**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

---------o0o---------

****

**ĐỒ ÁN TỐT NGHIỆP**

**XÂY DỰNG WEBSITE HỌC TỪ VỰNG TRỰC TUYẾN**

**Giáo viên hướng dẫn :**

*Ts.Trần Văn Dũng*

Bộ môn: Khoa học máy tính

**Sinh viên thực hiện**:

Trịnh Đức Thịnh (MSV: 1308975)

Lớp*:* Công nghệ thông tin 3 - K54

Hà Nội, ngày 18 tháng 05 năm 2017

**Nhận xét của giáo viên hướng dẫn**

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Giảng viên hướng dẫn

MỤC LỤC

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI VÀ CÔNG NGHỆ PHÁT TRIỂN 5](#_Toc482626348)

[**1.1. Bài toán** 5](#_Toc482626349)

[**1.2. Khảo sát** 5](#_Toc482626350)

[**1.3. Phân tích** 5](#_Toc482626351)

[**1.4. Công nghệ phát triển** 5](#_Toc482626352)

[1.4.1 Entity Framework 6](#_Toc482626353)

[1.4.2. AngularJS 12](#_Toc482626354)

[1.4.3. Tổng quan về Web Api 14](#_Toc482626355)

[1.4.3. Asp.Net MVC 15](#_Toc482626356)

[CHƯƠNG 2. THIẾT KẾ HỆ THỐNG 19](#_Toc482626357)

[**2.1. Xác định bài toán và phạm vi sơ bộ** 19](#_Toc482626358)

[2.1.1 Khảo sát hiện trạng 19](#_Toc482626359)

[2.2. Phân tích yêu cầu bài toán 21](#_Toc482626360)

[2.2.1 *Yêu cầu chức năng* 21](#_Toc482626361)

[**2.2 Đặc tả chức năng** 23](#_Toc482626362)

[2.2.1 Chức năng quản lý người dùng 23](#_Toc482626363)

[2.2.2 Chức năng quản lý từ vựng 24](#_Toc482626364)

[2.2.3 Chức năng thống kê 25](#_Toc482626365)

[2.2.4 Chức năng học và kiểm tra. 26](#_Toc482626366)

[2.2.5 Chức năng thông báo Bao gồm những chức năng sau: 27](#_Toc482626367)

[**2.3 Thiết kế cơ sở dữ liệu** 28](#_Toc482626368)

[2.3.1 Thông tin cần lưu trữ 28](#_Toc482626369)

[2.3.1 Mô tả cấu trúc cơ sở dữ liệu 28](#_Toc482626370)

[CHƯƠNG 3. CÀI ĐẶT VÀ THỬ NGHIỆM WEBSITE 32](#_Toc482626371)

[3.1. Cấu trúc của project 32](#_Toc482626372)

[3.2. Cài đặt các module 35](#_Toc482626373)

[3.2.1 Module đăng ký 35](#_Toc482626374)

[3.3.1 Xây dựng module đăng nhập 36](#_Toc482626375)

[3.4. Chức năng học từ vựng 38](#_Toc482626376)

**Lời mở đầu**

Bản báo cáo gồm 3 phần chính:

Phần 1: Giới thiệu công nghệ sử dụng

Phần 2: Khảo sát yêu cầu Web học từ vựng trực tuyến

Phần 3: Phân tích và thiết kế cơ sở dữ liệu

Tuy em đã rất cố gắng nhưng chắc chắn cũng không thể tránh khỏi những sơ suất sẽ mắc phải. Mong thầy góp ý thêm để báo cáo của em hoàn thiện.

Xin trân trọng cảm ơn!

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI VÀ CÔNG NGHỆ PHÁT TRIỂN

**1.1. Bài toán**

**1.2. Khảo sát**

**1.3. Phân tích**

**1.4. Công nghệ phát triển**

### 1.4.1 Entity Framework

* **Entity Framework (EF)** là một framework ánh xạ quan hệ đối tượng (ORM) dành cho ADO.NET, là 1 phần của .NET Framework. EF cho phép các nhà phát triển Web tương tác với dữ liệu quan hệ theo phương pháp hướng đối tượng đặc trưng. Lợi ích lớn nhất của EF là giúp lập trình viên giảm thiểu việc lập trình mã nguồn cần thiết để truy cập và tương tác với cơ sở dữ liệu. EF được Microsoft hỗ trợ phát triển lâu dài và bền vững, vì vậy EF là 1 framework mạnh nhất hiện nay để phát triển ứng dụng Web với sự hỗ trợ đông đảo của các nhà phát triển Web.
* **Vị trí của Entity Framework trong mô hình lập trình và phát triển Web**  
  Chắc hẳn, chúng ta đã biết về mô hình Web 3 tầng (n tầng) khi đọc sách, giáo trình, bài giảng về nội dung lập trình Web. Tuy nhiên, vấn đề chúng ta cần phải hiểu rõ là: việc định rõ số lượng các tầng và các mối liên hệ giữa các tầng trong mô hình phát triển Web cũng rất đa dạng, tùy theo cách hiểu các lập trình viên và dự án Web. Do đó, việc hiểu Entity Framework nằm ở đâu trong mô hình Web 3 tầng cũng không hoàn toàn dễ dàng. Trong phần này, tôi có tham khảo từ Stephen M. Redd nhằm giúp hiểu rõ bản chất vấn đề hơn.

Đầu tiên, chúng ta có vị trí của EF trong mô hình dự án Web ASP.NET MVC.Thật sự, khi bắt tay xây dựng nhiều dự án Web, vị trí thường thấy EF ở các dự án website trên thực tế như mô hình sau.

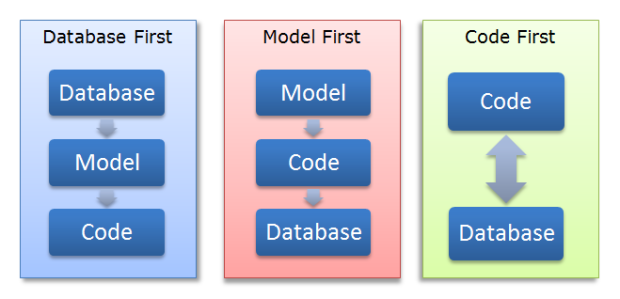


Hình 2. Mô hình dự án Web ASP.NET MVC trên thực tế.

Ta có thể hiểu 1 cách chung nhất EF có vị trí trung gian, đóng vai trò kết nối giữa cơ sở dữ liệu và các thành phần khác của 1 dự án Web khi cần đến.

* **Các thành phần trong Entity Framework**:
* Code là mã lệnh tạo thành các lớp đối tượng dữ liệu cho phép thao tác với dữ liệu.
* Model là sơ đồ gồm các hộp mô tả các thực thể và các đường nối kết mô tả các quan hệ.
* Database là cơ sở dữ liệu (có thể là SQL Server, Compact SQL Server, Local database, MySQL, Oracle,…

Có 3 cách sử dụng Entity Framework: [Code First](http://yinyangit.wordpress.com/2011/11/30/thiet-ke-entity-data-model-part-3-code-first/), [Models First](http://yinyangit.wordpress.com/2011/11/28/cac-huong-thiet-ke-entity-data-model-part-2-model-first/), [Database First](http://yinyangit.wordpress.com/2011/11/26/cac-huong-thiet-ke-entity-data-model-part-1-database-first/).

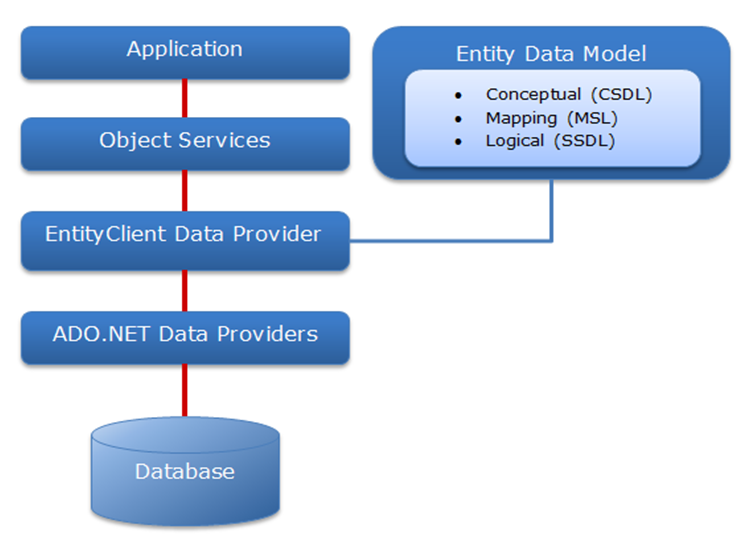
[](https://yinyangit.files.wordpress.com/2011/11/entity-framework-design-approaches.png?w=620&h=299)

* Database first: là phương pháp chỉ nên dùng khi ta đã có sẵn CSDL (không phải tạo), EF Wizard sẽ tạo Model và Code cho ta.
* Models first: nên dùng khi ta bắt đầu thiết kế CSDL từ đầu (từ chưa có gì). Ta sẽ thiết kế mô hình CSDL (Model) EF sẽ tự tạo code cho ta, sau đó nhờ EF Wizard tạo CSDL.
* Code first: nên dùng khi đã có mô hình CSDL, ta sẽ chỉ viết code từ đó tạo Database.

Dù cách nào thì cuối cùng cũng phải có Code để thao tác trong mã lệnh và Database để lưu trữ dữ liệu. Model chỉ là một thành phần trung gian.

**- Kiến trúc Entity Framework**

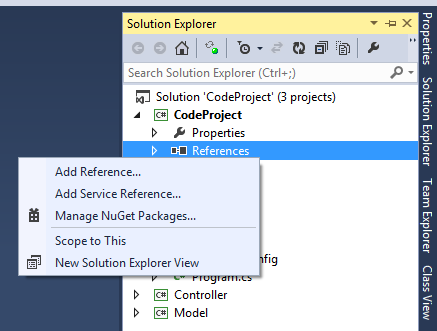
Mô hình kiến trúc của EF được mô tả đơn giản qua hình sau.

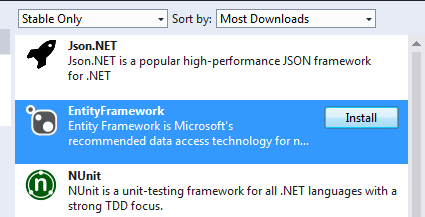


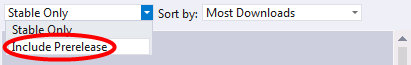
Hình 3. Mô hình kiến trúc của Entity Framework.

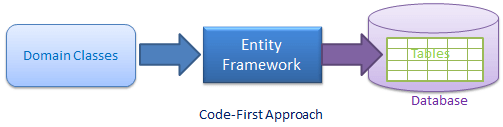
* **Application** (ứng dụng): là tầng chứa giao diện trang Web (HTML, CSS, Javascript, hình ảnh, …) và các đoạn mã nguồn (C#, VB) để tương tác dữ liệu với các tầng khác trong mô hình thông qua Object Services.
* **Object Services** (tạm dịch là **các dịch vụ đối tượng**): là tầng chứa quá trình tương tác giữa ứng dụng và database, hay nói cách khác nó là nơi chủ yếu để truy cập dữ liệu từ database và trả ngược kết quả về giao diện. Object Services cung cấp các tiện ích để truy vết các thay đổi và quản lý nhận dạng, đồng thời là các quan hệ và thay đổi ở database.
* **EntityClient Data Provider:** Đây là tầng cung cấp các kết nối, diễn dịch các truy vấn thực thể thành truy vấn nguồn dữ liệu (chuyển L2E – LINQ to Entity hay các truy vấn thực thể SQL thành truy vấn SQL), trả về data reader để EF dùng chuyển dữ liệu thực thể thành các đối tượng. Phần này kết nối ADO.NET Data Providers để gửi hoặc lấy dữ liệu từ database. Tầng này hoàn toàn khác với EDM (Entity Data Model) khi thực thi các truy vấn tương tự như cách thực hiện ở ADO.NET Provider. EntityClient Data Provider sử dụng ESQL (Entity SQL), một ngôn ngữ truy vấn độc lập dạng văn bản, tương tự như SQL.
* **ADO.NET Data Providers:** Đây là tầng thấp nhấp để dịch các truy vấn L2E (LINQ to Entity) thông qua cây lệnh thành các câu lệnh SQL và thực thi các câu lệnh trong hệ thống DBMS (database management system – hệ quản lý dữ liệu) nào đó. Tầng này kết với database sử dụng ADO.NET.
* **EDM (**tạm dịch là mô hình dữ liệu thực thể**)** chứa 3 phần chính: mô hình khái niệm (CSDL – Conceptual schema definition language), mô hình ánh xạ (MSL – mapping specification language) và mô hình lưu trữ (SSDL – store schema definition language). EDM khác với EntityClient Data Provider ở chỗ EDM sử dụng LINQ là ngôn ngữ truy vấn tương tác với database.
* **Mô hình khái niệm (**CSDL – Conceptual schema definition language**)**  
  Mô hình khái niệm chứa các lớp mô hình và mối quan hệ giữa các lớp này. Điều này để độc lập với mô hình quan hệ các bảng trong database.
* **Mô hình lưu trữ (**SSDL – store schema definition language**)** Mô hình lưu trữ là 1 mô hình thiết kế database bao gồm các bảng, view, stored procedure (thủ tục), và mối quan hệ giữa chúng và các khóa. Mô hình này thể hiện gần giống mô hình quan hệ các bảng trong database.
* **Mô hình ánh xạ (**MSL – mapping specification language**)** Mô hình ánh xạ gồm thông tin về cách mô hình khái niệm được ánh xạ đến mô hình lưu trữ.
* **L2E (LINQ to Entities)**: là 1 ngôn ngữ truy vấn được dùng để viết các truy vấn trái với mô hình đối tượng. L2E trả về các thực thể, được định nghĩa bởi mô hình khái niệm. Chúng ta có thể dùng LINQ trong ngôn ngữ này.
* **Cài đặt Entity Framework**

Phiên bản mới nhất của EF có sẵn tại gói EntityFramework tại Nuget (<http://nuget.org/packages/EntityFramework/>). Để cài Entity Framework, bạn nhấn chuột phải vào thư mục References (hoặc bin) trong dự án Web và chọn Manage NuGet Packages.

[](https://dammio.files.wordpress.com/2016/11/manage_nuget_package.png)

Sau đó, bạn tìm và cài đặt gói Entity Framework vào dự án Web.  
**[](https://dammio.files.wordpress.com/2016/11/ef_install.png)**

* Cài đặt bản EF mới nhất  
  Để cài đặt phiên bản EF mới nhất, bạn có thể chọn Include Prerelease trong cửa **sổ** Manage NuGet Package để hoàn tất cài đặt.  
  **[](https://dammio.files.wordpress.com/2016/11/ef_includeprerelease.png)**
* **Entity framework code first**
* Entity Framework giới thiệu cách tiếp cận Code-First từ Entity Framework 4.1. Code-First là cách tiếp cận thiết kế mô hình thao tác cơ sở dữ liệu.
* Với cách tiếp cận Code-First, chúng ta có thể tập trung vào việc thiết kế domain và bắt đầu tạo ra các lớp theo yêu cầu thiết kế domain mà không tập trung thiết kế cơ sở dữ liệu đầu tiên. Sau đó tạo ra các lớp ánh xạ đến thiết kế cơ sở dữ liệu. Code-First API sẽ tạo cơ sở dữ liệu dựa trên các lớp thực thể và lớp cấu hình.



* Là một nhà phát triển, đầu tiên chúng ta bắt đầu bằng cách viết các lớp và lớp context sử dụng C # hay VB.net. Khi chạy ứng dụng, Code-First API sẽ tạo cơ sở dữ liệu mới (nếu nó chưa tồn tại) và ánh xạ các lớp với cơ sở dữ liệu bằng cách sử dụng các quy ước code-first mặc định.

### 1.4.2. AngularJS

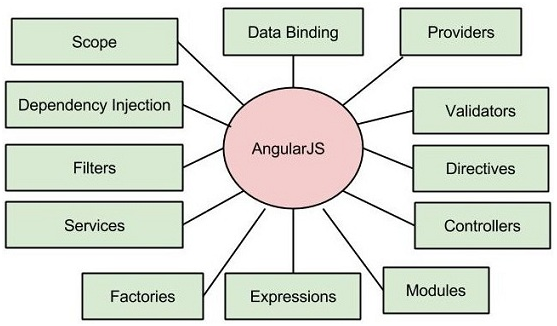
**AngularJS** là một bộ Javascript Framework rất mạnh và thường được sử dụng để xây dựng project **Single Page Application** (SPA). Nó hoạt động dựa trên các thuộc tính mở rộng HTML (các atributes theo quy tắc của Angular). Đây là một Framework mã nguồn mở hoàn toàn miễn phí và được hàng ngàn các lập trình viên trên thế giới ưa chuộng và sử dụng. Framework này được thế hệ Web 2.0 phát triển khá mạnh.

* **Các đặc tính của AngularJS**
* AngularJS là một Framwork phát triển dựa trên Javascript để tạo các ứng dụng web phong phú
* AngularJS thường dùng để phát triển frontend (giao diện khách hàng) thông qua các API để gọi data, sử dụng mô hình MVC rất mạnh mẽ
* Mã nguồn AngularJS tự động fix với các trình duyệt khác nhau nên bạn không cần phải lo vấn đề tương thích trình duyệt
* Angular là mã nguồn mở, hoàn toàn miễn phí và được phát triển bởi hàng ngàn các lập trình viên trên thế giới.

Chung quy lại có thể hiểu khi làm việc với AngularJS giống như là đang làm việc với Ajax, sử dụng cớ chế bind data, hoạt động theo mô hình MVC và sử dụng service để tương tác với dữ liệu từ server.

* **Các tính năng cố lõi quan trọng trong AngularJS**
* **Data-binding**: (liên kết dữ liệu) tự động đồng bộ dữ liệu giữa model và view
* **Scope**: (Phạm vi) Đây là những đối tượng kết nối giữa Controller và View
* **Controller**: Đây là những hàm javascript xử lý kết hợp với bộ điều khiển Scope
* **Service**: Như tôi đề cập ở trên, AngularJS sử dụng các API được xây dựng từ các web service (PHP, ASP) để thao tác với DB.
* **Filters**: Bộ lọc lọc ra các thành phẩn của một mảng và trả về mảng mới
* **Directives**:  đánh dấu vào các yếu tố của DOM, nghĩa là sẽ tạo ra các thẻ HTML tùy chỉnh
* **Templates**: hiển thị thông tin từ controller, đây là một thành phần của views
* **Routing**:  chuyển đổi giữa các action trong controller
* **MVC**: Mô hình chia thành phần riêng biệt thành Model, View, Controller. Đây là một mô hình khá hay nhưng trong Angular thì nó được chế biến lại một chút gần giốn với MVVM (Model View View Model)
* **Deep Linking:** Liên kết sâu, cho phép bạn mã hóa trạng thái của ứng dụng  trong các URL  để nó có thể đánh dấu được với công cụ tìm kiếm.
* **Dependency Injection:** Angular giúp các nhà phát triển tạo ứng dụng  dễ dàng hơn để phát triển, hiểu và thử nghiệm dễ dàng.

Sau đây là hình ảnh mô hình cac thành phần quan trọng trong AngularJS:



Hình -Các thành phần của AngularJS

* **Ưu điểm của AngularJS**:
* Angular cho phép tạo ra các ứng dụng một cách đơn giản, code sạch
* Angular sử dụng data bind giống .NET với tính năng liên kết với HTML nên giúp người dùng cảm thấy dễ chịu.
* Angular đang ở giai đoạn thử nghiệm
* Angular có thể chạy trên hầu hết các trình duyệt điện thoại thông minh.
* **Nhược điểm** **của AngularJS**:

Mặc dù angular có nhiều lợi thế ưu điểm nhưng nó cũng có mặt trái ngược lại:

* Không an toàn: Được phát triển từ javascript nên nó không an toàn, phía máy chủ phải thường xuyên xác nhận quyền để hệ thống chạy trơn tru.
* Phụ thuộc: Nếu người dùng vô hiệu hóa javascript thì ứng dụng không hoạt động được.

### 1.4.3. Tổng quan về Web Api

* **Giới thiệu Web Api**

Cùng với việc giới thiệu các công nghệ dịch vụ (service) trong thập niên gần đây, Microsoft đã tạo nên một chuỗi các công nghệ cho phép dễ dàng xây dựng và hỗ trợ các dịch vụ web trên nền tảng .NET.

Bắt đầu với XML Web Service, rồi sau đó được mở rộng thông qua  Windows Communication Foundation vài năm sau đó, Microsoft đã cho phép các nhà phát triển .NET có nhiều lựa chọn để xây dựng các dịch vụ trên cơ sở SOAP. Chỉ với một số thay đổi nhỏ trong cấu hình, sẽ cho phép bạn hỗ trợ rất nhiều thiết bị đa dạng về giao thức kết nối, cơ chế xác thực, định dạng thông báo cùng các chuẩn hỗ trợ bảo mật WS-\* với WCF.Nhưng khi thế giới các thiết bị kết nối phát triển, kéo theo nhu cầu về các giao tiếp đơn giản HTTP – chỉ sử dụng nền tảng dịch vụ - bỏ qua các yếu tố phức tạp như xác thực liên đoàn (federated authentication) hay mã hóa thông báo, cũng như không cần phải tạo phiên giao dịch (transaction), khám phá dịch vụ trên cơ sở Ngôn ngữ mô tả dịch vụ (Web Services Description Language – WSDL),…

Theo một cách đơn giản, đó là các giao tiếp dịch vụ di động (mobile-to-service) và trên cơ sở trình duyệt, các ứng dụng trang đơn lẻ (Single-Page Application – SPA). Khi đó các nhà phát triển có thể tạo các ứng dụng trên nền JavaScript, các ứng dụng mobile giá rất rẻ (nhỏ hơn 1 USD) – và các ứng dụng đó chỉ cần giao thức HTTP đơn giản, giao tiếp dữ liệu backend tương thích JSON.

Đáp lại đó, Microsoft đã tạo ra ASP.NET Web API, một nền tảng đầy hiệu quả mà rất đơn giản với giao thức HTTP và dịch vụ web theo mặc định sử dụng JSON. Các model làm việc  để xử lý và trả về được đóng gói qua các đối tượng đơn giản (Plain Old CLR Objects)

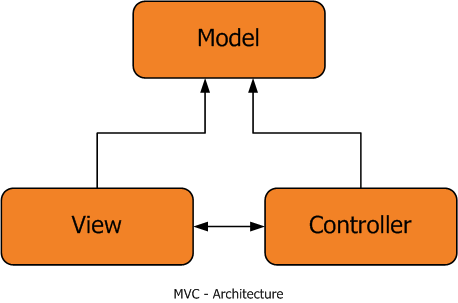
* **Những lợi ích khi sử dụng ASP.NET Web Api**
* **Cấu hình**: Ngay khi bạn xây dựng một website, không cần bất kỳ cấu hình gì chúng ta đã có thể cài đặt và sử dụng dịch vụ trên nền ASP.NET Web API. Khái niệm điểm đầu cuối (xuất hiện trong WCF) không còn, và cũng không cần contract. Theo đó bạn chỉ cần sử dụng REST (Representational State Transfer) URL, cùng các tham số đầu vào, và kết quả trả về là các bản tin JSON hay XML.
* **Mặc định sử dụng REST:** Việc xây dựng các dịch vụ với ASP.NET Web API cung cấp cho ta hầu hết những thành phần thiết yếu để xây dựng kiến trúc REST. Đó là bởi vì tính năng điều hướng (Route) URL tích hợp ngay trong nền tảng, các địa chỉ dịch vụ với ASP.NET Web API là các điều hướng RESTful ánh xạ với các phương thức của controller. Vì thế cho phép sử dụng các đường dẫn theo phong cách REST rất thân thiện.
* **Trìu tượng hóa với các Route:** Tương tự như các giao diện dịch vụ và sự thực thi chúng trên WCF, các route cho phép nhà phát triển ASP.NET Web API tạo ra lớp trìu tương giữa bên yêu cầu nhìn thấy và sự thực thi ở lớp dưới. Nói theo cách khác thì bạn có thể gắn kết bất cứ URL nào với bất cứ phương thức nào của các controller.

Với sự thay đổi về phiên bản dịch vụ, chỉ đơn giản bằng cách thêm một route mới (ví dụ “v2”) vào trong URL cho phép ta duy trì song song nhiều phiên bản dịch vụ trên hệ thống mà không ảnh hưởng đến các dị vụ đang có.

### 1.4.3. Asp.Net MVC

* **Kiến trúc của mô hình MVC**

Kiến trúc Model – View – Controller được sử dụng nhằm chia ứng dụng thành ba thành phần chính: model, view và controller. Nền tảng ASP.NET MVC giúp cho chúng ta có thể tạo được các ứng dụng web áp dụng mô hình MVC thay vì tạo ứng dụng theo mẫu ASP.NET Web Form. Đặc điểm nổi bật của ASP.NET MVC là nhẹ (lighweigt), dễ kiểm thử phần giao diện (so với ứng dụng Web Forms), tích hợp các tính năng có sẵn của ASP.NET. MVC là một mẫu thiết kế (design pattern) chuẩn mà nhiều lập trình viên đã quen thuộc. Một số loại ứng dụng web sẽ thích hợp với kiến trúc MVC. Một số khác vẫn thích hợp với ASP.NET Web Forms và cơ chế postbacks, một số ứng dụng kết hợp cả hai kiến trúc trên. Kiến trúc MVC bao gồm các thành phần dưới đây:

  
  
**Hình 1.2.** Mẫu Model – View – Controller

**Models**: Các đối tượng Models là một phần của ứng dụng, các đối tượng này thiết lập logic của phần dữ liệu của ứng dụng. Thông thường, các đối tượng model lấy và lưu trạng thái của model trong CSDL. Ví dụ như, một đối tượng Product (sản phẩm) sẽ lấy dữ liệu từ CSDL, thao tác trên dữ liệu và sẽ cập nhật dữ liệu trở lại vào bảng Products ở SQL Server.

Trong các ứng dụng nhỏ, model thường là chỉ là một khái niệm nhằm phân biệt hơn là được cài đặt thực thụ, ví dụ, nếu ứng dụng chỉ đọc dữ liệu từ CSDL và gởi chúng đến view, ứng dụng khong cần phải có tầng model và các lớp lien quan. Trong trường hợp này, dữ liệu được lấy như là một đối tượng model (hơn là tầng model).

**Views**: Views là các thành phần dùng để hiển thị giao diện người dùng (UI). Thông thường, view được tạo dựa vào thông tin dữ liệu model. Ví dụ như, view dùng để cập nhật bảng Products sẽ hiển thị các hộp văn bản, drop-down list, và các check box dựa trên trạng thái hiện tại của một đối tượng Product.

**Controllers**: Controller là các thành phần dùng để quản lý tương tác người dùng, làm việc với model và chọn view để hiển thị giao diện người dùng. Trong một ứng dụng MVC, view chỉ được dùng để hiển thị thông tin, controller chịu trách nhiệm quản lý và đáp trả nội dung người dùng nhập và tương tác với người dùng. Ví dụ, controller sẽ quản lý các dữ liệu người dùng gởi lên (query-string values) và gởi các giá trị đó đến model, model sẽ lấy dữ liệu từ CSDL nhờ vào các giá trị này.

Kiến trúc MVC giúp chúng ta tạo được các ứng dụng phân tách rạch ròi các thành phần (logic về nhập liệu, logic xử lý tác vụ và logic về giao diện). Kiến trúc MVC chỉ ra mỗi loại logic kể trên nên được thiếp lập ở đâu trên ứng dụng. Logic giao diện (UI logic) thuộc về views. Logic nhập liệu (input logic) thuộc về controller. Và logic tác vụ (Business logic – là logic xử lý thông tin, mục đích chính của ứng dụng) thuộc về model. Sự phân chia này giúp chúng ta giảm bớt được sự phức tạp của ứng dụng và chỉ tập trung vào mỗi khía cạnh cần được cài đặt ở mỗi thời điểm. Ví dụ như chúng ta chỉ cần tập trung vào giao diện (views) mà không phải quan tâm đến logic xử lý thông tin của ứng dụng.

Để quản lý sự phức tạp của ứng dụng, mẫu MVC giúp cho chúng ta có thể kiểm thử ứng dụng dễ dàng hơn hẳn so với khi áp dụng mẫu Web Forms. Ví dụ, trong một ứng dụng ASP.NET Web Forms, một lớp thường được sử dụng để hiển thị thông tin xuất ra cho người dùng và đồng thời xử lý thông tin người dùng nhập. Việc xây dựng các bộ test tự động cho ứng dụng Web Forms là rất phức tạp, bởi để kiểm thử mỗi trang web, chúng ta phải khởi tạo đối tượng trang, khởi tạo tất cả các control được sử dụng trong trang và các lớp phụ thuộc trong ứng dụng. Và bởi vì có quá nhiều lớp cần được khởi tạo để chạy được trang, thật khó để có thể viết các test chỉ tập trung vào một khía cạnh nào đó của ứng dụng. Và vì thế, kiểm thử đối với các ứng dụng dứa trên nền tảng Web Forms sẽ khó khăn hơn nhiều so với khi áp dụng trên ứng dụng MVC. Hơn thế nữa, việc kiểm thử trên nền tảng Web Forms yêu cầu phải sử dụng đến web server. Nền tảng MVC phân tách các thành phần và sử dụng các interface (khái niệm giao diện trong lập trình hướng đối tượng), và nhờ đó có thể kiểm thử các thành phần riêng biệt trong tình trạng phân lập với các yếu tố còn lại của ứng dụng.

Sự phân tách rạch ròi ba thành phần của ứng dụng MVC còn giúp cho việc lập trình diễn ra song song. Ví dụ như một lập trình viên làm việc với view, lập trình viên thứ hai lo cài đặt logic của controller và lập trình viên thứ ba có thể tập trung vào logic tác vụ của model tại cùng một thời điểm.

* **Lợi ích của ứng dụng web dựa trên mô hình MVC**

Chúng ta cần phải xem xét kỹ càng việc áp dụng mô hình ASP.NET MVC hay mô hình ASP.NET Web Forms khi xây dựng một ứng dụng. Mô hình MVC không phải là mô hình thay thế cho Web Forms, chúng ta có thể dùng một trong hai mô hình. Trước khi quyết định sử dụng MVC hay Web Forms cho một web site cụ thể, chúng tan cần phải phân tích lợi ích khi chọn một trong hai hướng.

Nền tảng ASP.NET MVC mang lại những lợi ích sau:

* Dễ dàng quản lý sự phức tạp của ứng dụng bằng cách chia ứng dụng thành ba thành phần model, view, controller
* Nó không sử dụng view state hoặc server-based form. Điều này tốt cho những lập trình viên muốn quản lý hết các khía cạnh của một ứng dụng.
* Nó sử dụng mẫu Front Controller, mẫu này giúp quản lý các requests (yêu cầu) chỉ thông qua một Controller. Nhờ đó chúng ta có thể thiết kế một hạ tầng quản lý định tuyến. Để có nhiều thông tin hơn, chúng ta có thể xem phần Front Controller trên web site MSDN.
* Hỗ trợ tốt hơn cho mô hình phát triển ứng dụng hướng kiểm thử (TDD)
* Nó hỗ trợ tốt cho các ứng dụng được xây dựng bởi những đội có nhiều lập trình viên và thiết kế mà vẫn quản lý được tính năng của ứng dụng
* **Các tính năng của nền tảng ASP.NET MVC**

Tách bạch các tác vụ của ứng dụng (logic nhập liệu, business logic, và logic giao diện), dễ dàng kiểm thử và mặc định áp dụng hướng phát triển TDD. Tất cả các tính năng chính của mô hình MVC được cài đặt dựa trên interface và được kiểm thử bằng cách sử dụng các đối tượng mocks, mock object là các đối tượng mô phỏng các tính năng của những đối tượng thực sự trong ứng dụng. Chúng ta có thể kiểm thử unit-test cho ứng dụng mà không cần chạy controller trong tiến trình ASP.NET, và điều đó giúp unit test được áp dụng nhanh chóng và tiện dụng. Chúng ta có thể sử dụng bất kỳ nền tảng unit-testing nào tương thích với nền tảng .NET.

MVC là một nền tảng khả mở rộng (extensible) & khả nhúng (pluggable). Các thành phần của ASP.NET MVC được thiết kể để chúng có thể được thay thế một cách dễ dàng hoặc dễ dàng tùy chỉnh. Chúng ta có thể nhúng thêm view engine, cơ chế định tuyến cho URL, cách kết xuất tham số của action-method và các thành phần khác. ASP.NET MVC cũng hỗ trợ việc sử dụng Dependency Injection (DI) và Inversion of Control (IoC). DI cho phép chúng ta gắn các đối tượng vào một lớp cho lớp đó sử dụng thay vì buộc lớp đó phải tự mình khởi tạo các đối tượng. IoC quy định rằng, nếu một đối tượng yêu cầu một đối tượng khác, đối tượng đầu sẽ lấy đối tượng thứ hai từ một nguồn bên ngoài, ví dụ như từ tập tin cấu hình. Và nhờ vậy, việc sử dụng DI và IoC sẽ giúp kiểm thử dễ dàng hơn.

ASP.NET MVC có thành phần ánh xạ URL mạnh mẽ cho phép chúng ta xây dựng những ứng dụng có các địa chỉ URL xúc tích và dễ tìm kiếm. Các địa chỉ URL không cần phải có phần mở rộng của tên tập tin và được thiết kế để hỗ trợ các mẫu định dạng tên phù hợp với việc tối ưu hóa tìm kiếm (URL) và phù hợp với lập địa chỉ theo kiểu REST.

Hỗ trợ sử dụng đặc tả (các thẻ) của các trang ASP.NET (.aspx), điều khiển người dùng (.ascx) và trang master page (.master). Chúng ta có thể sử dụng các tính năng có sẵn của ASP.NET như là sử dụng lồng các trang master page, sử dụng in-line expression (<% %>), sử dụng server controls, mẫu, data-binding, cục bộ hóa (localization) và hơn thế nữa.

Hỗ trợ các tính năng có sẵn của ASP.NET như cơ chế xác thực người dùng, quản lý thành viên, quyền, output caching và data caching, seession và profile, quản lý tình trạng ứng dụng, hệ thống cấu hình…

ASP.NET MVC 3 còn bổ sung một view engine mới là Razor View Engine cho phép thiết lập các view nhanh chóng, dễ dàng và tốn ít công sức hơn so với việc sử dụng Web Forms view engine.

# CHƯƠNG 2. THIẾT KẾ HỆ THỐNG

**2.1. Xác định bài toán và phạm vi sơ bộ**

Bài toán đặt ra: Xây dựng website học từ trực tuyến phục vụ cho nhiều đối tượng, những người mong muốn và yêu thích tiếng anh nói chung và từ vựng nói riêng.

Đối tượng của website: Người học, quản trị viên

* Đáp ứng nhu cầu học tập, học từ vựng theo từng chủ đề, từng bài học.
* Thực hiện việc quản lý người dùng, quản lý từ vựng, bài học bài kiểm tra cho người quản trị.

### 2.1.1 Khảo sát hiện trạng

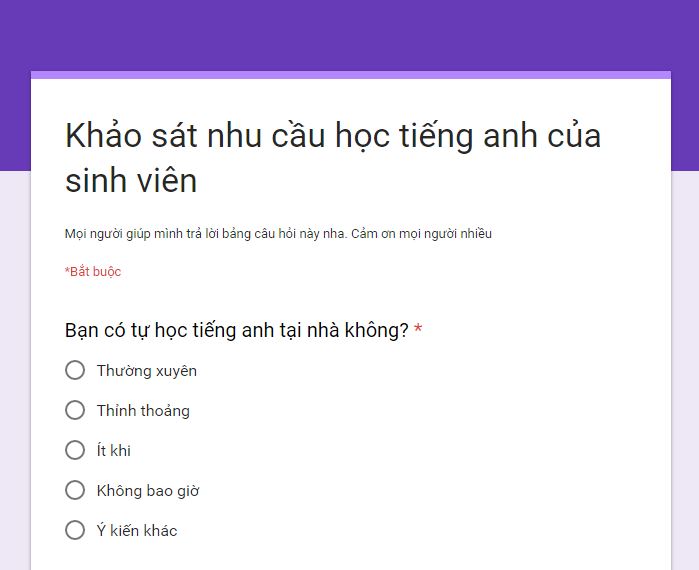
**- Mục tiêu khảo sát**

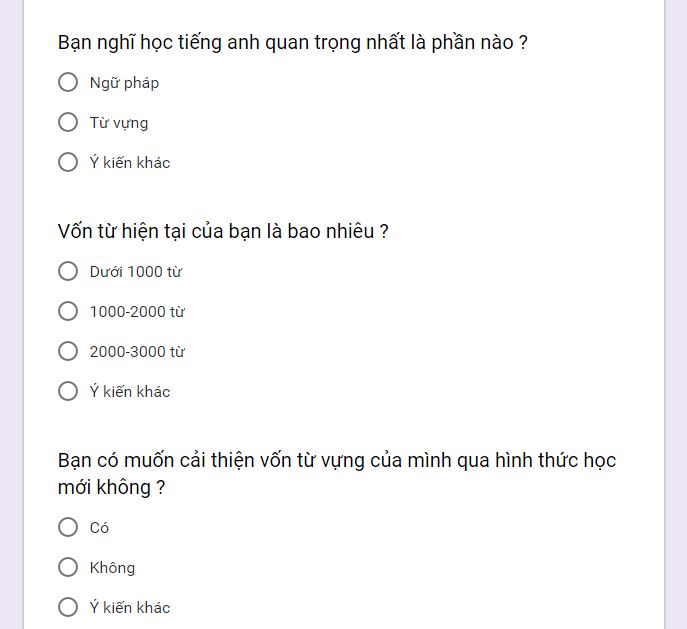
Địa điểm khảo sát: Sinh viên trong trường đại học, bạn bè xung quanh

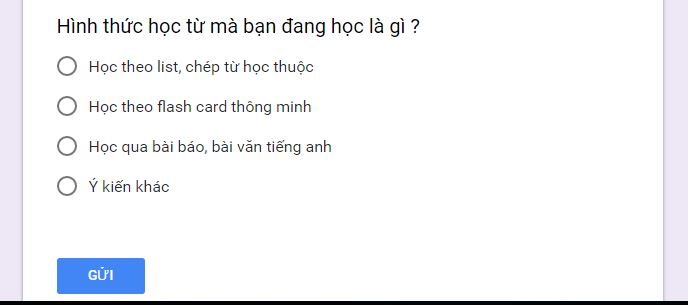
Quá trình khảo sát được thực hiện với các chủ đề sau:

* Tìm hiểu thực trạng và nhu cầu của từ vựng trong tiếng anh.
* Tìm hiểu phương pháp học mong muốn của người dùng.
* Định hướng xây dựng một web mới phù hợp trình độ cũng như mong muốn của đa số người dùng.

-**Một số câu hỏi khảo sát**

******

******

******

**- Hiện trạng**

Hiện trạng trình độ tiếng anh của đa phần học sinh, sinh viên hiện nay rất đáng lo ngại. Một trong những vấn đề được bàn đến tại các hội nghị là tình trạng học ngoại ngữ hiện nay của sinh viên, rằng việc dạy và học ngoại ngữ (nhất là Tiếng anh) đang trở nên “báo động”. Sinh viên mất nhiều kiến thức về từ vựng, ngữ pháp và kể cả “mất gốc” môn tiếng Anh ngay từ khi còn học THPT. Trong khi đó đa phần các công ty, nhà tuyển dụng hiện nay lại yêu cầu khả năng tiếng anh là điều kiện tiên quyết trong những tiêu chí của mình. Vì vậy vấn đề nâng cao trình độ tiếng anh là rất cấp thiết của mỗi học sinh, sinh viên nói riêng và toàn thể mọi người nói chung. Để góp phần giải quyết vấn đề đó, trước tiên ta cần cải thiện vốn từ vựng của bản thân mỗi ngày.

### 2.2. Phân tích yêu cầu bài toán

Từ những khảo sát sơ bộ và nhu cầu cũng như mong muốn đã khảo sát ở phần trước ta rút ra được một số yêu cầu chính sau:

### 2.2.1 *Yêu cầu chức năng*

- ***Về phía người dùng***: Cũng giống như những trang web khác người dùng cần có một tài khoản để đăng nhập thành viên của website. Khi đăng nhập vào website mà người dùng chưa được cấp tài khoản, sẽ có một giao diện đăng ký tài khoản gồm những trường thông tin cơ bản: Họ tên, email, mật khẩu… để người dùng có thể đăng ký tài khoản mới. Sau khi đã có tài khoản, người dùng có thể thay đổi hoặc chỉnh sửa lại thông tin cá nhân và bắt đầu quá trình học tập trên website.

Để bắt đầu học người dùng cần đăng nhập với tài khoản đã đăng ký và chọn bộ từ để bắt đầu (hệ thống sẽ có nhiều bộ từ, chia theo từng nhóm từ khác nhau ví dụ như: Từ vựng cho giao tiếp, từ vựng cho học sinh lớp 10,11,12…) cung cấp cho người dùng nhiều sự lựa chọn, trong bộ tự lại có nhiều chủ để (mỗi chủ đề lại gồm 20 từ vựng liên quan đến chủ đề đó) giúp người học có thể dễ dàng học tập theo từng chủ đề. Với mỗi chủ đề (tương ứng với từng bài học) người dùng sẽ có 3 bước bắt đầu.

**Bước 1** Sàng lọc từ. Bước này giúp người dùng có thể phân loại được những từ đã biết và chưa biết để người dùng tập trung vào những từ chưa biết, tránh lãng phí thời gian cho những từ đã biết.

**Bước 2** Học từ. Trong bước này người dùng có thể lựa chọn ôn tập ( đối với người đã học) và học từ đối với người chưa học. Với một giao diện flashcard được thiết kế để người dùng dễ dàng thao tác và ghi nhớ. Mỗi từ tương ứng với 3 mặt gồm các phần sau: mặt thứ nhất là hình ảnh và định nghĩa tiếng anh, mặt thứ hai gồm hình ảnh và phiên âm, nghĩa từ, mặt thứ 3 là hình ảnh và ví dụ cụ thể của hình ảnh đó. Sau khi học xong một từ bằng cách điền từ đang học vào ô kiểm tra, người dùng có thể chuyển sang từ tiếp theo để tiếp tục quá trình học cho đến khi hết bài học đó. Người dùng có thể học lại nhiều lần trước khi quyết định kiểm tra khả năng ghi nhớ của bài học vừa ôn tập.

**Bước 3** Kiểm tra: Sau quá trình học từ và ôn tập, người dùng có thể kiểm tra khả năng ghi nhớ của mình thông qua bài kiểm tra cho từng chủ đề. Cấu trúc bài kiểm tra có thể được chia làm 4 loại (5 câu nghe-viết, 5 câu chọn đáp án theo hình ảnh, 5 câu chọn đáp án theo định nghĩa, 5 câu ví dụ-chọn đáp án đúng).

Người dùng cũng thể xem được thông tin cá nhân, chủ đề mình đã học ,website cũng cung cấp một số tính năng xếp hạng, đưa ra danh sách 10 người dùng thành tích học tập xuất sắc nhất (dựa trên số lượng từ đã thuộc và chưa thuộc), và cũng đưa ra thứ hạng của người dùng trong danh sách người dùng trên website, tăng thêm sự hứng thú cho người học.

-**Về phía quản trị viên**:

Cập nhật thông tin tài khoản người dùng: quản trị có thể cấp quyền cho một số thành viên có thể cùng quản trị và xây dựng nội dung cho website

Quản trị có thể thêm sửa xóa một từ vựng mới cho phần từng bài học, trong từng chủ đề, cập nhật thông tin cho bài học(chủ đề), cập nhật thông tin cho bộ chủ đề tương ứng cho phù hợp.

Soạn bài và đăng tin thông báo mới cho người dùng.

Từ những phân tích sơ bộ trên ta xác định một số chức năng chính như hình dưới.

Hình 3.1: Sơ đồ chức năng tổng quan của hệ thống

**2.2 Đặc tả chức năng**

### 2.2.1 Chức năng quản lý người dùng

Bao gồm các chức năng con sau:

|  |  |
| --- | --- |
| **Bảng 2.1**. Đặc tả chức năng đăng ký tài khoản. | |
| Tên chức năng | **Đăng ký tài khoản** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Hệ thống hiển thị form “đăng ký” 2. Tác nhân nhập thông tin đăng ký trên form 3. Tác nhân nhấn nút “đăng ký”. 4. Hệ thống hiện thỉ kết quả đăng ký 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 2, tác nhân “Hủy bỏ”. Kết thúc chức năng. |
| Điều kiện tiên quyết | Không. |
| Điều kiện sau | Các thông tin trên form phải được nhập đầy đủ |

|  |  |
| --- | --- |
| **Bảng 2.2**. Đặc tả chức năng đăng nhập. | |
| Tên chức năng | **Đăng nhập** |
| Tác nhân chính | Quản trị viên, người dùng |
| Dòng sự kiện chính | 1. Hệ thống hiển thị form “đăng nhập” 2. Tác nhân nhập thông tin đăng nhập 3. Tác nhân nhấn nút “đăng nhập”. 4. Hệ thống chuyển sang màn hình thư viện hệ thống 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Không |
| Điều kiện tiên quyết | Tác nhân phải đăng ký tài khoản thành công. |
| Điều kiện sau | Không |

|  |  |
| --- | --- |
| Bảng 2.3. Đặc tả chức năng đăng xuất | |
| Tên chức năng | **Đăng xuất** |
| Tác nhân chính | Quản trị viên, người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “Đăng xuất”. 2. Hệ thống đăng xuất khỏi hệ thống và chuyển sang form đăng nhập 3. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 1, tác nhân “Hủy bỏ”.  Chức năng kết thúc. |
| Điều kiện tiên quyết | Người dùng phải đăng nhập thành công. |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.4. Đặc tả chức năng cập nhật thông tin tài khoản | |
| Tên chức năng | **Cập nhật thông tin tài khoản** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “Thông tin cá nhân”. 2. Hệ thống hiển thị form thông tin cá nhân 3. Tác nhân cập nhật thông tin cá nhân 4. Tác nhân nhấn nút “Cập nhật” 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 3, tác nhân “Hủy bỏ”.  Chức năng kết thúc. |
| Điều kiện tiên quyết | Người dùng phải đăng nhập thành công. |
| Điều kiện sau | Không có |

### 2.2.2 Chức năng quản lý từ vựng

Bao gồm những chức năng con sau:

|  |  |
| --- | --- |
| Bảng 2.5. Đặc tả chức năng cập nhật từ vựng | |
| Tên chức năng | **Cập nhật từ vựng** |
| Tác nhân chính | Quản trị viên |
| Dòng sự kiện chính | 1. Tác nhân chọn “cập nhật từ vựng”. 2. Hệ thống hiển thị form cập nhật từ vựng 3. Tác nhân cập nhật thông từ vựng 4. Tác nhân nhấn nút “Cập nhật” 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 3, tác nhân “Hủy bỏ”.  Chức năng kết thúc. |
| Điều kiện tiên quyết | Người dùng phải đăng nhập thành công với tài khoản admin hoặc tài khoản được ủy quyền. |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.6. Đặc tả chức năng cập nhật nhóm chủ đề (nhóm các bài học) | |
| Tên chức năng | **Cập nhật bộ chủ đề** |
| Tác nhân chính | Quản trị viên |
| Dòng sự kiện chính | 1. Tác nhân chọn “cập nhật nhóm chủ đề”. 2. Hệ thống hiển thị form cập nhật nhóm chủ đề 3. Tác nhân cập nhật thông về nhóm chủ đề cần cập nhật 4. Tác nhân nhấn nút “Cập nhật” 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 3, tác nhân “Hủy bỏ”.  Chức năng kết thúc. |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập thành công với tài khoản admin hoặc tài khoản được ủy quyền. |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.7. Đặc tả chức năng cập nhật nhóm chủ đề (nhóm các bài học) | |
| Tên chức năng | **Cập nhật chủ đề (bài học)** |
| Tác nhân chính | Quản trị viên |
| Dòng sự kiện chính | 1. Tác nhân chọn “cập nhật chủ đề”. 2. Hệ thống hiển thị form cập nhật chủ đề 3. Tác nhân cập nhật thông về chủ đề cần cập nhật 4. Tác nhân nhấn nút “Cập nhật” 5. Chức năng kết thúc. |
| Dòng sự kiện phụ | Nếu tại dòng sự kiện chính số 3, tác nhân “Hủy bỏ”.  Chức năng kết thúc. |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập thành công với tài khoản admin hoặc tài khoản được ủy quyền. |
| Điều kiện sau | Không có |

### 2.2.3 Chức năng thống kê

Bao gồm các chức năng chính sau:

|  |  |
| --- | --- |
| Bảng 2.8. Đặc tả chức năng thống kê tình trạng học theo chủ đề: | |
| Tên chức năng | **Cập nhật chủ đề (nhóm bài học)** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “nhóm chủ đề”. 2. Hệ thống hiển thị biểu đồ tròn thống kê tình trạng học của nhóm từ vựng đó của tác nhân. 3. Tác nhân chọn “nhóm chủ đề khác” 4. Chức năng kết thúc. |
| Dòng sự kiện phụ | Không |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.9. Đặc tả chức năng thống kê tình trạng học theo chủ đề (bài học): | |
| Tên chức năng | **Cập nhật chủ đề (bài học)** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “nhóm chủ đề”. 2. Hệ thống hiển thị biểu đồ tròn thống kê tình trạng học của nhóm từ vựng đó của tác nhân. 3. Tác nhân chọn “nhóm chủ đề khác” 4. Chức năng kết thúc. |
| Dòng sự kiện phụ | Không |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.10. Đặc tả chức năng thống kê tình trạng học theo chủ đề (bài học): | |
| Tên chức năng | **đánh giá kết quả kiểm tra cho từng bài kiểm tra** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “kiểm tra” bài học. 2. Hệ thống hiển thị bài kiểm tra. 3. Tác nhân chọn thực hiện trả lời câu hỏi. 4. Sau khi kết thúc các câu hỏi biểu đồ đánh giá kết quả xuất hiện. 5. Người dùng nhấn nút “Close” 6. Chức năng kết thúc. |
| Dòng sự kiện phụ | Không |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

### 2.2.4 Chức năng học và kiểm tra.

Bao gồm những chức năng sau:

|  |  |
| --- | --- |
| Bảng 2.11. Đặc tả chức năng thống kê tình trạng học theo chủ đề (bài học): | |
| Tên chức năng | **Học từ theo chủ đề** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “chủ đề” cần học. 2. Hệ thống hiển thị những câu hỏi của chủ đề đó. 3. Tác nhân chọn thực hiện trả lời câu hỏi. 4. Sau khi kết thúc các câu hỏi người dùng có thể học lại 5. Người dùng nhấn nút “Close” 6. Chức năng kết thúc. |
| Dòng sự kiện phụ | -Người dùng nhấn nút thoát. Chức năng kết thúc. |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

|  |  |
| --- | --- |
| Bảng 2.12. Đặc tả chức năng tra từ theo chủ đề | |
| Tên chức năng | **Học từ theo chủ đề** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Tác nhân chọn “chủ đề” cần học. 2. Hệ thống hiển thị những câu hỏi của chủ đề đó. 3. Tác nhân chọn thực hiện trả lời câu hỏi. 4. Sau khi kết thúc các câu hỏi người dùng có thể học lại 5. Người dùng nhấn nút “Close” 6. Chức năng kết thúc. |
| Dòng sự kiện phụ | -Người dùng nhấn nút thoát. Chức năng kết thúc. |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

### 2.2.5 Chức năng thông báo Bao gồm những chức năng sau:

|  |  |
| --- | --- |
| Bảng 2.12. Đặc tả chức năng tra từ theo chủ đề | |
| Tên chức năng | **Học từ theo chủ đề** |
| Tác nhân chính | Người dùng |
| Dòng sự kiện chính | 1. Sau khi tác nhân “đăng nhập”. 2. Hệ thống hiển thị số lượng chủ đề cần ôn tập trên tag notify. 3. Tác nhân nhấn “notify”. 4. Hệ thống hiển thị danh sách chủ đề cần ôn tập 5. Người dùng chọn chủ đề để ôn tập 6. Chức năng kết thúc. |
| Dòng sự kiện phụ | -Người dùng nhấn nút thoát. Chức năng kết thúc. |
| Điều kiện tiên quyết | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Điều kiện sau | Không có |

**2.3 Thiết kế cơ sở dữ liệu**

### 2.3.1 Thông tin cần lưu trữ

Thông qua quá trình khảo sát và phân tích trên ta xác định một số thông tin cần phải lưu trữ như sau:

**Người dùng**: Mỗi người dùng cần những thông tin lưu trữ sau: mã người dùng, tên đăng nhập, mật khẩu, email.

**Loại từ vựng**: Loại từ vựng cần lưu trữ thông tin sau: Mã loại từ, tên loại từ, mô tả chi tiết.

**Từ vựng:** Mỗi từ vựng bao gồm những thông tin cần lưu trữ như: Mã từ, từ vựng, phiên âm, dịch nghĩa, phát âm, hình ảnh minh họa, mã bài học, mã loại từ.

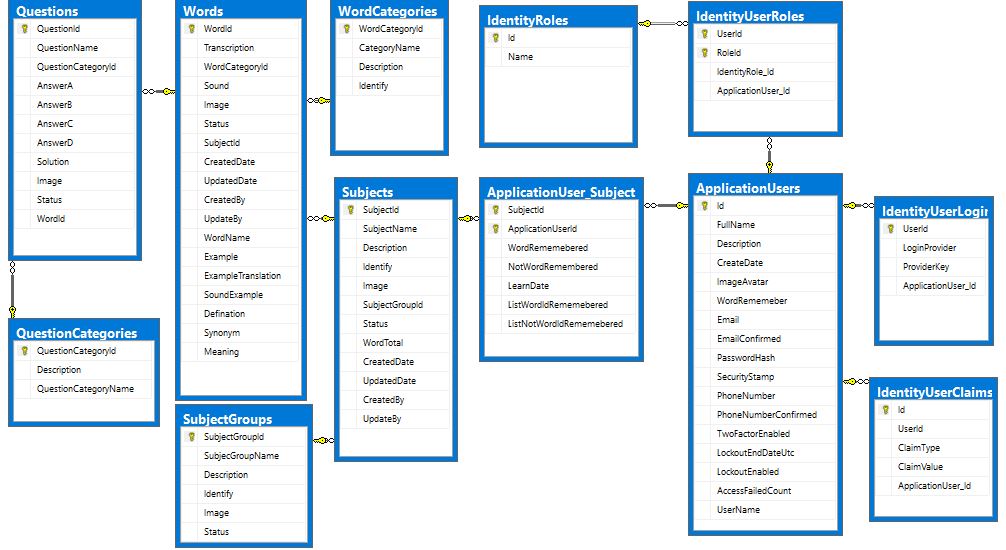
**Nhóm chủ đề:** Bao gồm những thông tin sau cần lưu trữ: Mã nhóm chủ đề, tên nhóm chủ đề, ảnh minh họa, mô tả chi tiết, số lượng từ vựng.

**Bài học:** Mỗi bài học bao gồm những thông tin sau: Mã bài học, tên bài học, mã nhóm chủ đề, số lượng từ vựng

Do mỗi người dùng có thể tham gia học nhiều bài học và một bài học cũng có thể có nhiều người học nên ta cần lưu trữ thêm về thông tin người dùng và bài học với những thông tin sau mã bài học, mã người dùng, thời gian học, danh sách từ đã thuộc, danh sách từ chưa thuộc (phục vụ cho việc xây dựng kết quả kiểm tra sau mỗi bài học).

**Người dùng-bài học:** Những thông tin cần lưu trữ như sau: Mã người dùng, mã bài học, số lượng từ đã thuộc, từ chưa thuộc, thời gian học, danh sách từ chưa thuộc, danh sách từ đã thuộc.

### 2.3.1 Mô tả cấu trúc cơ sở dữ liệu



**Hình 3.8**. Sơ đồ quan hệ thực thể

**Chi tiết các bảng**

- Tên bảng: **Questions**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | QuestionId (PK) | int | Mã câu hỏi |
| 2 | QuestionName | nchar(30) | Câu hỏi |
| 3 | QuestionCategoryId (FK) | int | Loại câu hỏi |
| 4 | AnswerA | nvarchar(200) | Phương án trả lời thứ nhất |
| 5 | AnswerB | nvarchar(200) | Phương án trả lời thứ hai |
| 6 | AnswerC | nvarchar(200) | Phương án trả lời thứ ba |
| 7 | AnswerC | nvarchar(200) | Phương án trả lời thứ tư |
| 8 | Solution | nvarchar(200) | Đáp án trả lời đúng |

- Tên bảng: **QuestionCategories**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | QuestionCategoryId (PK) | int | Mã loại câu hỏi |
| 2 | Description | nchar(300) | Mô tả |
| 3 | QuestionCategoryName | int | Tên loại câu hỏi |

- Tên bảng: **WordCategories**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | WordCategoryId(PK) | int | Mã loại từ |
| 2 | Description | nchar(300) | Mô tả |
| 3 | WordCategoryName | int | Tên loại từ |
| 4 | Identify | Nvarchar(300) | Viết tắt |

- Tên bảng: **Words**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | WordId(PK) | int | Mã từ |
| 2 | Transcription | nchar(300) | Phiên âm |
| 3 | WordCategoryId(FK) | int | Tên loại câu hỏi |
| 4 | Sound | nvarchar(300) | Đường dẫn phát âm |
| 5 | Image | nvarchar(300) | Đường dẫn ảnh |
| 6 | SubjectId(FK) | int | Mã bài học |
| 7 | WordName | nvarchar(300) | Từ vựng |
| 8 | Defination | nvarchar(300) | Định nghĩa tiếng anh |
| 9 | Meaning | nvarchar(300) | Nghĩa của từ |
| 10 | Example | nvarchar(300) | Ví dụ tiếng anh |
| 11 | ExampleTraslation | nvarchar(300) | Dịch nghĩa ví dụ |
| 12 | SounExample | nvarchar(300) | Đường dẫn đọc ví dụ |
| 13 | CreateDate | Datetime | Ngày tạo |
| 14 | UpdateDate | Datetime | Ngày chỉnh sửa |
| 15 | CreateBy | nvarchar(300) | Người tạo |
| 16 | UpdateBy | nvarchar(300) | Người chỉnh sửa |

- Tên bảng: **SubjectGroups**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | SubjectGroupId (PK) | int | Mã nhóm chủ đề |
| 2 | SubjectGroupName | nvarchar(300) | Tên nhóm |
| 3 | Description | nvarchar(300) | Mô tả |
| 4 | Title | nvarchar(300) | Tiêu đề |
| 5 | Image | nvarchar(300) | Đường dẫn ảnh minh họa |
| 6 | Status | Bit | Trạng thái |

- Tên bảng: **Subjects**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | SubjectId(PK) | int | Mã chủ đề (bài học) |
| 2 | SubjectName | nvarchar(300) | Tên chủ đề |
| 3 | Description | nvarchar(300) | Mô tả |
| 4 | Title | nvarchar(300) | Tiêu đề |
| 5 | Image | nvarchar(300) | Đường dẫn ảnh minh họa |
| 6 | WordTotal | Int | Tổng số lượng từ vựng |
| 7 | CreateDate | datetime | Ngày tạo |
| 8 | UpdateDate | datetime | Ngày chỉnh sửa |
| 9 | CreateBy | nvarchar(300) | Người tạo |
| 10 | UpdateBy | nvarchar(300) | Người chỉnh sửa |
| 9 | Status | Bit | Trạng thái |

- Tên bảng: **Subjects**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | SubjectId(PK) | int | Mã chủ đề (bài học) |
| 2 | SubjectName | nvarchar(300) | Tên chủ đề |
| 3 | Description | nvarchar(300) | Mô tả |
| 4 | Title | nvarchar(300) | Tiêu đề |
| 5 | Image | nvarchar(300) | Đường dẫn ảnh minh họa |
| 6 | WordTotal | int | Tổng số lượng từ vựng |
| 7 | CreateDate | datetime | Ngày tạo |
| 8 | UpdateDate | datetime | Ngày chỉnh sửa |
| 9 | CreateBy | nvarchar(300) | Người tạo |
| 10 | UpdateBy | nvarchar(300) | Người chỉnh sửa |
| 9 | Status | Bit | Trạng thái |

- Tên bảng: **ApplicationUsers**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | Id(PK) | int | Mã người dùng |
| 2 | UserName | nvarchar(300) | Tên đăng nhập |
| 3 | FullName | nvarchar(300) | Tên đầy đủ |
| 4 | Email | nvarchar(300) | Địa chỉ email |
| 5 | PasswordHash | nvarchar(300) | Mật khẩu đã được mã hóa |
| 6 | Số điện thoại | nvarchar(300) | Số điện thoại |
| 7 | WordRemember | Int | Số lượng từ đã thuộc |
| 8 | Address | nvarchar(300) | Địa chỉ |
| 9 | ImageAvatar | nvarchar(300) | Đường dẫn ảnh đại diện |

- Tên bảng: **ApplicationUser\_Subject**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tên cột** | **Kiểu dữ liệu** | **Ý nghĩa** |
| 1 | SubjectId(PK) | int | Mã chủ đề |
| 2 | ApplicationUserId(PK) | nchar(300) | Mã người dùng |
| 3 | WordRemembered | int | Từ đã thuộc |
| 4 | NotWordRemember | int | Từ chưa thuộc |
| 5 | LearnDate | datetime | Ngày học |
| 6 | ListWordIdRemember | nvarchar(100) | Danh sách mã từ đã thuộc |
| 7 | ListWordIdNotRemember | nvarchar(100) | Danh sách từ chưa thuộc |

Và một số bảng của Identity ASP.NET khác tự sinh khi sử dụng Identity ASP.NET

như trong sơ đồ quan hệ thực thể.

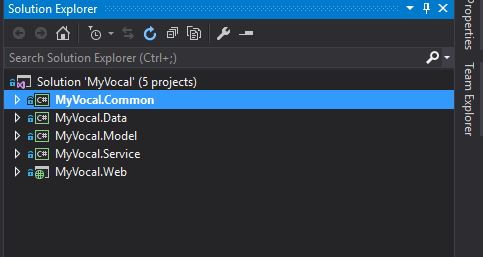
# CHƯƠNG 3. CÀI ĐẶT VÀ THỬ NGHIỆM WEBSITE

### 3.1. Cấu trúc của project

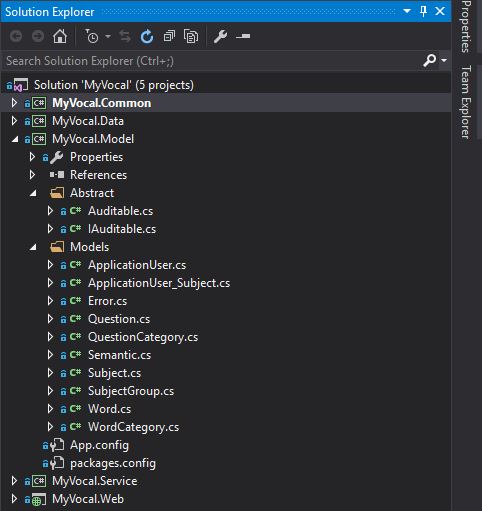
**Công nghệ triển khai**:

* Phần database: Sử dụng Entity Framework code first để truy cập và thao tác dữ liệu.
* Phần Service sử dụng WebApi
* Phần quản trị sử dụng: AngularJS+WebApi để xây dựng
* Phần hiển thị cho người dùng sử dụng: Asp.net MVC
* Chứng thực người dùng: ASP.NET Identity

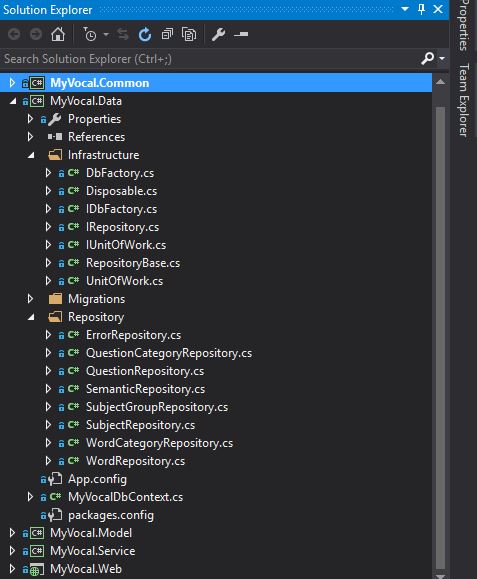
**Cấu trúc project: Solution** bao gồm 5 project như hình dưới



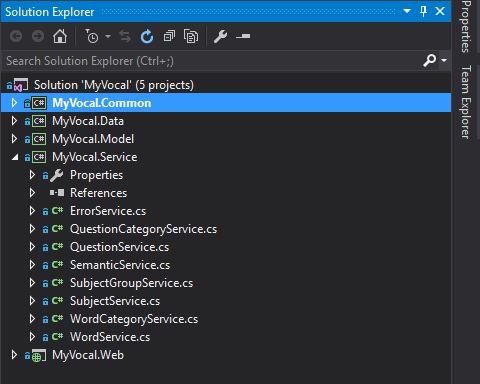
* Myvocal.Common: Chứa tiện ích dùng chung cho
* Myvocal.Model: Chứa các lớp Domain Entities



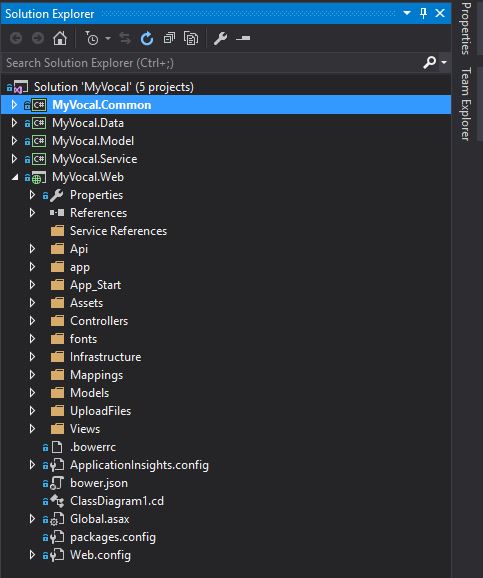
* Myvocal.Data: Chứa tầng truy cập dữ liệu sử dụng Entities Framework Code First



* MyVocal.Service: Chứa các service xử lý business logic



* MyVocal.Web: Project chính để hiển thị giao diện và tương tác người dùng



### 3.2. Cài đặt các module

### 3.2.1 Module đăng ký

|  |  |
| --- | --- |
| Bảng 2.12. Xây dựng module đăng ký đăng nhập. | |
| Tên modul | **Đăng ký đăng ký** |
| Bảng liên quan | ApplicationUser |
| Công nghệ sử dụng | Asp.net MVC, Asp.net Identity |
| Mô tả | Người dùng sẽ cần một tài khoản để đăng ký sử dụng thông tin trên website các bước để đăng ký   1. Người dùng mở form đăng ký 2. Người dùng điền đầy đủ thông tin trên form 3. Người dùng ấn đăng ký để hoàn thành quá trình đăng ký |
| Quy trình cài đặt | -Người dùng phải đăng nhập để có thể sử dụng chức năng |
| Giao diện | |
|  | |

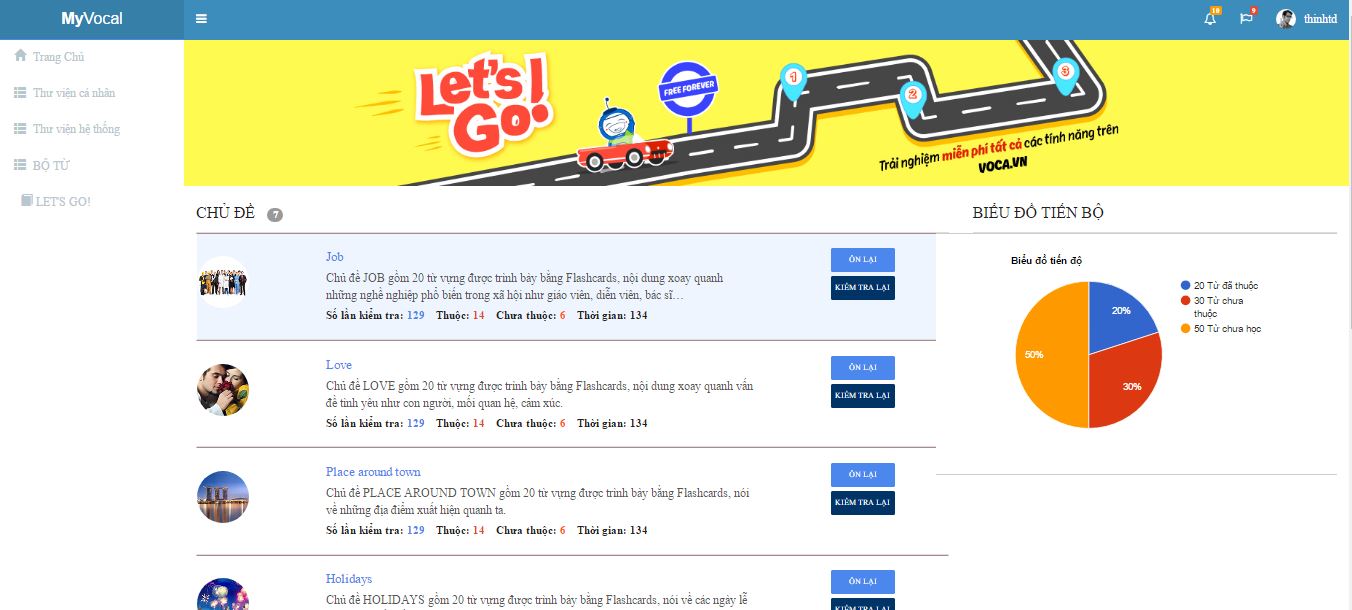
### 3.3.1 Xây dựng module đăng nhập

|  |  |
| --- | --- |
| Bảng 2.12. Xây dựng modul đăng ký đăng nhập | |
| Tên modul | **Đăng ký đăng ký** |
| Bảng liên quan | ApplicationUser |
| Công nghệ sử dụng | Asp.net MVC, Asp.net Identity |
| Mô tả | Khi người dùng đã đăng ký thành công, có thể sử dụng tài khoản đó để đăng nhập hệ thống.   1. Người dùng mở form đăng nhập 2. Người dùng điền đầy đủ thông tin trên form 3. Người dùng ấn đăng nhập để hoàn thành quá trình đăng nhập 4. Website chuyển sang trang thư viện hệ thống như giao diện bên dưới |
| Quy trình cài đặt |  |
| Giao diện | |
|  | |

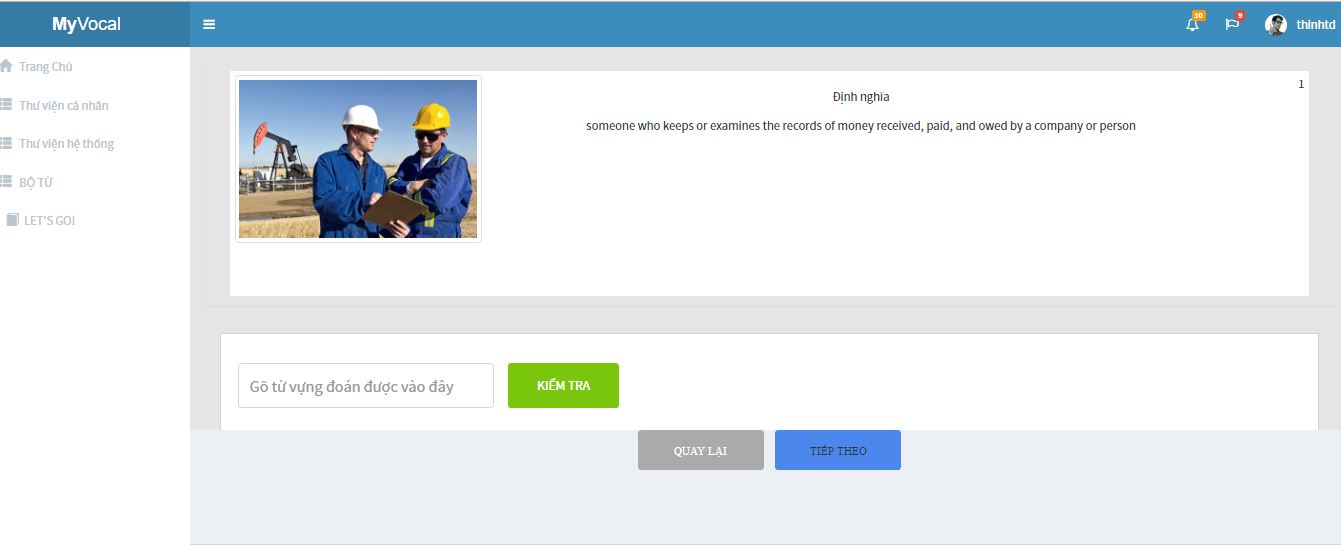
## 3.4. Chức năng học từ vựng

a. Mô tả: Khi người dùng đã đăng nhập thành công, có thể sử dụng tài khoản đó để học tập trên trang web. Từ trang thư viện hệ thống người dùng có thể chọn một chủ đề mình thích để bắt đầu học.

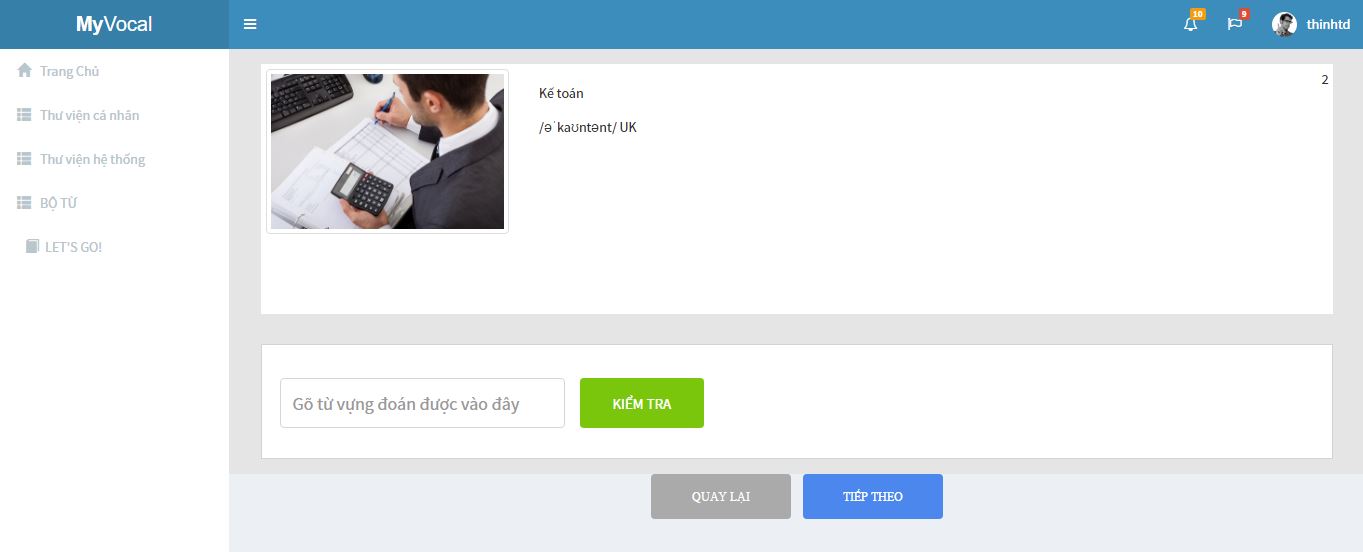
Trong một chủ đề sẽ có nhiều bài học, tùy vào mỗi chủ để giúp người học tập trung và dễ dàng hơn cho việc tiếp thu, tương ứng với mỗi chủ đề sẽ có một biểu đồ tiến độ nhỏ, giúp người học dễ dàng lắm bắt tiến độ học của mình trong chủ đề đó.



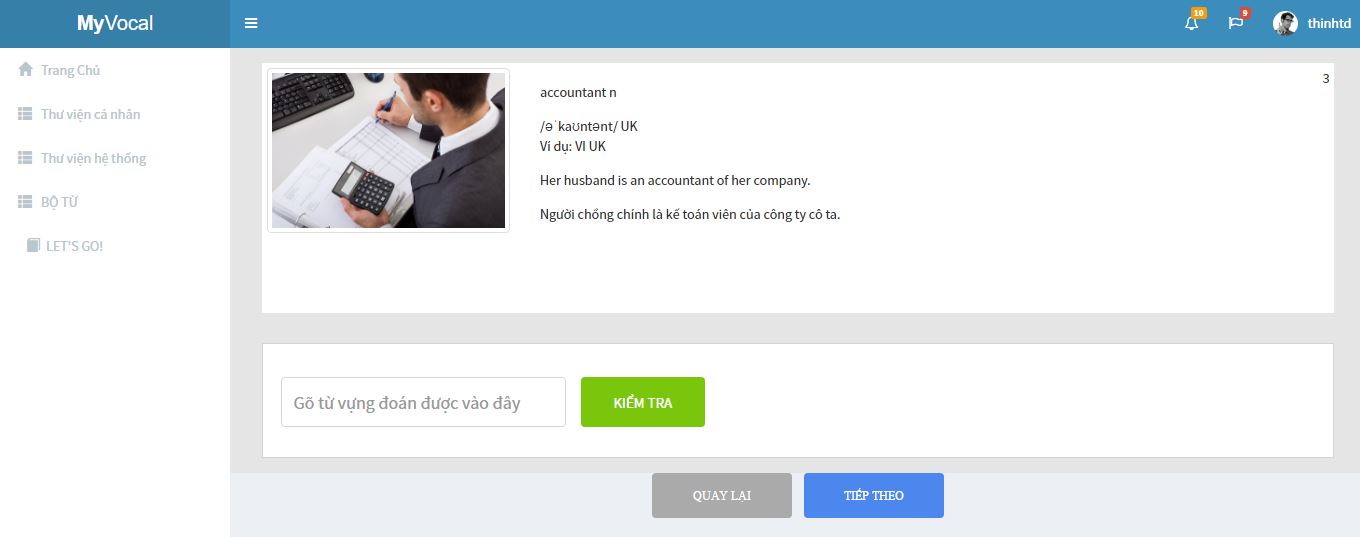
Khi người dùng click chọn vào một bài học sẽ có một giao diện học bài cho từng từ, với mỗi từ sẽ tương ứng với định nghĩa tiếng anh của từ đó



Phiên âm + hình ảnh của từ đó



Hình ảnh+ Ví dụ



Khi người dùng chuyển qua từng phát âm của từ đó sẽ được đọc lên

Người dùng có thể nhập vào ô ‘Gõ từ vựng đoán được vào đây’ và ấn kiểm tra để vừa học thuộc mặt chữ vừa kiểm tra đáp án mình vừa nhập có đúng hay không

b. Bảng liên quan: bảng Subject(Chủ đề), Word(Từ vựng).

c. Kỹ thuật cài đặt: Sử dụng Asp.Net MVC thuần, sử dụng Jquery+Ajax để lấy dữ liệu và xử lý trên View.

`