**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**Môn học: Phát triển ứng dụng web**

**ĐỀ TÀI**

**QUẢN LÝ CYPHER GAME**

**GV hướng dẫn : Bùi Chí Thành  
SV thực hiện : Phạm Ngọc Duy**

**Mục lục**

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 4](#_Toc151761566)

[1.1. Lý do chọn đề tài: 4](#_Toc151761567)

[1.2. Mục tiêu của đề tài 4](#_Toc151761568)

[1.3. Giới hạn và phạm vi của đề tài 4](#_Toc151761569)

[1.4. Tính khả thi 4](#_Toc151761570)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 5](#_Toc151761571)

[2.1. Tổng quan về ngôn ngữ Asp.NET MVC 5](#_Toc151761572)

[2.1.1. Giới thiệu Asp.NET MVC Framework 5](#_Toc151761573)

[2.1.2 Lý do chọn Asp.NET MVC 10](#_Toc151761574)

[2.2. Tổng quan về SQL SERVER 11](#_Toc151761575)

[2.2.1. Hệ quản trị CSDL SQL Server 11](#_Toc151761576)

[**2.2.2. Đặc điểm của SQL Server và đối tượng làm việc** 12](#_Toc151761577)

[CHƯƠNG 3: NỘI DUNG THỰC HIỆN 13](#_Toc151761578)

[3.1 . Đối tượng sử dụng website: 13](#_Toc151761579)

[3.2. Các chức năng chính của website: 13](#_Toc151761580)

[3.3 Phân tích thiết kế hệ thống 15](#_Toc151761581)

[3.3.1 Các yêu cầu chức năng chính của hệ thống 15](#_Toc151761582)

[3.3.2 Các biểu đồ Use Case 17](#_Toc151761583)

[3.4 Thiết kế cơ sở dữ liệu 18](#_Toc151761584)

[3.4.1 Mô hình cơ sở dữ liệu 18](#_Toc151761585)

[3.4.2 Danh sách các lớp đối tượng 18](#_Toc151761586)

[3.5 Thiết kế và đặc tả giao diện 21](#_Toc151761587)

[3.5.1 Giao diện trang Admin 22](#_Toc151761588)

[3.5.1.1 Giao diện trang chủ quản trị 22](#_Toc151761589)

[3.5.1.2 Giao diện trang quản lý thành viên 23](#_Toc151761590)

[3.5.1.3 Giao diện trang danh sách máy 24](#_Toc151761591)

[3.5.1.4 Giao diện trang danh sách đơn giá của máy 24](#_Toc151761592)

[3.5.1.5 Giao diện trang quản lý dịch vụ 25](#_Toc151761593)

[3.5.1.6 Giao diện trang đơn hàng 29](#_Toc151761594)

[3.5.1.7 Giao diện trang order 29](#_Toc151761595)

[CHƯƠNG 4: KẾT LUẬN 30](#_Toc151761596)

[4.1 Kết quả đạt được của đề tài 30](#_Toc151761597)

[4.2 Hạn chế của đề tài 30](#_Toc151761598)

[4.3 Kết luận 30](#_Toc151761599)

# TỔNG QUAN VỀ ĐỀ TÀI

## Lý do chọn đề tài:

Chương trình quản lý Cyber Game dành cho một Cyber Game nhỏ với khoảng 10 đến 20 máy chỉ được cài đặt trên máy chủ. Phần mềm này hoạt động như một bảng điều khiển, giúp quản trị viên xử lý các thông tin liên quan đến thanh toán tiền máy, ghi lại sự cố của các máy trong Cyber Game, và cung cấp thông tin tổng kết về tình hình hoạt động của quán NET sau một khoảng thời gian cụ thể.

## Mục tiêu của đề tài:

* Trau dồi kiến thức cho môn học, có kinh nghiệm phân tích, thiết kế, cụ thể hóa các bài toán thực tế.
* Tăng kỹ năng nhìn tổng quan về vấn đề.
* Công nghệ hóa quy trình quản lý tại cửa hàng.
* Ứng dụng xây dựng website quản lý Cyber Game với các tính năng sau: giao diện thân thiện, dễ sử dụng, xem thông tin dễ dàng, mở máy nhanh chóng.

## Giới hạn và phạm vi của đề tài:

* Nhỏ, trước mắt sẽ triển khai tại các quán net nhỏ ở thành phố Nha Trang hoặc các vùng lân cận
* Đối tượng sử dụng: Phần mềm được sử dụng bởi nhân viên quán net.

## Tính khả thi:

* Xây dựng được website quản lý quán net theo yêu cầu của đề tài.
* Xây dựng được hệ thống quản lý quán net có đủ các chức năng cơ bản.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tổng quan về ngôn ngữ Asp.NET MVC

Mô hình kiến trúc Model-View-Controller (MVC) được áp dụng để chia ứng dụng thành ba thành phần chính: model, view và controller. ASP.NET MVC là một nền tảng giúp tạo ra các ứng dụng web sử dụng mô hình MVC thay vì theo mô hình ASP.NET Web Forms. ASP.NET MVC được đánh giá vì tính nhẹ, dễ kiểm thử giao diện (so với Web Forms), và khả năng tích hợp tốt với các tính năng của AưSP.NET. Nó được định nghĩa trong namespace System.Web.Mvc và là một phần của namespace System.Web.

MVC là một mẫu thiết kế chuẩn mà nhiều lập trình viên quen thuộc. Một số ứng dụng web phù hợp với kiến trúc MVC, trong khi những ứng dụng khác có thể ưa chuộng ASP.NET Web Forms và cơ chế postbacks. Đôi khi, có các ứng dụng kết hợp cả hai kiến trúc để đáp ứng yêu cầu cụ thể của họ.

2.1.1. Giới thiệu Asp.NET MVC Framework

ASP.NET là một nền tảng ứng dụng web (web application framework) được phát triển và cung cấp bởi Microsoft, cho phép những người lập trình tạo ra những trang web động, những ứng dụng web và những dịch vụ web. - Dựa trên ASP.NET, ASP.NET MVC cho phép các nhà phát triển phần mềm xây dựng một ứng dụng web dựa trên mẫu thiết kế MVC. - MVC là một mẫu thiết kế (design pattern) chuẩn, được sử dụng nhằm chia ứng dụng thành ba thành phần chính: model, view và controller. - Phiên bản ASP.NET MVC: ASP.NET MVC 5 (10/2013), ASP.NET MVC 5.2.7 (11/2017).

Nền tảng MVC bao gồm các thành phần dưới đây :

*Hình 1: Mẫu Model - View - Controller*

**Models:** Các đối tượng Models là một phần của ứng dụng, 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, một đối tượng Product (sản phẩm) sẽ lấy dữ liệu từ CSDL, thao tác trên dữ liệu, và cập nhật dữ liệu trở lại vào bảng Products ở SQL Server.

Trong các ứng dụng nhỏ, model thường chỉ là một khái niệm nhằm phân biệt, chứ không nhất thiết phải được cài đặt thực tế. Ví dụ, nếu ứng dụng chỉ đọc dữ liệu từ CSDL và gửi chúng đến view, không cần phải có tầng model và các lớp liên quan. Trong trường hợp này, dữ liệu được lấy như là một đối tượng model (hơn là tầng model).

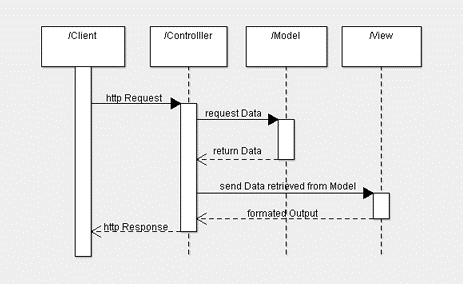
**Views:**  là các thành phần dùng để hiển thị giao diện người dùng (UI). Thông thường, view được tạo dựa vào thông tin dữ liệu model. Ví dụ như, view dùng để cập nhật bảng Products sẽ hiển thị các hộp văn bản, drop-down list, và các check box dựa trên trạng thái hiện tại của một đối tượng Product.

**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.

Mẫu MVC giúp bạn tạo được các ứng dụng mà chúng phân tách rạch ròi các khía cạnh của ứng dụng (logic về nhập liệu, logic xử lý tác vụ và logic về giao diện). Mẫu 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 bạn 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 bạn 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, bạn 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.



Hình 2: Mô hình tuần tự của MVC

Lấy ví dụ một GUI Component (thành phần đồ họa người dùng ) đơn giản là checkbox. Checkbox có thành phần Model để quản lý trạng thái của nó là check hay uncheck, thành phần View để thể hiện nó với trạng thái tương ưng lên màn hình, và thành phần Controller để xử lý nhưng sự kiện khi có sự tương tác của người sử dụng hoặc các đối tượng khác lên checkbox.

Khi ngươi sử dụng nhần chuột vào Check box , thành phần Controller của Checkbox sẽ xử lý sự kiện này, yêu cầu thành phần Model thay đổi dữ liệu trạng thái. Sau khi thay đổi trạng thái, thành phần Model phát thông điệp đến thành phần View và Controller. Thành phần View của Checkbox nhận được thông điệp sẽ cập nhật lại thể hiện của Checkbox, phản ánh chính sác trạng thái Checkbox do Model lưu giữ. Thành phần Controller nhận được thông điệp do Model gởi tới sẽ có nhưng tương tác phản hòi với người sử dụng nếu cần thiết.

**Tiềm hiểu thêm về Controler**

Controller có trách nhiệm chính là điều hướng các yêu cầu của người sử dụng. Như vậy trên toàn ứng dụng của ta, tất cả các request đều sẽ phải đi tới controller. Và tại đây, ứng với các tham số người sử dụng truyền mà ta đưa họ đến một tác vụ nào đó trên ứng dụng.

Tại các tác vụ này, chúng sẽ thông qua lớp model để làm việc và trả kết quả trở về controller. Cuối cùng controller sẽ đẩy dữ liệu thao tác tới view. View là thành phần cuối cùng mà người sử dụng nhận được khi họ giở request tới ứng dụng.

Có thể hiểu controller, giống với kỹ thuật đa cấp với các tác vụ chạy ứng dụng phân cấp theo từng nhánh riêng biệt như: Module, action,… Qua hình vẽ này, ta hiểu rằng. Để có thể thao tác với các action (hành động) ta cần phải đi qua file index.php. Lúc này file index đóng vai trò như một controller được dùng để điều hướng các request. Ứng với các request thì nó sẽ trả về một controller khác để xử lý tác vụ một cách cụ thể. Tại controller con, nó sẽ gọi các action riêng biệt. Ở đó, thông qua các action mà nó gọi tới các file xử lý giản đơn.

**Tìm hiểu Model**

Model là thành phần chủ yếu được sử dụng để thao tác xử lý dữ liệu. Trong các framework, Model vẫn thường sử dụng theo phương thức Active Record. Một trong những design pattern. Chúng có tác dụng rút ngắn thời gian viết câu truy vấn cho người sử dụng. Biến những câu truy vấn phức tạp trở nên gần gũi và thân thiện với người sử dụng thông qua các thư viện được định nghĩa sẵn.

Model thường sẽ là các phương thức có trách nhiệm xử lý các tác vụ như: select, insert, update, delete các record trong database. Ứng với các lấy dữ liệu, model thường sử dụng mảng để gởi trả kết quả về. Vì mảng có thể cho phép model lưu trữ nhiều thông tin hơn, nên thường các record khi bóc tách chúng sẽ mang các dữ liệu của database một cách chi tiết.  
Khi sử dụng models, ta cũng cần tuân theo nguyên tắc chính của chúng là không xuất giá trị trực tiếp trong model. Mà tất cả những dữ liệu ấy, phải đưa vào mảng và trả về theo phương thức. Và tiếp tục ở view ta sẽ sử dụng nó để lấy dữ liệu ra.

**Tìm hiều về View**

View là phần hiển thị thông tin tương phản khi gởi và nhận request. Trước đây, khi người lập trình chưa nghĩ tới view. Họ thường thao tác xử lý dữ liệu ngay trực tiếp trên ứng dụng và đổ cả dữ liệu ngay trên file PHP đó. Điều này làm cho ứng dụng trở nên cồng kềnh, và đặc biệt rất khó cho việc bảo trì nâng cấp sau này. Nhất là đối với designer, việc thay đổi giao diện của một website luôn làm cho họ cảm thấy đau đầu vì phải vọc thẳng vào core. Trước đây, để giải quyết tình huống này. Người ta thường sử dụng template để phân tách website thành 2 mảng riêng biệt. Một là giao diện và một là core. Việc chỉnh sửa giao diện trở nên đơn giản hơn đối với họ so với cách viết thập cẩm kia. Tuy nhiên, các thư viện này thực chất sẽ làm cho ứng dụng của chúng ta trở nên chậm chạp hơn bao giờ hết. Bởi chúng phải phiên dịch nhiều lần các kịch bản.

Chẳng hạn: Để dễ thao tác, smarty sẽ dịch ngược các yêu cầu của bạn sang ngôn ngữ của nó. Sau đó chúng sẽ chuyển ngôn ngữ đó sang PHP và thao tác xử lý trên nó.  
Việc này sẽ làm ứng dụng chậm chạp, do cứ phải dịch qua, dịch lại một kịch bản. Trong khi, với sự kết hợp của PHP thuần, ứng dụng của bạn sẽ nhanh và ổn định hơn nhiều. Và view cũng là một phần trong việc nâng cấp những hạn chế ấy. Chúng giúp giảm thiểu tối đa quá trình biên dịch nhiều lần. Và làm cho ứng dụng trở nên mạnh mẽ và chuyên nghiệp hơn nhiều so với cách lập trình thuần.

2.1.2 Lý do chọn Asp.NET MVC

* Các tính năng của 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. Bạn 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. Bạn 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. Bạn 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 bạn 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 bạn 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 (.mater) Bạn có thể dụng các tình năng có sẵn của ASP.NET như dùng lồng các trang master page sử dụng in-line expression (<%=%>), sử dụng server controls ,mẫu, data -binding, địa phương 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 MVC5 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.

2.2. Tổng quan về SQL SERVER

2.2.1. Hệ quản trị CSDL SQL Server

Khái niệm hệ quản trị cơ sở dữ liệu SQL Server

SQL Server là một hệ thống quản lý cơ sở dữ liệu RDBMS sử dụng Transact-SQL để trao đổi dữ liệu giữa Client computer và SQL Server computer. Một RDBMS bao gồm Databases, Database engine và các ứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS.

SQL Server được tối ưu để có thể chạy trên môi trường cơ sở dữ liệu rất lớn (Very Large Database Environment) lên đến Tera -Byte và có thể phục vụ cùng lúc cho hàng ngàn User. SQL Server 2000 có thể kết hợp "ăn ý" với các Server khác như Microsoft Internet Information Server (IIS), ECommerce Server, Proxy Server...

**2.2.2. Đặc điểm của SQL Server và đối tượng làm việc**

* SQL là ngôn ngữ tựa tiếng Anh.
* SQL là ngôn ngữ phi thủ tục, nó không yêu cầu ta cách thức truy nhập CSDL như thế nào. Tất cả các thông báo của SQL đều rất dễ sử dụng và ít khả năng mắc lỗi .
* SQL cung cấp tập lệnh phong phú cho các công việc hỏi đáp dữ liệu
* Chèn, cập nhật, xoá các hàng trong một quan hệ.
* Tạo, sửa đổi, thêm và xoá các đối tượng trong của CSDL.
* Điều khiển việc truy nhập tới cơ sở dữ liệu và các đối tượng của CSDL để đảm bảo tính bảo mật của cơ sở dữ liệu.
* Đảm bảo tính nhất quán và sự ràng buộc của CSDL.
* Yêu cầu duy nhất để sử dụng cho các hỏi đáp là phải nắm vững được các cấu trúc CSDL của mình.

# CHƯƠNG 3: NỘI DUNG THỰC HIỆN

## . Đối tượng sử dụng website:

* Đối với Admin: Admin có quyền quản lý (thêm/sửa/xóa) các máy cho khách hàng sử dụng, dịch vụ, hóa đơn, mở máy.
* Đối với thành viên: có quyền gọi dịch vụ trên trang web.

## 3.2. Các chức năng chính của website:

Chức năng đối với Admin

1. ***Quản lý tài khoản***

Quản trị viên đăng nhập vào hệ thống bằng cách cung cấp tài khoản, password vào form đăng nhập được xây dựng sẵn. Kiểm tra thông tin vào có hợp lệ hay không. Nếu có chuyển đến trang quản trị ngược lại thông báo lỗi và yêu cầu nhập lại.

1. ***Quản lí mở máy***

Quản trị viên mở máy cho khách hàng khi khách hàng đã đăng kí thành viên

1. ***Quản lý máy tính***

Quản trị viên có quyền thêm, sửa hoặc xóa máy tính:

* Thêm:
* Kiểm tra tính hợp lệ của thông tin máy.
* Kiểm tra số ID máy có tồn tại hay chưa?
* Nếu chưa thêm mới máy vào CSDL.
* Sửa:
* Kiểm tra tính hợp lệ của thông tin máy.
* Kiểm tra số ID máy có tồn tại hay chưa?
* Nếu chưa cập nhật thông tin máy vào CSDL.
* Xóa:
* Nếu máy đang hoạt động hoặc tồn tại thì không được phép xóa.
* Ngược lại cho phép xóa máy tính.

1. ***Quản lý dịch vụ***

Quản trị viên có quyền thêm, sửa hoặc xóa các loại món ăn, nước uống.

1. ***Quản lý đơn hàng***

Quản trị viên có thể xem, xóa và giải quyết đơn hàng.

* Xem:
* Hiển thị tất cả các hóa đơn, phân theo tình trạng: chưa thanh toán và đã thanh toán.
* Xóa:
* Chỉ được phép xóa các hóa đơn chưa thanh toán.
* Các hóa đơn đã thanh toán thì không được xóa.
* Thanh toán hóa đơn
* Hóa đơn chờ thanh toán được cập nhật thành hóa đơn đã thanh toán.

1. ***Quản lý đăng ký thành viên***

Khách hàng có thể đăng ký làm thành viên thông qua quản trị. Khi đó, khách hàng sẽ cung cấp các thông tin như tài khoản, password. Khi hoàn thành quản trị viên sẽ đưa tài khoản, password cho khách.

Chức năng đối với Thành viên

1. ***Xem dịch vụ***

Khách hàng lựa chọn dịch vụ mình muốn để order.

1. ***Gọi dịch vụ***

Sau khi chọn dịch vụ vào giỏ hàng, khách hàng có thể gọi dịch vụ đó. Hệ thống sẽ xử lý thông tin dịch vụ của khách hàng để ghi nhận hóa đơn vào cơ sở dữ liệu. Hiển thị thông tin về dịch vụ mà khách hàng đã đặt hàng.

1. ***Đăng nhập website***

Thành viên sau khi có tài khoản thì sẽ đăng nhập vào website bằng cách gõ tài khoản, password đã đăng ký. Sau đó hệ thống sẽ kiểm tra cơ sở dữ liệu để xác nhận.

3.3 Phân tích thiết kế hệ thống

3.3.1 Các yêu cầu chức năng chính của hệ thống

***a)Yêu cầu lưu trữ***

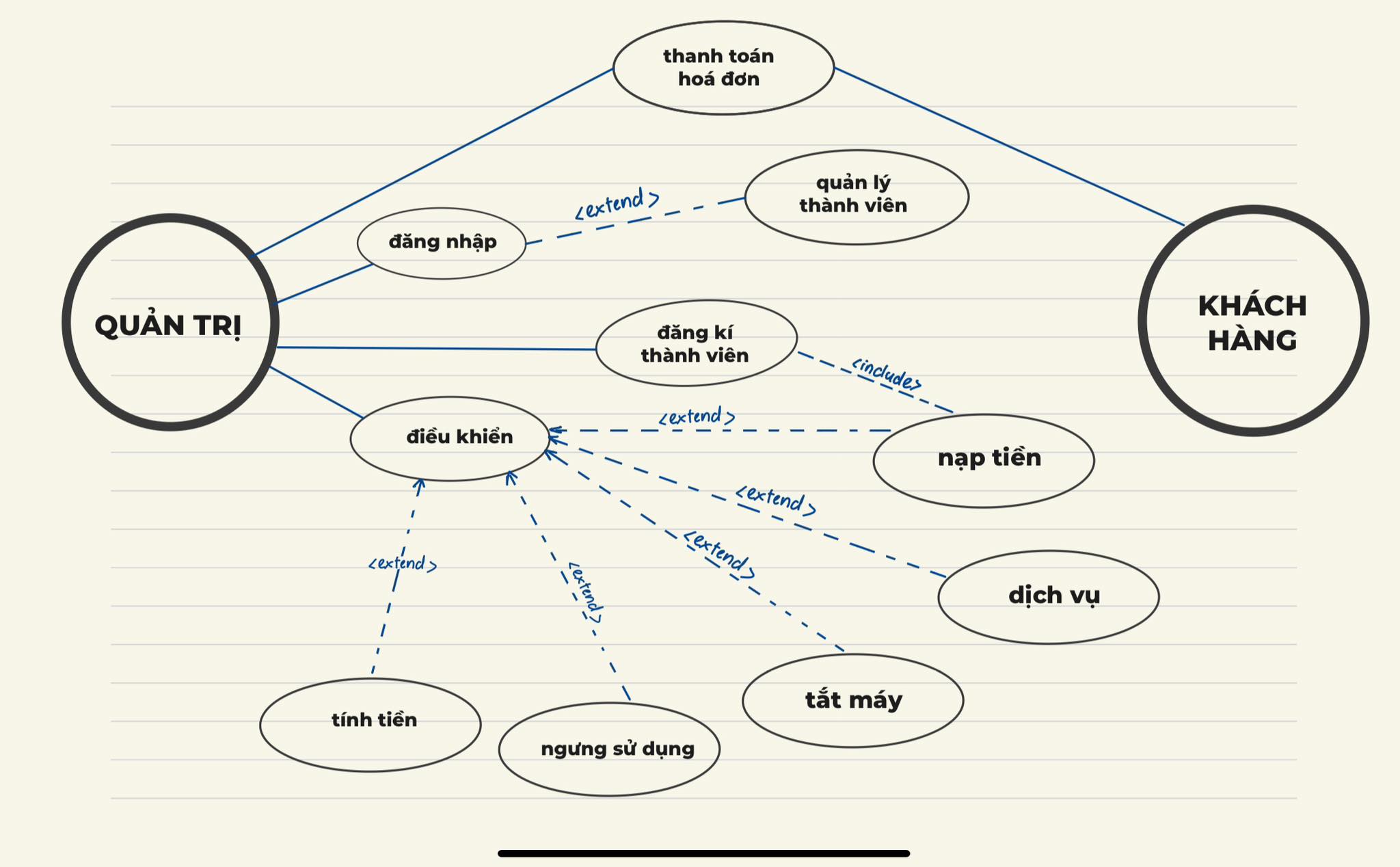
* Hệ thống cần lưu trữ các thông tin liên quan đến máy, dịch vụ, thành viên.
* Thông tin đăng nhập
  + Phần quản trị của website cần đảm bảo tính bảo mật do có thể truy cập tới toàn bộ thông tin về quán net và hoạt động kinh doanh.
* Thông tin DonGia (ID\_gia,GiaMay
* Thông tin May (ID\_May, TenMay, TinhTrangMay, ID\_gia, Hoatdong)
* Thông tin DichVu (ID\_DV, TenDV, AnhSP, GiaBan)
* Thông tin HoaDon (ID\_HoaDon, ID\_ThanhVien, ID\_TTDV, ID\_May, ThoiGianMo, ThoiGianTat, TinhTrangHD, TongTien)
* Thông tin ThanhVien (ID\_ThanhVien, TenThanhVien, TenDN,Matkhau, SoTien)
* Thông tin donHang(maDH,ID\_ThanhVien, ID\_DV,tongGia,ghiChu,ngayDatHang)
* Thông tin ThanhToanDV(ID\_TTDV, ID\_ThanhVien)

***b) Yêu cầu tính toán***

* Tính tiền thời gian chơi của khách hàng, tính tiền dịch vụ khách hàng order dịch vụ.
* Kết quả tính toán: Tổng số tiền khách hàng phải trả cho mỗi dịch vụ, thời gian chơi của khách hàng.

***c)Yêu cầu tra cứu***

* Chọn thông tin tra cứu về máy, dịch vụ.
* Kết quả xuất ra sau khi tra cứu: Danh sách các dịch vụ đang có, máy chưa hoạt động (máy chưa được chơi).

3.3.2 Các biểu đồ Use Case

***Hình 3: Biểu đồ Use case tổng quát***

3.4 Thiết kế cơ sở dữ liệu

3.4.1 Mô hình cơ sở dữ liệu

- A computer screen shot of a computer

Description automatically generated

###### ***Hình 4: Biểu đồ dữ liệu quan hệ***

3.4.2 Danh sách các lớp đối tượng

1. ***Lớp Máy***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_May | Varchar(5) | Khóa chính | Mã máy |
| **2** | TenMay | Nvarchar(30) |  | Tên máy |
| **3** | TinhTrangMay | Nvarchar(50) |  | Tình trạng máy |
| **4** | ID\_gia | Varchar(5) |  | Mã đơn giá |
| **5** | HoatDong | Bit |  | Tình trạng hoạt động |

*Bảng 3-1: Danh sách các thuộc tính lớp Máy*

1. ***Lớp Dịch vụ***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_DV | Varchar(5) | Khóa chính | Mã dịch vụ |
| **2** | TenDV | Nvarchar(30) |  | Tên dịch vụ |
| **3** | AnhSP | Nvarchar(50) |  | Ảnh sản phẩm |
| **4** | GiaBan | Money |  | Giá bán |

*Bảng 3-2: Danh sách các thuộc tính lớp Dịch vụ*

1. ***Lớp Đơn giá***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_gia | Varchar(5) | Khóa chính | Mã giá máy |
| **2** | GiaMay | money |  | Giá máy |

*Bảng 3-3: Danh sách các thuộc tính lớp Đơn giá*

1. ***Lớp Thanh toán dịch vụ***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_TTDV | Varchar(5) | Khóa chính | Mã thanh toán dịch vụ |
| **2** | ID\_ThanhVien | Varchar(5) |  | Mã thành viên sử dụng dịch vụ |

*Bảng 3-4: Danh sách các thuộc tính lớp Thanh toán dịch vụ*

1. ***Lớp Đơn hàng***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | maDH | Varchar(5) | Khóa chính | Mã chi tiết thông tin dịch vụ |
| **2** | ID\_ThanhVien | Varchar(5) |  | Mã thành viên |
| **3** | ID\_DV | Varchar(5) |  | Mã dịch vụ |
| **4** | tongGia | Decimal (10,2) |  | Tổng giá |
| **5** | ghiChu | Varchar(255) |  | Ghi chú |
| **6** | ngayDatHang | Date |  | Ngày đặt hàng |

*Bảng 3-5: Danh sách các thuộc tính lớp Đơn hàng*

1. ***Lớp Thành viên***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_ThanhVien | Varchar(5) | Khóa chính | Mã thành viên |
| **2** | TenThanhVien | Nvarchar(30) |  | Tên thành viên |
| **3** | TenDN | Nvarchar(20) |  | Tên đăng nhập |
| **4** | Matkhau | Varchar(20) |  | Mật khẩu |
| **5** | SoTien | Money |  | Số tiền hiện có của thành viên |

*Bảng 3-6: Danh sách các thuộc tính lớp Thành viên*

1. ***Lớp Hóa Đơn***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên trường | Kiểu dữ liệu | Ràng buộc | Mô tả |
| **1** | ID\_HoaDon | Varchar(10) | Khóa chính | Mã hóa đơn |
| **2** | ID\_ThanhVien | Varchar(5) |  | Mã thành viên |
| **3** | ID\_TTDV | Varchar(5) |  | Mã thông tin dịch vụ |
| **4** | ID\_May | Varchar(5) |  | Mã máy |
| **5** | ThoiGianMo | Datetime |  | Thời gian mở máy |
| **6** | ThoiGianTat | Datetime |  | Thời gian tắt máy |
| **7** | TongTien | Money |  | Tổng tiền |
| **8** | TinhTrangHD | Nvarchar(50) |  | Tình trạng hóa đơn |

*Bảng 3-7: Danh sách các thuộc tính lớp Hóa đơn*

3.5 Thiết kế và đặc tả giao diện

A screenshot of a login page

Description automatically generated

###### ***Hình 5: Giao diện trang đăng nhập***

* Mục đích: để thực hiện việc đăng nhập của quản trị quản lí các chức năng và thành viên thì có chức năng của thành viên.
* Khi đăng nhập thành công:
* **Đối với quản trị:** Tên Admin sẽ được hiện thị góc trái dưới của trang ngược lại nếu đăng nhập sai thì hệ thống sẽ yêu cầu nhập lại.
* **Đối với thành viên:** Nếu là thành viên thì sẽ hiển thị tên khi đã được đăng kí thành viên.
* Khi hoàn tất quá trình đăng nhập:
* **Đối với quản trị:** Thì hệ thống sẽ truy cập trực tiếp vào trang quản trị.
* **Đối với thành viên:** Thì hệ thống sẽ trở lại trang chủ để cho bạn tìm kiếm hay order dịch vụ.

3.5.1 Giao diện trang Admin

3.5.1.1 Giao diện trang chủ quản trị

A screenshot of a computer

Description automatically generated

***Hình 6: Giao diện trang quản trị***

* Mục đích: hiển thị các chức năng của hệ thống như : quản lý máy, quản lý dịch vụ, quản lý thành viên, quản lý đơn giá, quản lý đơn hàng , order .

3.5.1.2 Giao diện trang quản lý thành viên

A screenshot of a computer

Description automatically generated

***Hình 7: Giao diện trang quản lý thành viên***

* Mục đích: thực hiện nhập, sửa, xóa thông tin thành viên.
* Ràng buộc: đăng nhập dưới quyền quản trị.
* Mô tả:
* Nhập đầy đủ thông tin và nhấn nút “Thêm” để lưu thông tin thành viên đăng kí.
* Sửa, xóa: chọn thành viên rồi nhấn nút “Sửa” hoặc “Xóa” để sửa hoặc xóa thông tin của thành viên.

3.5.1.3 Giao diện trang danh sách máy

A screenshot of a computer

Description automatically generated  
***Hình 8: Giao diện trang danh sách máy***

* Mục đích: thực hiện nhập, sửa, xóa thông tin máy.
* Ràng buộc: đăng nhập dưới quyền là người quản trị.

3.5.1.4 Giao diện trang danh sách đơn giá của máy

A screenshot of a computer

Description automatically generated

***Hình 9: Giao diện trang danh sách quản lý đơn giá máy***

* Mục đích: thực hiện nhập, sửa, xóa thông tin đơn giá nếu có thay đổi.
* Ràng buộc: đăng nhập dưới quyền là người quản trị.
* Mô tả:
* Thực hiện nhập đơn giá bạn nhập mã đơn giá, giá máy, sau đó nhấn nút “Thêm” hệ thống sẽ cập nhật thông tin mà bạn vừa nhập vào hệ thống.
* Thực hiện sửa, xóa đơn giá bạn chọn thông tin muốn sửa, xóa sau đó thông tin sẽ hiện lên cho bạn thực hiện. Khi hoàn tất nhấn nút “Xác nhận chỉnh sửa”, “Xác nhận xóa”.

3.5.1.5 Giao diện trang quản lý dịch vụ

A screenshot of a computer

Description automatically generated

***Hình 10: Giao diện trang quản lý dịch vụ***

A screenshot of a computer

Description automatically generated

***Hình 11: Giao diện trang tìm kiếm dịch vụ***

***A screenshot of a computer

Description automatically generated***

***Hình 12: Giao diện trang thêm dịch vụ***

***A screenshot of a computer

Description automatically generated***

***A screenshot of a computer

Description automatically generated***

***Hình 13: Giao diện trang sửa dịch vụ***

***A close up of a sandwich

Description automatically generated***

***Hình 14: Giao diện trang xóa dịch vụ***

* Mục đích: thực hiện việc thêm, sửa, xóa tìm kiếm dịch vụ..
* Ràng buộc: đăng nhập dưới quyền của người quản trị.
* Mô tả:
* Thêm: Thực hiện nhập dịch vụ bạn nhập mã dịch vụ, tên dịch vụ, giá, sau đó nhấn nút “thêm” hệ thống sẽ cập nhật thông tin mà bạn vừa nhập vào hệ thống.
* Sửa: Sửa thông tin dịch vụ.
* Xóa: Xóa dịch vụ.

3.5.1.6 Giao diện trang đơn hàng

A screenshot of a computer

Description automatically generated

***Hình 15: Giao diện trang đơn hàng***

* Mục đích: thực hiện việc thêm, sửa, xóa tìm kiếm đơn hàng
* Ràng buộc: đăng nhập dưới quyền là người quản trị.

3.5.1.7 Giao diện trang order

A screenshot of a computer

Description automatically generated

***Hình 16: Giao diện trang order***

* Mục đích: thực hiện việc order dịch vụ cho thành viên
* Ràng buộc: đăng nhập dưới quyền là người quản trị.

# CHƯƠNG 4: KẾT LUẬN

4.1 Kết quả đạt được của đề tài

* Xây dựng website Quản lý quán net.
* Giao diện đẹp, màu sắc hài hòa, dễ nhìn, dễ sử dụng.
* Xây dựng được các chức năng cần thiết cho hệ thống: thêm, sửa, xóa thông tin.

4.2 Hạn chế của đề tài

* Một số chức năng còn chưa tối ưu, thực hiện được.

4.3 Kết luận

Trong quá trình nguyên cứu và vận dụng các kiến thức đã được học. Chúng em đã hoàn thành phần mềm “Quản lý quán net” nhằm mục đích hoạt động kinh doanh và ôn lại những kiến thức đã học. Chúng em hy vọng sau hoạt động nhóm này là tiền đề để chúng em phát triển mạnh hơn về kiến thức và chất lượng phần mềm.

Trong quá trình hoàn thành phần mềm với kinh nghiệm thực tế chưa nhiều nên báo cáo không tránh khỏi những sai sót, rất mong sự góp ý của thầy.

Chúng em xin chân thành cảm ơn.

**TÀI LIỆU THAM KHẢO**

* Bùi Chí Thành , *Giáo trình Phát triển ứng dụng web.*
* Phạm Hữu Khang, Lập trình ASP.Net MVC. Hà Nội, Việt Nam: Nhà xuất bản Lao Động Xã Hội, 2005.
* Phạm Hữu Khanh, [Microsoft SQL Server 2008 - Quản Trị Cơ Sở Dữ Liệu](https://www.vinabook.com/microsoft-sql-server-2008-quan-tri-co-so-du-lieu-tap-1-p37708.html), NXB Lao động - Xã hội,2004.
* Đoàn Thiện Ngân, Phạm Hữu Khang, Lập trình ASP.NET MVC , Nhà xuất bản lao động xã hội, 2004.