BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH

BÁO CÁO THỰC TẬP TỐT NGHIỆP ĐẠI HỌC

Đề tài:

QUẢN LÝ ĐẶT VÉ XE KHÁCH ONLINE

Giáo viên hướng dẫn : TS. HỒ MẠNH TÀI
Sinh viên thực hiện : NGUYỄN VÕ HOÀNG DUY
Mã số sinh viên : N14DCCN046
Lớp : D14CQCP01-N
Khóa : 2014
Hệ : Đại học chính quy

TPHCM, Tháng 8/2018

LỜI CẢM ƠN

Trong quá trình thực hiện đề tài "***Quản lý đặ vé xe khách online***", em đã nhận được rất nhiều sự giúp đỡ, tạo điều kiện của tập thể Ban Giám Hiệu, Khoa Công Nghệ Thông Tin, giảng viên, cán bộ các phòng ban chức năng Học Viện Công Nghệ Bưu Chính Viễn Thông tại TP.HCM. Em xin bày tỏ lòng cảm ơn chân thành về sự giúp đỡ đó.

Em xin bày tỏ lòng biết ơn sâu sắc tới TS Hồ Mạnh Tài – thầy giáo trực tiếp hướng dẫn và chỉ bảo em hoàn thành đề tài này.

Sau cùng, em xin kính chúc quý Thầy Cô trong Khoa Công Nghệ Thông Tin và TS Hồ Mạnh Tài thật dồi dào sức khỏe, niềm tin để tiếp tục sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

Trân trọng!

TP. Hồ Chí Minh, ngày 3 tháng 8 năm 2018

Sinh viên thực hiện

Nguyễn Võ Hoàng Duy

MỤC LỤC

MỞ ĐẦU 1
CHƯƠNG I: TỔNG QUAN 2
1.1 Lý do chọn đề tài 2
1.2 Mục tiêu và những yêu cầu khi xây dựng hệ thống 2
1.2.1 Mục tiêu xây dựng hệ thống 2
1.2.2 Những yêu cầu đặt ra cho hệ thống 2
1.2.3 Hoạt động của hệ thống 2
1.2.4 Hoạt động của thành viên 2
1.2.5 Hoạt động của quản trị viên 2
1.3 Phương pháp giải quyết các vấn đề 2
CHƯƠNG II: ĐÁNH GIÁ HIỆN TRẠNG 3
2.1 Tìm hiểu về website 3
2.1.1 Các mô hình website 3
2.2 Thực trạng website đặt vé xe khách online 5
2.2.1 Tiềm năng 5
CHƯƠNG III. CƠ SỞ LÝ THUYẾT 6
3.1 Tổng quan về Node JS 6
3.1.1 Lập trình không đồng bộ 6
3.1.2 Đặc điểm môi trường 6
3.2 Tổng quan về React JS 8
3.2.1 React JS là gì? 8
3.2.2 Đặc điểm của React 8
3.2.3 Các tính năng chính 8
3.2.4 Đặc trưng 8
3.3 Hệ quản trị cơ sở dữ liệu MongoDB 10
3.3.1 Giới thiệu về NoSQL 10
3.3.2 Hệ quản trị cơ sở dữ liệu MongoDB 11
3.4 Tổng quan về ExpressJS 12
3.4.1 ExpressJS là gì? 12
3.4.2 Các tính năng chính 12
CHƯƠNG IV: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 13
4.1 Tổng quan ứng dụng 13
4.1.1 Đối tượng sử dụng ứng dụng 13
4.1.2 Các chức năng chính của hệ thống 13
4.2 Đặc tả Use Case 13
4.2.1 Đăng nhập đối với quản trị viên 13
4.2.2 Quản lý danh mục lịch trình 14

4.2.3 Quản lý lịch trình 15
4.2.4 Quản lý chi tiết lịch trình 15
4.2.5 Quản lý nơi lên xe 15
4.2.6 Quản lý đơn hàng 15
4.2.7 Quản lý danh sách liên hệ 15
4.2.8 Đặt vé xe 15
4.2.9 Thanh toán xe 15
4.2.10 Gửi liên hệ 16
4.2.11 Xem danh mục lịch trình 16
4.2.12 Xem tất cả lịch trình của nhà xe 16
4.3 Lược đồ Use Case 17
4.3.1 Lược đồ Use Case khách hàng 17
4.3.2 Lược đồ Use Case Admin 17
4.4 Lược đồ tuần tự 18
4.4.1 Lược đồ tuần tự khách hàng xem lịch trình 18
4.4.2 Lược đồ tuần tự khách hàng đặt vé 18
4.4.3 Lược đồ tuần tự khách hàng thanh toán sau khi đặt vé 19
4.4.4 Lược đồ tuần tự khách hàng gửi liên hệ đến hãng xe 19
4.4.5 Lược đồ tuần tự quản trị viên đăng nhập 20
4.4.6 Lược đồ tuần tự quản trị viên thêm / sửa thông tin 20
4.5 Thiết kế cơ sở dữ liệu 21
4.5.1 Tổng quan cơ sở dữ liệu 21
4.5.2 Chi tiết cơ sở dữ liệu 21
CHƯƠNG V: XÂY DỰNG WEBSITE ĐẶT VÉ XE KHÁCH ONLINE 25
5.1 Giới thiệu các công nghệ sử dụng 25
5.1.1 Các công nghệ sử dụng trong việc xây dựng website 25
5.1.2 Các bước thực hiện xây dựng website 25
5.2 Kết quả 25
5.3 Một số hình ảnh giao diện của ứng dụng và cách giải quyết 25
5.4 Hướng dẫn cài đặt 34
CHƯƠNG VI: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 37
6.1 Kết quả đạt được 37
6.2 Hạn chế 37
6.3 Hướng phát triển 37
TÀI LIỆU THAM KHẢO 38

THUẬT NGỮ, CHỮ VIẾT TẮT

IPC	:	InterProcess Communication.
HTTP	:	Hypertext Transfer Protocol .
MIME	:	Multpurpose Internet Mail Extension .
HTML	:	Hypertext Markup language.
CSS	:	Cascading Style Sheet.
DOM	:	Document Object Model.
JSON	:	JavaScript Object Notation.
MVC	:	Model - View – Controller.
SPA	:	Single Page Application.
NoSQL	:	Not Only SQL.
API	:	Application Programming Interface.
CSDL	:	Cơ sở dữ liệu.
CMT	:	Chứng minh thư.

DANH MỤC HÌNH

Hình 2.1: Mô hình Client/Server 3
Hình 3.1: NodeJS 6
Hình 3.2 Mô hình MVC 9
Hình 3.3 Single Page App 10
Hình 3.4 MongoDB 10
Hình 3.5 Minh họa Bộ sưu tập 11
Hình 4.1 Lược đồ Use Case khách hàng 17
Hình 4.2 Lược đồ use case admin 17
Hình 4.3 Lược đồ tuần tự khách hàng xem lịch trình 18
Hình 4.4: Lược đồ tuần tự khách hàng đặt vé 18
Hình 4.5 Lược đồ tuần tự khách hàng thanh toán 19
Hình 4.6 Lược đồ tuần tự khách hàng gửi liên hệ 19
Hình 4.7 Lược đồ tuần tự quản trị viên đăng nhập 20
Hình 4.8 Lược đồ tuần tự quản trị viên thêm/sửa thông tin 20
Hình 4.9 Database 21
Hình 5.1 Giao diện trang chủ 26
Hình 5.2 Giao diện trang đặt vé 26
Hình 5.3 Giao diện trang thanh toán 27
Hình 5.4 Giao diện thanh toán online 27
Hình 5.5 Giao diện mail gửi về khách hàng sau khi đặt vé 28
Hình 5.6 Giao diện trang dashboard Stripe 28
Hình 5.7 Giao diện trang tất cả lịch trình của hãng xe 29
Hình 5.8 Giao diện trang lịch trình theo tỉnh 29
Hình 5.9 Giao diện trang đăng nhập của quản trị viên 30
Hình 5.10 Hình danh sách danh mục lịch trình 30
Hình 5.11 Thêm danh mục lịch trình 31
Hình 5.12 Danh sách lịch trình 31
Hình 5.13 Thêm lịch trình 32
Hình 5.14 Danh sách chi tiết lịch trình 32
Hình 5.15 Thêm chi tiết lịch trình 33
Hình 5.16 Danh sách nơi lên xe 33
Hình 5.17 Danh sách đơn đặt vé 34
Hình 5.18 Cài đặt NodeJS 34
Hình 5.19 Cấu trúc thư mục chương trình 35
Hình 5.20 Cài đặt và chạy server 35
Hình 5.21 Cài đặt và chạy client 36

DANH MỤC BẢNG

Bảng 3.1: So sánh chấm điểm benchmark khi chạy trên Node Js và PHP.....7
Bảng 4.1: Bảng mô tả thuộc tính của thực thể group 22
Bảng 4.2: Bảng mô tả thuộc tính của thực thể user.....22
Bảng 4.3 Bảng mô tả thuộc tính của thực thể province.....22
Bảng 4.4 Bảng mô tả thuộc tính của thực thể category_schedule.....22
Bảng 4.5 Bảng mô tả thuộc tính của thực thể schedule.....23
Bảng 4.6 Bảng mô tả thuộc tính của thực thể schedule_detail.....23
Bảng 4.7 Bảng mô tả thuộc tính của thực thể route_departure 23
Bảng 4.8 Bảng mô tả thuộc tính của thực thể contact 24

LỜI MỞ ĐẦU

Ngày nay với tốc độ phát triển của khoa học kỹ thuật trên thế giới ngày càng mạnh mẽ. Cuộc cách mạng công nghệ thông tin đã và đang diễn ra trên hầu hết các nước tiên tiến trên thế giới. Cùng với sự phát triển không ngừng của công nghệ thông tin nhu cầu về ứng dụng phần mềm càng được tăng cao. Có thể nói phần mềm ngày nay đóng vai trò hết sức quan trọng trong cuộc sống hàng ngày của mỗi con người từ việc ăn gì, xem gì ở đâu cho đến vấn đề cổ phiếu tăng giá hay giảm, những vấn đề quan trọng của thế giới hay phần mềm giúp con người làm việc nhanh hơn, thậm chí thay con người làm việc.

Nhìn nhận vấn đề đầu tư và phát triển phần mềm ở thế giới nói chung và nước ta nói riêng đang rất phát triển. Đặc biệt là những phần mềm có thể đáp ứng được lượng thông tin cần thiết và được tích hợp trong những chiếc điện thoại thông minh nhỏ gọn có thể dùng bất cứ khi nào. Trong thời đại bận rộn hiện nay càng khó có thể có thời gian đi chọn những quyển báo hay cuốn sách hay để đọc.

Chính vì vậy: Đề tài ***"Quản lý đặt vé xe khách online"*** là một website được thiết kế và thực hiện nhằm hỗ trợ người dùng có thể đặt vé xe online thông qua website. Ở đó người dùng có thể biết được những vé xe nào hiện đang còn ở khung giờ nào để từ đó người dùng có thể tùy chỉnh đặt vé. Người quản trị cũng có thể quản lý thông tin của các tuyến xe, và xác nhận thông tin của những người đã đặt vé.

CHƯƠNG I: TỔNG QUAN

1.1 Lý do chọn đề tài

Do nhu cầu của việc đi lại bằng xe giường năm hiện nay khá cao. Nên thành ra việc quản lý đặt vé cũng trở nên khó khăn hơn với những người chủ xe. Chính vì những lý do như vậy em đã chọn đề tài này để làm.

1.2 Mục tiêu và những yêu cầu khi xây dựng hệ thống

1.2.1 Mục tiêu xây dựng hệ thống

- Đáp ứng được phần đặt vé cho khách hàng
- Người chủ xe có thể cập nhật quản lý thông tin của các tuyến xe, quản lý phần đặt vé của người dùng

1.2.2 Những yêu cầu đặt ra cho hệ thống

- Tạo ra sự thân thiện, tiện lợi, dễ sử dụng cho người dùng..

1.2.3 Hoạt động của hệ thống

Trong hoạt động của hệ thống có 2 đối tượng và có các hoạt động chính

- Hoạt động của người chủ xe.
- Hoạt động của người dùng bình thường.

1.2.4 Hoạt động của thành viên

Người dùng bình thường là đối tượng sử dụng hệ thống nhằm mục đích xem được những vé xe của các chuyến xe có còn chỗ để họ được đặt hay không. Nếu như còn người dùng có thể đặt vé thông qua trang web. Sau đó sẽ có người quản trị liên lạc lại.

1.2.5 Hoạt động của quản trị viên

Website có các chức năng của quản trị viên:

Tạo danh mục lịch trình, lịch trình, chi tiết lịch trình.

Quản lý thông tin các vé xe mà người dùng đặt để liên hệ

1.3 Phương pháp giải quyết các vấn đề

Từ các yêu cầu, phân tích hoạt động đặt ra của đề tài, em tiến hàng phân tích và đưa ra phương hướng giải quyết theo các bước như sau:

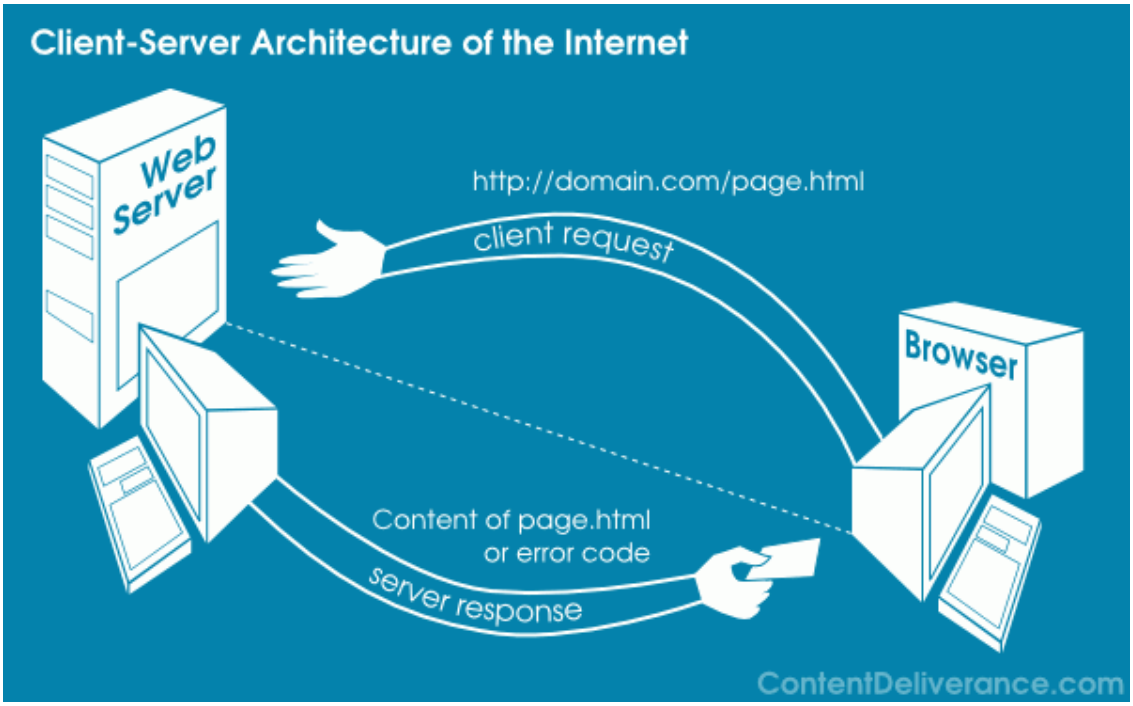
- Tìm hiểu các yêu cầu cần thiết của một web đặt vé xe khách, từ đó phân tích các chức năng sẽ đưa vào trong website
- Tìm hiểu về các ngôn ngữ lập trình, môi trường, các dạng cơ sở dữ liệu từ đó đưa ra giải pháp phù hợp để tiến hành xây dựng.
- Xây dựng kế về API, database, UI để làm rõ thiết kế của website.
- Tiến hành xây dựng website theo thiết kế.
- Tiến hành kiểm tra và chạy thử ứng dụng website.
- Thay đổi, bổ sung, khắc phục các sự cố, lỗi, từ đó đưa website vào sử dụng trong thực tiễn.

CHƯƠNG II: ĐÁNH GIÁ HIỆN TRẠNG

2.1 Tìm hiểu về website

2.1.1 Các mô hình website

Web được xây dựng và hoạt động theo mô hình Client/Server. Do đó, trước tiên ra sẽ xem về mô hình Client/Server.



Hình 2.1: Mô hình Client/Server

- ***Mô hình Client/Server***

- *Khái niệm*

Thuật ngữ Server được sử dụng cho những chương trình thi hành như một dịch vụ trên toàn mạng. Các chương trình Server này chấp nhận tất cả các yêu cầu hợp lệ đến từ mọi nơi trên mạng, sau đó thi hành dịch vụ và trả kết quả về nơi yêu cầu. Một máy chứa chương trình Server được gọi là máy chủ hay máy phục vụ (Server).

Môt máy tính được gọi là Client khi nó gửi các yêu cầu tới máy Server và được câu trả lời từ Server. Máy có chứa chương trình từ Client được gọi là máy khách (Client).

Chương trình Client và Server nói chuyện với nhau bằng các thông điệp (message) thông qua một cổng truyền liên tác IPC (InterProcess Communication) theo một chuẩn nói chuyện có sẵn được gọi là giao thức (Protocol).

Trên thực tế phân biệt giữa Client và Server chỉ là tương đối. Một Client có thể gửi yêu cầu đến một hoặc nhiều Server, Server không những đáp ứng yêu cầu của Client mà còn có thể gửi yêu câu tới Server khác, trong trường hợp này Server đã trở thành một Client.

Với mô hình trên ta nhận thấy mô hình Client/Server chỉ mang đặc điểm của phần mềm chứ không liên quan gì đến phần cứng mặc dù trên thực tế yêu cầu về phần cứng của máy Server là cao hơn nhiều so với một máy Client, do máy Server phải quản lý rất nhiều các yêu cầu từ các Client khác nhau trên mạng.

- ***Ưu và nhược điểm chính của mô hình Client/Server:***

- *Ưu điểm*

Trong mô hình Client/Server, dữ liệu và tài nguyên được chia sẻ giữa hai máy, tăng hiệu quả của hệ thống. Có thể nói rằng với mô hình Client/Server, mọi thứ dường như nằm trên bàn tay của người sử dụng, dữ liệu được truy nhập tử xa với nhiều dịch vụ đa dạng mà các mô hình cũ không có được.

- *Nhược điểm*

Sự tập trung cao độ trên Server là điểm yếu của mô hình Client/Server. Nhiều Server gặp sự cố mà không có dự phòng thì mạng có nguy cơ ngừng hoạt động. Hơn nữa, khi một lúc có nhiều yêu cầu đến Server thì sẽ sinh ra tắc nghẽn trên đường truyền.

- ***Mô hình Web Client/Server:***

Mô hình Client/Server ứng dụng vào các Web được gọi là mô hình Web Client/Server. Giao thức chuẩn được sử dụng để giao tiếp giữa Client và Server là giao thức HTTP (Hypertext Transfer Protocol).

Web Browser hay trình duyệt Web là công cụ truy xuất dữ liệu trên Internet, là phần mềm giao diện trực tiếp với người sử dụng. Nó có khả năng yêu cầu thông tin từ Web Server, sau đó Web Browser sẽ đợi thông tin trả lời từ Web Server và hiển thị thông tin cho người sử dụng. Thông tin hiển thị có thể được lưu trữ trên nhưng trang Web riêng được tạo ra trước đó khi yêu cầu là trang Web động.

Nhiều hãng phần mềm đã đưa ra các sản phẩm phần mềm Web Browser trên thị trường như Navigator của Netscape, Internet Explorer của Microsoft, Web Access của Novell.

Web Server đóng vai trò phục vụ trong dịch vụ Web, đáp ứng yêu cầu do các Web Browser gửi tới. Khi nhận được các yêu cầu, Web Server sẽ phân tích xem dữ liệu, thông tin yêu cầu là gì, thực hiện và kết quả về nơi yêu cầu nếu không có lỗi, ngược lại báo lỗi về cho trình duyệt.

Khi được thi hành, nó nạp vào bộ nhớ và đợi yêu cầu (Request) từ nơi khác đến. Các yêu cầu có thể đến từ một người sử dụng phần mềm Web Browser hoặc cũng có thể từ một Web Server khác. Khi nhận được yêu cầu, nó phân tích xem dữ liệu, thông tin mà người dùng yêu cầu là gì, sau đó nó lây tư liệu và gửi cho người dùng.

Tuỳ theo yêu cầu của khách, trang Web được Web Server trả về có thể là một trạng Web tĩnh (là một trang Web được tạo ra và lưu trữ từ trước khi có yêu cầu, Web Server chỉ đơn thuần lấy nó trả về) hoặc là một trang Web động (được Web Server tạo ra khi có yêu cầu và thông tin của khách).

Giao thức HTTP (Hypertext Transfer Protocol) là giao thức dùng trong việc trao đổi thông tin giữa trình duyệt Web và Web Server. Giao thức này hỗ trợ và truyền các thông tin dưới dạng nhiều dạng khác nhau như văn bản, hình ảnh, âm thanh,... Theo chuẩn MIME (Multpurpose Internet Mail Extension).

Hoạt động của giao thức:

- Ban đầu trình duyệt Web trên Client có một văn bản HTML và hiển thị lên màn hình với đầy đủ các mối liên kết.
- Khi người sử dụng bấm vào một liên kết nào đó trong văn bản trên thì trình duyệt sẽ sử dụng giao thức HTTP gửi yêu cầu lên mạng cho Web Server để truy nhập tới trang Web mới hay một dịch vụ nào đó.
- Sau khi nhận thông tin từ trình duyệt Server có thể tự xử lý thông tin hoặc gửi cho bộ phận khác (Data Server hoặc CGI,...) rồi chờ lây kết quả để gửi về cho Client. Web Server sử dụng giao thức HTTP để trả lời yêu cầu của trình duyệt bằng trang Web hoặc dữ liệu mà Client yêu cầu.
- Trình duyệt nhận và định dạng dữ liệu chuẩn của trang Web để hiển thị lên màn hình. Quá trình tiếp diễn liên tục như vậy được gọi là quá trình duyệt Web trên mạng.

2.2 Thực trạng website đặt vé xe khách online

2.2.1 Tiềm năng

Hiện nay, nhu cầu đi lại giữa các tỉnh ngày càng tăng cao. Vì thế website này được hình thành để việc giải quyết những khó khăn trên.

Thêm vào đó, sự bùng nổ này đang làm thay đổi thói quen sử dụng các phương tiện truyền thông của người Việt Nam.

Kết quả khảo sát gần đây của TNS Media cho thấy tại Việt Nam thời gian dành cho các phương tiện thông tin đại chúng truyền thống như truyền hình miễn phí, báo chí, video đang có xu hướng giảm dần, ngược lại thời gian dành cho các phương tiện truyền thông mới tăng lên, đặc biệt là tăng mạnh đối với Internet.

Chẳng hạn, tại TP.HCM thời gian truy cập Internet hàng ngày của người dân từ 31 phút vào năm 2007 đã tăng lên 36,8 phút vào năm 2008 và có khả năng sẽ tăng lên 60 phút vào năm 2010, trong khi thời gian bình quân xem truyền hình hàng ngày đã giảm từ 272,5 phút (2007) xuống còn 243,5 phút (2008).

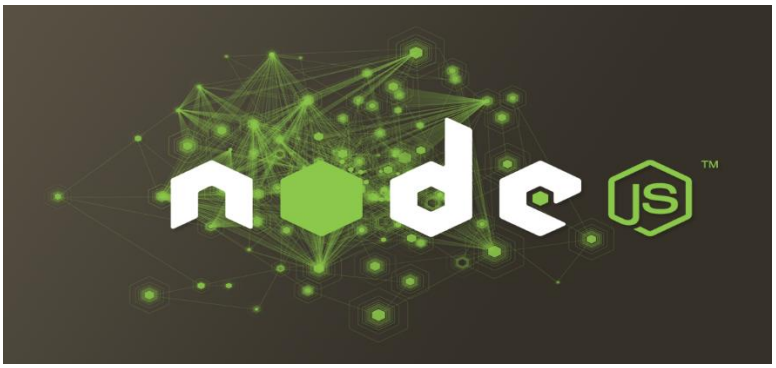
Theo Trung tâm Internet Việt Nam (VNNIC) , đến tháng 10/2008 đã có trên 20,5 triệu người sử dụng Internet, chiếm tới 24% dân số của cả nước.

Một nghiên cứu của Công ty IDC cũng cho biết trong vòng bốn năm tới số người sử dụng Internet tại Việt Nam sẽ lên tới con số 27 triệu, nói cách khác cứ ba người dân thì sẽ có một người sử dụng Internet.

Vì thế, website này sẽ tiếp cận được nhiều người trong thực trạng Internet hiện nay.

CHƯƠNG III. CƠ SỞ LÝ THUYẾT

3.1 Tổng quan về Node JS



Hình 3.1: NodeJS

Node.js là một hệ thống phần mềm được thiết kế để viết các ứng dụng internet có khả năng mở rộng, đặc biệt là máy chủ web. Chương trình được viết bằng JavaScript, sử dụng kỹ thật điều khển theo sự kiện, nhập/xuất không đồng bộ để tối tiểu tổng chi phí và tối đại khả năng mở rộng. Node.js bao gồm có V8 JavaScript engine của Google, libUV, và vài thư viện khác.

Node.js được tạo bởi Ryan Dahl từ năm 2009, và phát triển dưới sự bảo trợ của Joyent.

Mục tiêu ban đầu của Dahl là làm cho trang web có khả năng push như trong một số ứng dụng web như Gmail. Sau khi thử với vài ngôn ngữ Dahl chọn Javascript vì một API Nhập/Xuất không đầy đủ. Điều này cho phép anh có thể định nghĩa một quy ước Nhập/Xuất điểu khiển theo sự kiện, non-blocking.

3.1.1 Lập trình không đồng bộ

Là lợi thế nếu bạn đã quen thuộc với các phương pháp lập trình không đồng bộ. Tất cả các hàm trong Node.js là không đồng bộ. Do đó, tất cả chạy như các block thread thông thường thay vì chạy nền. Đây là điều quan trọng nhất để nhớ về Node.js. Ví dụ, nếu bạn đang đọc một tập tin trên hệ thống tập tin, bạn phải chỉ định một chức năng gọi lại đó là thực hiện khi đã hoàn thành các hoạt động đọc.

3.1.2 Đặc điểm môi trường

Node.js chỉ là môi trường – điều đó có nghĩa là bạn phải tự làm tất cả. Đó không phải là một server http ngầm định hoặc là bất cứ server nào khác. Điều này có thể là hơi khó hiểu với người mới, nhưng thành công thực sự của nó là đưa lại một hiệu năng đáng kinh ngạc . Một scrits có thể điều phối mọi kết nối với các client. Điều này làm sử dụng ít tài nguyên đưa đến một hiệu quả rất cao. Ví dụ sau về một ứng dụng Node.js.

```
var i, a, b, c, max;
max = 1000000000;
var d = Date.now();
```

```
for (i = 0; i < max; i++) {
    a = 1234 + 5678 + i;
    b = 1234 * 5678 + i;
    c = 1234 / 2 + i;
}
console.log(Date.now() - d);
```

Và đây là tương đương với mã PHP:

```
$a = null; $b = null;
$c = null; $i = null;
$max = 1000000000;
$start = microtime(true);
for ($i = 0; $i < $max; $i++) {
    $a = 1234 + 5678 + $i;
    $b = 1234 * 5678 + $i;
    $c = 1234 / 2 + $i;
}
var_dump(microtime(true) - $start);
```

Và giờ ta xem chấm điểm benchmark cho hai đoạn code trên khi chạy trên hai môi trường khác nhau:

Bảng 3.1: So sánh chấm điểm benchmark khi chạy trên Node Js và PHP

Number of iterations	Node.js	PHP
100	2.00	0.14
10’000	3.00	10.53
1’000’000	15.00	1119.24
10’000’000	143.00	10621.46
1’000’000’000	11118.00	1036272.19

Tôi thực hiện chạy hai đoạn code trên từ command line (console command) nên không có trễ thao tác thực thi. Tôi chạy từng thử nghiệm 10 lần và lấy kết quả trung bình. PHP nhanh hơn trong các lần chạy với số lượng nhỏ vòng lặp. Nhưng vấn đề thay đổi khi số lượng vòng lặp tăng lên, số lần xử lý tăng lên thì PHP chạy chậm hơn rất nhiều trong khi Node.js có tốc độ đáng kinh ngạc. Sau tất cả thao tác, PHP chậm hơn 93% so với Node.js.

3.2 Tổng quan về React JS

3.2.1 React JS là gì?

React.js là một thư viện Javascript đang nổi lên trong những năm gần đây với xu hướng Single Page Application. Trong khi những framework khác cố gắng hướng đến một mô hình MVC hoàn thiện thì React nổi bật với sự đơn giản và dễ dàng phối hợp với những thư viện Javascript khác. Nếu như AngularJS là một Framework cho phép nhúng code javasscript trong code html thông qua các attribute như ng-model, ng-repeat...thì với react là một library cho phép nhúng code html trong code javascript nhờ vào JSX, bạn có thể dễ dàng lồng các đoạn HTML vào trong JS.Tích hợp giữa javascript và HTML vào trong JSX làm cho các component dễ hiểu hơn.

3.2.2 Đặc điểm của React

- Phát triển dựa trên Javascript
- Khả năng tương thích cao, tự động xử lý mã javascript để phù hợp vứi mỗi trình duyệt.
- Mã nguồn mở, miễn phí hoàn toàn và được sủ dụng rộng rãi.

3.2.3 Các tính năng chính

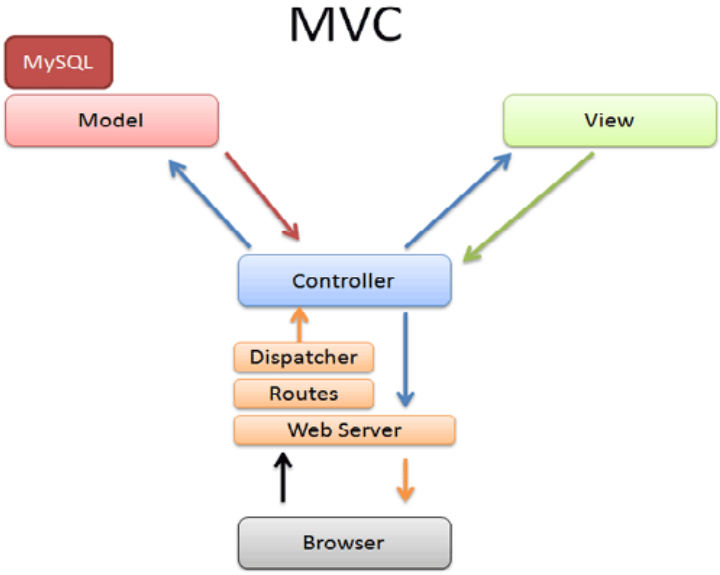
- Prop : Props là properties của một component . Chúng ta có thể thay đổi giá trị props của một component bằng cách truyền dữ liệu từ ngoài vào. Khi một props được truyền vào component thì giá trị của nó là không thay đổi .
- Component : Về mặt khái niệm components giống như các function của JavaScript . Chúng có các đầu vào gọi là "props" và trả về các yếu tố phản hồi mô tả điều gì sẽ xuất hiện trên màn hình . Các component có thể tham khảo các component khác trong đầu ra của chúng.
- State : State biểu hiện trạng thái của component, state là private và chỉ thay đổi bên trong bản thân component đó
- Redux : là một thư viện Javascript giúp tạo ra thành một lớp quản lý trạng thái của ứng dụng
- JSX: Đây đơn giản là một syntax extension của Javascript. Với nó bạn có thể viết Javascript với những tag giống như XML (XML-like). Về bản chất, các tag thực sự là những lời gọi hàm, sẽ được chuyển đổi trong React code và end up dưới dạng HTML và Javascript trong cây DOM.

3.2.4 Đặc trưng

Có 3 đặc điểm chính để nói về React đó là:

- Just the UI Chỉ sử dụng React để xây dựng giao diện người dùng, hiểu đơn giản thì trong mô hình MVC, React được coi như là phần V. Vì vậy nếu bạn muốn xây dựng một trang web hoàn chỉnh thì không thể dùng mỗi React được, mà nó cần thêm phần M và C nữa
- Virtual DOM Công nghệ DOM ảo giúp tăng hiệu năng cho ứng dụng.
- Data Flow React sử dụng luồng dữ liệu 1 chiều giúp kiểm soát dữ liệu dễ dàng hơn.

3.2.4.1 Mô hình MVC



Hình 3.2 Mô hình MVC

Mô hình MVC (Model - View - Controller) là một kiến trúc phần mềm hay mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. Nó giúp cho các developer tách ứng dụng của họ ra 3 thành phần khác nhau Model, View và Controller. Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần khác.

Mô hình MVC được giới thiệu từ những năm 70 như một phần của Smalltalk, nhưng đối với nền tảng web, thì nó mới được thịnh hành gần đây.

Ý tưởng đằng sau MVC là để chia rõ 3 thành phần chính là model(xử lý, truy xuất database), view(giao diện), và controller(điều hướng yêu cầu từ người dùng).

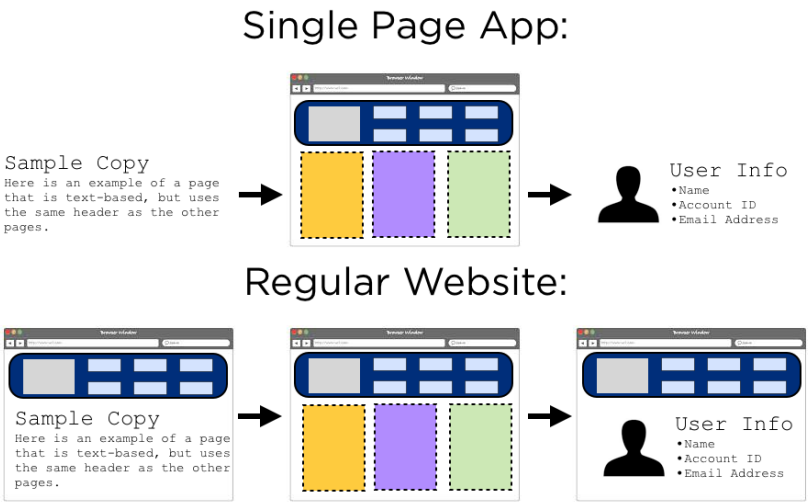
MVC thể hiện tính chuyên nghiệp trong lập trình, phân tích thiết kế. Do được chia thành các thành phần độc lập nên giúp phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp, bảo trì. Đối với Angular, View sẽ là DOM, Controller là các lớp JavaScript, còn Model sẽ là dữ liệu được lưu ở thuộc tính của các đối tượng trong JS.

Sau khi chứng kiến nhiều tranh luận về MV*, một tác giả của Angular đã tuyên bố AngularJS là một MVW framework (Model – View – Whatever, trong đó Whatever là viết tắt của whatever works for you).

3.2.4.2 SPA – Single Page Application

Một single page application hay còn được gọi là single page interface, là một web app hay website hiển thị vừa vặn trên một mặt của trang web với mục đích giúp người dùng có trải nghiệm giống như đang dùng ứng dụng trên desktop.

Là ứng dụng chạy bên trong trình duyệt, không yêu cầu phải tải lại toàn bộ trang web mỗi lần sử dụng.



Hình 3.3 Single Page App

3.3 Hệ quản trị cơ sở dữ liệu MongoDB

Hình 3.4 MongoDB

3.3.1 Giới thiệu về NoSQL

Với hầu hết các thời kỳ web, Hệ quản trị cơ sở dữ liệu quan hệ dựa trên SQL đã thống trị hầu hết các hệ Quản trị Cơ sở dữ liệu. Tuy nhiên, thời gian gần đây, một cách tiếp cận mới đã bắt đầu biết đến là NoSQL, tạo ra sự thay thế cho các hệ quản trị cơ sở dữ liệu quan hệ truyền thống.

NoSQL còn có nghĩa là Non-Relational - không ràng buộc. Tuy nhiên, thuật ngữ đó ít phổ dụng hơn và ngày nay người ta thường dịch NoSQL thành Not Only SQL - Không chỉ SQL. NoSQL ám chỉ đến những cơ sở dữ liệu không dùng mô hình dữ liệu quan hệ để quản lý dữ liệu trong lĩnh vực phần mềm

Để tìm hiểu sâu hơn về các CSDL hiện đại NoSQL, chúng ta đi nghiên cứu chi tiết CSDL đặc trưng là MongoDB.

3.3.2 Hệ quản trị cơ sở dữ liệu MongoDB

Trong những gương mặt góp phần làm suy tàn đế chế SQL thì MongoDB nổi lên là một CSDL đáng tin cậy và dễ dùng nhất. Mongo viết bằng C++. Nó thích hợp cho các ứng dụng tầm trung trở lên. Nếu tỉ lệ lượng dữ liệu ghi vào CSDL của ứng dụng lớn hơn lượng đọc thì đây càng là lựa chọn hợp lý.

MongoDB là một CSDL có khả năng mở rộng, hiệu suất cao, mã nguồn mở và hướng văn bản.

Trước khi đi vào tìm hiểu kỹ hơn về MongoDB, chúng ta làm quen với một số khái niệm cơ bản của MongoDB:

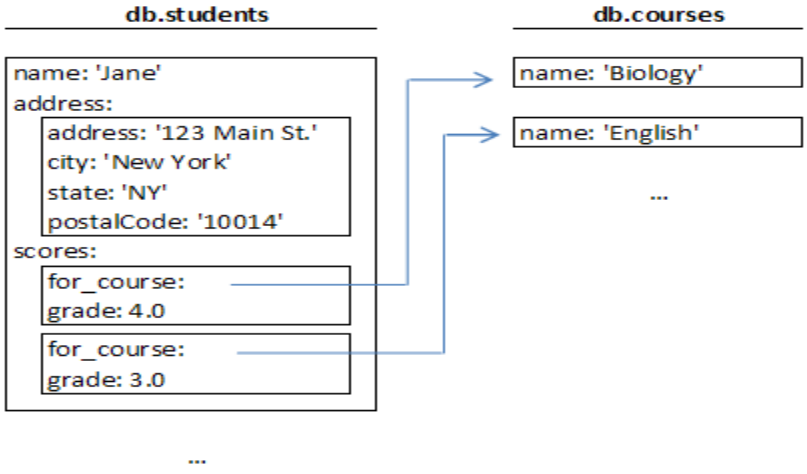
- Văn bản (Document) là đơn vị cơ bản của dữ liệu trong MongoDB, nó tương đương với một dòng trong CSDL quan hệ.
- Bộ sưu tập (Collection) có thể được coi như tương đương với một bảng.
- MongoDB có thể lưu trữ nhiều CSDL độc lập, mỗi CSDL này có các bộ sưu tập và điều khoản riêng của mình.
- MongoDB đi kèm với một trình tiện ích JavaScript đơn giản nhưng mạnh mẽ, nó hữu ích trong quản trị và thao tác dữ liệu.
- Mỗi văn bản có một khóa đặc biệt, đó là "_id", nó là duy nhất trong bộ sưu tập của văn bản.

3.3.2.1 Thiết kế lược đồ

Với MongoDB, chúng ta ít phải "chuẩn hóa" hơn so với khi làm việc với lược đồ quan hệ vì trong MongoDB không có khái niệm liên kết (join). Nói chung, với mỗi đối tượng (object) mức cao nhất, ta sẽ có một bộ sưu tập (collection) dữ liệu.

Một bộ sưu tập không phải cho tất cả các lớp (class), thay vào đó, các đối tượng sẽ được nhúng vào đó.

Hình 3.5 minh họa có 2 bộ sưu tập: students và courses. Các văn bản student được nhúng văn bản address và văn bản score. Trong đó, văn bản Score được tham chiếu đến Courses.



Hình 3.5 Minh họa Bộ sưu tập

So sánh với lược đồ quan hệ: ta cần lưu Score vào bảng riêng và dùng khóa ngoài liên kết với Student.

3.3.2.2 Truy vấn

Một trong những tính năng tốt nhất của MongoDB là hỗ trợ truy vấn động (ad hoc). Hệ thống hỗ trợ truy vấn động không yêu cầu bất cứ chỉ mục nào để tìm dữ liệu. Người dùng có thể tìm dữ liệu với việc sử dụng bất kỳ tiêu chuẩn nào. Với CSDL quan hệ, truy vấn động là chuẩn hóa.

*** Đối tượng biểu thức truy vấn**

MongoDB hỗ trợ một số các đối tượng truy vấn để lấy dữ liệu. Ví dụ, giả sử chúng ta muốn sử dụng trình MongoDB để trả về mọi văn bản trong bộ sưu tập *users.* Truy vấn sẽ được viết như sau:

```
db.users.find({})
```

Trong trường hợp này, lựa chọn (điều kiện) của chúng ta là trống, nó phù hợp với mọi văn bản trong bộ sưu tập. Chúng ta xem thêm một số ví dụ:

```
db.users.find({'last_name': 'Smith'})
```

Ở đây, lựa chọn của chúng ta là tất cả các văn bản mà thuộc tính *last_name* là *Smith.*

3.4 Tổng quan về ExpressJS

3.4.1 ExpressJS là gì?

Express là một framework nhỏ và tiện ích để xây dựng các ứng dụng web, cung cấp một lượng lớn của tính năng mạnh mẽ để phát triển các ứng dụng web và mobile. Nó rất dễ dàng để phát triển các ứng dụng nhanh dựa trên Node.js cho các ứng dụng Web. Dưới đây là các tính năng cơ bản của Express framework.

3.4.2 Các tính năng chính

- Framework giúp cho việc phát triển ứng dụng được rút ngắn đi rất nhiều. Cũng như các framework dựa trên những ngôn ngữ khác như Rails (Ruby); Django (Python); Laravel, CakePHP (PHP)... Express được xây dựng dựa trên Node.js. Vậy nó có ưu điểm gì để ta lựa chọn cho việc phát triển ứng dụng.
- Express hỗ trợ việc phát triển ứng dụng theo mô hình MVC, mô hình phổ biến cho việc lập trình web hiện nay.
- Cho phép định nghĩa Middleware hỗ trợ cho việc tổ chức và tái sử dụng code.
- Định nghĩa routes và các request method đến server một cách dễ dàng.
- Hỗ trợ REST API.
- Và còn rất nhiều thứ mà framework này hỗ trợ, còn bây giờ hãy thử cài đặt và xây dựng một ứng dụng đơn giản nhé.

⇨ Với ExpressJS, việc phát triển ứng dụng trở nên dễ dàng hơn.

CHƯƠNG IV: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

4.1 Tổng quan ứng dụng

4.1.1 Đối tượng sử dụng ứng dụng

❖ **Người dùng**

Người dùng là đối tượng sử dụng website để thực hiện các yêu cầu như là xem thông tin tuyến xe, các vé đã đặt, còn trống để đặt vé.

❖ **Quản lý hệ thống**

Quản lý hệ thống là đối tượng sử dụng website để quản lý các chuyến xe, cũng như là quản lý thông tin khách hàng đã đặt vé ở bên mình

4.1.2 Các chức năng chính của hệ thống

❖ **Đối với người dùng**

Hệ thống cần có các chức năng để hỗ trợ và giải quyết các nhu cầu của người dùng như sau:

- Hiển thị các chuyến xe đang có của nhà xe .
- Xem thông tin về vé đặt của các chuyến xe.
- Đặt vé
- Liên hệ lại nhà xe

❖ **Đối với quản lý hệ thống**

Hệ thống cần có các chức năng để hỗ trợ việc quản lý của nhà xe như sau:

- Quản lý thông tin lịch trình của các chuyến xe
- Quản lý thông tin của các chuyến xe.
- Quản lý thông tin của các trạm dừng
- Quản lý thông tin của các liên hệ
- Quản lý thông tin của các khách hàng đã đặt vé xe

4.2 Đặc tả Use Case

4.2.1 Đăng nhập đối với quản trị viên

❖ Tác nhân: quản trị viên

❖ Mô tả: Cho phép người quản trị đăng nhập vào hệ thống.

❖ Dòng sự kiện chính:

- Quản trị viên khi muốn vào những trang quản trị sẽ được đẩy qua trang đăng nhập này
- Form đăng nhập hiển thị
- Nhập email và mật khẩu vào
- Hệ thống sẽ kiểm tra email và mật khẩu
- Nếu hệ thống kiểm tra email và mật khẩu đúng thì sẽ đăng nhập thành công còn nếu sai chuyển sang luồng nhánh A1.
- Use case kết thúc.

❖ Dòng sự kiện rẽ nhánh A1.

- Quản trị viên đăng nhập không thành công.
- Hệ thống thông báo quá trình đăng nhập không thành công.
- Hệ thống yêu cầu nhập lại tên và mật khẩu
- Nếu khách xem đồng ý thì quay về bước 3 của luồng sự kiện chính, nếu không đồng ý thì use case kết thúc.

4.2.2 Quản lý danh mục lịch trình

❖ Tác nhân: quản trị viên

❖ Mô tả: Cho phép người quản trị thêm, xóa, thay đổi thông tin của danh mục lịch trình.

❖ Tiền điều kiện: người quản trị phải đăng nhập vào hệ thống.

❖ Dòng sự kiện chính:

- Người quản lý chọn kiểu tác động lên danh mục lịch trình: thêm danh mục lịch trình, thay đổi thông tin danh mục lịch trình, xóa danh mục lịch trình.
- A thêm danh mục lịch trình

+ Hệ thống hiển thị form nhập thông tin danh mục lịch trình, người quản trị nhập thông tin danh mục lịch trình vào.

+ Nhấn nút lưu thông tin.

+ Nếu nhập thành công thực hiện bước tiếp theo dưới đây. Nếu sai thực hiện luồng sự kiện rẽ nhánh A1.

+ Lưu thông tin danh mục lịch trình vào database.

- B thay đổi thông tin danh mục lịch trình

+ Hệ thống hiển thị form sửa thông tin danh mục lịch trình

+ Người quản trị nhập thông tin cần thay đổi.

+ Nhấn nút lưu thông tin.

+ Nếu việc thay đổi thành công thì thực hiện bước tiếp theo. Nếu sai thực hiện luồn sự kiện rẽ nhánh A1.

+ Lưu thông tin danh mục lịch trình vào database.

- C xóa danh mục lịch trình.

+ Người quản trị chọn danh mục lịch trình cần xóa.

+ Nhấn nút xóa sẽ thực hiện loại bỏ danh mục lịch trình (Mặc dù là xóa nhưng mặc định sẽ update status của danh mục đó lại thành inactive. Vì ở ngoài trang chủ chỉ có những danh mục nào active mới được load ra ngoài)

+ Thông báo danh mục lịch trình đã được xóa.

+ Hệ thống cập nhật lại danh sách danh mục lịch trình.

❖ Dòng sự kiện rẽ nhánh A1.

- Hệ thống báo việc nhập dữ liệu không hợp lệ.
- Người quản trị nhập lại thông tin.
- Quay lại bước 3 của luồng sự kiện chính

4.2.3 Quản lý lịch trình

(Tương tự như quản lý danh mục lịch trình)

4.2.4 Quản lý chi tiết lịch trình

(Tương tự như quản lý danh mục lịch trình)

4.2.5 Quản lý nơi lên xe

(Tương tự như quản lý danh mục lịch trình)

4.2.6 Quản lý đơn hàng

(Tương tự như quản lý danh mục lịch trình. Nhưng ở đây chỉ là xem danh sách. Chứ không có thêm, xóa, sửa)

4.2.7 Quản lý danh sách liên hệ

(Tương tự như quản lý danh mục lịch trình. Nhưng ở đây chỉ là xem danh sách. Chứ không có thêm, xóa, sửa)

4.2.8 Đặt vé xe

❖ Tác nhân: khách hàng

❖ Mô tả: Cho phép khách hàng tiến hành đặt vé xe .

❖ Tiền điều kiện: Phải biết là của chuyến xe nào, ngày nào.

❖ Dòng sự kiện chính:

- Hệ thống hiển thị form cho phép khách hàng lựa chọn khung giờ lên xe, nơi lên xe. Và phía bên phải là một màn hình form để chọn chính xác vị trí ngồi
- Khách hàng nhấn vào nút đặt vé
- Hệ thống kiểm tra thông tin đầu vào, nếu thất bại chuyển hướng đến nhánh A1
- Kiểm tra thành công, hệ thống hiển thị giao diện thanh toán.
- Use case kết thúc.

❖ Dòng sự kiện rẽ nhánh A1.

- Hệ thống yêu cầu phải lựa chọn vị trí ngồi.
- Quay lại bước 2.
- Khách hàng không chọn vị trí ngồi, kết thúc giao dịch.

4.2.9 Thanh toán xe

❖ Tác nhân: khách hàng

❖ Mô tả: Cho phép khách hàng tiến hành thanh toán, đặt vé xe đã chọn

❖ Tiền điều kiện: Vé xe cần phải được đặt

❖ Dòng sự kiện chính:

- Hệ thống hiển thị form cho phép khách hàng điền thông tin liên hệ cá nhân, phương thức thanh toán
- Khách hàng nhấn vào nút thanh toán.
- Hệ thống kiểm tra thông tin đầu vào, nếu thất bại chuyển hướng đến nhánh A1
- Kiểm tra thành công, hệ thống hiển thị lại giao diện trang chủ, đồng thời sẽ gửi mail xác nhận lại cho khách hàng
- Use case kết thúc.

❖ Dòng sự kiện rẽ nhánh A1.

- Hệ thống yêu cầu khách hàng nhập đầy đủ thông tin.
- Quay lại bước 2.
- Khách hàng không điền thông tin, kết thúc giao dịch.

4.2.10 Gửi liên hệ

❖ Tác nhân: khách hàng

❖ Mô tả: Cho phép khách hàng gửi những liên hệ, thắc mắc đến nhà xe

❖ Dòng sự kiện chính:

- Hệ thống hiển thị form cho phép khách hàng điền thông tin liên hệ cá nhân, chủ đề liên hệ, nội dung liên hệ
- Khách hàng nhấn vào nút gửi liên hệ.
- Hệ thống kiểm tra thông tin đầu vào, nếu thất bại chuyển hướng đến nhánh A1
- Kiểm tra thành công, hệ thống sẽ thông báo gửi liên hệ thành công, đồng thời sẽ gửi mail xác nhận lại cho khách hàng
- Use case kết thúc.

❖ Dòng sự kiện rẽ nhánh A1.

- Hệ thống yêu cầu khách hàng nhập đầy đủ thông tin.
- Quay lại bước 2.
- Khách hàng không điền thông tin, kết thúc use case.

4.2.11 Xem danh mục lịch trình

❖ Tác nhân: khách hàng

❖ Mô tả: Cho phép khách hàng xem thông tin lịch trình theo tỉnh của nhà xe

❖ Dòng sự kiện chính:

- Khách hàng lựa chọn chức năng xem lịch trình theo từng tỉnh, thành phố.
- Hệ thống sẽ hiển thị thông tin chi tiết lịch trình của từng tỉnh cho khách hàng.
- Khách hàng xem thông tin chi tiết lịch trình của từng tỉnh
- Use case kết thúc

❖ Dòng sự kiện khác.

4.2.12 Xem tất cả lịch trình của nhà xe

❖ Tác nhân: khách hàng

❖ Mô tả: Cho phép khách hàng xem thông tin tất cả các lịch trình đang có của nhà xe

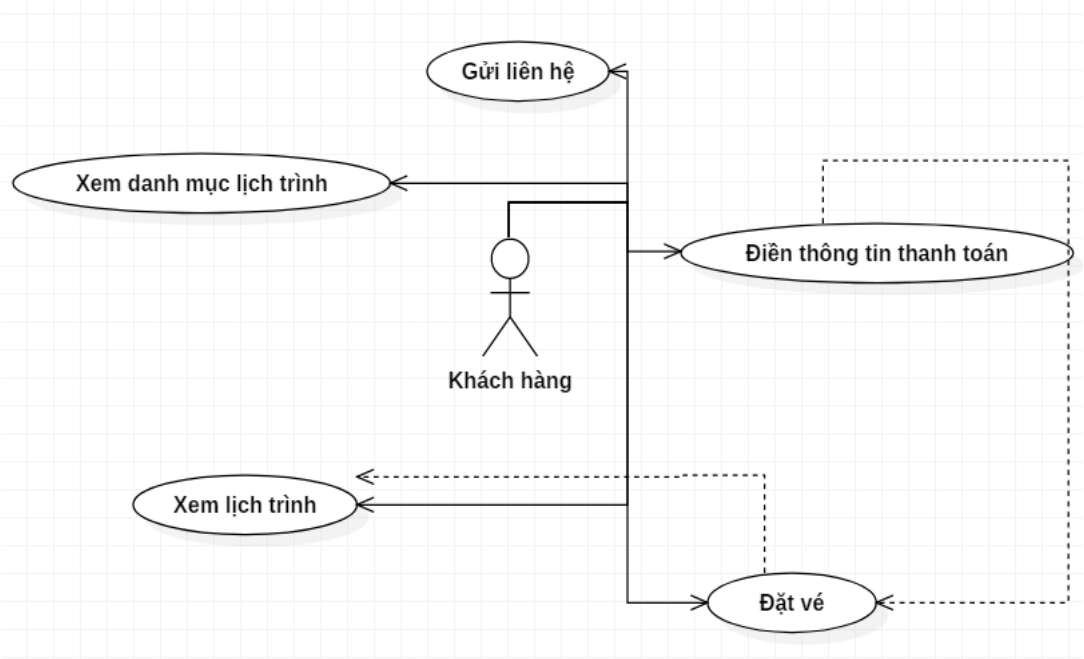
❖ Dòng sự kiện chính:

- Khách hàng lựa chọn chức năng xem lịch trình.
- Hệ thống sẽ hiển thị thông tin chi tiết lịch trình của từng tỉnh cho khách hàng.
- Khách hàng xem thông tin tất cả các lịch trình của nhà xe
- Use case kết thúc

❖ Dòng sự kiện khác.

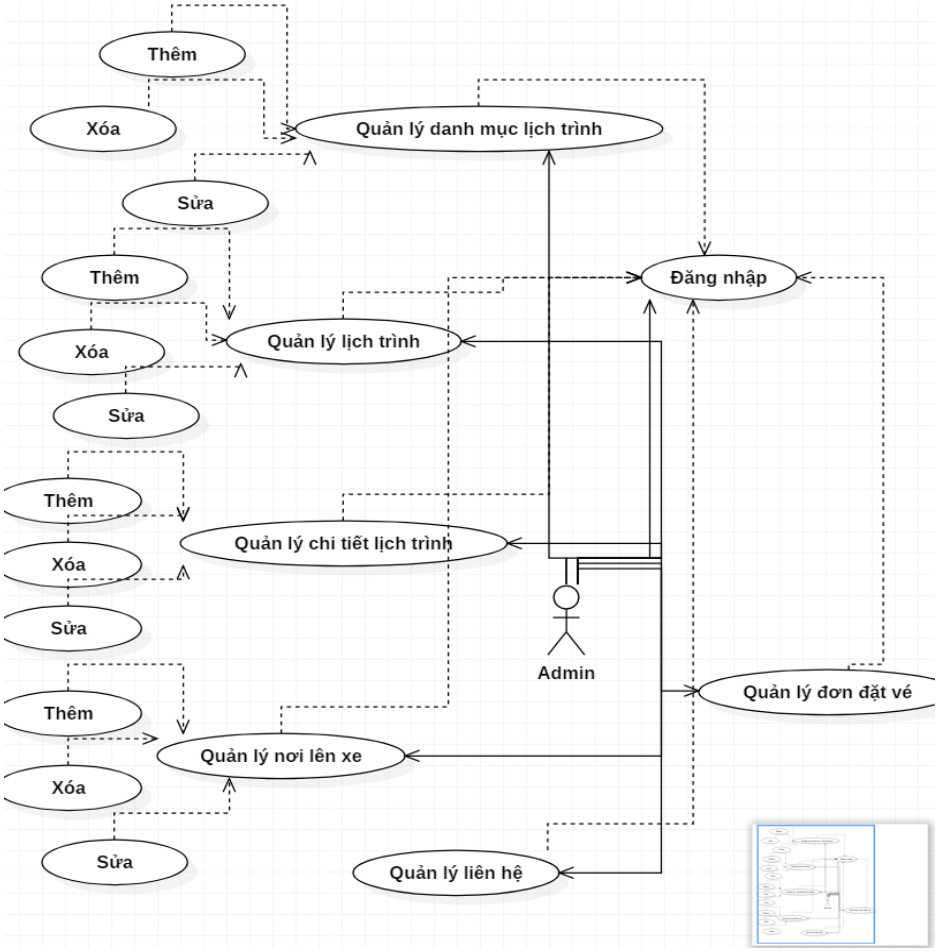
4.3 Lược đồ Use Case

4.3.1 Lược đồ Use Case khách hàng



Hình 4.1 Lược đồ Use Case khách hàng

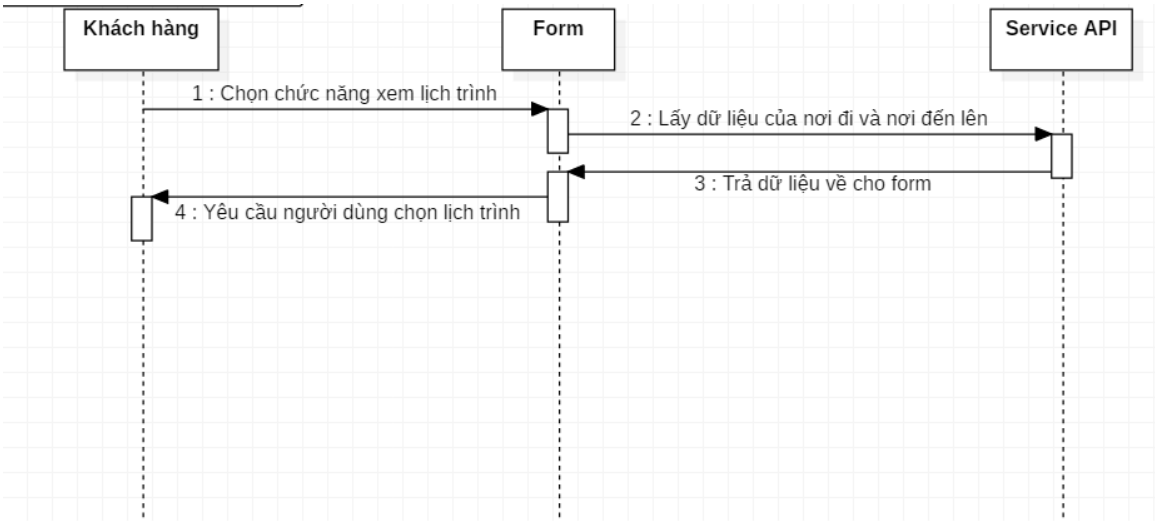
4.3.2 Lược đồ Use Case Admin



Hình 4.2 Lược đồ use case admin

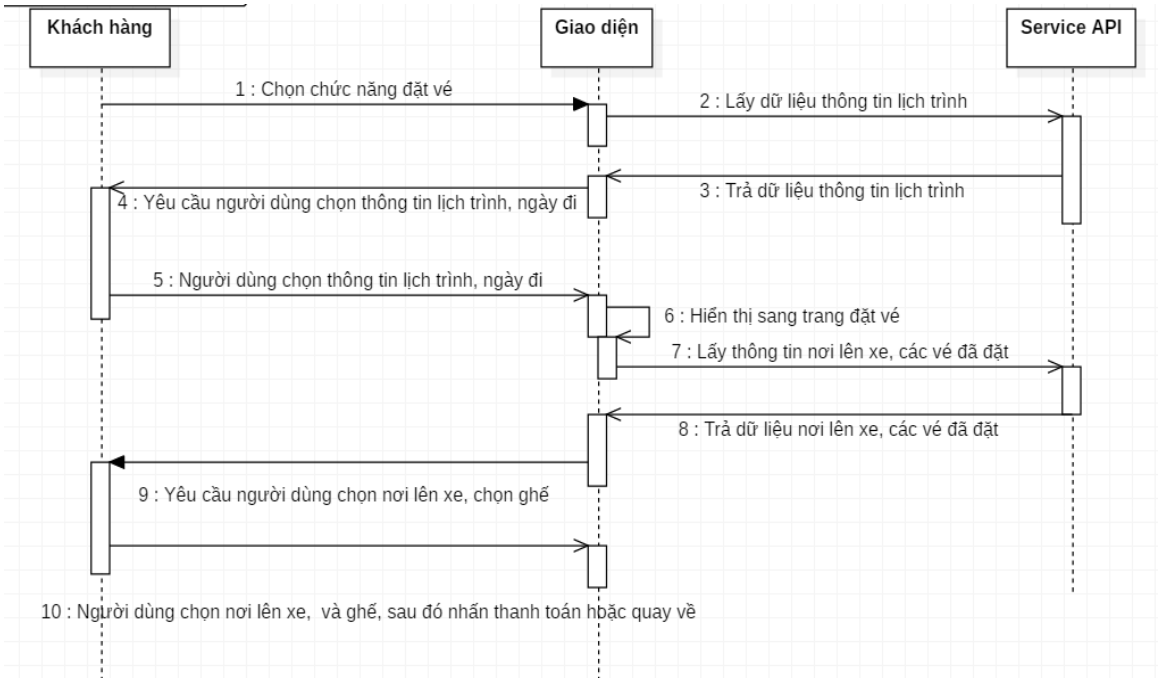
4.4 Lược đồ tuần tự

4.4.1 Lược đồ tuần tự khách hàng xem lịch trình



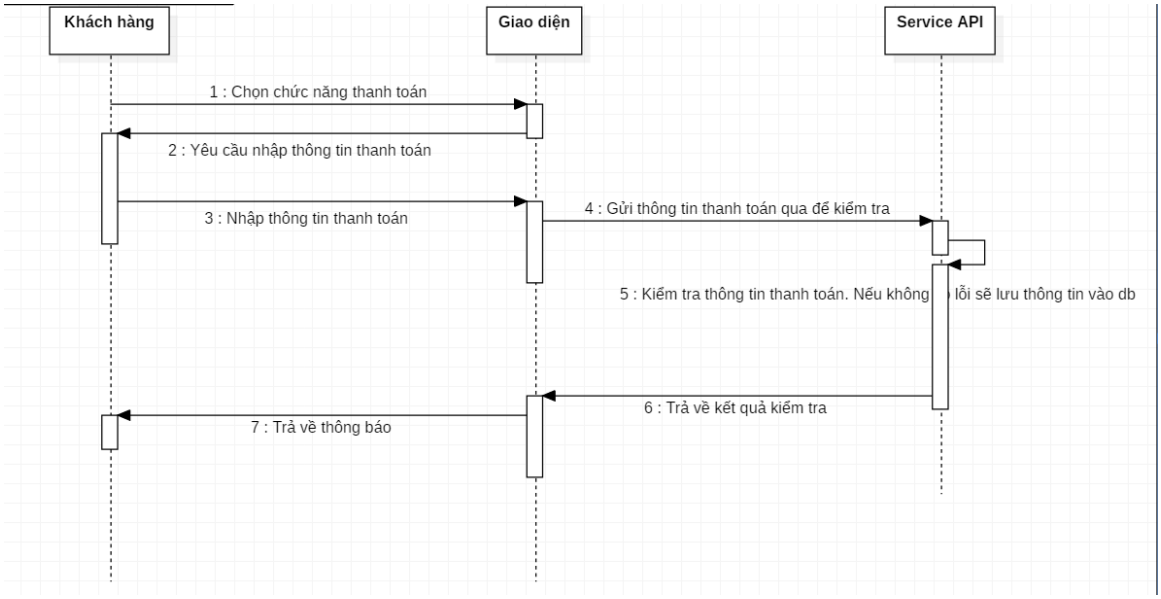
Hình 4.3 Lược đồ tuần tự khách hàng xem lịch trình

4.4.2 Lược đồ tuần tự khách hàng đặt vé



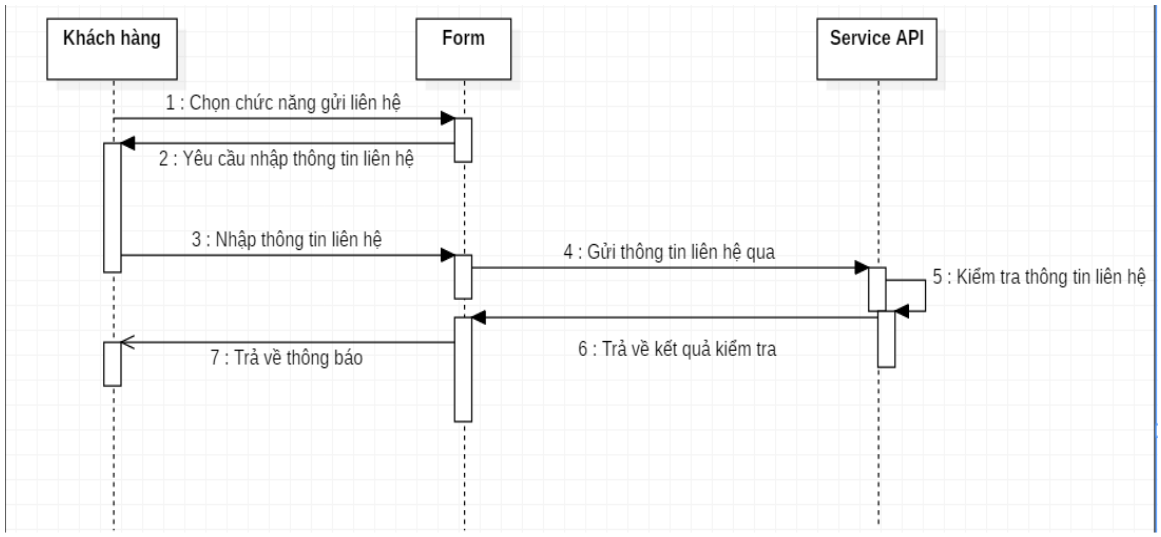
Hình 4.4: Lược đồ tuần tự khách hàng đặt vé

4.4.3 Lược đồ tuần tự khách hàng thanh toán sau khi đặt vé



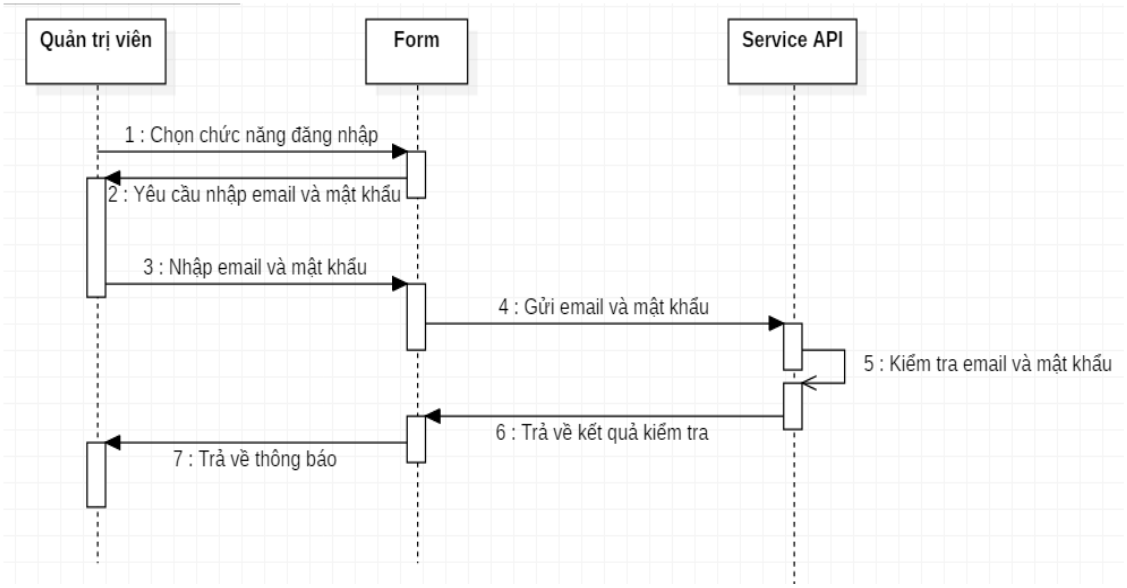
Hình 4.5 Lược đồ tuần tự khách hàng thanh toán

4.4.4 Lược đồ tuần tự khách hàng gửi liên hệ đến hãng xe



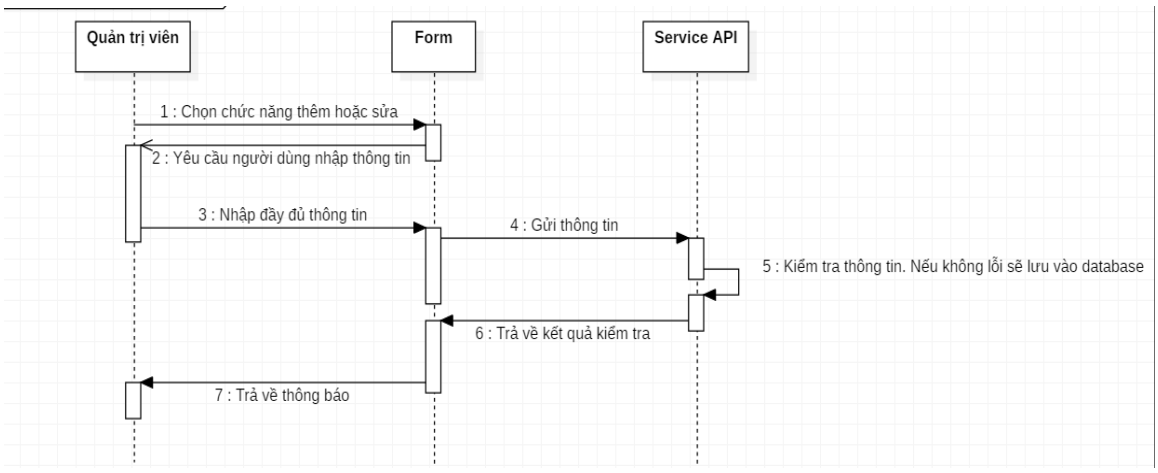
Hình 4.6 Lược đồ tuần tự khách hàng gửi liên hệ

4.4.5 Lược đồ tuần tự quản trị viên đăng nhập



Hình 4.7 Lược đồ tuần tự quản trị viên đăng nhập

4.4.6 Lược đồ tuần tự quản trị viên thêm / sửa thông tin



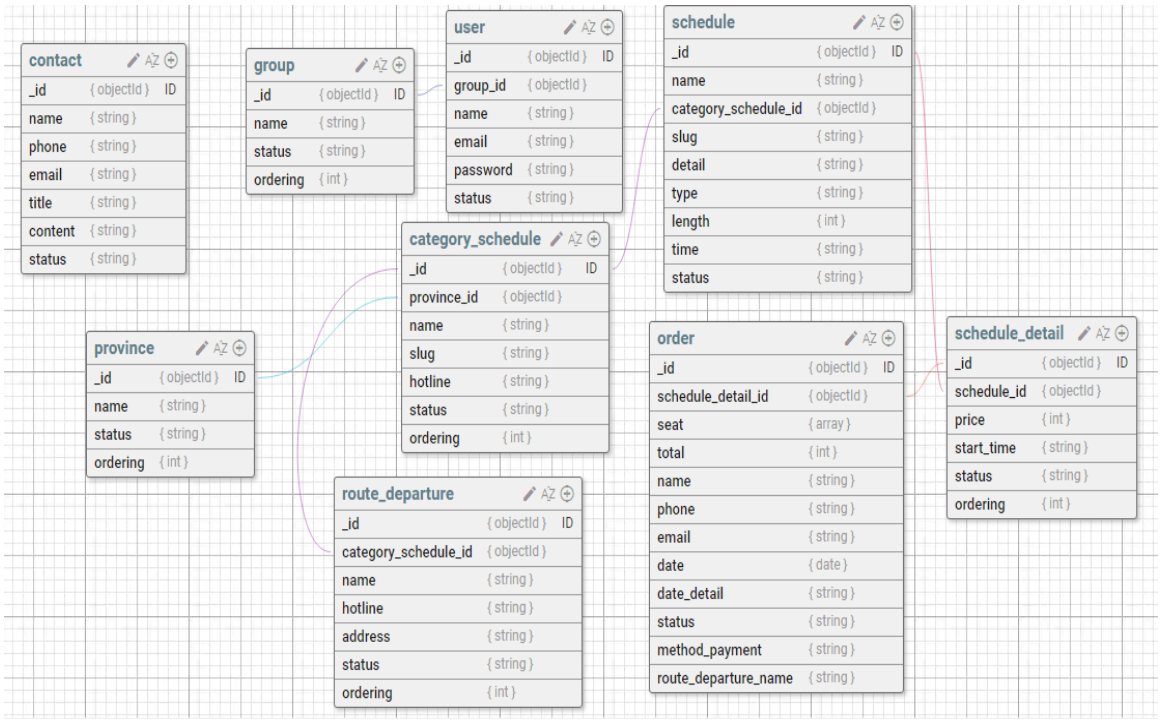
Hình 4.8 Lược đồ tuần tự quản trị viên thêm/sửa thông tin

4.5 Thiết kế cơ sở dữ liệu

4.5.1 Tổng quan cơ sở dữ liệu

Từ những phân tích tổng quán của hệ thống, đưa đến việc phát triển cơ sở dữ liệu nhằm lưu trữ và truy xuất dữ liệu.

Hệ quản trị cơ sở dữ liệu sử dụng: Mongo DB



Hình 4.9 Database

4.5.2 Chi tiết cơ sở dữ liệu

Danh sách các thực tể có trong cơ sở dữ liệu :

- *group*: Thông tin nhóm có trong hệ thống.
- *user*: Thông tin người dùng có trong hệ thống.
- *province*: Thông tin các tỉnh và thành phố.
- *category_schedule*: Thông tin của các lịch trình theo từng tỉnh hoặc thành phố
- *schedule*: Thông tin của các chuyến xe
- *schedule_detail*: Thông tin chi tiết của các chuyến xe.
- *route_departure*: Thông tin các bến đỗ xe.
- *transhipment_office*: Thông tin của các trạm dừng theo bến đỗ.
- *order:* Thông tin của các đơn đặt vé.
- *contact*: Thông tin mà người dùng nhập trong trang liên hệ

Thực thể group lưu trữ thông tin :

Bảng 4.1: Bảng mô tả thuộc tính của thực thể group

STT	Thuộc tính	Mô tả
1	_id	Mã nhóm
2	name	Tên nhóm
3	status	Trạng thái của nhóm
4	ordering	Sắp xếp của nhóm

Thực thể user lưu trữ thông tin :

Bảng 4.2: Bảng mô tả thuộc tính của thực thể user

STT	Thuộc tính	Mô tả
1	_id	Mã người dùng
2	group_id	Mã nhóm (ObjectID liên kết với bảng group)
3	name	Tên người dùng
4	email	Email người dùng
5	password	Mật khẩu người dùng
6	status	Trạng thái của người dùng
7	ordering	Sắp xếp của người dùng

Thực thể province lưu trữ thông tin :

Bảng 4.3 Bảng mô tả thuộc tính của thực thể province

STT	Thuộc tính	Mô tả
1	_id	Mã tỉnh
2	name	Tên tỉnh
3	status	Trạng thái của tỉnh
4	ordering	Sắp xếp tỉnh

Thực thể category_schedule lưu trữ thông tin :

Bảng 4.4 Bảng mô tả thuộc tính của thực thể category_schedule

STT	Thuộc tính	Mô tả
1	_id	Mã danh mục lịch trình
2	province_id	Mã tỉnh(ObjectID liên kết với bảng province)
3	name	Tên danh mục lịch trình
4	slug	Slug của danh mục lịch trình
5	hotline	Hotline của danh mục lịch trình
6	status	Trạng thái của danh mục lịch trình
7	ordering	Sắp xếp của danh mục lịch trình

Thực thể schedule lưu trữ thông tin :

Bảng 4.5 Bảng mô tả thuộc tính của thực thể schedule

STT	Thuộc tính	Mô tả
1	_id	Mã lịch trình
2	category_schedule_id	Mã danh mục lịch trình(ObjectID liên kết với bảng category_schedule)
3	name	Tên lịch trình
4	slug	Slug của lịch trình
5	detail	Tên chi tiết của lịch trình
6	type	Kiểu xe của lịch trình
7	length	Độ dài của lịch trình
8	status	Trạng thái của lịch trình
9	time	Thời gian của lịch trình

Thực thể schedule_detail lưu trữ thông tin :

Bảng 4.6 Bảng mô tả thuộc tính của thực thể schedule_detail

STT	Thuộc tính	Mô tả
1	_id	Mã chi tiết lịch trình
2	schedule_id	Mã lịch trình(ObjectID liên kết với bảng schedule)
3	price	Giá tiền
4	start_time	Thời gian bắt đầu chuyến xe
5	status	Trạng thái của chuyến xe
6	ordering	Sắp xếp của chuyến xe

Thực thể route_departure lưu trữ thông tin :

Bảng 4.7 Bảng mô tả thuộc tính của thực thể route_departure

STT	Thuộc tính	Mô tả
1	_id	Mã bến xe
2	category_schedule_id	Mã danh mục lịch trình (ObjectID liên kết với bảng category_schedule)
3	name	Tên của các bến xe
4	hotline	Hotline của các bến xe
5	status	Trạng thái của bến xe
6	ordering	Sắp xếp của bến xe
7	address	Địa chỉ của bến xe

Thực thể contact lưu trữ thông tin :

Bảng 4.8 Bảng mô tả thuộc tính của thực thể contact

STT	Thuộc tính	Mô tả
1	_id	Mã liên hệ
2	title	Tiêu đề liên hệ
3	name	Tên của khách hàng
4	phone	Số điện thoại của các khách hàng
5	status	Trạng thái của trạm dừng
6	content	Chi tiết liên hệ
7	email	Email của khách hàng

Thực thể order lưu trữ thông tin :

Bảng 4.9 Bảng mô tả thuộc tính của thực thể order

STT	Thuộc tính	Mô tả
1	_id	Mã đơn hàng
2	schedule_detail_id	Mã chuyến xe (ObjectID liên kết với bảng schedule_detail)
3	seat	Các ghế đã đặt
4	total	Tổng tiền
5	name	Tên người đặt vé
6	phone	Số điện thoại của người đặt vé
7	status	Trạng thái của trạm dừng
8	date	Ngày đặt (theo kiểu date hệ thống)
9	method_payment	Phương thức thanh toán
10	route_departure_name	Nơi lên xe
11	email	Email của người đặt vé
12	Status	Trạng thái đơn hàng

CHƯƠNG V: XÂY DỰNG WEBSITE ĐẶT VÉ XE KHÁCH ONLINE

5.1 Giới thiệu các công nghệ sử dụng

5.1.1 Các công nghệ sử dụng trong việc xây dựng website

- Môi trường: Node JS
- Cơ sở dữ liệu: Mongo DB
- Server: Express JS
- Client: React JS

5.1.2 Các bước thực hiện xây dựng website

Bước 1: Phân tích, thiết kế, mô tả các chức năng cần thiết của website.

Bước 2: Phân tích, thiết kế cơ sở dữ liệu để phù hợp với các chức năng và thực hiện hiệu quả việc truy vấn dữ liệu.

Bước 3: Viết API sau khi đã thiết kế xong cơ sở dữ liệu.

Bước 4: Thiết kế MockUp giao diện người dùng sơ bộ theo mô tả ban đầu và theo chức năng cần có của website.

Bước 5: Thiết kế hoàn chỉnh bằng các phần mềm hỗ trợ đồ họa để có bản thiết kế cho website.

Bước 6: Sử dụng React JS, Boostraps, HTML, CSS, JavaScript..v..v. để tiến hành việc lập trình theo thiết kế hoàn chỉnh.

Bước 7: Chạy thử website và sửa lỗi.

5.2 Kết quả

Sau khi thực hiện tuần tự các bước như đã đặt ra, website đã hoạt động và cơ bản đảm bảo các chức năng yêu cầu đặt ra ban đầu, bên cạnh đó cũng có một số chức năng cải tiến mới và đảm bảo thuận lợi nhất cho người sử dụng khi truy cập vào website

5.3 Một số hình ảnh giao diện của ứng dụng và cách giải quyết

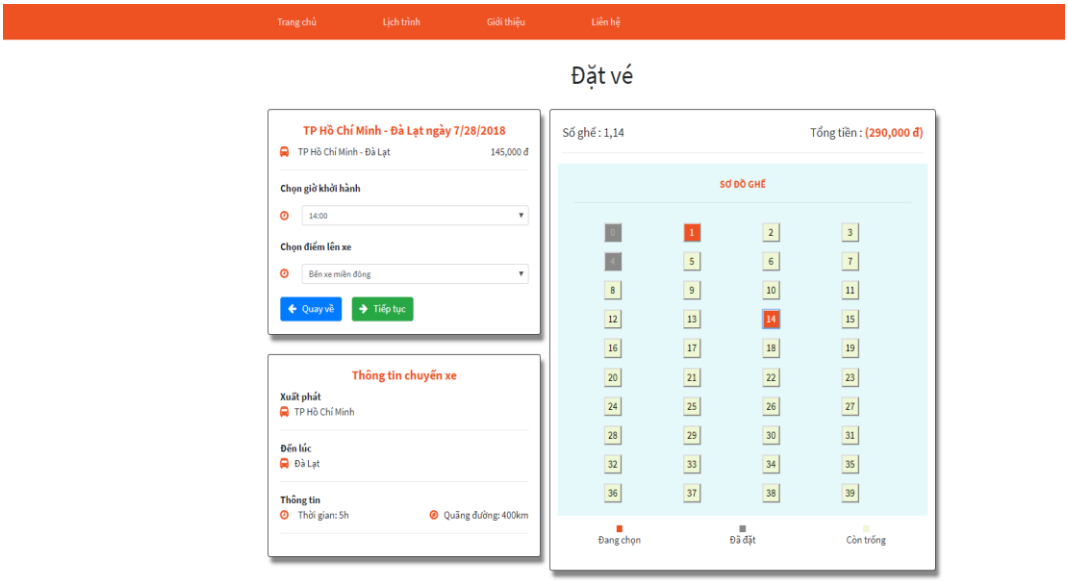
5.3.1 Trang chủ

- Đây là trang dành cho khách hàng để xem thông tin lịch trình để đặt vé.
- Ở đây khách hàng cần phải xem thông tin chuyến xe mà mình đi. Sau đó click vào button mua vé. Sẽ được nhảy sang trang đặt vé.
- Ở phía dưới trang chủ cũng show ra một số thông tin để khách hàng xem về hãng xe.

Hình 5.1 Giao diện trang chủ

5.3.2 Trang đặt vé

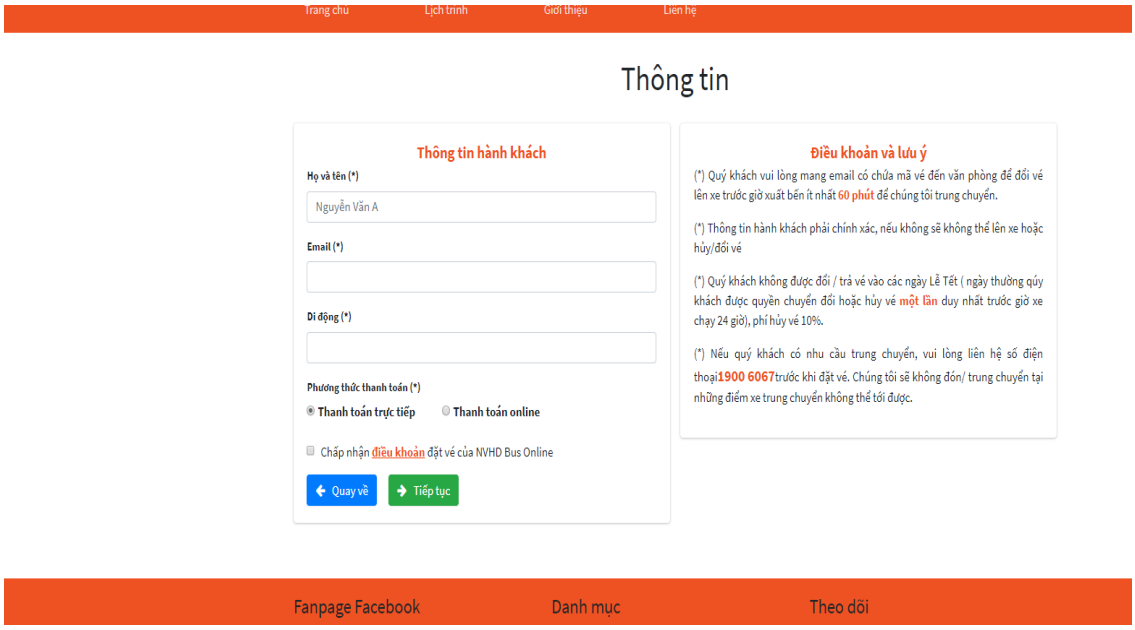
- Đây là trang đặt vé. Khách hàng cần phải lựa chọn đúng khung giờ xe chạy và nơi khách hàng lên xe, đồng thời là số ghế khách hàng chọn.
- Như ở box đặt vé. Những ô màu xám có nghĩa là đã có khách hàng khác đã đặt.
- Những ô màu trắng thì hiện tại vẫn trống. Còn những ô màu cam thì là khách hàng đang chọn ghế của mình.
- Button Quay về có nghĩa là sẽ quay về lại trang chủ để khách hàng có thể chọn lại thông tin chuyến xe khác.
- Button Tiếp tục có nghĩa là khách hàng sau khi đã chọn chỗ ngồi, khung giờ xe chạy và nơi lên xe thì mới chuyển sang trang lấy thông tin để thanh toán.



Hình 5.2 Giao diện trang đặt vé

5.3.3 Trang thông tin khách hàng để thanh toán

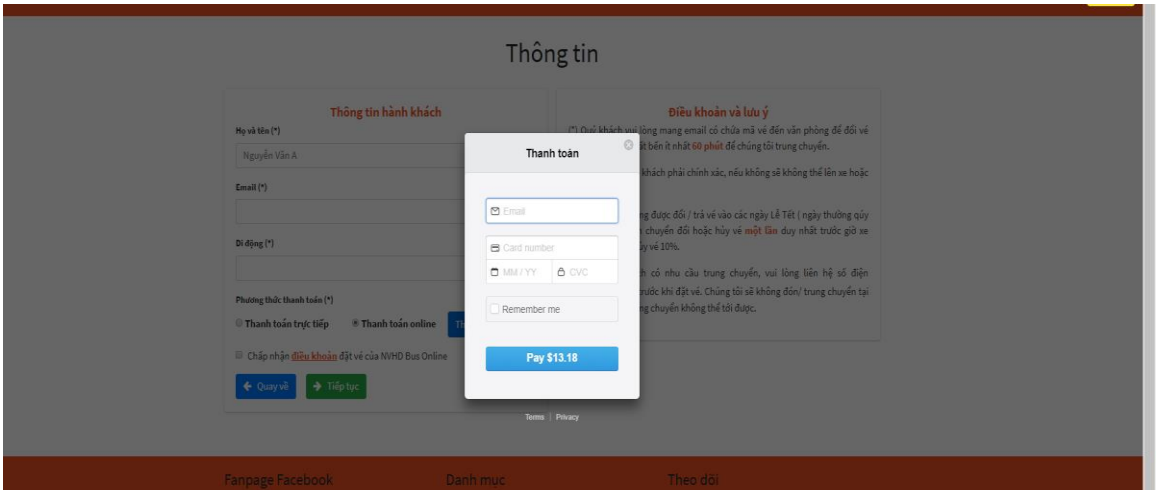
- Đây là trang khách hàng cần phải nhập thông tin chính xác để nhà xe có thể lưu trữ thông tin cũng như là sẽ liên lạc lại với bạn thông qua trang này. Ở đây có 2 phần là thanh toán trực tiếp và thanh toán online.



Hình 5.3 Giao diện trang thanh toán

5.3.4 Trang thanh toán online

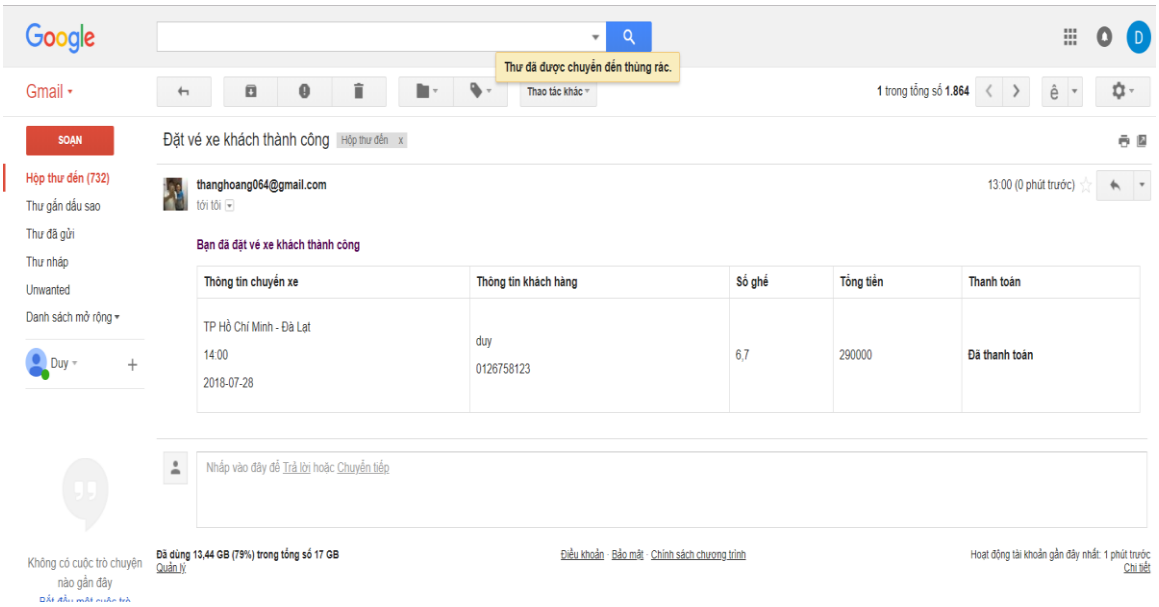
- Ở phần thanh toán online. Em có sử dụng một cổng thanh toán Stripe. Đây cũng là một trong số các cổng thanh toán online hiện nay. Vì không hỗ trợ tiền việt như Paypal. Nên em sẽ quy đổi tiền việt thành tiền đô. Rồi người dùng sẽ thanh toán



Hình 5.4 Giao diện thanh toán online

5.3.5 Mail gửi về khách hàng sau khi đặt vé

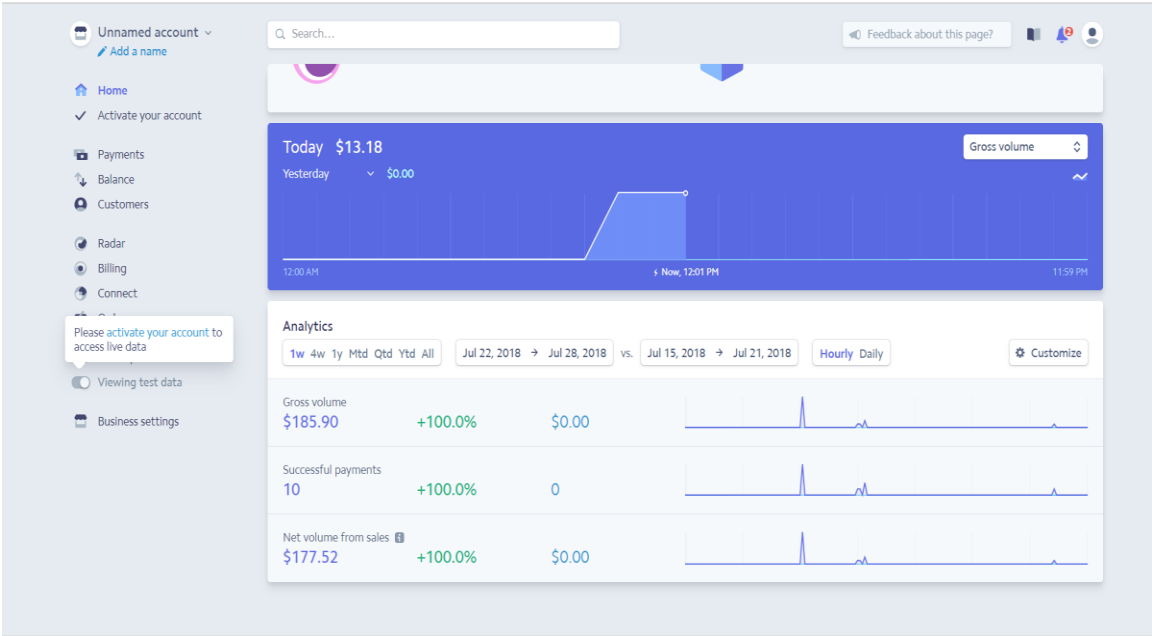
- Đây là giao diện mail gửi về khách hàng sau khi đặt vé thành công.



Hình 5.5 Giao diện mail gửi về khách hàng sau khi đặt vé

5.3.6 Trang stripe

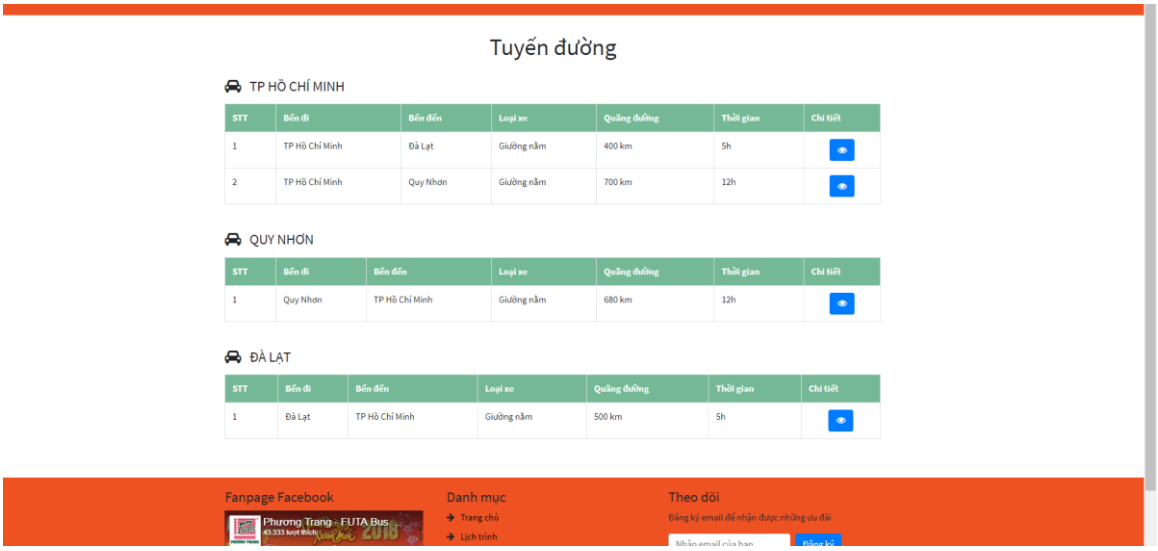
- Đây là trang để xem tài khoản trên trang chủ stripe. Ở đây người quản trị có thể xem tổng số các giao dịch.



Hình 5.6 Giao diện trang dashboard Stripe

5.3.7 Trang lịch trình

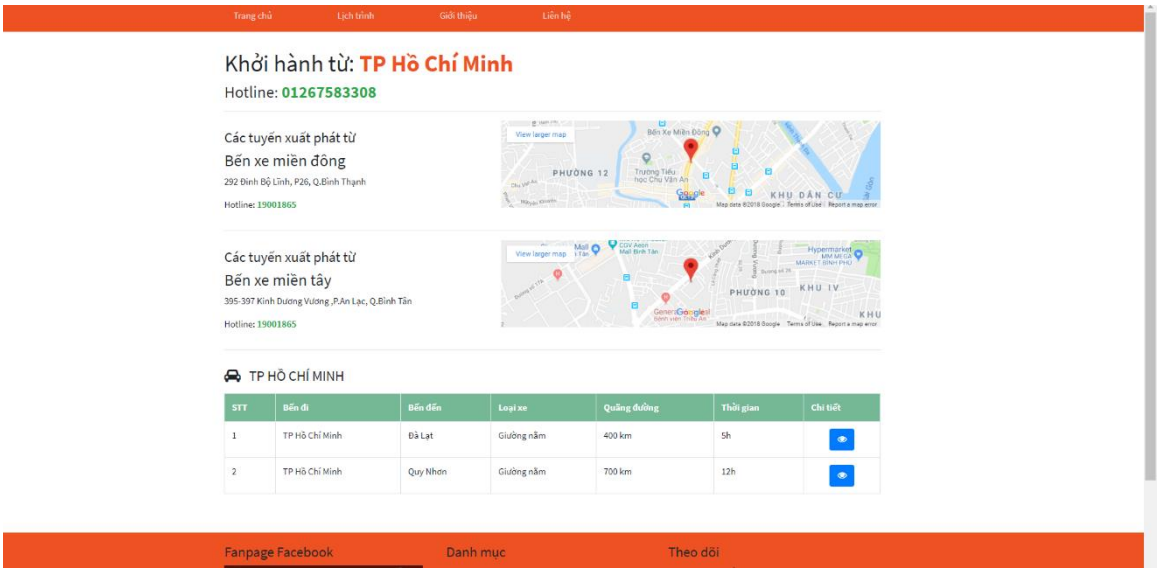
- Đây là trang hiển thị tất cả các tuyến đường mà hãng xe hiện đang có.
- Ở button Chi tiết của từng dòng. Khi nhấn vào đó sẽ hiển thị ra chi tiết chuyến xe đó. Kiểu như 1 tuyến đường thì sẽ có nhiều khung giờ khác nhau
- Ví dụ như: Tuyến xe Quy Nhơn – TP Hồ Chí Minh có 2 khung giờ là 17:00 và 19:00.



Hình 5.7 Giao diện trang tất cả lịch trình của hãng xe

5.3.8 Trang danh mục lịch trình cụ thể

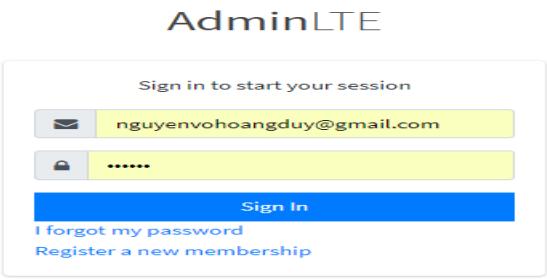
- Đây là trang hiển thị cụ thể danh mục lịch trình. Vì ở đây danh mục lịch trình được cho là theo từng tỉnh. Nên giả sử những khách hàng xuất phát từ TP Hồ Chí Minh thì sẽ có 2 điểm lên xe là Bến xe miền đông và Bến xe miền tây .Ở phần google map như trong hình. Thì ở module admin chỉ cần điền đúng địa chỉ nơi xuất phát. Phần này em có sử dụng google map API để lấy ra tọa độ lat, lng rồi in ra được phần map như hình.



Hình 5.8 Giao diện trang lịch trình theo tỉnh

5.3.9 Trang quản trị đăng nhập

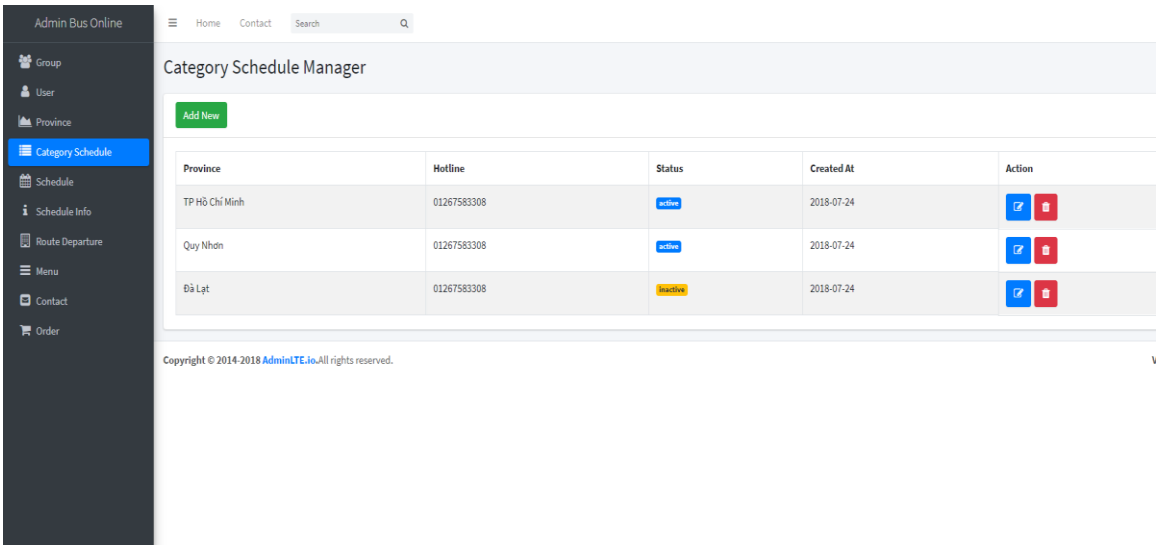
- Người quản trị cần nhập thông tin email và mật khẩu vào. Nếu sai sẽ báo lỗi hiển thị lên.
- Nếu đúng email và password sẽ được gửi lên hệ thống để check. Nếu đúng thì ở hệ thống sẽ gửi về 1 token. Em có lưu token đó vào localStorage. Và ở phía client thì nó sẽ kiểm tra token này. Nếu đúng thì mới vào được trang quản trị



Hình 5.9 Giao diện trang đăng nhập của quản trị viên

5.3.10 Trang quản trị danh mục lịch trình

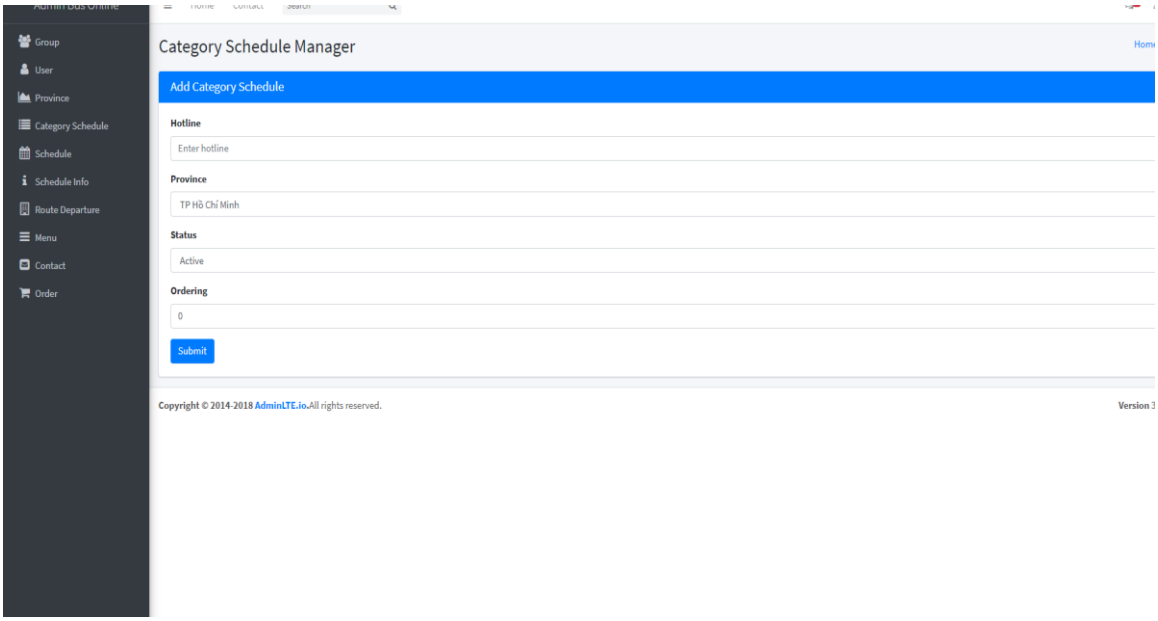
- Đây là trang load ra tất cả các danh mục lịch trình. Vì em mặc định là 1 danh mục lịch trình sẽ đi theo 1 tỉnh. Nên ở ngoài client. Chỉ có những danh mục lịch trình nào có status là active thì mới được load ra trang chủ.



Hình 5.10 Hình danh sách danh mục lịch trình

5.3.11 Trang quản trị thêm danh mục lịch trình

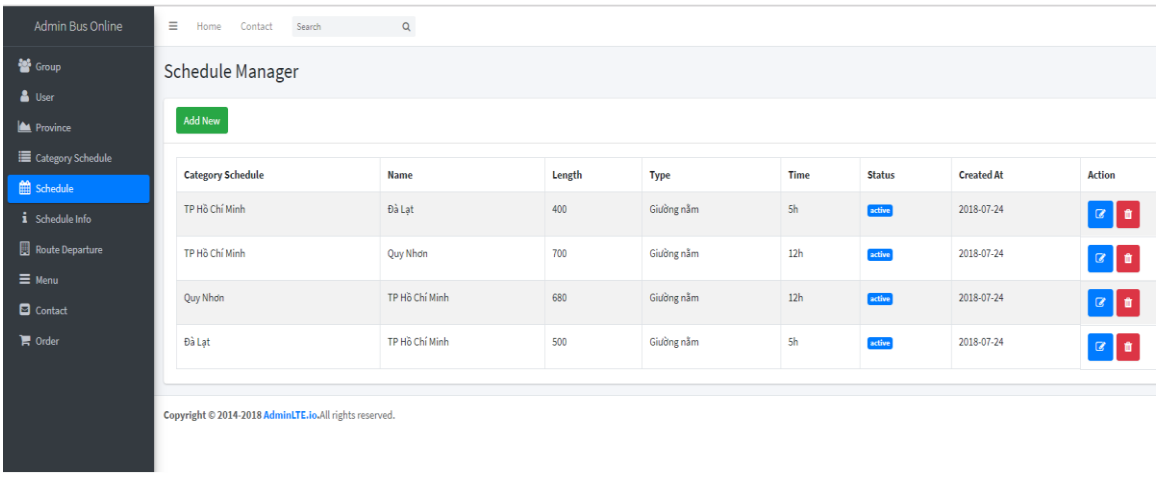
- Ở đây là phần thêm vào danh mục lịch trình. Ở phần province em đã làm 1 bảng riêng để có thể lấy ra 64 tỉnh thành. Sau khi nhập thành công dữ liệu sẽ được lưu vào database. Còn nếu nhập sai hoặc trống thì sẽ được thông báo lỗi.



Hình 5.11 Thêm danh mục lịch trình

5.3.12 Trang quản trị lịch trình

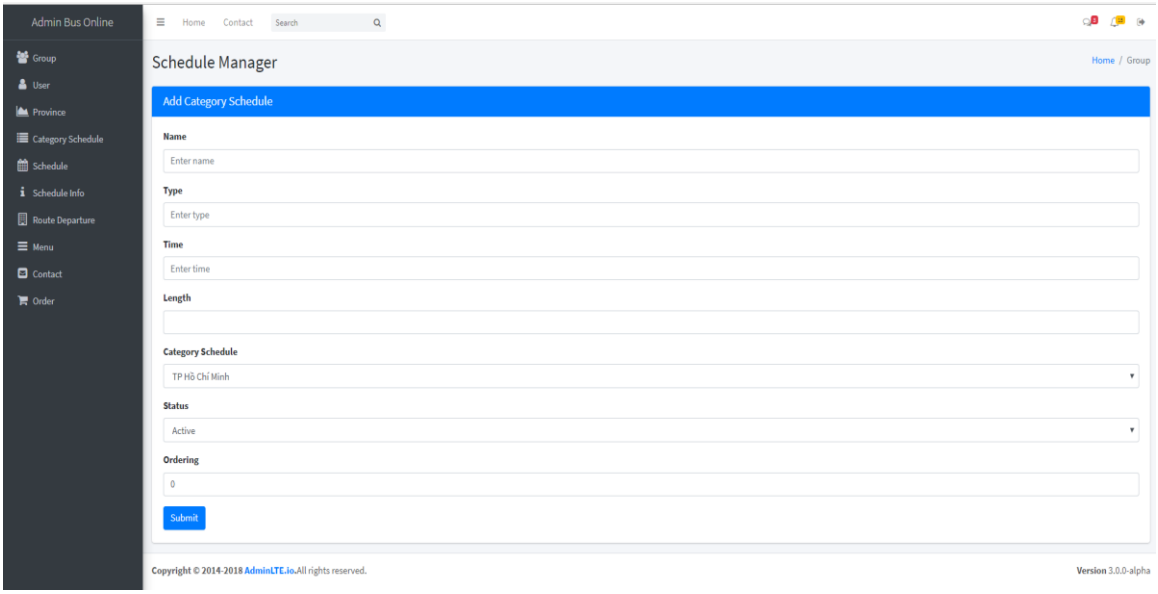
- Trang này thì cũng tựa như trang danh mục lịch trình. Category Schedule là field được lấy từ database danh mục lịch trình qua. Phần name có nghĩa là nơi đến.



Hình 5.12 Danh sách lịch trình

5.3.13 Trang quản trị thêm lịch trình

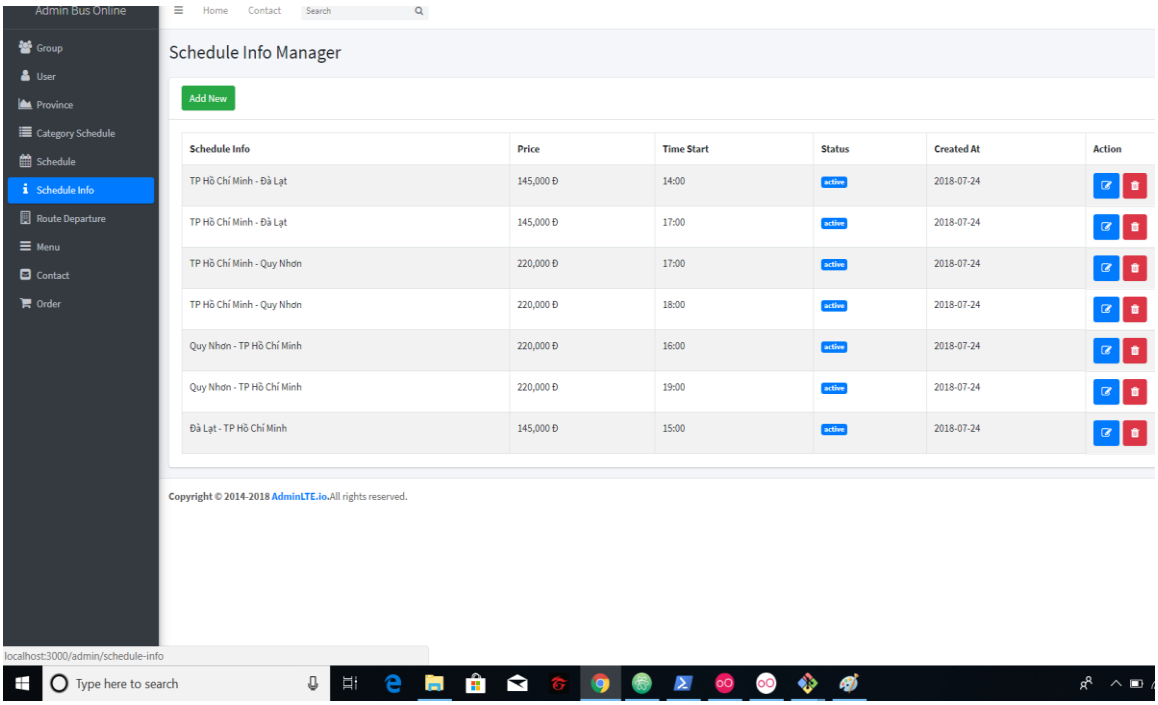
- Cái này thì cũng tương tự như phần thêm ở danh mục lịch trình



Hình 5.13 Thêm lịch trình

5.3.14 Trang quản trị chi tiết lịch trình

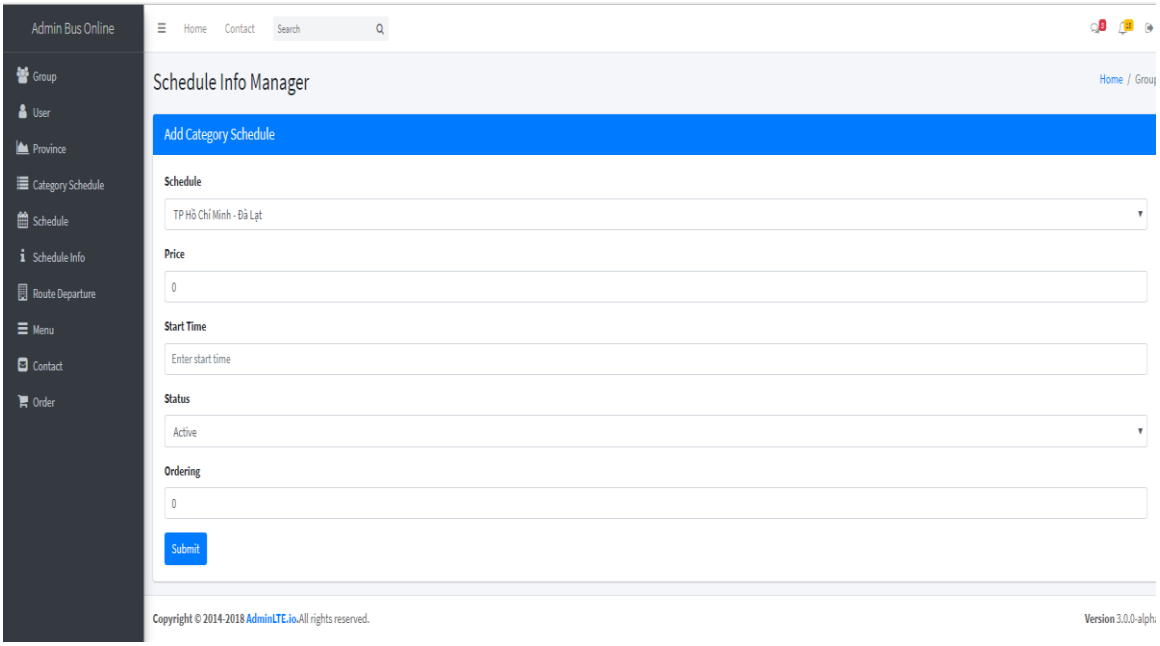
- Trang này thì cũng tựa như trang lịch trình.



Hình 5.14 Danh sách chi tiết lịch trình

5.3.15 Trang quản trị thêm chi tiết lịch trình

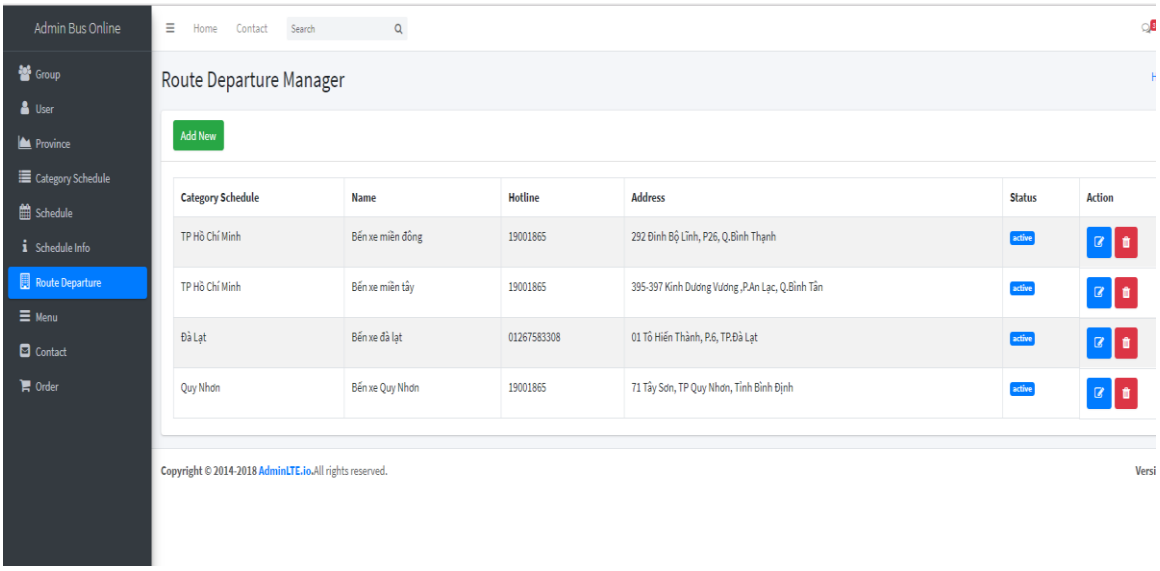
- Tương tự như trang thêm danh mục lịch trình.



Hình 5.15 Thêm chi tiết lịch trình

5.3.16 Trang quản trị nơi lên xe

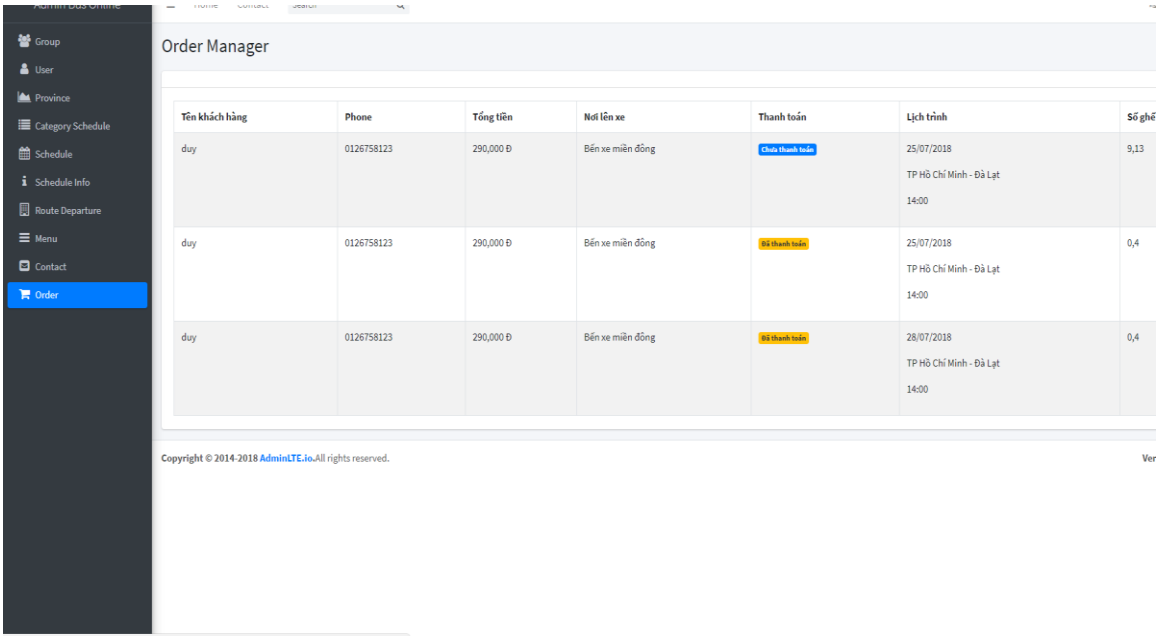
- Tương tự như trang quản lý lịch trình. Người dùng cần nên nhập đúng địa chỉ để có thể hiện thị ngoài trang chủ google map cho chính xác.



Hình 5.16 Danh sách nơi lên xe

5.3.17 Trang quản trị đơn đặt vé

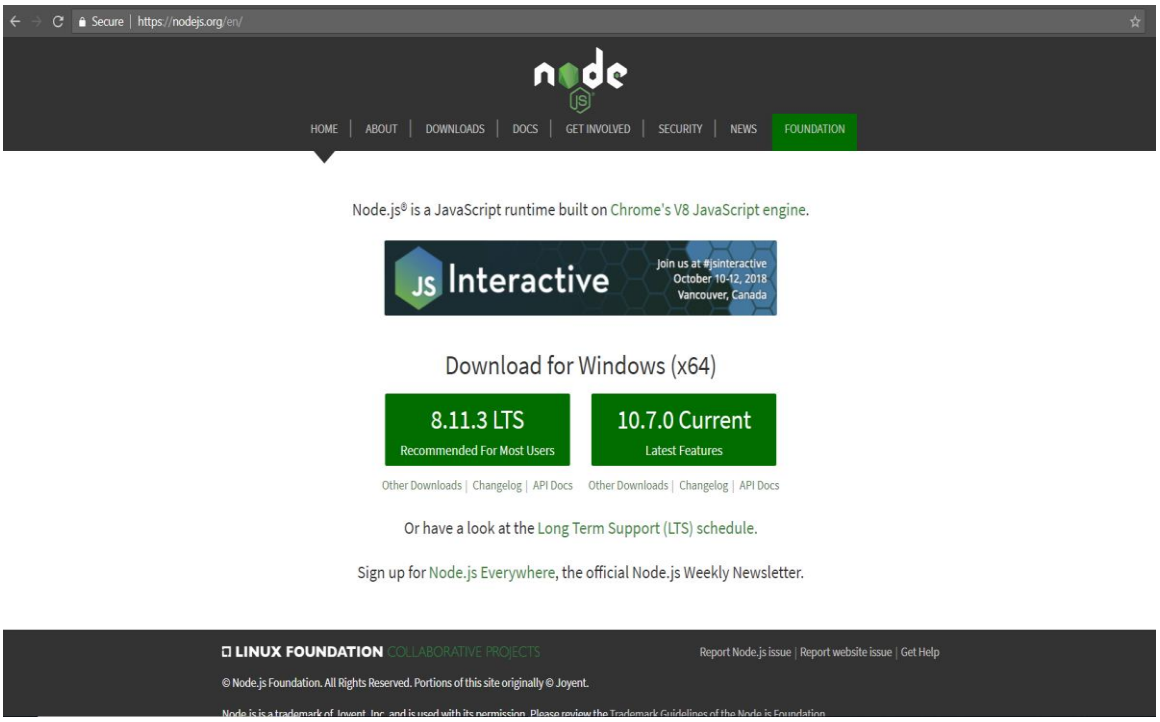
- Đây là trang người quản trị xem thông tin khách hàng đặt vé.



Hình 5.17 Danh sách đơn đặt vé

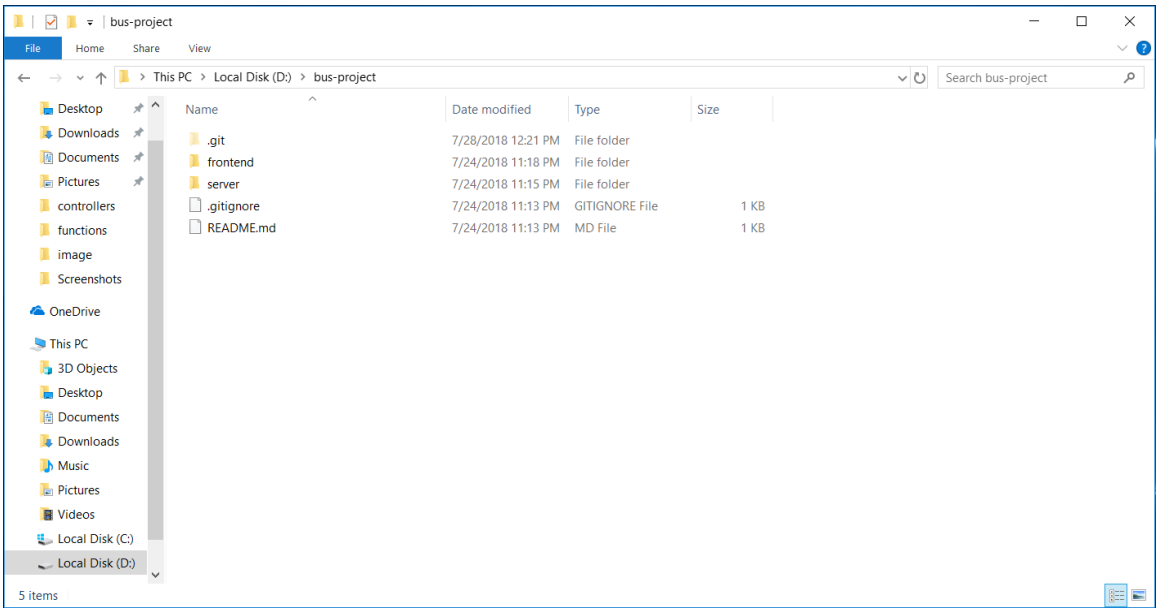
5.4 Hướng dẫn cài đặt

⇨ Cài đặt Node



Hình 5.18 Cài đặt NodeJS

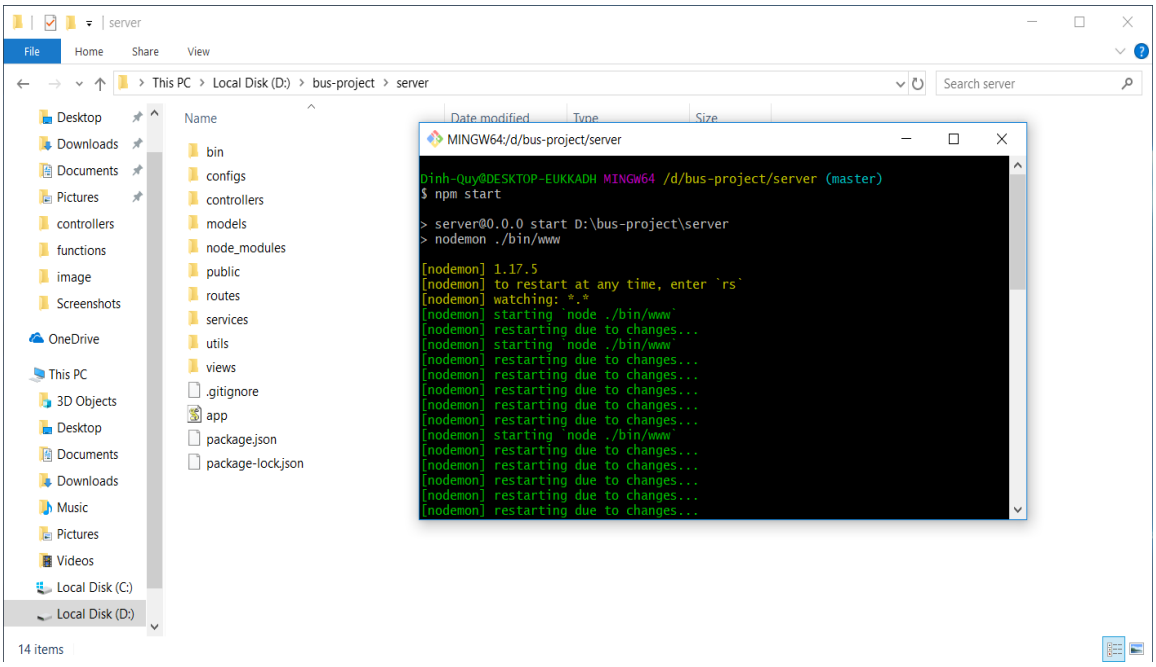
⇨ Đây là cấu trúc thư mục chương trình



Hình 5.19 Cấu trúc thư mục chương trình

⇨ Ở thư mục server

- Chạy command: npm install –save để cài đặt những gói cần thiết cho server
- Tiếp theo sau khi đã chạy xong lệnh trên. Chúng ta chạy tiếp command: npm start như hình sau

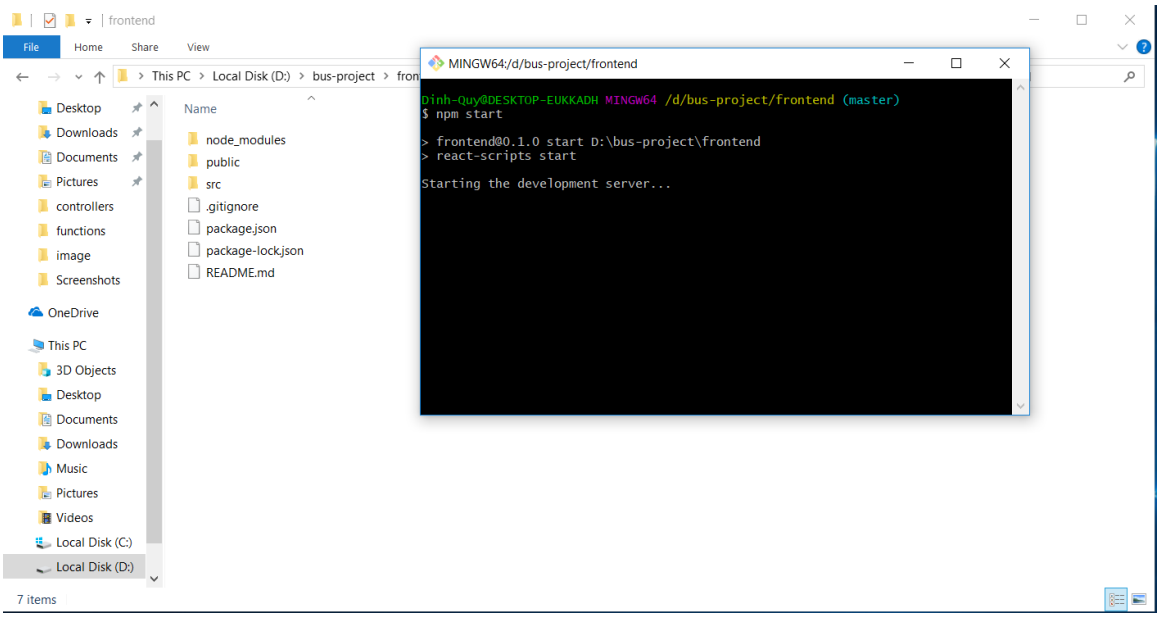


Hình 5.20 Cài đặt và chạy server

- Sau khi ở terminal của bạn hiển thị được như trong hình và có dòng chữ Successful connect database thì bạn đã bật server thành công.

⇨ Ở thư mục frontend

- Chạy command: npm install –save để cài đặt những gói cần thiết cho client
- Tiếp theo sau khi đã chạy xong lệnh trên. Chúng ta chạy tiếp command: npm start như hình sau



Hình 5.21 Cài đặt và chạy client

- Vì project này em lưu database trực tuyến trên server mlab.com nên sẽ không cần phải cài đặt hay cấu hình database

CHƯƠNG VI: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Kết quả đạt được

Website đặt vé xe khách online về cơ bản đã giải quyết một phần nào khó khăn hiện nay của các chủ nhà xe nhỏ bao gồm các chức năng cho phép người dùng dễ dàng trong việc đặt vé mình mong muốn.

Về các yêu cầu phi chức năng, website có tốc độ hiển thị khá tốt và giao diện bắt mắt, chuyên nghiệp.

Trong quá trình làm đồ án, bản thân em cũng đã học hỏi và tìm hiểu được rất nhiều các công nghệ mới cũng như áp dụng vào thực tế đề tài nhiều công nghệ như đã trình bày ở trong báo cáo.

6.2 Hạn chế

Tuy nhiên, do thời gian ngắn và số lượng công việc cũng như tìm hiểu khá nhiều công nghệ cho nên việc xảy ra lỗi xảy ra nhiều làm cho quá trình hoàn thành bị chậm. Bên cạnh đó một số các chức năng dự kiến thêm vào vẫn chưa được triển khai như quản lý danh mục hàng hóa được gửi đi thì vẫn chưa hoàn thành.

6.3 Hướng phát triển

- Quản lý về phần danh sách hàng hóa được gửi đi.
- Có thể dựng thêm 1 module nữa. Để có thể mở rộng ra cho 1 website nhưng có thể có nhiều nhà xe.

Có thể sẽ làm vé ticket điện tử QR Code. Để khi đặt vé thành công sẽ được gửi về mail hãng xe có thể scan vé này để xác nhận vé của hãng mình.

TÀI LIỆU THAM KHẢO

Tiếng Việt

1. https://viblo.asia/tags/reactjs
2. https://nodejs.vn/

Tiếng Anh

3. Michael Mikowski, Josh Powell Single page web applications JavaScript end-to-end 2013
4. HTML & CSS by Jon Duckett.
5. JavaScript: The Good Parts by Douglas Crockford.

Website:

1. https://vi.wikipedia.org/wiki/
2. https://nodejs.org/en/docs/
3. https://docs.mongodb.org/manual/
4. http://www.w3schools.com/html/html5_geolocation.asp
5. https://www.mapbox.com
6. http://getbootstrap.com/
7. https://reactjs.org/