Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Hệ thống Quản lý cửa hàng sử dụng Microsoft .NET WEB API

Nguyễn Mạnh Duy

Hà Nội, 12/2019

Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Hệ thống Quản lý cửa hàng sử dụng Microsoft .NET WEB API

|  |  |
| --- | --- |
| Sinh viên thực hiện | Nguyễn Mạnh Duy |
| Người hướng dẫn | PGS. TS. Trương Diệu Linh |

Hà Nội, 12/2019

# Lời cam kết

Họ và tên sinh viên: Nguyễn Mạnh Duy

Điện thoại liên lạc: 01696510249 Email: nguyenduysoict.2310@gmail.com

Lớp: CNTT1.03-K60 Hệ đào tạo: Kỹ sư

Em –Nguyễn Mạnh Duy – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân em dưới sự hướng dẫn của PGS.TS. Trương Diệu Linh. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng em, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Em xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 28 tháng 5 năm 2018*  Tác giả ĐATN  *Nguyễn Mạnh Duy* Nhận xét của giáo viên hướng dẫn ……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………… |

# Lời cảm ơn

Trong thời gian làm đồ án tốt nghiệp, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè.  Với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến tất cả các cá nhân và cơ quan đã tạo điều kiện giúp đỡ trong quá trình học tập và hoàn thành đồ án.

Em xin cảm ơn PGS.TS. Trương Diệu Linh – Phó trưởng Bộ môn Truyền thông và Mạng máy tính đã tận tình hướng dẫn em trong suốt quá trình làm Đồ án Tốt Nghiệp này.

Em xin cảm ơn anh Tạ Tô Minh Chí – sinh viên K59 Bách khoa chuyên ngành Công nghệ phần mềm đã hỗ trợ em hoàn thành báo cáo này.

Xin cảm ơn gia đình, bạn bè đã động viên, quan tâm và là nguồn động lực lớn lao giúp em trong quá trình làm đồ án.

# Tóm tắt

ĐATN của em có nội dung về công nghệ Microsoft .NET API và ứng dụng nó trong hệ thống quản lý cửa hàng. Việc quản lý hàng hóa, nhân viên, các khoản doanh thu cũng chi phí của một cửa hàng vẫn luôn là bài toán cần có giải pháp để giúp cho việc quản lý trở lên tối ưu nhất. Bên cạnh đó việc phát triển các hệ thống chuỗi cửa hàng hiện nay cũng như việc sử dụng các ứng dụng trên mọi nền tảng lại làm cho việc phát triển một hệ thống quản lý trở nên khó khăn hơn. Các hệ thống quản lý cửa hàng còn sử dụng các công nghệ cũ như MVC, khiến cho hệ thống không đạt hiệu quả cao. Công nghệ .NET API là một công nghệ mới của Microsoft, giải quyết tốt vấn đề cân tải ở hệ thống cũng như hệ thống đa nền tảng. Với mong muốn đưa ra một giải pháp tối ưu hơn cho bài toán quản lý, em quyết định sử dụng công nghệ này trong đồ án của mình nhằm xây dựng một hệ thống quản lý cửa hàng tốt hơn cũng như mở rộng ra để giải quyết các bài toán khác như tra cứu điểm thi đại học, thương mại điện tử, các ứng dụng đa nền tảng. Giúp cho công việc quản lý cửa hàng, nhất là hệ thống chuỗi cửa hàng trở nên dễ dàng hơn, cùng với đó là ứng dụng rộng rãi trong các lĩnh vực khác trong đời sống, như là hệ thống tra cứu điểm thi đại học quá tải mỗi mùa tuyển sinh, các ứng dụng đa nền tảng chạy được cả trên web, desktop và mobile, …

# Mục lục

1. **Contents**

[**Lời cam kết iii**](#_gjdgxs)

[**Nhận xét của giáo viên hướng dẫn iv**](#_30j0zll)

[**Lời cảm ơn v**](#_1fob9te)

[**Tóm tắt vi**](#_3znysh7)

[**Mục lục vii**](#_tyjcwt)

[**Danh mục hình vẽ**](#_3dy6vkm) **viii**

[**Danh mục bảng ix**](#_1t3h5sf)

[**Danh mục các từ viết tắt x**](#_4d34og8)

[**Danh mục thuật ngữ xi**](#_2s8eyo1)

[**Chương 1 Giới thiệu đề tài 1**](#_3rdcrjn)

[**1.1 Đặt vấn đề 1**](#_26in1rg)

[**1.2 Mục tiêu và phạm vi đề tài 2**](#_lnxbz9)

[**1.3 Định hướng giải pháp 3**](#_35nkun2)

[**1.4 Bố cục đồ án 4**](#_1ksv4uv)

[**Chương 2 Khảo sát và phân tích yêu cầu 6**](#_44sinio)

[**2.1 Khảo sát hiện trạng 6**](#_2jxsxqh)

[**2.2 Tổng quan chức năng 8**](#_4i7ojhp)

[2.2.1 Biểu đồ use case tổng quan 8](#_2xcytpi)

[2.2.2 Biểu đồ use case phân rã 10](#_3as4poj)

[2.2.3 Quy trình nghiệp vụ 13](#_147n2zr)

[**2.3 Đặc tả chức năng 16**](#_32hioqz)

[**2.4 Yêu cầu phi chức năng 37**](#_sqyw64)

[**Chương 3 Công nghệ sử dụng 39**](#_1rvwp1q)

[**3.1 Danh sách các công nghệ sử dụng 39**](#_4bvk7pj)

[**3.2 HTML, CSS, Javascript 40**](#_2r0uhxc)

[**3.3 JQuery và jQueryUI 40**](#_1664s55)

[**3.4 C# và .NET Framework 41**](#_3q5sasy)

[**3.5 RESTful Web Service 42**](#_25b2l0r)

[**3.6 Microsoft .NET Web API 45**](#_43ky6rz)

[**Chương 4 Phát triển và triển khai ứng dụng 52**](#_pkwqa1)

[**4.1 Thiết kế kiến trúc 52**](#_39kk8xu)

[4.1.1 Lựa chọn kiến trúc 52](#_1opuj5n)

[4.1.2 Thiết kế tổng quan 54](#_2nusc19)

[4.1.3 Thiết kế chi tiết gói 55](#_3mzq4wv)

[**4.2 Thiết kế chi tiết 57**](#_319y80a)

[4.2.1 Thiết kế giao diện 57](#_1gf8i83)

[4.2.2 Thiết kế lớp 58](#_upglbi)

[4.2.3 Thiết kế cơ sở dữ liệu 61](#_meukdy)

[**4.3 Xây dựng ứng dụng 63**](#_zu0gcz)

[4.3.1 Thư viện và công cụ sử dụng 63](#_3jtnz0s)

[4.3.2 Kết quả đạt được 63](#_4iylrwe)

[4.3.3 Minh hoạ các chức năng chính 64](#_1d96cc0)

[**4.4 Triển khai 66**](#_3x8tuzt)

[**Chương 5 Các giải pháp và đóng góp nổi bật 67**](#_2ce457m)

[**5.1 .NET WEB API được ứng dụng như thế nào trong Project 67**](#_rjefff)

[**5.2 Tại sao phải áp dụng kiến trúc REST? 70**](#_1qoc8b1)

[**Chương 6 Kết luận và hướng phát triển 76**](#_243i4a2)

[**6.1 Kết luận 76**](#_j8sehv)

[**6.2 Hướng phát triển 76**](#_338fx5o)

[**Tài liệu tham khảo 77**](#_1idq7dh)

# Danh mục hình vẽ

# Danh mục bảng

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| API | Application Programming Interface  Giao diện lập trình ứng dụng |
| **EUD** | End-User Development  Phát triển ứng dụng người dùng cuối |
| **GWT** | Google Web Toolkit  Công cụ lập trình Javascript bằng Java của Google |
| **HTML** | HyperText Markup Language  Ngôn ngữ đánh dấu siêu văn bản |
| **CNTT** | Công nghệ thông tin |
| **ĐATN** | Đồ án tốt nghiệp |
| **SV** | Sinh viên |

# Danh mục thuật ngữ

|  |  |
| --- | --- |
| Browser | Trình duyệt |
| **Cache memory** | Bộ nhớ đệm |
| **E-commerce** | Thương mại điện tử |
| **Bloatware** | Ứng dụng nhà sản xuất tích hợp vào thiết bị |
| **Interpreter** | Trình thông dịch |
| **Compiler** | Trình biên dịch |

# Giới thiệu đề tài

## Đặt vấn đề

Thưa thầy cô và các bạn sinh viên Bách Khoa thân mến, với một sự quan tâm nhất định đến lĩnh vực kinh doanh, bản thân em nhận định việc giải quyết các bài toán quản lý hàng hóa, nhân viên, doanh thu cũng như các khoản chi phí trong quản lý cửa hàng là hết sức quan trọng. Thay vì sử dụng các loại sổ sách giấy tờ một cách kém hiệu quả, hiện nay cũng đã có một vài ứng dụng đáp ứng được việc quản lý nhưng hầu như dừng lại ở mức độ quản lý một cửa hàng đơn lẻ. Với mong muốn giải quyết vấn đề quản lý một hệ thống chuỗi nhiều cửa hàng, nhận thấy hiện nay trên chưa có nhiều ứng dụng một cách toàn diện và hiệu quả cũng như mong muốn được áp dụng công nghệ WEB API để phát triển ứng dụng nên em mong muốn được xây dựng một ứng dụng quản lý một cách toàn diện, hiệu quả.

Thêm vào đó, sự bùng nổ của các thiết bị di động cũng như sự phát triển mạnh mẽ của ứng dụng desktop (như Appstore của Microsoft từ Windows 8) cũng đặt ra một vấn đề là phát triển các ứng dụng đa nền tảng, chạy trên nhiều thiết bị khác nhau để mang đến trải nghiệm và chất lượng phục vụ tốt nhất cho người dùng.

Việc sử dụng công nghệ Microsoft .NET WEB API sẽ giúp cho hệ thống có hiệu năng cao hơn, hệ thống có thể chia thành nhiều module nhỏ, linh hoạt và giảm tải tốt hơn. Ngoài ra công nghệ này còn khiến cho việc truy cập từ điện thoại và desktop cũng tương tự như việc truy cập bằng website, giúp tạo nên các hệ thống đa nền tảng một cách dễ dàng hơn, chúng ta chỉ cần tạo 1 back-end cho tất cả các thiết bị và thiết kế front-end cho mỗi kiểu thiết bị khác nhau. Ứng dụng vào hệ thống quản lý cửa hàng, nó sẽ khiến cho việc xử lý truy cập dễ dàng hơn cũng như thiết kế một hệ thống quản lý cửa hàng đa nền tảng, linh hoạt hơn.

## Mục tiêu và phạm vi đề tài

Hiện nay cũng đã một vài ứng dụng đáp ứng cho nhu cầu quản lý cửa hàng ở mức độ cơ bản, tuy nhiên, như em đã trình bày ở trên, hầu hết các hệ thống đều được xây dựng với mục tiêu quản lý cửa hàng đơn lẻ và thường có hạn chế về nền tảng sử dụng.

Trong quá trình thực tập tại công ty MISA, rất may mắn cho em được tiếp cận với các công nghệ của công ty. Một trong số đó là Microsoft .NET Web API – công nghệ mới nhất của Microsoft cho việc thiết kế hệ thống HTTP Service. Từ đó, em nảy ra ý tưởng “Tại sao không mang công nghệ này thử giải quyết vấn đề thực tế trong cuộc sống xem sao”.

Việc sử dụng kiến trúc và công nghệ mà MISA đã áp dụng, tận dụng kinh nghiệm từ những người đi trước sẽ giúp tạo nên một hệ thống có hiệu năng tốt hơn, linh hoạt hơn.

Tuy nhiên, với khả năng và thời gian của mình, em không thể thực hiện được một hệ thống hoàn chỉnh, vậy nên em thực hiện một phần của hệ thống, mục tiêu là đáp ứng các yêu cầu về nghiệp vụ và áp dụng công nghệ xây dựng hệ thống hoạt động trên nền tảng web – desktop.

## Định hướng giải pháp

Bằng những tìm hiểu đã kể trên, em quyết định xây dựng hệ thống quản lý cửa hàng, về mặt nghiệp vụ vẫn tham khảo theo nghiệp vụ của ứng dụng quản lý cửa hàng của công ty MISA mà cụ thể là sản phẩm MshopKeeper, hệ thống vẫn chạy trên nền tảng web như hiện tại vì đây là nền tảng thân thuộc và dễ sử dụng cũng như đảm bảo thực hiện đầy đủ các chức năng nghiệp vụ.

Về front-end, Angular là framework được em sử dụng, cụ thể hơn hệ thống sử dụng các công nghệ thông dụng và phổ biến hiện nay như html, css, javascript, ngoài ra sử dụng them các thư viện hỗ trợ khác như Jquery, Jquery UI để giúp việc hoàn thiện giao diện dễ dàng hơn và đơn giản hơn cho việc viết code.

Hệ thống back-end được xây dựng trên nền tảng chính là framework .NET API của Microsoft, sử dụng kiến trúc REST (RESTful API) trong xây dựng API, Enity Framework cho việc đóng gói thực thể đối tượng, viết trên ngôn ngữ C#.

Kết quả đạt được là 1 hệ thống quản lý cửa hàng đơn giản gồm các module chính: bán hàng, quản lý thu, chi, xuất, nhập kho, báo cáo và tính lương Bao gồm các chức năng xem, thêm, sửa, xóa, tìm kiếm cho mỗi module.

## Bố cục đồ án

Chương 1 của đồ án giới thiệu về đề tài. Tại sao em lại chọn đề tài này, nguyên nhân, thực trạng ra sao và hướng giải pháp nào em đưa ra để giải quyết các vấn đề đấy.

Chương 2 là các khảo sát của em về đề tài này, cùng với đó là phân tích các yêu cầu mà đề tài đặt ra như các use-case, quy trình nghiệp vụ, các luồng chính của sự kiện, các yêu cầu chức năng và phi chức năng.

Chương 3 là các công nghệ em sử dụng trong chương trình, bao gồm các công nghệ về front-end (tương tác với người dùng) và back-end (tương tác trong hệ thống).

Chương 4 nói về thiết kế và triển khai chương trình. Trong chương này em sẽ nói về kiến trúc của chương trình, trình bày cụ thể về các tầng của kiến trúc, cùng với đó là các thiết kế giao diện, CSDL, và việc triển khai chương trình.

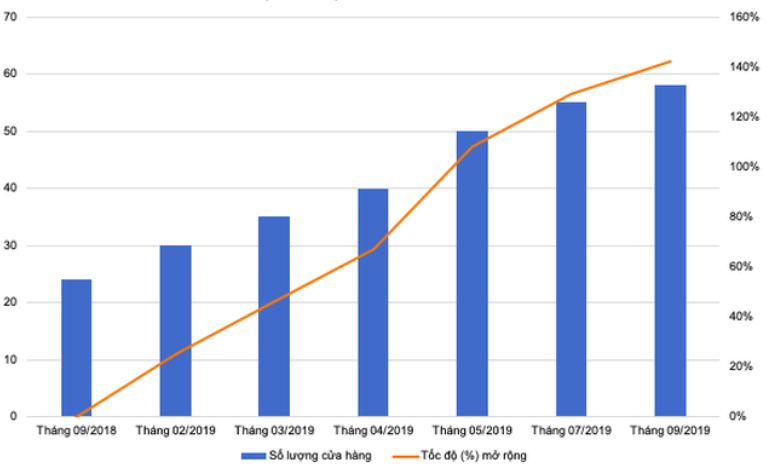
Chương 5 là đóng góp và giải pháp, em sẽ nêu cụ thể tác dụng của các công nghệ mà em sử dụng được áp dụng cho vấn đề nào, giải quyết được những gì, những khó khăn em gặp phải trong quá trính làm đồ án và cách khắc phục.

Chương 6 là đánh giá của em về kết quả đạt được sau quá trình làm đồ án, ưu điểm và nhược điểm cũng như phương hướng phát triển trong tương lai.

# Khảo sát và phân tích yêu cầu

## Khảo sát hiện trạng

**Biểu đồ tốc độ gia tăng mở mới cửa hàng, chuỗi cửa hàng trên địa bàn thành phố hà nội.**



Hình 1: Biểu đồ tốc độ gia tăng mở mới cửa hàng, chuỗi cửa hàng quần áo trên địa bàn thành phố hà nội.Nguồn: <https://dantri.com.vn/>

**Biểu đồ mức độ đáp ứng yêu cầu của các phần mềm đối với đối tượng là quản lý cửa hàng.**

.

Hình 2 Biểu đồ thể hiện mức độ đáp ứng yêu cầu của các sản phẩm phần mềm quản lý với đối tượng là quản lý cửa hàng. Nguồn: <https://www.kiotviet.vn/>

Qua quá trình tìm hiểu em nhận thấy sự phát triển của nền kinh tế và sự gia tăng đời sống của người dân luôn đi kèm với những nhu cầu mua bán hàng hóa, nhu cầu về việc mua sắm, cải thiện đời sống. Điều này đã kéo theo việc phát triển mạnh số lượng các cửa hàng ăn uống, quần áo.

Trên cơ sở đó, em thấy việc phát triển một hệ thống mới cho việc quản lý cửa hàng là cần thiết. Hệ thống cần cung cấp các chức năng cho việc bán hàng, quản lý các khoản doanh thu chi phí, kiểm kê, báo cáo và giải quyết được một số vấn đề về hiệu năng.

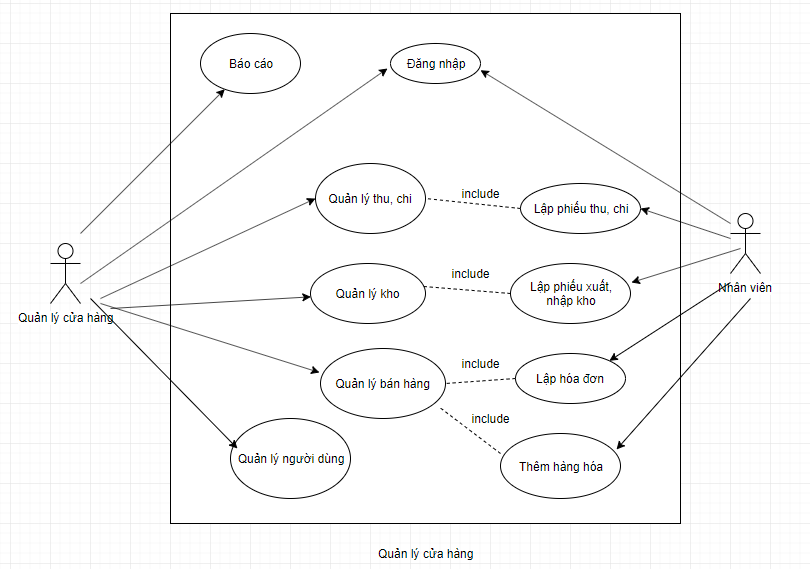
## Tổng quan chức năng

Trong phần này, em xin được phác hoạ các chức năng tổng quan của hệ thống quản lý cửa hàng thông qua việc xây dựng các biểu đồ trường hợp sử dụng (use-case diagrams).

Hệ thống có 2 tác nhân chính là:

* Quản lý
* Nhân viên

### Biểu đồ use case tổng quan



Hình 4 Biểu đồ use case tổng quan

Hệ thống có các use-case chính: Quản lý thu chi, Quản lý kho, Quản lý bán hàng, Báo cáo, kiểm kê đối với đối tượng quản lý và Chấm công, tra cứu lương đối với đối tượng nhân viên. Ngoài ra còn các use-case phụ khác như đăng nhập, nhà cung cấp, khách hàng, nhân viên...

Vì là hệ thống nghiệp vụ nội bộ nên sẽ không có đăng ký tài khoản, toàn bộ việc quản lý tài khoản, phân quyền… được thực hiện bởi người Quản lý thông qua use-case Quản lý người dùng.

Bảng 1 Bảng mô tả quyền hạn của Quản lý trong các use-case

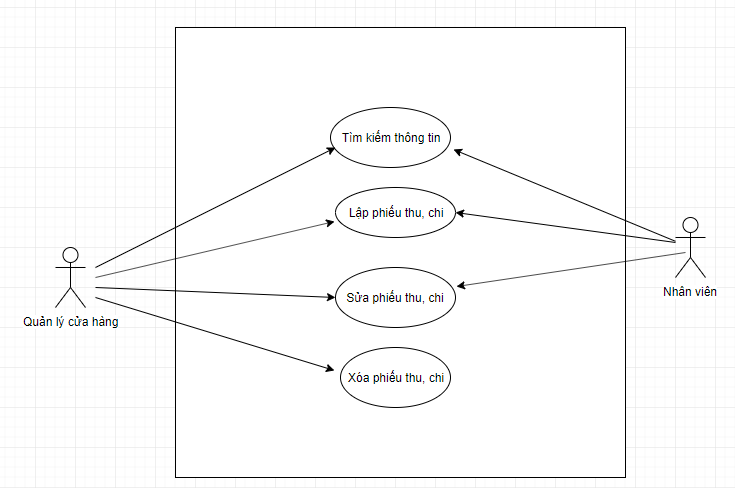
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tìm kiếm | Thêm | Sửa | Xóa |
| Module Quản lý thu chi | x | x | x | x |
| Module Quản lý kho | x | x | x | x |
| Module Quản lý bán hàng | x | x | x | x |
| Báo cáo | x | x | x | x |

Bảng 2 Bảng mô tả quyền hạn của Nhân viên trong các use-case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tìm kiếm | Thêm | Sửa | Xóa |
| Module Quản lý thu chi |  | x | x |  |
| Module Quản lý kho |  | x | x |  |
| Module Quản lý bán hàng |  | x | x |  |
| Tra cứu lương | x |  |  |  |

### Biểu đồ use case phân rã

1. Biểu đồ use case phân rã Quản lý thu, chi



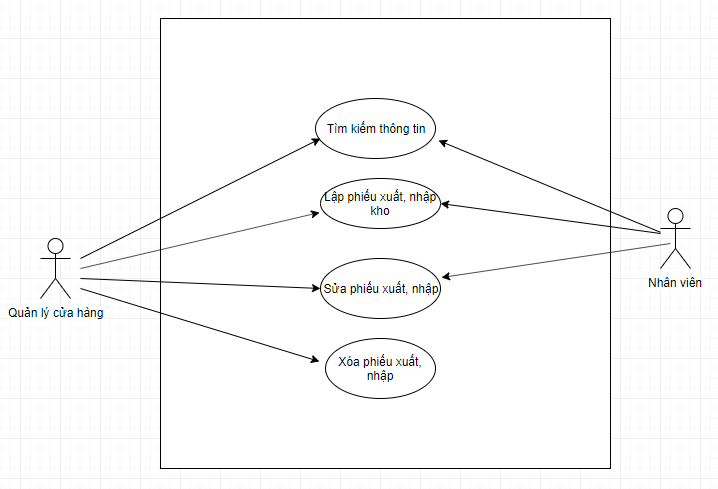
Hình 5 Biểu đồ use case phân rã quản lý thu, chi

Bao gồm 2 tác nhân chính: quản lý, nhân viên

4 use case chính: thêm, tìm kiếm, sửa, xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Use case phân rã cho Quản lý kho



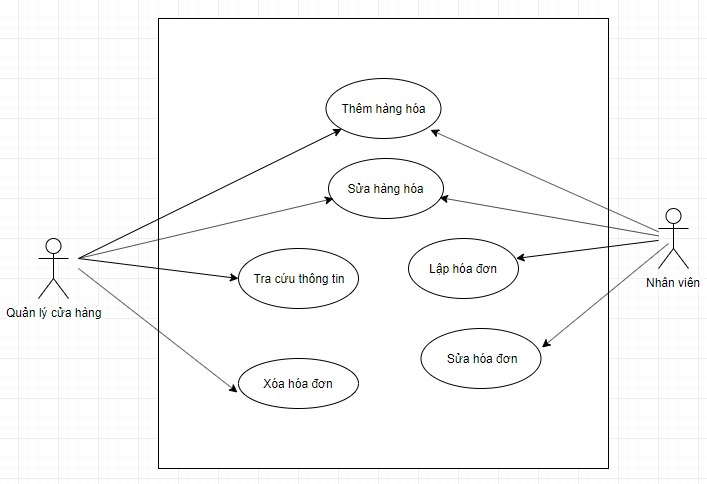
Hình 6 Biểu đồ use case phân rã cho quản lý kho

Bao gồm 2 tác nhân chính: quản lý, nhân viên

4 use case chính: thêm, tìm kiếm, sửa, xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Use case phân rã cho Quản lý bán hàng



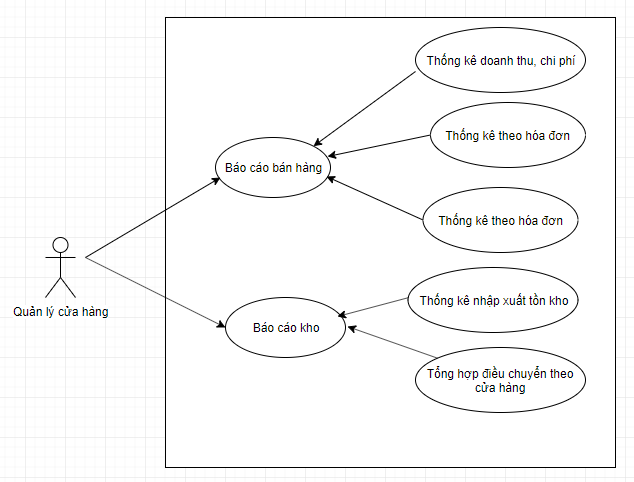
Hình 7 Biểu đồ use case phân rã cho quản lý bán hàng

Bao gồm 2 tác nhân chính: quản lý, nhân viên

4 use case chính: thêm, tìm kiếm, sửa ,xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Usecase phân rã cho Module Kiểm kê, Báo cáo

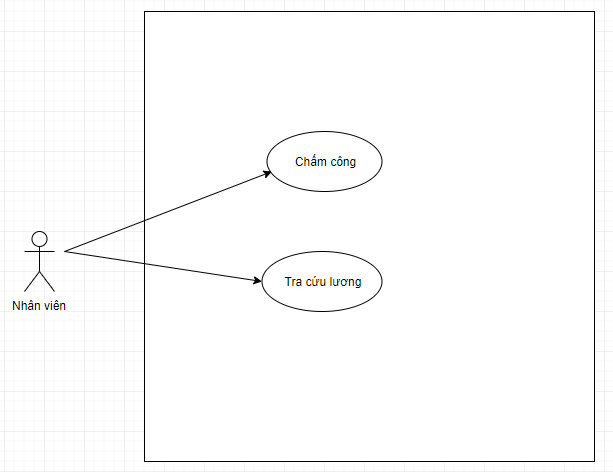


Bao gồm 1 tác nhân chính: quản lý.

2 use case chính: xem báo cáo bán hàng và báo cáo kho

Tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Usecase phân rã cho Module Chấm công, Tra cứu lương



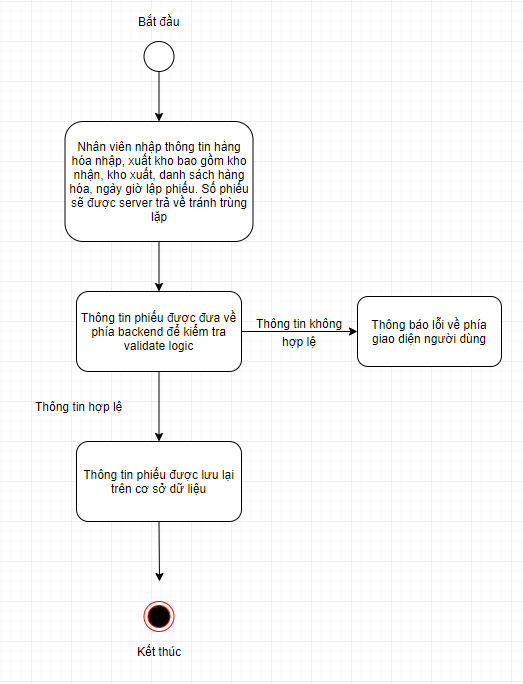
Bao gồm 1 tác nhân chính là nhân viên.

2 use case chính: chấm công và Tra cứu lương

Tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

### Quy trình nghiệp vụ

##### Quy trình nghiệp vụ thêm phiếu xuất, nhập kho:



Hình 8 Biểu đồ quy trình nghiệp vụ thêm/sửa phiếu xuất, nhập kho

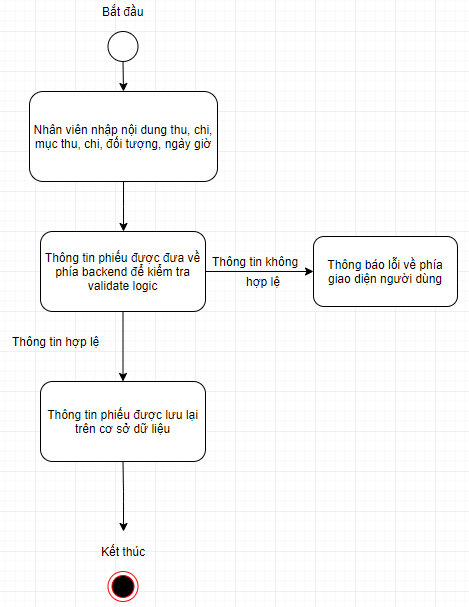
Mô tả quy trình:

Bước 1: Nhân viên thực hiện tạo mới hoặc sửa phiếu xuất, nhập kho đã có, nhập đầy đủ các trường thông tin

Bước 2: Dữ liệu được đẩy về phía Server để kiểm tra thông tin phiếu có hợp lệ

Bước 3: Dữ liệu không hợp lệ sẽ được gửi thông báo về phía giao diện người dùng, với dữ liệu hợp lệ sẽ được lưu lại vào cơ sở dữ liệu.

##### Quy trình nghiệp vụ thêm mới phiếu thu, chi



Hình 9 Biểu đồ quy trình nghiệp vụ thêm mới phiếu thu/chi

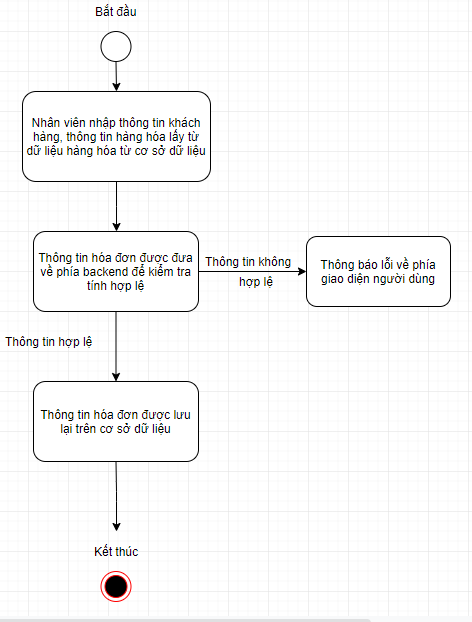
Mô tả quy trình:

Bước 1: Nhân viên lập phiếu thu/chi nhập đầy đủ nội dung các trường thông tin

Bước 2: Dữ liệu được đẩy về phía Server để kiểm tra thông tin phiếu có hợp lệ

Bước 3: Dữ liệu không hợp lệ sẽ được gửi thông báo về phía giao diện người dùng, với dữ liệu hợp lệ sẽ được lưu lại vào cơ sở dữ liệu.

##### Quy trình nghiệp vụ bán hàng



Hình 10 Biểu đồ quy trình nghiệp vụ bán hàng

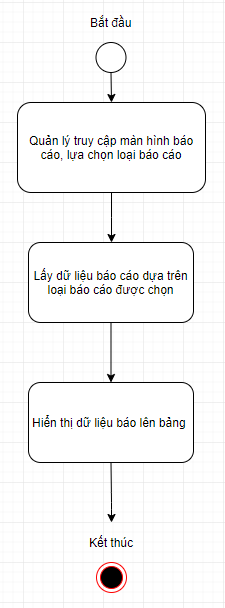
Mô tả quy trình:

Bước 1: Nhân viên lập hóa đơn nhập đầy đủ nội dung các trường thông tin

Bước 2: Dữ liệu được đẩy về phía Server để kiểm tra thông tin hóa đơn có hợp lệ

Bước 3: Dữ liệu không hợp lệ sẽ được gửi thông báo về phía giao diện người dùng, với dữ liệu hợp lệ sẽ được lưu lại vào cơ sở dữ liệu.

##### Quy trình xem báo cáo



Hình 11 Biểu đồ quy trình xem báo cáo

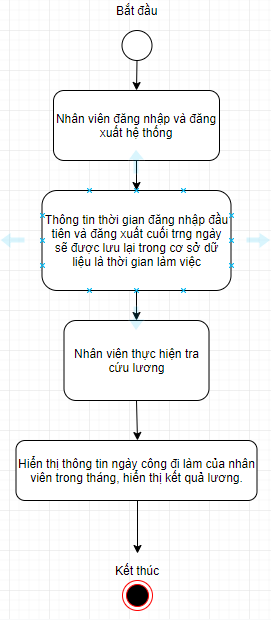
Mô tả quy trình:

Bước 1: Nhân viên lập hóa đơn nhập đầy đủ nội dung các trường thông tin

Bước 2: Dữ liệu được đẩy về phía Server để kiểm thông tin hóa đơn có hợp lệ

Bước 3: Dữ liệu không hợp lệ sẽ được gửi thông báo về phía giao diện người dùng, với dữ liệu hợp lệ sẽ được lưu lại vào cơ sở dữ liệu.

##### Quy trình chấm công, tra cứu lương



Hình 12 Biểu đồ quy trình chấm công, tra cứu lương

Mô tả quy trình:

Bước 1: Nhân viên thực hiện đăng nhập và đăng xuất trên hệ thống vào hệ thống.

Bước 2: Dữ liệu lần đăng nhập đầu tiên và cuối cùng của nhân viên được lưu lại trong cơ sở dữ liệu và lấy là thông tin giờ làm việc của nhân viên

Bước 3: Nhân viên chọn tab tra cứu lương, thông tin ngày công đi làm của nhân viên sẽ được tổng hợp và hiển thị kết quả lương tương ứng.

## Đặc tả chức năng

### Đặc tả use case Thêm phiếu xuất, nhập kho

##### Mô tả

Nhân viên thực hiện

##### Luồng sự kiện và phát sinh

Bảng 4 Đặc tả use case thêm sinh viên

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCTPK | **Tên UseCase** | Thêm phiếu xuất nhập kho |
| **Tác nhân** | Người dùng (Quản lý, Nhân viên) | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Người dùng | Vào tab Kho chọn Nhập/Xuất | |
| 2 | Client | Hiển thị giao diện danh sách phiếu xuất/nhập kho | |
| 3 | Người dùng | Chọn chức năng thêm phiếu xuất/nhập kho | |
| 4 | Client | Hiển thị popup thêm mới phiếu xuất nhập kho | |
| 5 | Người dùng | Nhập thông tin phiếu | |
| 6 | Client | Kiểm tra thông tin có hợp lệ hay không | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị kết quả truy vấn cho Nhân viên | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Hiển thị cảnh báo nếu thông tin nhập vào không hợp lệ | |
| 6b | Client | Hiển thị thông báo nếu thêm không thành công | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của phiếu xuất/nhập kho

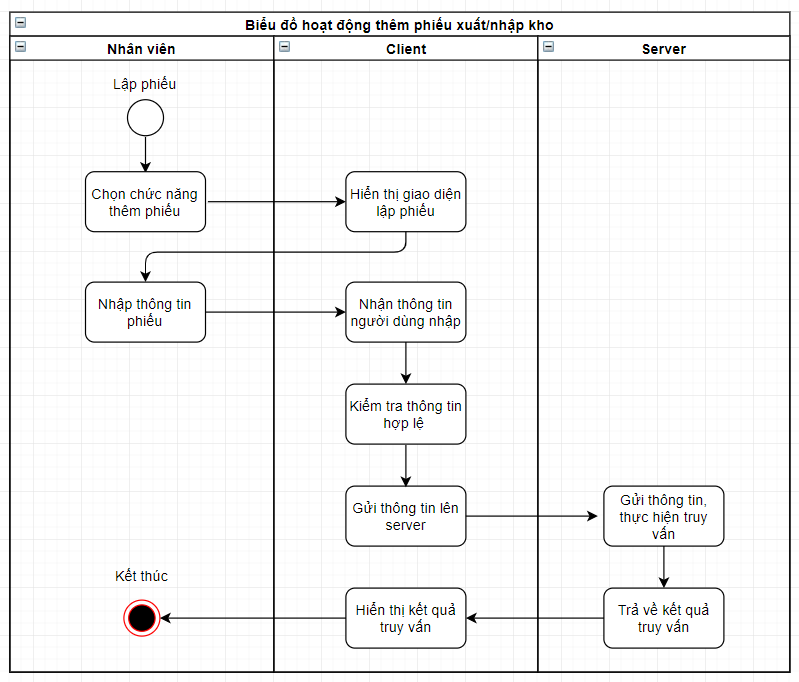
Bảng 5 Mô tả dữ liệu đầu vào Thêm phiếu xuất nhập kho

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã nhân viên |  | Có | Là 1 dãy số có Mã phiếu | XK0000364 |
| 2 | Ngày lập phiếu |  | Có | Ngày | 02/10/2019 |
| 3 | Giờ lập phiếu |  | Có | Giờ | 11:00 |
| 4 | Người lập phiếu |  | Có | Tên người | Nguyễn Mạnh Duy |
| 5 | Người nhập/xuất |  | Có | Tên người | Nguyễn Văn A |
| 6 | Mã hàng hóa | Thông tin chi tiết phiếu | Có | Dãy số với Mã ở đầu | AK-045023 |
| 7 | Tên hàng hóa | Thông tin chi tiết phiếu | Có | Là 1 chuỗi | Áo khoác Kaki |
| 8 | Tên kho xuất/nhập | Thông tin chi tiết phiếu | Có | Là 1 chuỗi | Chi nhánh Giải Phóng |
| 9 | Tên kho nhập/xuất | Thông tin chi tiết phiếu | Có | Là 1 chuỗi | Chi nhánh Cầu Giấy |
| 10 | Đơn vị tính | Thông tin chi tiết phiếu | Có | Là 1 chuỗi | Chiếc |
| 11 | Đơn giá | Thông tin chi tiết phiếu | Có | Dạng tiền | 340.000 |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động thêm phiếu xuất/nhập kho



Hình 11 Biểu đồ hoạt động thêm phiếu xuất/nhập kho

### Đặc tả use case Thêm phiếu thu/chi

##### Mô tả

Thực hiện công việc thêm phiếu thu/chi vào hệ thống

##### Luồng sự kiện và phát sinh

Bảng 6 Đặc tả use case Thêm phiếu thu/chi

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCTTC | **Tên UseCase** | Thêm phiếu thu/chi |
| **Tác nhân** | Người dùng (quản lý, nhân viên) | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Người dùng | Vào trang quản lý cửa hàng -> Chọn mục Thu/Chi | |
| 2 | Client | Hiển thị giao diện quản lý thu chi | |
| 3 | Người dùng | Nhấn thêm mới phiếu | |
| 4 | Client | Hiển thị form thêm mới | |
| 5 | Người dung | Nhập đầy đủ các trường thông tin | |
| 6 | Client | Gửi thông tin lên server | |
| 7 | Server | Nhận thông tin, thực hiện truy vấn | |
| 8 | Server | Trả về kết quả truy vấn cho client | |
| 9 | Client | Hiển thị kết quả truy vấn cho người dùng | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 4a | Client | Đóng gói dữ liệu | |
| 5a | Client | Hiển thị icon Loading | |
| 5b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 8a | Client | Hiển thị thông báo nếu không lấy được dữ liệu | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của thông tin phiếu thu/chi

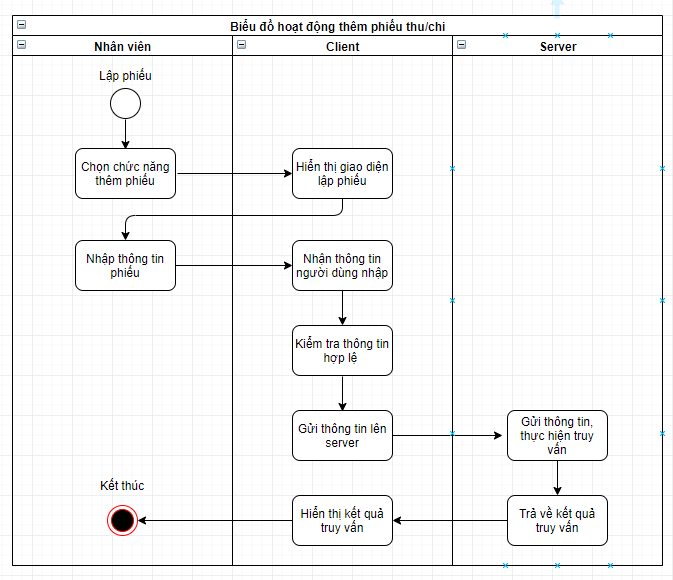
Bảng 7 Mô tả dữ liệu đầu vào phiếu thu/chi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Số phiếu | Có | Là 1 dãy số kèm Mã phiếu | PT00423 |
| 2 | Ngày thu/chi | Có | Dạng ngày tháng | 02/10/2019 |
| 3 | Đối tượng nộp | Có | Tên người | Nguyễn Mạnh Duy |
| 4 | Lý do thu/chi | Có | Chuỗi | Thu tiền nợ |
| 5 | Nhân viên thu | Có | Tên người | Nguyễn Văn A |
| 6 | Diễn giải | Có | Là 1 chuỗi | Thu tiền nợ |
| 7 | Số tiền | Có | Định dạng tiền | 234.000 |
| 8 | Mục thu | Có | Là 1 chuỗi | Thu tiền gửi |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động thêm phiếu thu/chi



Hình 12 Biểu đồ hoạt động thêm phiếu thu/chi

### Đặc tả use case Sửa phiếu thu/chi

##### Mô tả

Thực hiện công việc sửa thông tin phiếu thu/chi trong hệ thống

##### Luồng sự kiện và phát sinh

Bảng 8 Đặc tả use case Sửa thông tin phiếu

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCSPTC | **Tên UseCase** | Sửa thông tin phiếu thu/chi |
| **Tác nhân** | Người dung (Quản lý, Nhân viên) | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Người dùng | Vào mục quản lý phiếu thu/chi | |
| 2 | Client | Hiển thị giao diện quản lý phiếu thu/chi | |
| 3 | Người dùng | Chọn giao diện Sửa thông tin | |
| 4 | Client | Hiển thị giao diện Sửa thông tin | |
| 5 | Người dùng | Nhập các thông tin cần sửa | |
| 6 | Client | Lấy nội dung các trường cần sửa | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị kết quả truy vấn cho Người dùng | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị loading | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của thông tin cá nhân

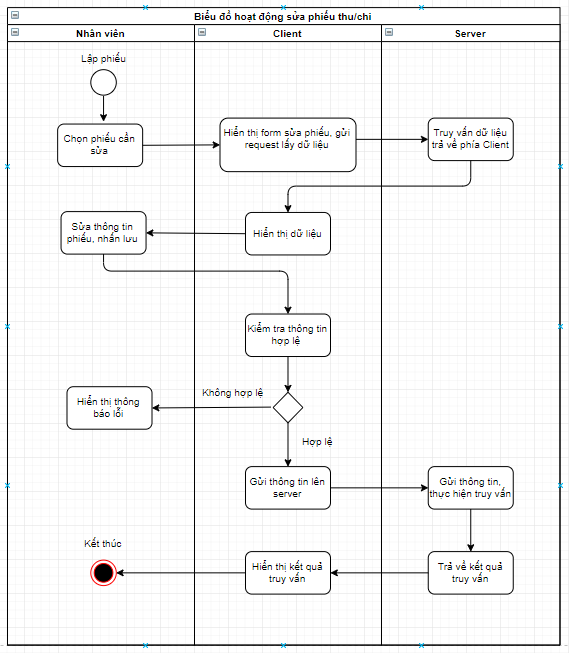
Bảng 9 Mô tả dữ liệu đầu vào Sửa phiếu thu/chi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Số phiếu |  | Có | Là 1 dãy số kèm Mã phiếu | PT00423 |
| 2 | Ngày thu/chi |  | Có | Dạng ngày tháng | 02/10/2019 |
| 3 | Đối tượng nộp |  | Có | Tên người | Nguyễn Mạnh Duy |
| 4 | Lý do thu/chi |  | Có | Chuỗi | Thu tiền nợ |
| 5 | Nhân viên thu |  | Có | Tên người | Nguyễn Văn A |
| 6 | Diễn giải | Chi tiết phiếu | Có | Là 1 chuỗi | Thu tiền nợ |
| 7 | Số tiền | Chi tiết phiếu | Có | Định dạng tiền | 234.000 |
| 8 | Mục thu | Lựa chọn | Có | Là 1 chuỗi | Thu tiền gửi |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động sửa thông tin sinh viên



Hình 13 Biểu đồ hoạt động sửa phiếu thu/chi

### Đặc tả use case Tìm kiếm, lọc dữ liệu phiếu xuất/nhập kho, thu/chi

##### Mô tả

Nhân viên thực hiện tìm kiếm thông tin phiếu xuất/nhập kho hoặc thu/chi

Use case tìm kiếm lọc dữ liệu phiếu xuất/nhập kho và thu/chi có luồng tương tự nhau nên em xin phép thể hiện chung trong phần này.

Người dùng (Quản lý, Nhân viên) thực hiện nhập thông tin và chọn điều kiện tìm kiếm dữ liệu.

##### Luồng sự kiện và phát sinh

Bảng 10 Đặc tả use case Tìm kiếm phiếu xuất/nhập kho, thu/chi

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCTKP | **Tên UseCase** | Tìm phiếu |
| **Tác nhân** | Người dùng (Quản lý/ Nhân viên) | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Người dùng | Vào trang quản lý phiếu | |
| 2 | Client | Hiển thị giao diện trang quản lý | |
| 5 | Người dùng | Nhập các thông tin phiếu cần tìm kèm điều kiện tìm kiếm | |
| 6 | Client | Lấy nội dung các trường cần tìm kèm điều kiện | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị danh sách học phần | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị loading | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của phiếu

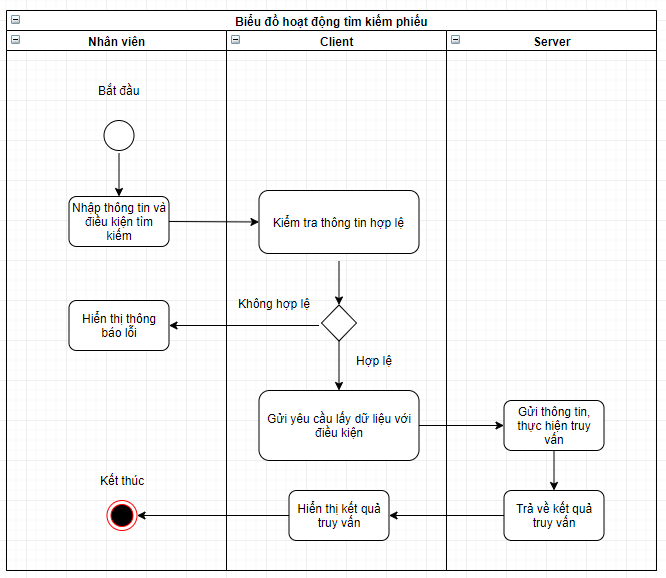
Bảng 11 Mô tả dữ liệu Phiếu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Số phiếu |  | Không | Là 1 chuỗi | PT000324 |
| 2 | Ngày xuất phiếu |  | Không | Ngày tháng | 02/10/2019 |
| 3 | Người xuất phiếu |  | Không | Tên người | Nguyễn Mạnh Duy |
| 4 | Tổng tiền |  | Không | Tiền | 340.000 |
| 5 | Loại chứng từ |  | Không | Là 1 chuỗi | Phiếu thu nợ |
| 6 | Mô tả |  | Không | Là 1 chuỗi | Mua quạt |
| 8 | Điều kiện tìm kiếm | Mỗi trường có 1 điều kiện tìm kiếm | Có | Kiểu enum | ‘\* ứng với chứa  = ứng với bằng  + ứng với bắt đầu bằng  ‘- ứng với kết thúc bằng  ! ứng với không chứa |

##### Yêu cầu chức năng

Đây là một trong những phần dễ gây tắc nghẽn nhất hệ thống, cần thực hiện với hiệu năng tốt, tốc độ đạt yêu cầu, đảm bảo chính xác và an toàn cho hệ thống

##### Mô tả biểu đồ hoạt động tìm kiếm phiếu



Hình 14 Biểu đồ hoạt động tìm kiếm phiếu

### Đặc tả use case Báo cáo, Kiểm kê

##### Mô tả

Quản lý Thực hiện xem báo cáo kho, báo cáo thu/chi

##### Luồng sự kiện và phát sinh

Bảng 12 Đặc tả use case xem báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCBC | **Tên UseCase** | Báo cáo |
| **Tác nhân** | Quản lý | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Quản lý | Vào mục báo cáo | |
| 2 | Client | Hiển thị giao diện báo cáo | |
| 3 | Quản lý | Chọn loại báo cáo muốn xem | |
| 4 | Client | Gửi yêu cầu lên server | |
| 5 | Client | Kiểm tra dữ liệu có hợp lệ | |
| 6 | Server | Kiểm tra hợp lệ của dữ liệu | |
| 7 | Server | Nhận thông tin, thực hiện truy vấn | |
| 8 | Server | Trả về kết quả truy vấn cho client | |
| 9 | Client | Hiển thị kết quả báo cáo về phía client | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị loading | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của báo cáo

Bảng 13 Mô tả dữ liệu báo cáo

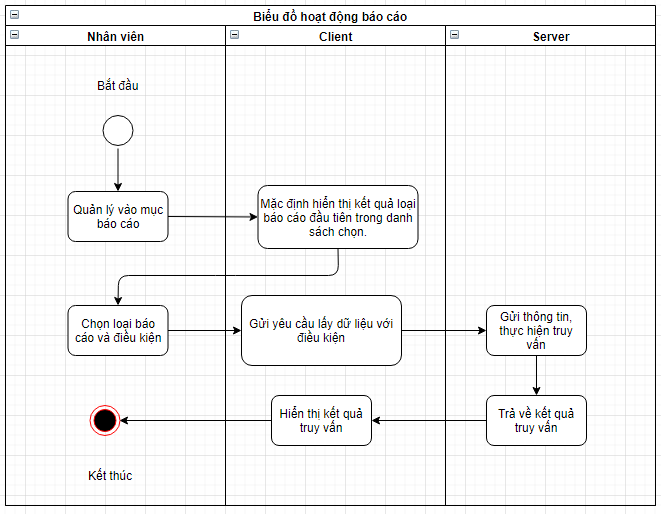
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Tên báo cáo |  | Có | Là 1 chuỗi | Bảng kê hóa đơn |
| 2 | Kỳ báo cáo |  | Có | Là 1 chuỗi | Hôm nay |
| 3 | Từ ngày |  | Có | Ngày tháng | 02/10/2019 |
| 4 | Đến ngày |  | Có | Ngày tháng | 02/10/2019 |
| 5 | Nhân viên |  | Không | Tên người | Nguyễn Văn A |
| 6 | Khách hàng |  | Không | Tên người | Nguyễn Văn A |
| 7 | Thống kê theo |  | Không | Là 1 chuỗi | Mẫu mã |
| 8 | Đơn vị tính |  | Không | Là 1 chuỗi | Chiếc |
| 9 | Kho |  | Không | Là 1 chuỗi | Chi nhánh Giải Phóng |
| 10 | Cửa hàng xuất |  | Không | Là 1 chuỗi | Chi nhánh Cầu Giấy |
| 11 | Cửa hàng nhận |  | Không | Là 1 chuỗi | Chi nhánh Bưởi |

##### Yêu cầu chức năng

Dữ liệu trả về đúng với loại báo cáo kèm theo các điều kiện.

Hệ thống hoạt động chính xác, bảo đảm toàn vẹn dữ liệu và hiệu năng.

##### Mô tả biểu đồ hoạt động báo cáo



Hình 15 Biểu đồ hoạt động báo cáo

### Đặc tả use case chấm công và tra cứu lương

##### Mô tả

Nhân viên thực hiện đăng nhập và đăng xuất trên hệ thống để lưu lại ngày công, vào mục tra cứu lương để tra cứu lương

##### Luồng sự kiện và phát sinh

Bảng 14 Đặc tả use case chấm công và tra cứu lương

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCCC | **Tên UseCase** | Chấm công, tra cứu lương |
| **Tác nhân** | Nhân viên | | |
| **Điều kiện trước** |  | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Nhân viên | Nhân viên vào trang quản lý cửa hàng | |
| 2 | Client | Hiển thị trang đăng nhập | |
| 3 | Nhân viên | Nhân viên thực hiện đăng nhập vào hệ thống | |
| 4 | Client | Gửi thông tin lên server | |
| 5 | Server | Kiểm tra tính hợp lệ trong thông tin đăng nhập của client. | |
| 6 | Server | Trả về lỗi nếu thông tin không hợp lệ, lưu lại thông tin nếu hợp lệ. | |
| 7 | Nhân viên | Vào mục Tra cứu lương | |
| 8 | Client | Gửi thông tin người dùng lên server lấy dữ liệu. | |
| 9 | Server | Nhận thông tin, thực hiện truy vấn | |
| 10 | Server | Trả về kết quả truy vấn cho client | |
| 11 | Client | Hiển thị ngày công trong tháng, bảng lương | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 4a | Client | Đóng gói dữ liệu | |
| 4b | Client | Hiển thị icon Loading | |
| 8b | Client | Gửi thông tin người dùng lên server | |
| 9a | Server | Kiểm tra tính hợp lệ của thông tin | |
| 10b | Server | Trả về lỗi nếu truy vấn không thành công | |
| 10a | Client | Hiển thị lỗi nếu truy vấn không thành công | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của Chấm công, tra cứu lương

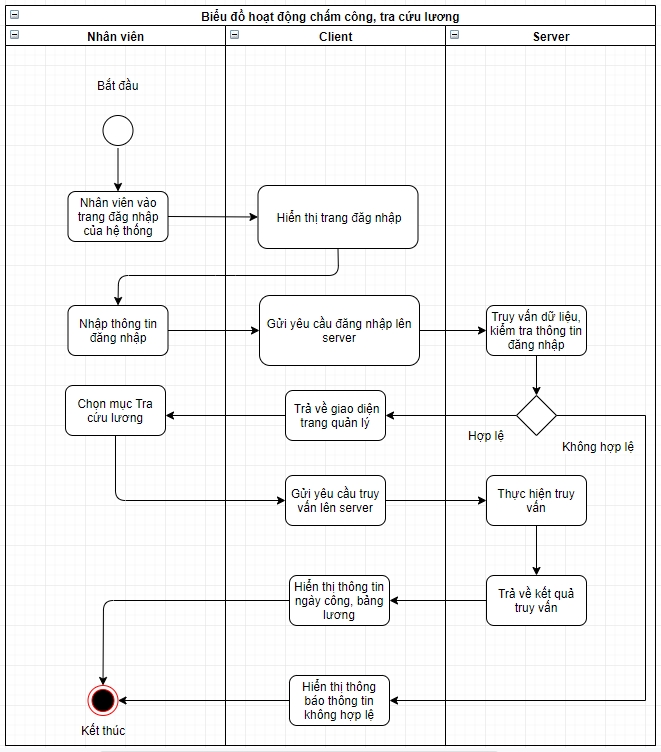
Bảng 15 Mô tả dữ liệu Chấm công

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Username |  | Có | Là 1 chuỗi | admin |
| 2 | Mật khẩu |  | Có | Là 1 chuỗi | Lovegirl1 |

##### Yêu cầu chức năng

Hệ thống xử lý nhanh, kịp thời, chính xác.

##### Mô tả biểu đồ hoạt động chấm công, tra cứu lương



Hình 16 Biểu đồ hoạt động chấm công, tra cứu lương

## Yêu cầu phi chức năng

Hệ thống được xây dựng cần phải đáp ứng được các yêu cầu sau:

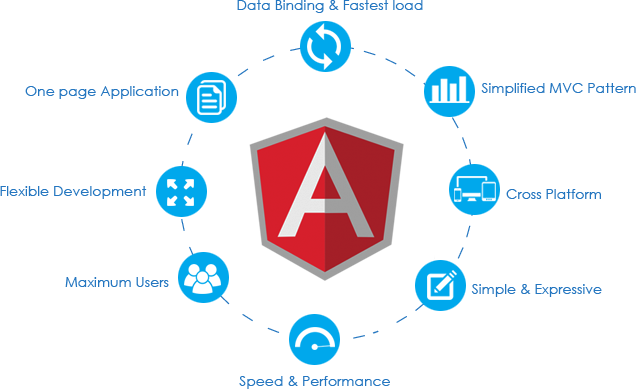
Bảng 16 Danh sách yêu cầu phi chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Yêu cầu | Phân loại yêu cầu | Cụ thể |
| 1 | Hiệu năng hoạt động | Yêu cầu về thời gian | Thời gian truy cập và gửi dữ liệu tới người sử dụng hệ thống cần phải diễn ra một cách nhanh chóng.  Đỗ trễ của hệ thống không quá 5s  Khi trễ quá 5s, hiển thị thông báo lỗi |
|  |  | Tài nguyên sử dụng | Hệ thống sẽ sử dụng các phần cứng khác nhau khi được khởi chạy trên các thiết bị khác nhau.  Có thể triển khai được trên cả Web, Desktop và Moblie |
| 2 | Tương thích | Cùng tồn tại | Hệ thống khi được khởi chạy trên nhiều thiết bị khác nhau cần phải đảm bảo tất cả đều khởi chạy bình thường, không quan tâm tới ứng dụng được khởi chạy trên môi trường nào |
|  |  | Tương tác liên thông | Hệ thống cần đảm bảo các thông tin được cập nhật liên tục lên cơ sở dữ liệu chung của ứng dụng. Tất cả các dữ liệu được gửi tới các thiết bị có trong hệ thống cần phải được đồng nhất. |
| 3 | Tính khả dụng | Phù hợp nhu cầu | Hệ thống được xây dựng lên cần phải đảm bảo theo sát yêu cầu đề bài |
|  |  | Tính dễ sử dụng | Hệ thống cần phải được thiết kế với giao diện đơn giản, giúp người dùng thực hiện ít thao tác nhất khi thực hiện điều hướng đến một nội dung nào đó có trong hệ thống |
|  |  | Khả năng truy cập | Hệ thống cần phải hỗ trợ được người dùng có thể thực hiện đăng nhập tại bất cứ đâu, trên cả 3 nền tảng là Web, Desktop và Android |
| 4 | Tính tin cậy | Trưởng thành | Hệ thống được thiết kế sao cho việc nâng cấp, phát triển các tính năng của hệ thống sẽ được thực hiện một cách dễ dàng nhất |
|  |  | Khả năng chịu lỗi | Khi xảy ra một số vấn đề không mong muốn: số lượng các yêu cầu được gửi lên quá nhiều; có lỗi không mong muốn xảy ra, hệ thống vẫn có thể hoạt động bình thường và không bị gián đoạn |
|  |  | Khả năng phục hồi | Khi hệ thống gặp phải các lỗi nguy hiểm làm hệ thống bị tạm ngưng hoạt động, hệ thống cần phải có khả năng phục hồi và trở lại hoạt động trong thời gian sớm nhất có thể |
| 5 | An toàn thông tin | Bảo mật | Các thông tin có trong hệ thống cần được phân cấp cụ thể những người có thể truy cập tới các thông tin đó:   * Người quản lý của hệ thống có toàn quyền xem các thông tin có trong hệ thống * Nhân viên có quyền lập phiếu bán hàng, các loại phiếu thu/chi và tra cứu lương |

# Công nghệ sử dụng

## Danh sách các công nghệ sử dụng

##### Front end



Ngôn ngữ lập trình: HTML, CSS, TypeScript.

**Angular** là framework được em lựa chọn, được phát triển bởi **Google** để xây dựng các Single Page Application bằng HTML, Typescript hỗ trợ phát triển linh hoạt và giúp cải thiện hiệu năng với các hệ thống lớn có nhiều module nhỏ. Ngoài ra Angular có cấu trúc khá rõ ràng, giao diện sử dụng đơn giản. Có lực lượng hỗ trợ đông đảo, tài liệu hướng dẫn đầy đủ. Code được viết bằng TypeScript và biên dịch thành Javascript và hiển thị tương tự lên trình duyệt.

##### Back end

Ngôn ngữ lập trình: C#

Nền tảng chính**: Microsoft .NET Framework**

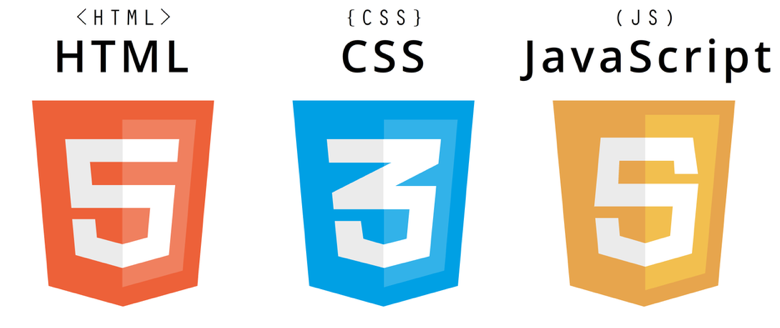
Các công nghệ sử dụng: Microsoft .NET Web API, .NET HTTP, RESTful Webservice, N-Tier Architecture.

##### Database



**SQL Server 2014**: đây là hệ quản trị cơ sở dữ liệu rất phổ biến và dễ sử dụng được phát triển bởi Microsoft. Có nhiều phiên bản phù hợp với đối tượng sử dụng khác nhau bao gồm phiên bản miễn phí được em sử dụng trong đồ án này. Việc lựa chọn phiên bản 2014 là do sự ổn định và có nhiều công cụ hỗ trợ miễn phí.

## HTML, CSS, Javascript



HTML, CSS, Javascript là 3 công nghệ cơ bản của lập trình website

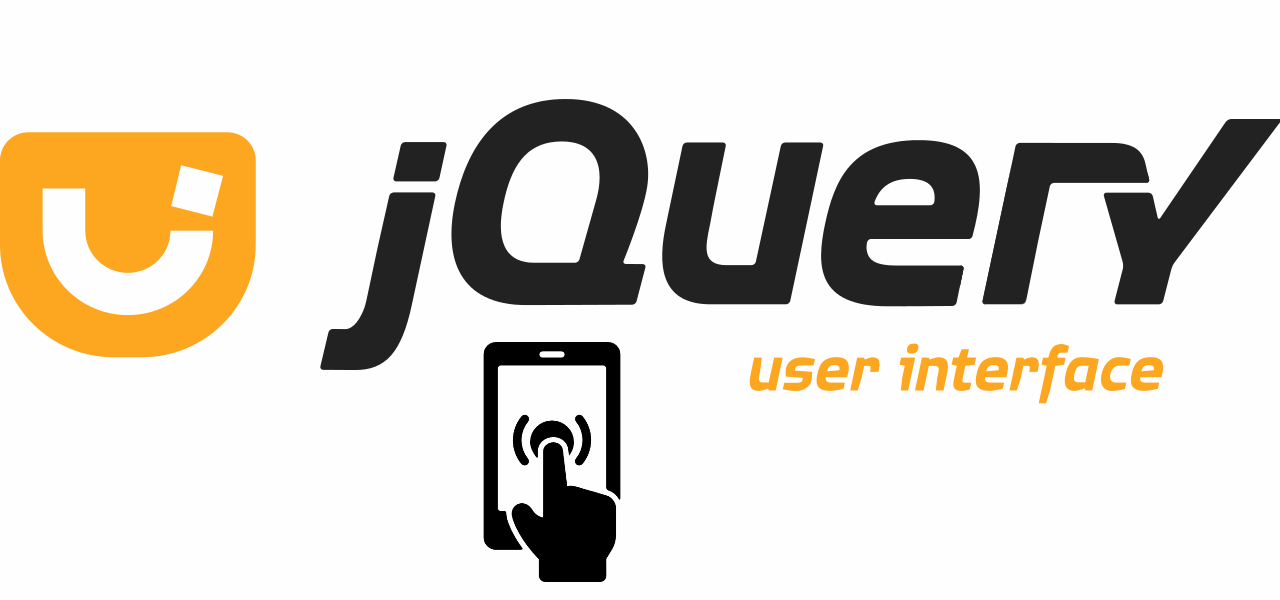
Đây là 3 công nghệ phổ biến và cơ bản nhất để xây dựng website, em sẽ không đi sâu chi tiết vào phần này. Tài liệu tham khảo có thể tìm thấy rõ ràng và chi tiết từ W3C [1].

## JQuery và jQueryUI



##### JQuery

JQuery là một [thư viện JavaScript](https://vi.wikipedia.org/w/index.php?title=Th%C6%B0_vi%E1%BB%87n_JavaScript&action=edit&redlink=1) [đa trình duyệt](https://vi.wikipedia.org/w/index.php?title=%C4%90a_tr%C3%ACnh_duy%E1%BB%87t&action=edit&redlink=1) được thiết kế để đơn giản hóa [lập trình phía máy người dùng](https://vi.wikipedia.org/w/index.php?title=L%E1%BA%ADp_tr%C3%ACnh_ph%C3%ADa_m%C3%A1y_ng%C6%B0%E1%BB%9Di_d%C3%B9ng&action=edit&redlink=1) của [HTML](https://vi.wikipedia.org/wiki/HTML), phát hành vào tháng 1 năm 2006 bởi [John Resig](https://vi.wikipedia.org/w/index.php?title=John_Resig&action=edit&redlink=1). Được sử dụng bởi hơn 52% trong 10.000 truy cập nhiều nhất các trang web, jQuery là [thư viện JavaScript](https://vi.wikipedia.org/w/index.php?title=Th%C6%B0_vi%E1%BB%87n_JavaScript&action=edit&redlink=1) phổ biến nhất được sử dụng ngày nay. [2]



##### JQueryUI

JQueryUI là một bộ các giao diện, hiệu ứng, tiện ích và chủ đề. Được xây dựng trên thư viện JQuery và JavaScript. Sử dụng để xây dựng những trang web có tính tương tác cao [3]. JqueryUI cung cấp thư viện về các tương tác và tiện ích như Draggable, Droppable, Resizeable, Dialog, Menu, Slider…

## C# và .NET Framework

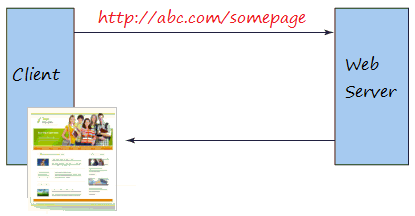


**C#** là một ngôn ngữ lập trình đơn giản, hiện đại, mục đích tổng quát, hướng đối tượng được phát triển bởi Microsoft và được phê chuẩn bởi European Computer Manufacturers Association (ECMA) và International Standards Organization (ISO). Cấu trúc C# khá gần với các ngôn ngữ high-level truyền thống, C và C++, và là một ngôn ngữ lập trình hướng đối tượng [4].

**.NET Framework** là một nền tảng lập trình và cũng là một nền tảng thực thi ứng dụng chủ yếu trên hệ điều hành Microsoft Windows, được phát triển bởi [Microsoft](https://vi.wikipedia.org/wiki/Microsoft). NET framework bao gồm tập các thư viện lập trình lớn, và những thư viện này hỗ trợ việc xây dựng các chương trình phần mềm như lập trình giao diện; truy cập, kết nối [cơ sở dữ liệu](https://vi.wikipedia.org/wiki/C%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u); [ứng dụng web](https://vi.wikipedia.org/wiki/%E1%BB%A8ng_d%E1%BB%A5ng_web); các [giải thuật](https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n), [cấu trúc dữ liệu](https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u); giao tiếp mạng... [5]

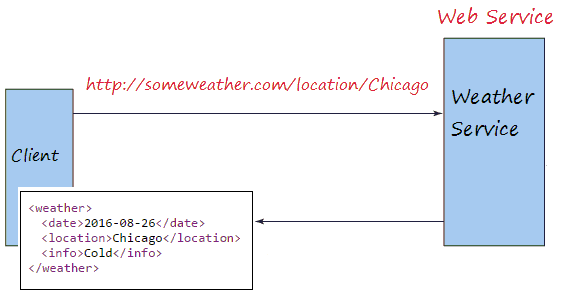
## RESTful Web Service

##### Web service



Hình 17 Web server

Với các web server thông thường, khi chúng ta truy cập server sẽ trả về cho ta nội dung của 1 trang web dưới dạng html, css và javascript.



Hình 18 Web service

Trong khi đó, Web service là 1 dịch vụ Web, khi chúng ta truy cập vào Web service, nó sẽ chỉ trả về cho ta dữ liệu thô, thường dưới dạng JSON hoặc XML.

Ví dụ như trong ảnh trên, khi chúng ta gửi yêu cầu thời tiết Chicago vào Webservice thông tin thời tiết, nó sẽ trả về cho chúng ta dữ liệu là thời tiết của Chicago thay vì cả trang web.

##### RESTful Service [6]

RESTful Web Service là các Web Service được viết dựa trên kiến trúc REST. REST định nghĩa các quy tắc kiến trúc để bạn thiết kế Web services, chú trọng vào tài nguyên hệ thống, bao gồm các trạng thái tài nguyên được định dạng như thế nào và được truyền tải qua HTTP, và được viết bởi nhiều ngôn ngữ khác nhau. REST là một bộ quy tắc để tạo ra một ứng dụng Web Service, mà nó tuân thủ 4 nguyên tắc thiết kế cơ bản sau:

* Sử dụng các phương thức HTTP một cách rõ ràng
* Phi trạng thái
* Hiển thị cấu trúc thư mục như các URls
* Truyền tải JavaScript Object Notation (JSON), XML hoặc cả hai.

1. **Sử dụng các phương thức HTTP một cách rõ ràng**

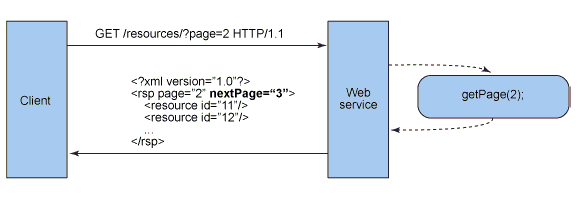
REST đặt ra một quy tắc đòi hỏi lập trình viên xác định rõ ý định của mình thông qua các phương thức của HTTP. Thông thường ý định đó bao gồm lấy dữ liệu, trèn dữ liệu, cập nhập dữ liệu hoặc xóa dữ liệu. Vậy khi bạn muốn thực hiện một trong các ý định trên hãy lưu ý các quy tắc sau:

* Để tạo một tài nguyên trên máy chủ, bạn cần sử dụng phương thức POST.
* Để truy xuất một tài nguyên, sử dụng GET.
* Để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó, sử dụng PUT.
* Để huỷ bỏ hoặc xoá một tài nguyên, sử dụng DELETE.

Chú ý rằng các nguyên tắc ở trên là không bắt buộc, thực tế có thể sử dụng phương thức GET để yêu cầu lấy dữ liệu, chèn, sửa hoặc xóa dữ liệu trên Server. Tuy nhiên REST đưa ra các nguyên tắc ở trên mục đích đưa mọi thứ trở lên rõ ràng và dễ hiểu.

1. **Phi trạng thái**

Một đặc điểm của REST là phi trạng thái (stateless), có nghĩa là nó không lưu giữ thông tin của client. Chẳng hạn bạn vừa gửi yêu cầu để xem trang thứ 2 của một tài liệu, và bây giờ bạn muốn xem trang tiếp theo (sẽ là trang 3). REST không lưu trữ lại thông tin rằng trước đó nó đã phục vụ bạn trang số 2. Điều đó có nghĩa là REST không quản lý phiên làm việc (Session).



Hình 19 Phi trạng thái (Stateless)

Như ở hình trên, Client phải gửi yêu cầu rõ ràng, bao gồm số thự tự của trang cần xem. Như vậy, các thành phần máy chủ phi trạng thái ít phức tạp hơn để thiết kế, viết và phân bổ thông qua máy chủ được cân bằng tải. Dịch vụ phi trạng thái không chỉ hoạt động tốt hơn, nó còn chuyển hầu hết vai trò duy trì trạng thái sang ứng dụng ở máy khách. Trong một dịch vụ mạng RESTful, máy chủ chịu trách nhiệm đưa ra các phản hồi và cung cấp một cách thức cho phép Client duy trì trạng thái ứng dụng của chính nó.

1. **Đưa ra cấu trúc thư mục giống các URI**

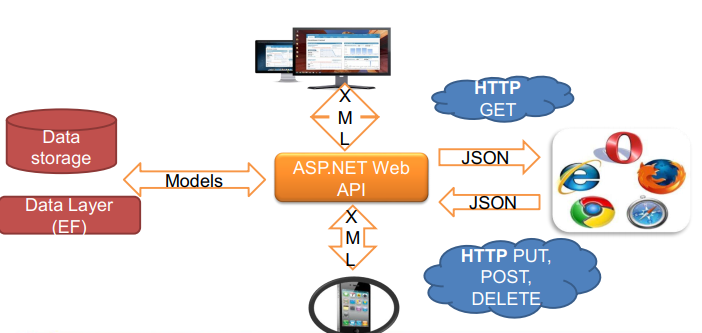
REST đưa ra một cấu trúc để người dùng có thể truy cập vào tài nguyên của nó thông qua các URL, tài nguyên ở đây là dữ liệu về bán hàng, phiếu thu/chi và xuất/nhập kho…

Các địa chỉ REST service cần phải thật trực quan đến mức người dùng dễ đoán. Hãy nghĩ về một địa chỉ (URI) giống như một gợi ý rõ ràng, dễ đoán rằng nó đang trỏ tới cái gì và cung cấp tài nguyên gì. Tóm lại, cấu trúc của một URI nên được đơn giản, có thể dự đoán, và dễ hiểu.

1. **Truyền tải XML, JSON hoặc cả hai**

Dữ liệu gửi lên hoặc trả về có thể ở dạng XML hoặc JSON hoặc cả 2.

## Microsoft .NET Web API



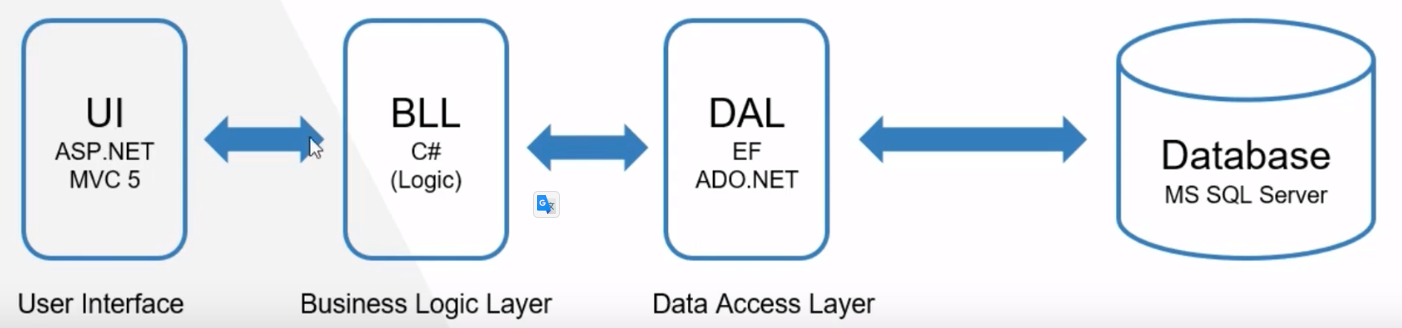
Hình 20 Web API

Đây là một framework mới giúp cho việc xây dựng các HTTP service rất đơn giản và nhanh chóng, Open Source và có thể được sử dụng bởi bất kì client nào hỗ trợ XML, JSON. Hỗ trợ đầy đủ các thành phần HTTP: URI, request/response headers, caching, versioning, content formats, có thể host trong ứng dụng hoặc trên IIS, là kiến trúc lý tưởng cho các thiết bị có băng thông giới hạn như smartphone, tablet và định dạng dữ liệu có thể là JSON, XML hoặc một kiểu dữ liệu bất kỳ [7].

Web API có các ưu điểm:

* Cấu hình hết sức đơn giản khi so với WCF
* Performance cao
* Hỗ trợ RESTful đầy đủ
* Hỗ trợ đầy đủ các thành phần MVC như: routing, controller, action result, filter, model binder, IoC container, dependency injection, unit test
* Open Source

## N-Tier Architecture



Hình 21 Kiến trúc đa tầng

Kiến trúc đa tầng phân tách hệ thống thành các lớp riêng biệt chịu trách nhiệm độc lập:

* **User Interface**: Giao diện người dùng, có chức năng hiển thị giao diện và trao đổi dữ liệu một cách trong suốt giữa người dung cuối và server.
* **Controller**: Đầu API tiếp nhận request từ client, thực hiện điều phối đến thành phần BusinessLayer
* **BussinessLayer**: Tầng xử lý logic, nghiệp vụ từ Controller, gọi đến tầng dữ liệu để truy xuất dữ liệu
* **DataAccessLayer**: Tầng xử lý việc kết nối và truy vấn đến cơ sở dữ liệu, lấy dữ liệu từ Database

Việc phân tách hệ thống thành nhiều lớp có những lợi ích:

* Việc phân chia thành từng lớp giúp cho code được tường minh hơn. Nhờ vào việc chia ra từng lớp đảm nhận các chức năng khác nhau và riêng biệt như giao diện, xử lý, truy vấn thay vì để tất cả lại một chỗ. Nhằm giảm sự kết dính.
* Dễ bảo trì khi được phân chia, thì một thành phần của hệ thống sẽ dễ thay đổi. Việc thay đổi này có thể được cô lập trong 1 lớp, hoặc ảnh hưởng đến lớp gần nhất mà không ảnh hưởng đến cả chương trình.
* Dễ phát triển, tái sử dụng: khi chúng ta muốn thêm một chức năng nào đó thì việc lập trình theo một mô hình sẽ dễ dàng hơn vì chúng ta đã có chuẩn để tuân theo. Và việc sử dụng lại khi có sự thay đổi giữa hai môi trường ( Winform sang Webfrom ) thì chỉ việc thay đổi lại lớp GUI.
* Dễ bàn giao. Nếu mọi người đều theo một quy chuẩn đã được định sẵn, thì công việc bàn giao, tương tác với nhau sẽ dễ dàng hơn và tiết kiệm được nhiều thời gian.
* Dễ phân phối khối lượng công việc. Mỗi một nhóm, một bộ phận sẽ nhận một nhiệm vụ trong mô hình 3 lớp. Việc phân chia rõ ràng như thế sẽ giúp các lập trình viên kiểm soát được khối lượng công việc của mình.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

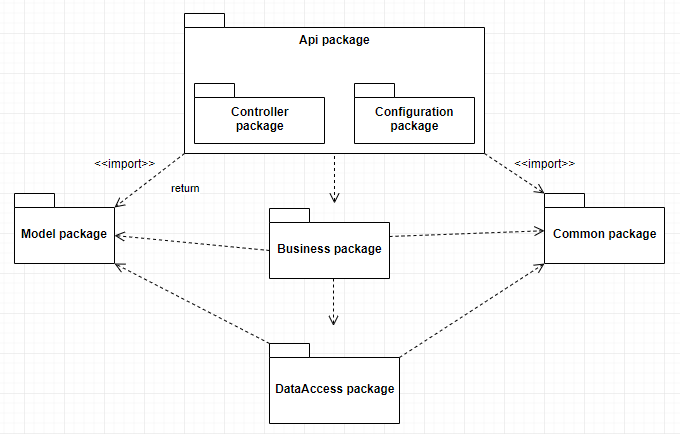
### Lựa chọn kiến trúc

Phần mềm sử dụng kiến trúc 5 tầng dựa trên .NET Web API bao gồm:

Bảng 17 Mô tả Kiến trúc chương trình

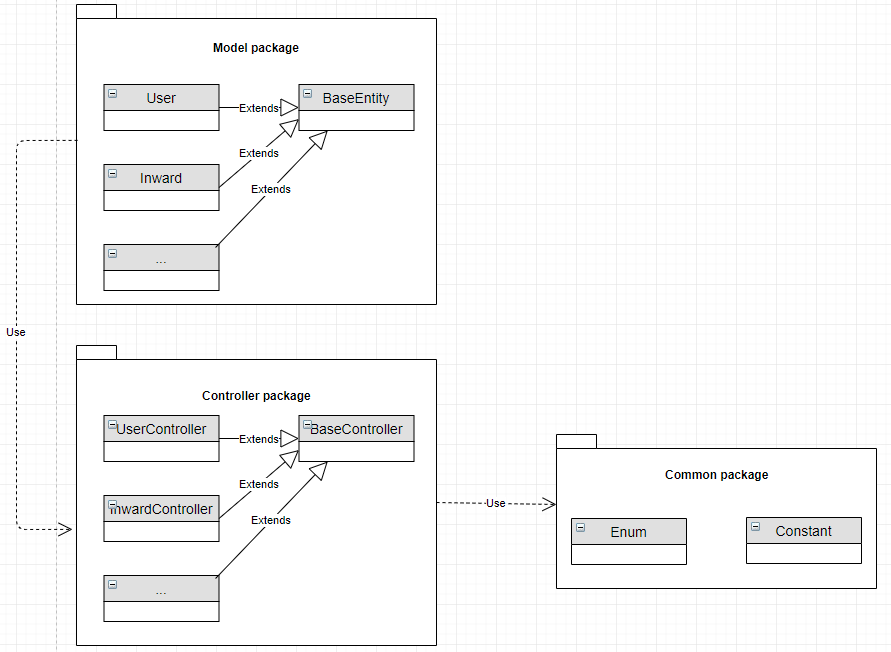
|  |  |  |
| --- | --- | --- |
| Tầng | Chức năng | Các thành phần |
| View | Hiển thị giao diện |  |
| Controller | Nhận yêu cầu người dùng và điều hướng yêu cầu người dùng | Class Controller: Điều hướng cho các yêu cầu từ phía client gửi lên, thường là gọi đến tầng Service |
| WebConfig: Chứa cấu hình về các trang web. VD: map các trang và url, địa chỉ kết nối đến CSDL, cấu hình trang chủ,... |
| Common | Chứa các hằng số, các kiểu định nghĩa | Class Constant: chứa các hằng số |
| Class Enumeration: chứa các kiểu Enum như Loại báo cáo |
| Model | Chứa các lớp khuôn mẫu để đổ dữ liệu | Class User: khuôn mẫu chứa các dữ liệu của người dùng |
| BusinessLayer | Chứa các lớp thực thi nghiệp vụ | Class UserBL: chứa các phương thức xử lý nghiệp vụ cho đối tượng người dùng như Thêm, sửa ,xóa... |
| DataAccessLayer | Liên kết với cơ sở dữ liệu, thực thi các truy vấn để | DatabaseAccess: Chứa các class thực thi kết nối tới database |
| BaseDL: chứa các class thực thi truy vấn chung |

### Thiết kế tổng quan

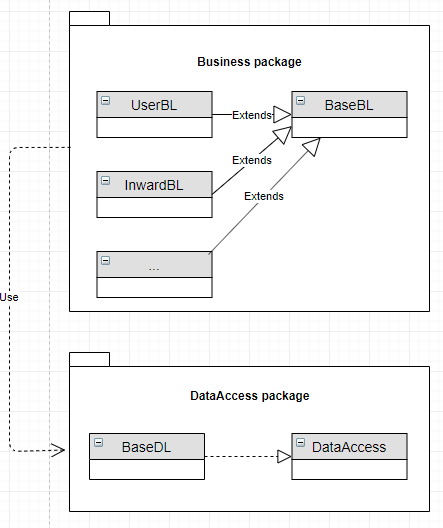


Hình 26 Package diagram

### Thiết kế chi tiết gói



Hình 27 Biểu đồ chi tiết package Controller, Model và Common



Hình 28 Biểu đồ chi tiết package Business và DataAccess

BaseBL: Chứa các phương thức thực thi Logic chung.

BaseDL: Chứa các phương thức tương tác với Cơ sở dữ liệu, định nghĩa các phương thức thực thi truy vấn, xử lý tham số truyền vào Thủ tục ở cơ sở dữ liệu.

DataAccess: xử lý việc kết nối xử lý transaction trong quá trình trao đổi dữ liệu.

## Thiết kế chi tiết

### Thiết kế giao diện

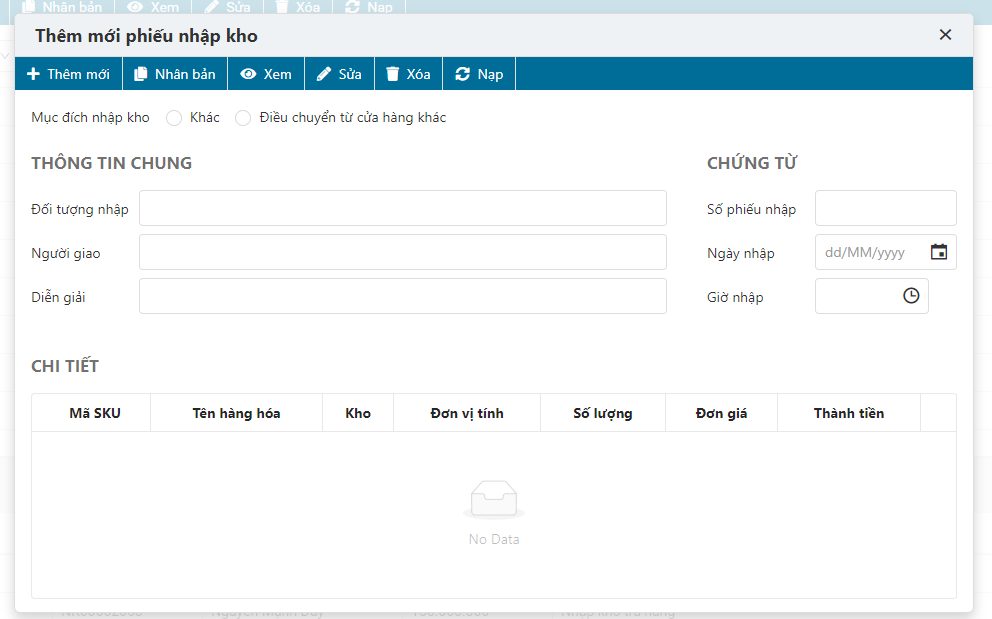
Giao diện được thiết kế trên nền tảng web. Vì là một hệ thống nghiệp vụ nên giao diện cần đơn giản, gọn gàng và tiện ích.

Nền tảng chính là website trên máy tính để bàn. Kích thước màn hình tối thiểu 1024x768. Giao diện cũng được thiết kế reponsive để tương thích với nhiều kích cỡ màn hình khác nhau.



Hình 29 Form đăng nhập

Form đăng nhập được thiết kế gọn gàng, đơn giản.



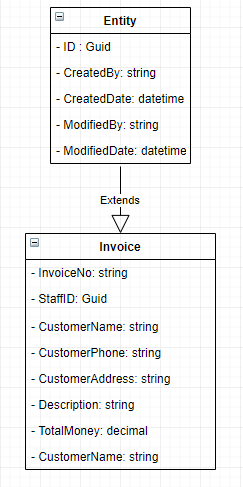
Hình 30 Form thêm mới phiếu nhập kho

Form Thêm phiếu nhập kho được thiết kế ngắn gọn, đủ ý, sẽ hiện ra khi người dùng click vào chức năng thêm phiếu nhập form. Form có đầy đủ các trường thông tin của một phiếu

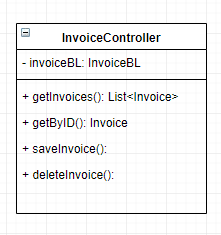
Đối với Form sửa cũng tương tự.

### Thiết kế lớp

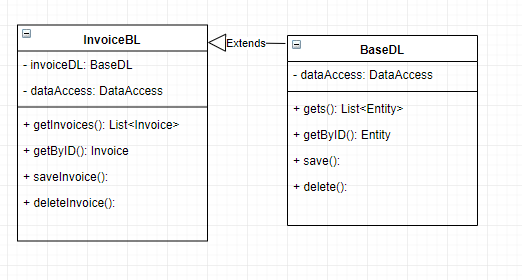
Biểu đồ một số lớp quan trọng



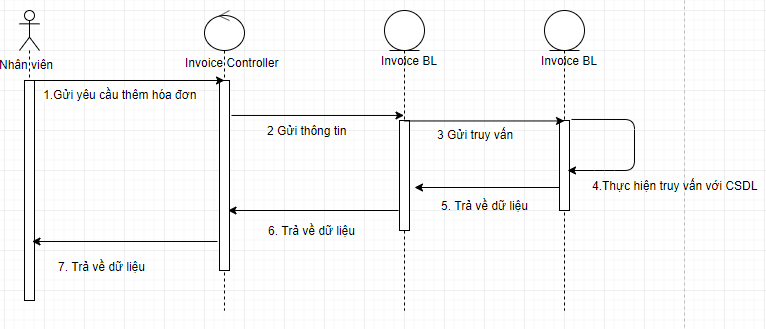
Hình 31 Biểu đồ lớp của lớp Hóa đơn



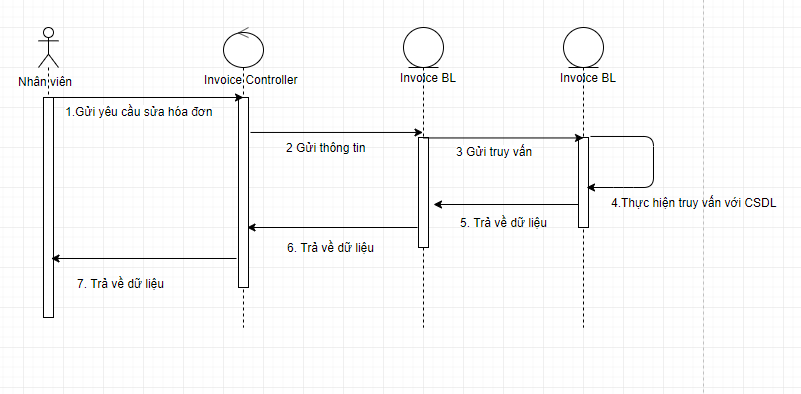
Hình 32 Biểu đồ lớp Student Controller



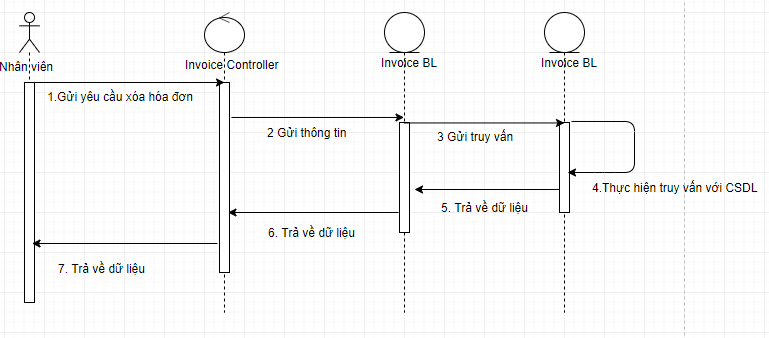
Hình 33 Biểu đồ lớp Invoice Business Layer



Hình 35 Biểu đồ tuần tự thêm mới hóa đơn

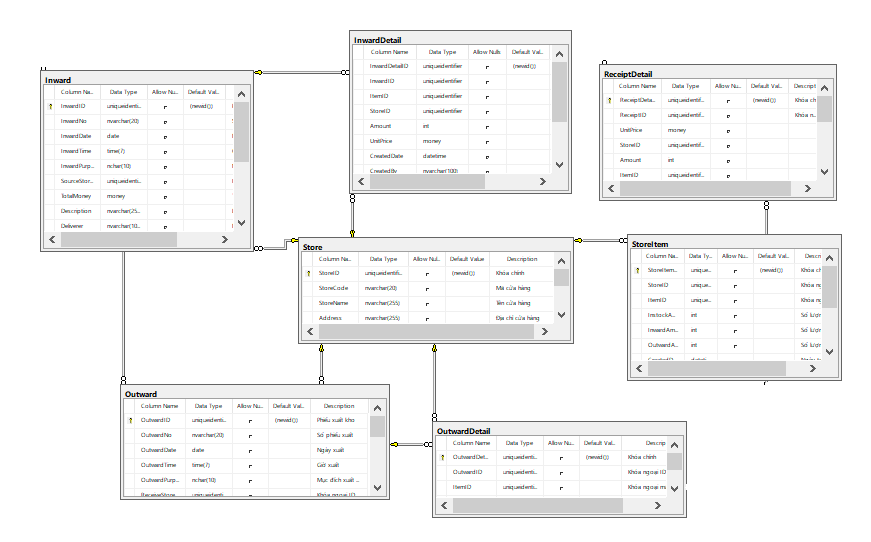


Hình 36 Biểu đồ tuần tự sửa hóa đơn



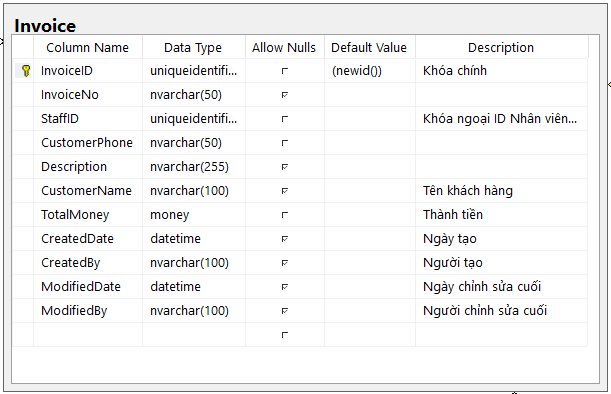
Hình 37 Biểu đồ tuần tự Xóa hóa đơn

### Thiết kế cơ sở dữ liệu

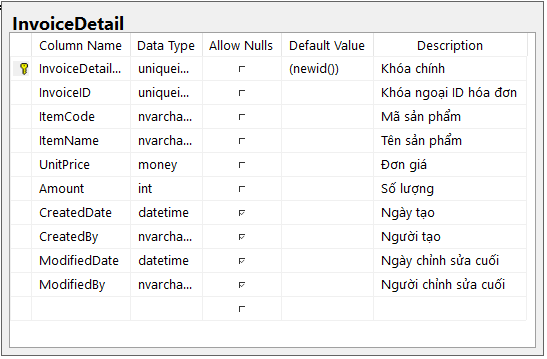


Hình 38 Biểu đồ cơ sở dữ liệu (một phần)

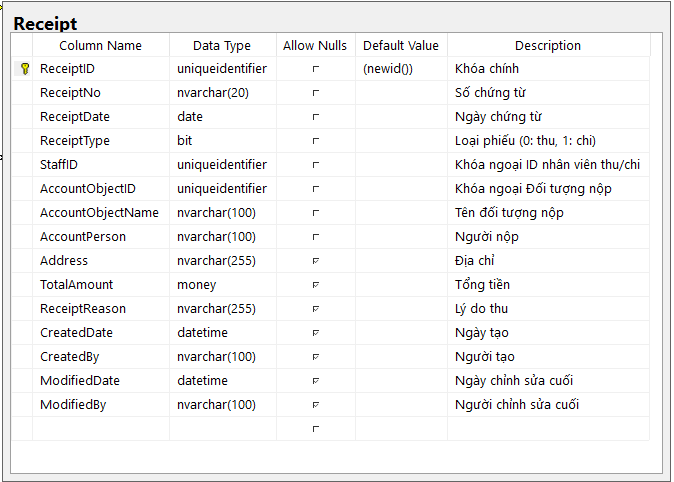
Dưới đây là mô tả cho một số bảng quan trọng

Bảng 18 Mô tả CSDL bảng Hóa đơn 

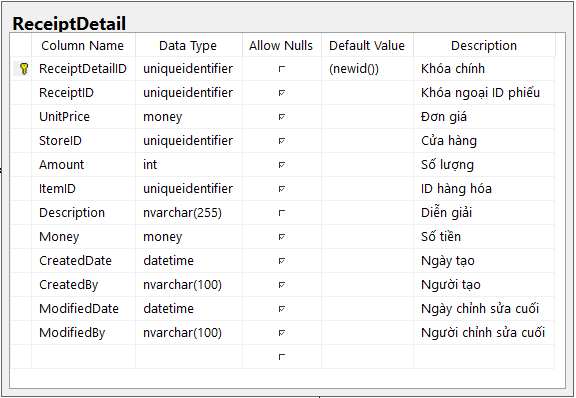
Bảng 19 Mô tả CSDL bảng Chi tiết hóa đơn



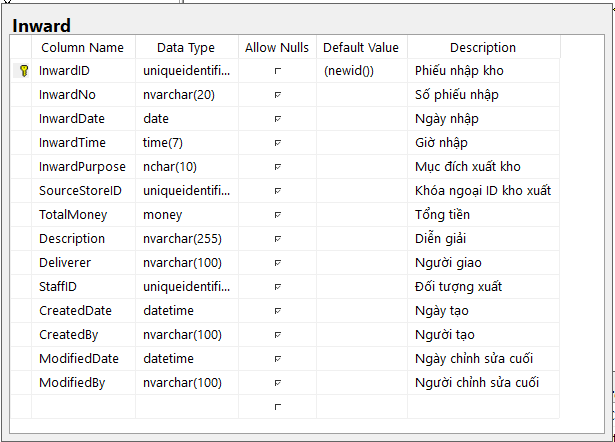
Bảng 20 Mô tả CSDL bảng Phiếu thu,chi



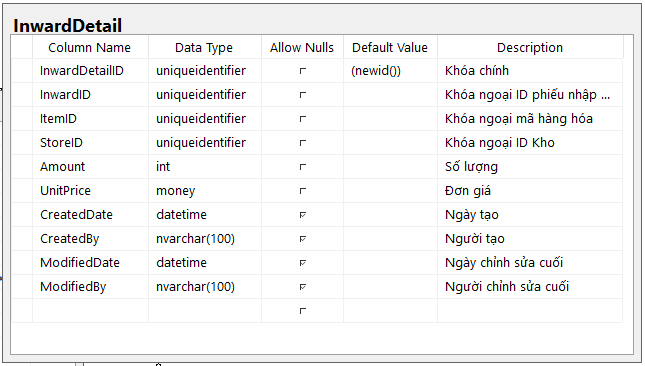
Bảng 21 Mô tả CSDL bảng chi tiết Phiếu thu,chi



Bảng 22 Mô tả CSDL bảng Phiếu xuất kho



Bảng 23 Mô tả CSDL bảng Chi tiết Phiếu xuất kho



## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Bảng 21 Thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Visual Studio 2015 | https://www.visualstudio.com/ |
| Nền tảng | .NET Framework 4.5 | https://www.microsoft.com/en-us/download/details.aspx?id=30653 |
| Quản trị CSDL | SQL Server 2014 | https://www.microsoft.com/en-us/sql-server/sql-server-2014 |
| Server | IIS 10.0 Express | https://www.microsoft.com/en-us/download/details.aspx?id=48264 |
| Hiển thị website | Google Chrome | https://www.google.com/intl/vi\_vn/chrome/ |
| Gửi request | Postman 6.0.10 | https://www.getpostman.com/ |
| Vẻ biểu đồ | Visual Paradigm CE 15.0 | https://www.visual-paradigm.com/download/ |

### Kết quả đạt được

Kết quả đạt được là 1 hệ thống website gồm module chính: Đăng nhập, quản lý Kho, quản lý Thu, chi. Quản lý bán hàng

Bảng thống kê chi tiết

Bảng 22 Thống kê kết quả đạt được

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Thông tin | Số lượng | Ghi chú |
| 1 | Package | 5 |  |
| 2 | Số lớp | 23 |  |
| 4 | Dung lượng mã nguồn | 39MB | Dung lượng project trên Visual Studio |
| 5 | Số module | 4 |  |
| 6 | Số webpage | 5 |  |
| 7 | Kích cỡ dữ liệu | 18MB |  |

### Minh hoạ các chức năng chính

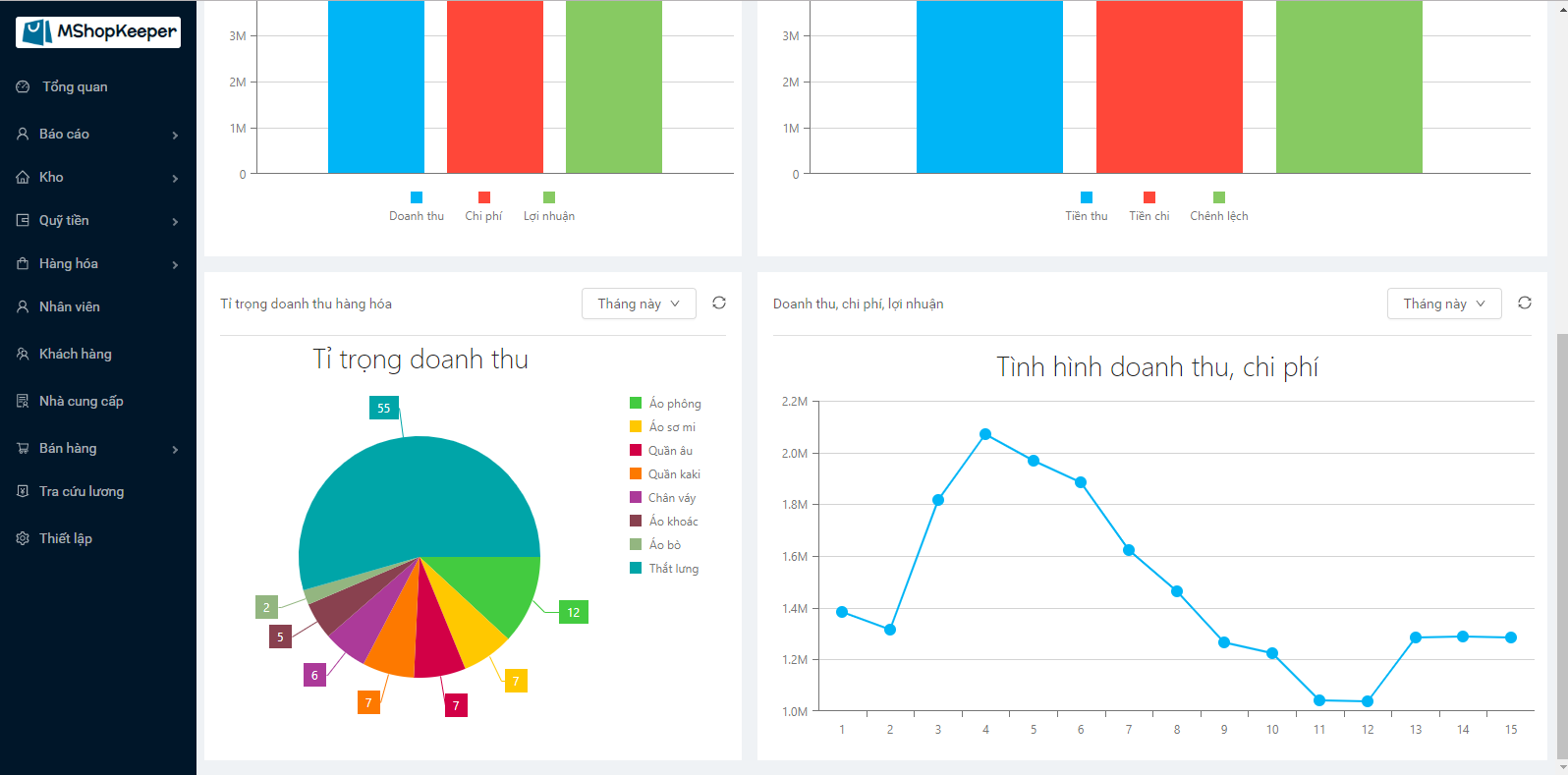
Sau khi truy cập vào trang web, người dùng sẽ được đưa đến trang đăng nhập



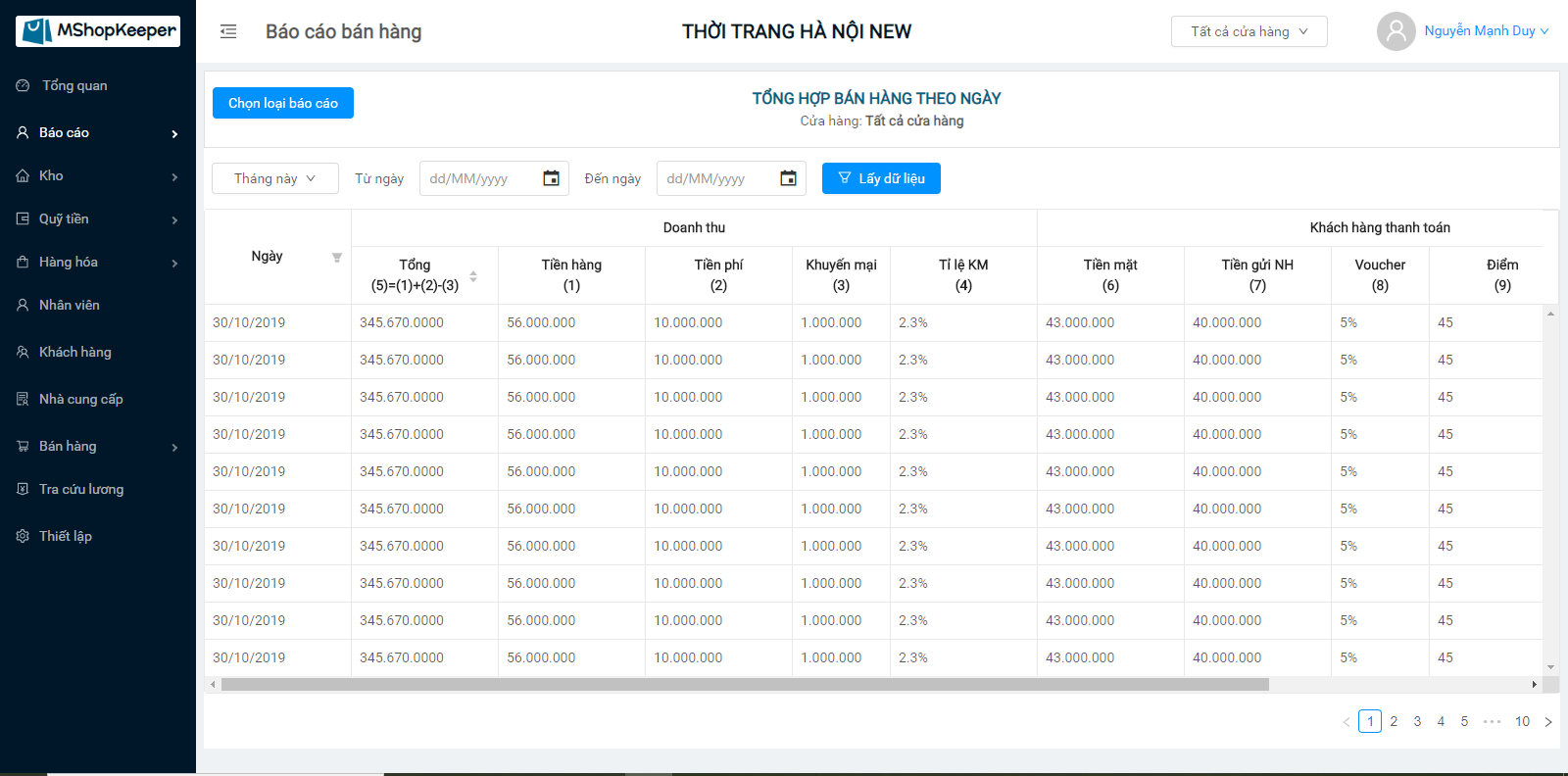
Hình 39: Giao diện trang đăng nhập

Sau khi đăng nhập:

**Với đối tượng quản lý**: Giao diện trang chủ là tab tổng quan hiển thị báo cáo chung tình hình doanh thu, chi phí, lợi nhuận để người quản lý có thể nắm được một cách tổng quan. Bên trái là menu để người quản lý lựa chọn các mục khác như báo cáo cụ thể, quản lý thông tin nhân viên, quản lý quỹ tiền, kho.

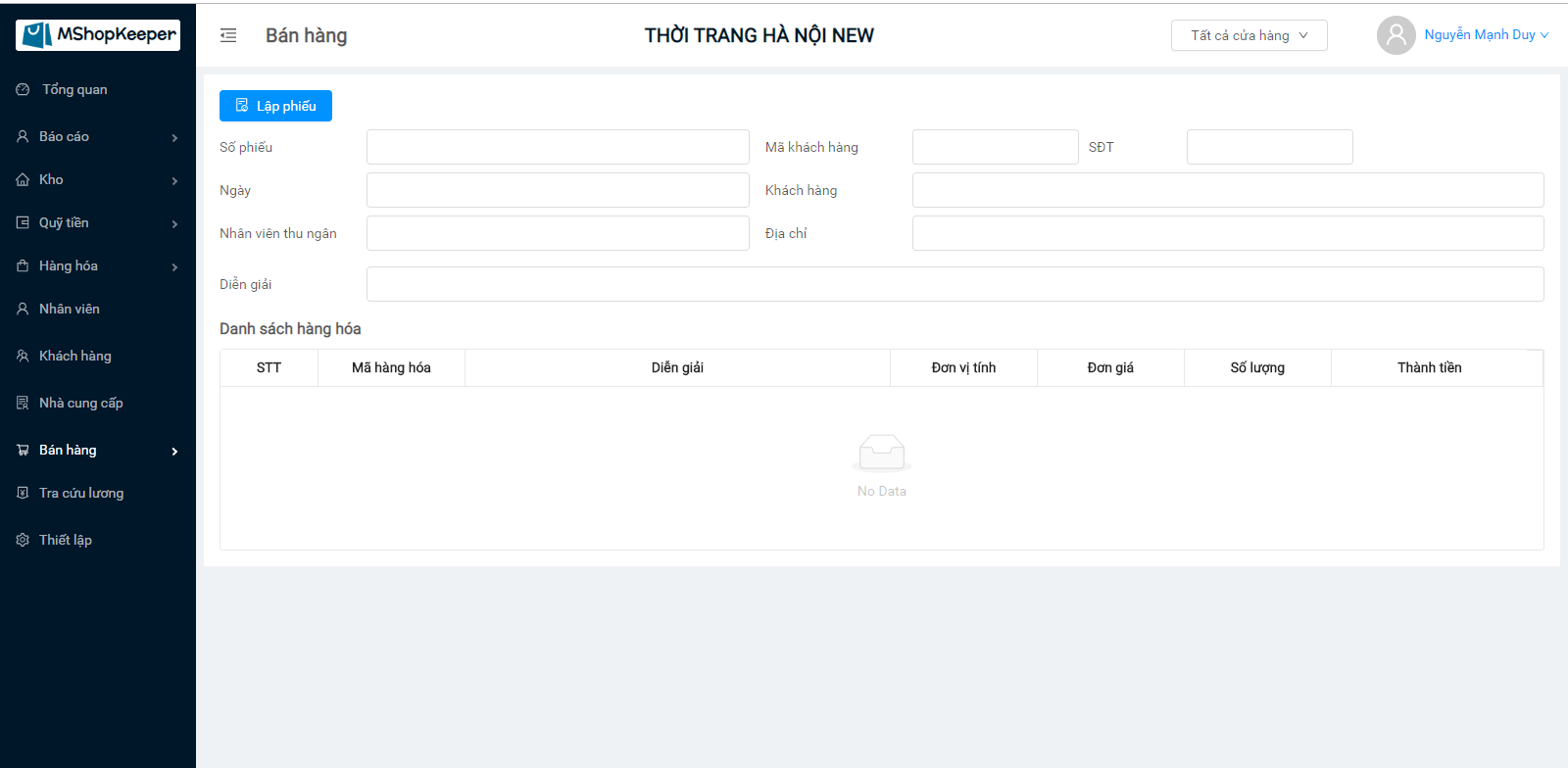


Hình 40: Giao diện trang tổng quan

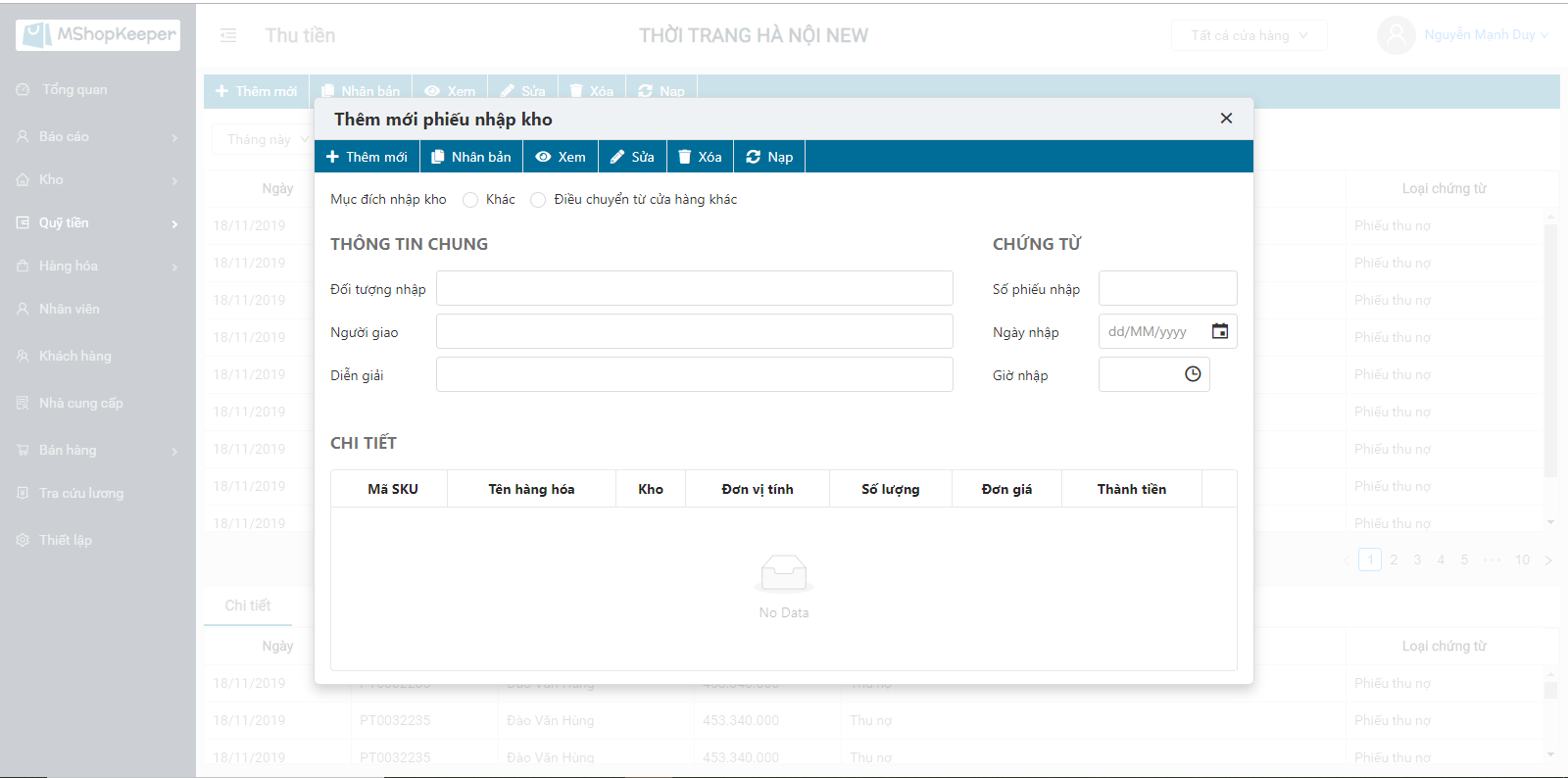


Hình 41: Giao điện trang báo cáo

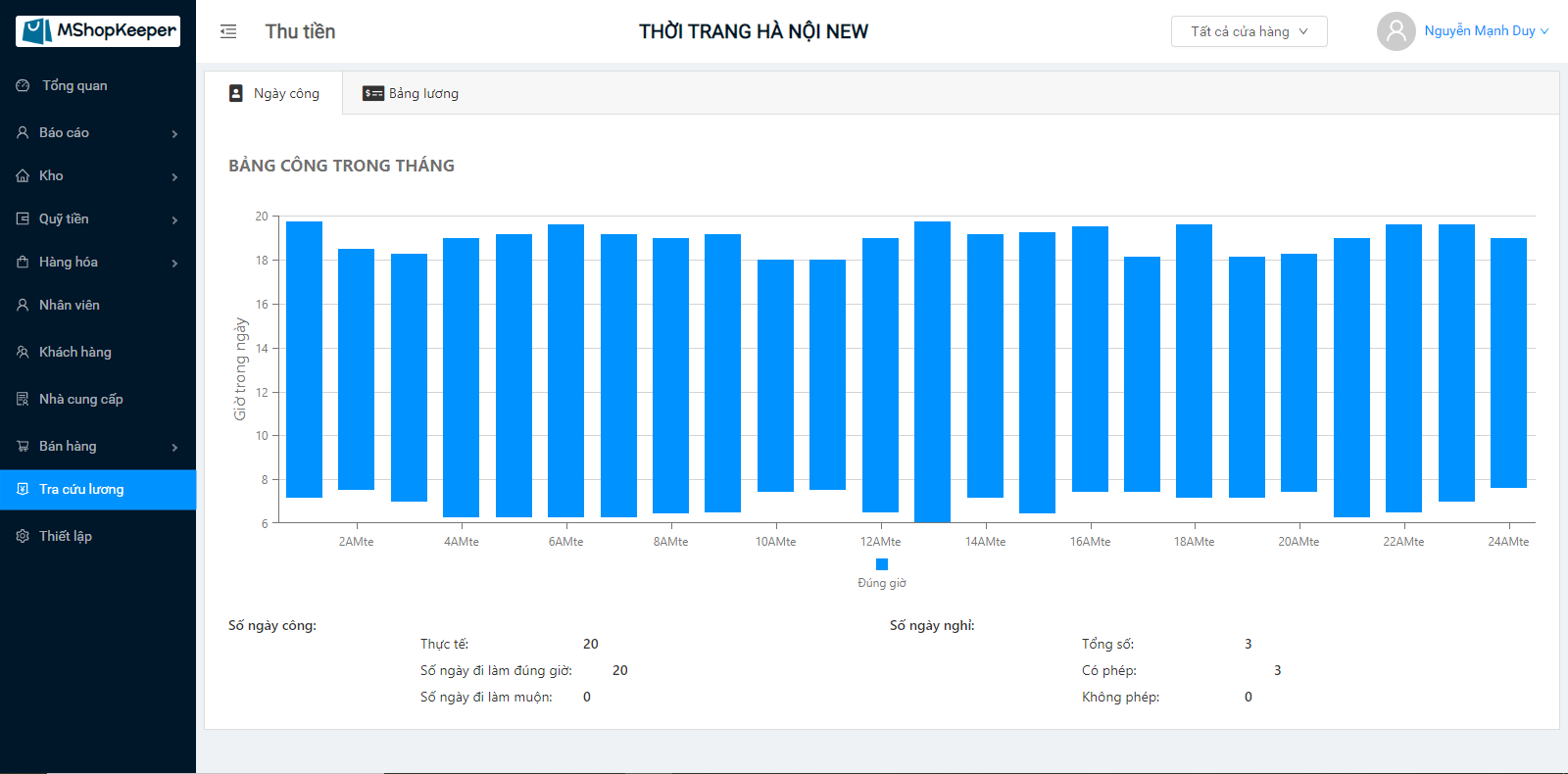
**Với đối tượng nhân viên:** Giao diện hiển thị mặc địch là màn hình lập hóa đơn, ngoài ra nhân viên có thể lập các loại phiếu, xem bảng công, tra cứu lương:



Hình 42: Màn hình lập hóa đơn



Hình 43:Form thêm phiếu nhập kho



Hình 44:Theo dõi ngày công trong tháng

## Triển khai

Hệ thống được triển khai trên server IIS ảo trên localhost.

Các truy cập được tiến hành từ Browser hoặc Postman

# Các giải pháp và đóng góp nổi bật

Trong chương này, em xin được tóm lược lại một số đóng góp cơ bản của đồ án và những phần mà em cảm thấy quan trọng nhất khi thực hiện đồ án. Về cơ bản, trong đồ án này, em đã thay thế mô hình MVC truyền thống bằng cách sử dụng WEB API để cải tiến về mặt hiệu năng. Sau đây là một số tóm lược về các đóng góp cơ bản.

## .NET WEB API được ứng dụng như thế nào trong Project

Không giống như các website PHP hay các website sử dụng framework như WordPress, Magneto... rất phổ biến và có thể tìm thấy mã nguồn ngập tràn trên mạng, .NET Web API chỉ mới được tung ra vài năm gần đây và cũng không phổ biến vì tính phức tạp, chỉ phù hợp cho các ứng dụng nghiệp vụ và các hệ thống lớn của nó. Đây là một nền tảng khá mới, còn rất ít tài liệu tham khảo, chủ yếu là tài liệu của Microsoft và các trang nước ngoài khiến cho việc tiếp cận tài liệu của em rất khó khăn.

Hầu hết các tài liệu đều chỉ mang tính chất giới thiệu hoặc minh họa, việc áp dụng nó vào một dự án thực tế như Quản lý cửa hàng là rất khó. Nhưng rất may mắn là trong quãng thời gian thực tập tại MISA, em có cơ hội tiếp cận với những công nghệ mới như này, và được sự tận tình hướng dẫn của các anh chị nhiều kinh nghiệm ở đây. Em đã tháo gỡ những khó khăn trong chuyên môn để hoàn thành đồ án này.

Để nói đến công dụng lớn nhất của Web API, chúng ta có thể hiểu đơn giản là thay vì cung cấp cho người dùng toàn bộ nội dung trang web từ giao diện, hình ảnh, dữ liệu, xử lý dữ liệu như Web Server truyền thống, API Web Service chỉ thực hiện công việc xử lý dữ liệu. Nó nhận đầu vào là dữ liệu (XML hoặc JSON hoặc cả 2), sau đó trả về dữ liệu cho người dùng, không hề gửi về một dòng HTML, CSS, JS nào cả. Chính vì thế chúng ta hoàn toàn có thể chia nhỏ 1 hệ thống thành nhiều API khác nhau VD trong 1 trang cửa hàng sẽ có các API Quản lý kho, Quản lý thu, chi, Quản lý bán hàng,... hoặc chúng ta có thể chia nhỏ nó ra thành nhiều API nữa như Thêm thông tin, sửa thông tin, xóa thông tin,... và các API này hoàn toàn có thể được tách ra hoặc gộp vào với nhau, triển khai trên cũng 1 service hoặc nhiều service, chúng ta hoàn toàn có thể từ 1 view gọi đến các Web Service khác nhau như khi vào tab Lập hóa đơn, ta gọi đến danh sách hàng hóa Từ Service khác

Việc này đảm bảo 2 yếu tố: thứ nhất là khi service này bị sập, service kia vẫn hoạt động bình thường, thứ 2 là hoàn toàn có thể gọi truy vấn HTTP Request từ ứng dụng di động, desktop hay thậm chí là các hệ thống nhúng nên nếu muốn tạo ra một hệ thống đa nền tảng, chúng ta chỉ cần viết lại phần giao diện cho mỗi loại thiết bị còn phần hệ thống hoàn toàn không ảnh hưởng. Như vậy trong tương lai, khi muốn phát triển thêm ứng dụng cho các nền tảng mobile như iOS, Android, chúng ta có thể tái sử dụng

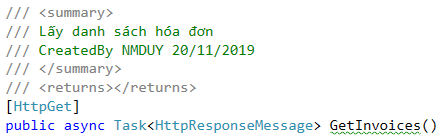
## Tại sao phải áp dụng kiến trúc REST?

Khi chúng ta sử dụng HTTP Request, chúng ta chỉ truyền lên dữ liệu và địa chỉ API, vậy làm sao hệ thống biết gọi đến hàm nào ở trong Controller? Đó chính là lúc kiến trúc REST phát huy tác dụng, các hàm sẽ được gọi theo phương thức và tham số đầu vào của nó.   
Ví dụ khi gửi lên request GET với đường dẫn chỉ bao gồm tên Controller và không có tham số đầu vào, hệ thống sẽ gọi đến phương thức GetInvoice() được cấu hình với phương thức GET và không có tham số đầu vào, còn nếu em vẫn gửi lên request GET với tham số là InvoiceID, hệ thống sẽ gọi đến phương thức GetInvoiceByID(GUID invoiceID) đã được cấu hình với phương thức GET và tham số invoiceID.

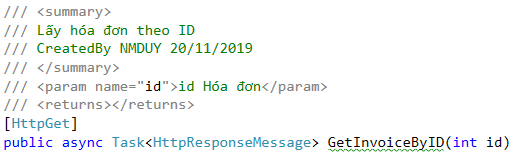
Tất nhiên, 2 phương thức với cùng 1 kiểu request và 1 kiểu tham số sẽ sinh ra lỗi.

Như ở đoạn code bên dưới, cả 2 đều là giao thức Get nhưng khác nhau ở tham số đầu vào, 1 phương thức không có tham số còn 1 phương thức có tham số.

Không có tham số:



Có tham số



# Kết luận và hướng phát triển

## Kết luận

Sản phẩm đã giải quyết được một số vấn đề đặt ra, tuy nhiên vẫn chỉ ở mức độ lý thuyết và chưa có ứng dụng thực tế hoàn toàn. Còn rất nhiều vấn đề thiếu sót như Hệ thống Bảo mật yếu kém, nghiệp vụ chưa hoàn toàn sát với thực tế, giao diện chưa bắt mắt...

Nguyên nhân do trình độ của em còn hạn chế và một hệ thống nghiệp vụ như thế này cần sự làm việc của rất nhiều người trong một khoảng thời gian dài, với các quy trình đầy đủ và sự tham vấn của nhiều bên.

## Hướng phát triển

Để phát triển sản phẩm mang tính thực tiễn cao hơn và có thể áp dụng thực tế, trước hết cần phải bám sát nghiệp vụ hơn, làm sao cho các yêu cầu nghiệp vụ quản lý đặt ra phải được thực hiện đầy đủ, chính xác.

Tiếp sau đó là mô hình hệ thống cần được cái tiến để đạt hiệu suất cao hơn. Mô hình hiện tại vẫn còn một số nhược điểm như chưa phân hóa rõ nhiệm vụ của các tầng, còn sự trùng lặp phương thức, trùng lặp code.

Cuối cùng là các biện pháp bảo mật như token cho API, chống tấn công hệ thống,...

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | World Wide Web Consortium (W3C), "W3C," World Wide Web Consortium (W3C), [Online]. Available: https://www.w3.org/. [Accessed 25 05 2018]. |
| [2] | D. K. Taft, jQuery Eases JavaScript, AJAX Development, Eweek, 2006. |
| [3] | D. Wellman, jQuery UI 1.8: The User Interface Library for jQuery (1st ed.), ISBN 978-1849516525, 2011. |
| [4] | J. G. Andrew Stellman, Head First C#, 3rd Edition, O'Reilly Media, ISBN-13: 978-1449343507, 2013. |
| [5] | J. Richter, Applied Microsoft® .NET Framework Programming (Developer Reference), ISBN-13: 978-0735614222. |
| [6] | S. R. Leonard Richardson, RESTful Web Services, O'Reilly Media, ISBN-13: 978-0596529260, 2008. |
| [7] | B. Lakshmiraghavan, Practical ASP.NET Web API 1st ed. Edition, ISBN-13: 978-1430261759. |
| [8] | Microsoft, "docs.microsoft.com," Microsoft, 04 27 2010. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff649690(v=pandp.10). [Accessed 25 05 2018]. |
| [9] | [Online]. |