CRUD). Không có các fuction joins phức tạp như MySQL.
* Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau như trong Relational Database Management System (RDBMS), khi insert, xóa hay update nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các bảng liên quan .
* Dữ liệu trong MongoDB được đánh chỉ mục(index) nên truy vấn nhanh gọn.
* Dễ dàng mở rộng hệ thống do có cluster là cụm các node chứa dữ liệu giao tiếp với nhau.
* Hỗ trợ replica set nhằm đảm bảo việc sao lưu và khôi phục dữ liệu.
* Không cần thiết conversion/mapping (chuyển đổi / ánh xạ) các đối tượng ứng dụng đến các đối tượng cơ sở dữ liệu.
* Sử dụng bộ nhớ trong để lưu trữ, cho phép truy cập dữ liệu nhanh hơn.

3. Nhược điểm MongoDB

* MongoDB không có các tính chất ràng buộc như trong RDBMS nên dễ bị làm sai dữ liệu
* Không hỗ trợ join giống như RDBMS nên khi viết function join trong code phải tự làm khiến cho tốc độ truy vấn bị giảm.
* Sử dụng nhiều bộ nhớ: do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên sẽ bị dư thừa dữ liệu.
* Bị giới hạn kích thước bản ghi: mỗi document không được có kích thước lớn hơn 16Mb.

4. Khi nào nên sử dụng MongoDB

* Danh mục sản phẩm thương mại điện tử
* Blog và quản lý các nội dung.
* Hệ thống realtime (thời gian thực) yêu cầu phản hồi nhanh.
* Lưu trữ dữ liệu không gian địa lý.
* Các trang web mạng di động và mạng xã hội.
* Các hệ thống bigdata với yêu cầu truy vấn nhanh.
* Làm search engine

VIII. Tổng kết và đánh giá:

1. Tổng kết

MongoDB không sử dụng mô hình quan hệ ràng buộc(non-relational)

Mô hình Lưu trữ dạng {“key : value”}

MongoDB là 1 CSDL phân tán lưu chữ dữ liệu trên nhiều server khác nhau, đa dạng hóa dữ liệu, là dạng dữ liệu lớn( BigData).

Độ chịu tải cao ,chịu lỗi cao

Có khả năng nâng cấp theo chiều ngang, dễ dàng bảo trì, thay đổi,sửa chữa , phát triển

1. Đánh giá

Chính vì những đặc điểm đó ta nên dùng nó cho các hệ thống, ứng dụng đòi hỏi lượng người dùng và dữ liệu lưu trữ đa dạng, lớn đến cực lớn.Ví dụ như mạng xã hội(facebook, Instagram, twitter…), các trang lưu trữ dữ liệu lớn như google, github, …..

TÀI LIỆU KHAM KHẢO

[1]. **Ebook NoSQL** - Nhữ Đình Thuận

[2]. **Kristina Chodorow - Micheal Dirolf, MongoDB**: The Definitive Guide, O'reilly, 2010

[3]. Eeclo Plugge, Peter Membrey and Tim Hawkins, **The Definitive Guide to MongoDB the NoSQL Database for Cloud and Desktop Computing.**

[4]. **NoSQL resources** - *http://nosql-database.org*

[5]. **MongoDB wikipedia** *- https://en.wikipedia.org/wiki/MongoDB*

[6]. *https://www.mongodb.com*

[7]. *https://bigsonata.wordpress.com/2014/06/05/mongodb/*

[8]. *http://kienthucweb.net/gioi-thieu-ve-co-so-du-lieu-mongodb.html*

[9]. *https://vietjack.com/mongodb/*

[10]. *https://freetuts.net/tong-quan-ve-mongodb-203.html*