

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

PHẠM TÂN LONG

KHÓA LUẬN TỐT NGHIỆP
NGHIÊN CỨU VỀ UNITY VÀ XÂY DỰNG PHẦN MỀM
HỌC LÁI XE AN TOÀN

KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM

TP. HỒ CHÍ MINH, 2015

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

PHẠM TÂN LONG – 10520163

KHÓA LUẬN TỐT NGHIỆP
NGHIÊN CỨU VỀ UNITY VÀ XÂY DỰNG PHẦN MỀM
HỌC LÁI XE AN TOÀN

KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM

GIẢNG VIÊN HƯỚNG DẪN
THẠC SĨ NGUYỄN TRÁC THÚC

TP. HỒ CHÍ MINH, 2015

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số
ngày của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. – Chủ tịch.
2. – Thư ký.
3. – Ủy viên.
4. – Ủy viên.

LỜI CẢM ƠN

Trước hết nhóm xin gửi lời cảm ơn chân thành đến thầy, ThS. Nguyễn Trác Thức, khoa Công Nghệ Phần Mềm, Trường Đại học Công Nghệ Thông Tin. Trong suốt thời gian thực hiện khóa luận, thầy đã dành rất nhiều thời gian và tâm huyết trong việc hướng dẫn nhóm. Thầy đã luôn có những định hướng, góp ý, sửa chữa những chỗ sai sót giúp nhóm có thể đi được đúng hướng. Khoa luận có được kết quả ngày hôm nay là nhờ sự nhắc nhở, đôn đốc và giúp đỡ nhiệt tình của thầy.

Nhóm cũng xin trân trọng cảm ơn tất cả thầy cô trong khoa Công Nghệ Phần Mềm, cũng như các thầy cô đã giảng dạy, giúp đỡ chúng em trong suốt thời gian học tập ở trường. Những kiến thức nền tảng và chuyên môn mà chúng em học được từ các thầy cô đã giúp chúng em có thể hoàn thành khóa luận này.

Con xin nói lên lòng biết ơn sâu sắc đối với cha mẹ, người đã chăm sóc, nuôi dạy chúng con thành người.

Xin chân thành cảm ơn các anh chị và bạn bè đã ủng hộ, giúp đỡ và động viên chúng em trong thời gian học tập và nghiên cứu ở trường cũng như ngoài xã hội.

Mặc dù đã rất cố gắng để hoàn thành nhưng vẫn không thể tránh khỏi những thiếu sót. Chúng em kính mong nhận được sự góp ý của quý thầy cô.

Nhóm khóa luận

Phạm Tân Long

MỤC LỤC

Chương 1. TỔNG QUAN	3
1.1. Bối cảnh nghiên cứu.....	3
1.2. Động lực nghiên cứu	3
1.3. Giới hạn đề tài	4
Chương 2. CƠ SỞ CÔNG NGHỆ.....	6
2.1. Game engine.....	6
2.1.1. Các loại game engine	6
2.1.2. Một số game engine trên thị trường	7
2.2. Unity.....	9
2.2.1. Giới thiệu.....	9
2.2.2. Các tính năng chính.....	10
2.2.3. Ngôn ngữ lập trình	13
2.2.4. Nền tảng Mono.....	14
2.2.4.1. Kiến trúc	15
2.2.4.2. Các thành phần	15
2.2.4.3. Các tính năng.....	16
2.2.5. Kiến trúc Component	17
2.2.6. Các Component trong Unity	19
2.2.7. Script	24
2.2.8. Physics.....	27
2.2.8.1. Các Component vật lý trong Unity	29
2.3. Laravel.....	33
2.3.1. Giới thiệu.....	33

2.3.2. Một số ưu điểm	34
2.3.3. Request LifeCycle	34
2.3.4. Route	35
2.3.5. View	36
2.3.6. Controller	37
2.3.7. Filter	38
2.4. MongoDB.....	39
2.4.1. NoSQL	39
2.4.2. MongoDB.....	40
2.4.2.1. Các khái niệm.....	40
2.4.2.2. Tính năng chính.....	41
2.4.2.3. Sử dụng	42
Chương 3. PHÂN TÍCH THIẾT KẾ.....	44
3.1. Tổng quan chức năng	44
3.2. Kiến trúc tổng quát.....	45
3.3. Game	46
3.3.1. Yêu cầu.....	46
3.3.2. Phân tích.....	47
3.3.3. Cấu trúc dữ liệu	50
3.3.4. Sơ đồ use case	51
3.3.5. Sơ đồ tuần tự	53
3.3.6. Giải thuật.....	55
3.3.7. Danh sách các lỗi	58
3.4. Map Editor	60

3.4.1. Sơ đồ use case	60
3.4.2. Sơ đồ tuần tự	61
3.4.3. Thiết kế giao diện.....	63
3.4.4. Cấu trúc dữ liệu	65
3.5. CMS	67
3.5.1. Sơ đồ use case	68
3.5.2. Sơ đồ tuần tự	69
3.5.3. Thiết kế cơ sở dữ liệu	70
3.5.4. Thiết kế giao diện.....	72
Chương 4. CÀI ĐẶT MINH HOA	78
4.1. Game	78
4.1.1. Giao diện	78
4.1.2. Hướng dẫn sử dụng	79
4.2. Map Editor	81
4.2.1. Giao diện	81
4.2.2. Hướng dẫn sử dụng	81
4.3. CMS	82
4.3.1. Yêu cầu phần cứng	82
4.3.2. Giao diện	82
Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	85
5.1. Kết luận	85
5.2. Hướng phát triển	85
TÀI LIỆU THAM KHẢO	86

DANH MỤC HÌNH VẼ

Hình 2.1 Kiến trúc Mono	15
Hình 2.2 Thực thể theo kiến trúc component	17
Hình 2.3 Các hệ thống xử lý các component	18
Hình 2.4 Các loại Physic Material mặc định trong Unity	30
Hình 2.5 Các thuộc tính chung của các Physic Material	30
Hình 2.6 Tỷ lệ sử dụng các framework PHP năm 2013	33
Hình 2.7 Vòng đời một Request trong Laravel	35
Hình 2.8 Request Filter trong Laravel	38
Hình 3.1 Kiến trúc tổng quát hệ thống	45
Hình 3.2 Làn đường (bên trái) và đường (bên phải)	47
Hình 3.3 Biển báo giao thông là nơi chứa thông tin ràng buộc	49
Hình 3.4 Làn đường là nơi chứa thông tin ràng buộc	49
Hình 3.5 Vấn đề kiểm tra lỗi người chơi	50
Hình 3.6 Sơ đồ use case của Game	52
Hình 3.7 Sơ đồ tuần tự use case Login	53
Hình 3.8 Sơ đồ tuần tự use case Logout	54
Hình 3.9 Sơ đồ tuần tự use case Show history	54
Hình 3.10 Sơ đồ tuần tự use case Select map	54
Hình 3.11 Sơ đồ tuần tự use case Post score	55
Hình 3.12 Vuốt đèn đỏ	55
Hình 3.13 Sơ đồ logic phát hiện lỗi vuốt đèn đỏ	56
Hình 3.14 Lần tuyển	56
Hình 3.15 Sơ đồ logic phát hiện lỗi lần tuyển	57
Hình 3.16 Sơ đồ logic lỗi chuyển hướng không tín hiệu	57
Hình 3.17 Sơ đồ use case Map Editor	60
Hình 3.18 Sơ đồ tuần tự use case New map	61
Hình 3.19 Sơ đồ tuần tự use case Save map	62
Hình 3.20 Sơ đồ tuần tự use case Open map	62

Hình 3.21 Giao diện chính Map Editor.....	63
Hình 3.22 Giao diện New map và Edit map	65
Hình 3.23 Sơ đồ use case CMS	68
Hình 3.24 Sơ đồ tuần tự use case Upload accounts	69
Hình 3.25 Sơ đồ tuần tự use case Update accounts	70
Hình 3.26 Giao diện chung các trang của CMS	73
Hình 3.27 Giao diện Đăng nhập	74
Hình 3.28 Giao diện Update accounts	75
Hình 3.29 Giao diện Update maps	76
Hình 3.30 Giao diện Scores	77
Hình 4.1 Giao diện Game	78
Hình 4.2 Các thành phần giao diện Game	79
Hình 4.3 Giao diện Map Editor	81
Hình 4.4 Giao diện Home / Scores trong CMS	82
Hình 4.5 Giao diện Update accounts trong CMS	83
Hình 4.6 Giao diện Upload map trong CMS	83
Hình 4.7 Giao diện Update maps trong CMS	84

DANH MỤC BẢNG

Bảng 2.1 Tỷ lệ sử dụng các ngôn ngữ trong Unity	13
Bảng 2.2 Các thuộc tính của MonoBehaviour	25
Bảng 2.3 Các phương thức của MonoBehaviour	25
Bảng 2.4 Các phương thức cần thừa kế của MonoBehaviour	27
Bảng 2.5 So sánh tốc độ xử lý vật lý của CPU và GPU	28
Bảng 2.6 Các thuộc tính của Rigidbody	29
Bảng 2.7 Các loại Collider.....	32
Bảng 2.8 So sánh các thuật ngữ trong RDBMS và MongoDB.....	41
Bảng 3.1 Danh sách Actor trong sơ đồ use case của Game.....	52
Bảng 3.2 Danh sách use case trong Game	53
Bảng 3.3 Danh sách các lỗi vi phạm.....	59
Bảng 3.4 Danh sách Actor trong Map Editor	60
Bảng 3.5 Danh sách use case của Map Editor	61
Bảng 3.6 Chú giải giao diện chính Map Editor	64
Bảng 3.7 Chú giải giao diện New map	65
Bảng 3.8 Danh sách Actor trong CMS	68
Bảng 3.9 Danh sách use case CMS.....	69
Bảng 3.10 Các thuộc tính mặc định trong MongoDB	70
Bảng 3.11 Collection accounts	71
Bảng 3.12 Collection maps	71
Bảng 3.13 Collection scores	72
Bảng 3.14 Các giao diện trong CMS	72
Bảng 3.15 Chú giải giao diện chung các trang CMS.....	73
Bảng 3.16 Chú giải giao diện Đăng nhập	74
Bảng 3.17 Chú giải giao diện Update accounts	75
Bảng 3.18 Chú giải giao diện Update maps.....	77
Bảng 3.19 Chú giải giao diện Scores	77
Bảng 4.1 Các thành phần giao diện Game	79

Bảng 4.2 Các phím điều khiển trong Game 80

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Điễn giải
API	Application Programming Interface
CMS	Content Management System
GUI	Graphical User Interface
MVC	Model-View-Controller
CSDL	Cơ sở dữ liệu
RDBMS	Relational Database Management System
SQL	Structured Query Language
NoSQL	Not Only Structured Query Language
2D, 3D	2-Dimensional, 3-Dimensional
PC	Personal Computer
ORM	Object-Relational Mapping
T-SQL	Transact-Structured Query Language

TÓM TẮT KHÓA LUẬN

Với sự phát triển mạnh mẽ của đất nước trong những năm gần đây, ngoài những thay đổi tích cực về đời sống con người, Việt Nam cũng đang phải đổi mới với nhiều vấn đề lớn, trong đó có vấn đề giao thông. Với mong muốn sử dụng công nghệ thông tin góp phần xây dựng ý thức tuân thủ luật giao thông bằng những hình thức mới lạ, dễ nhớ, khóa luận đã được hình thành ý tưởng như vậy.

Từ kết quả nghiên cứu, tìm hiểu về Unity, nhóm đã sử dụng công nghệ này để xây dựng nền phần mềm học luật giao thông nhằm tạo ra một công cụ học tập trực quan sinh động hơn, nhằm nâng cao ý thức người dân khi tham gia giao thông, từ đó hạn chế tai nạn giao thông, cải thiện tình trạng giao thông Việt Nam.

Trước hết, quá trình nghiên cứu Unity bao gồm tìm hiểu về các vấn đề cơ bản: game engine là gì, so sánh các game engine đang phổ biến, và lý do lựa chọn Unity. Tiếp theo là các vấn đề chuyên sâu của engine này, bao gồm: công nghệ Mono, kiến trúc Component, script, physic system và terrain.

Cuối cùng là việc áp dụng Unity vào việc xây dựng phần mềm học luật giao thông. Bên cạnh việc sử dụng Unity, để có thể hoàn thành hệ thống, nhóm đã tìm hiểu thêm về Laravel và MongoDB để xây dựng trang quản trị nội dung (CMS).

Kết quả cuối cùng, khóa luận tạo ra một hệ thống bao gồm 3 sản phẩm: Map Editor, Game và CMS.

Map Editor là một phần mềm độc lập, cho phép người quản trị xây dựng tùy biến một khu vực giao thông có hầu hết các thành phần giao thông trong thực tế gồm: làn đường, vỉa hè, vạch kẻ đường, biển báo, đèn giao thông, trạm xe buýt... Bên cạnh là các thành phần phối cảnh: nhà cửa, cây cối, người qua đường, các xe tham gia giao thông...

Game là phần mềm chạy trên web, thông qua Unity Web Player. Đây là nơi người chơi có thể tham gia chơi và học luật giao thông.

CMS là trang web cho phép nhà quản trị quản lý nội dung của game, quản lý người chơi, xem kết quả...

MỞ ĐẦU

Trong những năm gần đây, tình hình tai nạn giao thông nước ta đang trở thành vấn đề vô cùng nghiêm trọng. Thống kê trong năm 2014, cả nước xảy ra 25.322 vụ tai nạn giao thông, làm 8.996 người chết và 24.417 người bị thương. Đồng thời, lực lượng cảnh sát giao thông toàn quốc đã xử lý hơn 4,3 triệu trường hợp vi phạm giao thông đường bộ với số tiền phạt lên đến 2.577,84 tỷ đồng. Số liệu trên đã cho thấy, bên cạnh những lý do về cơ sở hạ tầng thì ý thức, kiến thức của người tham gia giao thông cũng là nguyên nhân quan trọng dẫn đến tình trạng tai nạn giao thông ở mức cao như vậy.

Trên thực tế, đã có rất nhiều nỗ lực tuyên truyền, giáo dục của nhà nước trên báo chí, truyền hình hay trực tiếp tại các trường học, tổ dân phố. Bên cạnh đó cũng đã có nhiều ứng dụng học luật giao thông trên máy tính, điện thoại di động để mọi người dễ dàng tiếp cận. Tuy nhiên, nhược điểm của những biện pháp này là chỉ mang tính đối phó, thiếu trực quan, không thu hút người dùng do đó khó có thể đem lại những thay đổi tích cực.

Cần có một hệ thống giáo dục luật giao thông mới trực quan, sinh động hơn, cho phép người học tham gia trực tiếp vào các tình huống giao thông, tương tác như trong thực tế, có thể dễ dàng thay đổi, cập nhật nội dung học tập, và hơn nữa là đa nền tảng để có thể đến được với mọi đối tượng người học.

Một game 3D là một ý tưởng tuyệt vời để thực hiện điều này. Các tình huống giao thông sẽ trở nên thực tế hơn, sinh động hơn trong môi trường 3D. Người tham gia có thể tương tác trực tiếp, tạo ra các tình huống vi phạm và bị xử phạt. Những kiến thức giao thông sẽ dễ dàng tiếp cận hơn khi việc học trở thành một trò chơi.

Unity đang là một game engine mạnh, phổ biến, đặc biệt khi Unity còn hỗ trợ phiên bản miễn phí. Người dùng cá nhân hoặc các tổ chức phi lợi nhuận có thể download, sử dụng và phát hành sản phẩm hoàn toàn miễn phí.

Với tất cả những khảo sát đó, nhóm xin lựa chọn đề tài "Nghiên cứu về Unity và Xây dựng phần mềm học lái xe an toàn".

Chương 1. TỔNG QUAN

1.1. Bối cảnh nghiên cứu

Theo thống kê của Bộ Giao thông Vận tải, tính đến tháng 3/2014, số lượng xe máy được sử dụng tại Việt Nam xấp xỉ 37 triệu chiếc. Với dân số hiện tại 90 triệu người thì cứ khoảng 2,4 người thì có 1 người có xe máy. Tuy nhiên, tại các khu vực đô thị, tỷ lệ người lao động với xe máy đạt hệ số 1:1. Xe máy đang chiếm tới hơn 85% tổng số phương tiện giao thông hiện nay, một con số vô cùng lớn. Bình quân mỗi năm thị trường Việt Nam tiêu thụ thêm hơn 3 triệu chiếc xe máy. Với đà tăng trưởng như vậy, đến năm 2020, tổng số lượng xe máy trên cả nước có khả năng đạt tới con số 60 triệu chiếc.

Theo Uỷ ban An toàn giao thông quốc gia, mỗi năm, nạn ùn tắc giao thông đã làm lãng phí 30.000 tỷ đồng của nhân dân. Theo tính toán, thời gian ùn tắc giao thông trung bình tại Hà Nội và TP.HCM là 45 phút/ngày, tương đương 15 giờ/tháng hay 180 giờ/năm. Với con số đó, tổng thiệt hại tương ứng chỉ tính riêng Hà Nội và TP.HCM đã lên tới 18.800 tỉ đồng/năm.Thêm vào đó, mỗi năm số tiền chi ra cho khắc phục tai nạn giao thông tại Việt Nam lên tới 50.000 tỉ đồng, bằng 30% ngân sách chi cho giáo dục. Điều đáng chú ý ở đây là 70% các vụ tai nạn đó có thủ phạm chính là xe máy.

Với những số liệu đó, vấn đề nâng cao ý thức, kiến thức cho người điều khiển xe gắn máy tham gia giao thông là một vấn đề cấp bách.

1.2. Động lực nghiên cứu

Tuyên truyền về kiến thức giao thông là hoạt động cần được đẩy mạnh trong bối cảnh trên, tuy nhiên vấn đề đặt ra là phương pháp nào sẽ hiệu quả.

Các hoạt động tuyên truyền phổ biến hiện nay như: áp phích, băng rôn, chương trình truyền hình, cuộc thi trong trường học, tuyên truyền thông qua loa phát thanh ở từng khu phố... Nhược điểm có thể thấy của các hoạt động này là chỉ có thể đề cập đến lý thuyết hay trình bày hiện trạng nhằm răn đe. Người nghe có thể thay đổi

ý thức, kiến thức, nhưng chưa có cơ hội để có thể thực hành và kiểm tra trong các điều kiện giống với thực tế. Chính những thiếu sót đó dẫn đến người dân lúng túng khi xử lý các tình huống thực tế dẫn đến gây ùn tắc giao thông hoặc tệ hơn là gây tai nạn giao thông.

Điều đó đặt ra yêu cầu cần có một giải pháp tuyên truyền khác hiệu quả hơn. Giải pháp mới đòi hỏi phải trực quan, giống với thực tế, cho phép người chơi tương tác như đang điều khiển phương tiện tham gia giao thông như trong thực tế và hơn nữa, các tình huống vi phạm phải được phát hiện kịp thời và cảnh báo để người tham gia có thể ghi nhớ.

1.3. Giới hạn đề tài

Nghiên cứu tập trung vào kiến thức game 3D và môi trường phát triển của Unity. Bên cạnh đó nghiên cứu cũng bao hàm các kiến thức về Laravel và cơ sở dữ liệu MongoDB.

Mục tiêu nghiên cứu nhằm xây dựng ứng dụng học luật giao thông chạy trên nền web và hệ thống hỗ trợ quản trị nội dung CMS.

Mục tiêu cụ thể bao gồm:

- Nghiên cứu các kiến thức 3D và môi trường phát triển Unity.
- Nghiên cứu Laravel
- Nghiên cứu cơ sở dữ liệu MongoDB
- Xây dựng ứng dụng học luật giao thông với các tính năng sau:
 - o Game
 - Đăng nhập
 - Chơi game
 - Đăng tải điểm
 - Xem lịch sử
 - o CMS
 - Quản lý nội dung ứng dụng

- Thiết kế nội dung ứng dụng
 - Quản lý tài khoản người dùng
 - Báo cáo
- Map Editor
 - Tạo mới bản đồ
 - Xuất ra file bản đồ
 - Mở lại bản đồ đã lưu

Chương 2. CƠ SỞ CÔNG NGHỆ

2.1. Game engine

Game engine là một bộ ứng dụng được xây dựng để phát triển game, hỗ trợ các tính năng để quá trình phát triển game trở nên đơn giản, dễ dàng và nhanh chóng. Các tính năng được hỗ trợ bởi các game engine thường bao gồm:

- Dựng hình 2D, 3D
- Cảnh sửa mã nguồn
- Âm thanh
- Vật lý
- Tạo hiệu ứng động
- Quản lý bộ nhớ
- Xây dựng trí tuệ nhân tạo
- Kết nối mạng
- ...

Game engine giúp tiết kiệm thời gian và chi phí cho các dự án rất đáng kể.

2.1.1. Các loại game engine

Việc phân loại game engine rất phức tạp vì có rất nhiều tiêu chí để so sánh giữa chúng. Tuy nhiên, dựa vào mức độ chuyên biệt, mức độ hỗ trợ của các engine mà ta chia chúng thành 3 cấp độ khác nhau, từ thấp lên cao: Roll-Your-Own, Most-Ready và Point-And-Click.

❖ Roll-Your-Own

Roll-Your-Own engine được xem như những engine ở mức thấp nhất. Thường thì chúng không phổ biến do được phát triển ở mỗi công ty với những mục đích riêng. Những engine này sử dụng những ứng dụng giao diện được công bố rộng rãi, các API như XNA, DirectX, OpenGL, SDL để phát triển phần đồ họa.Thêm vào đó, chúng có thể dùng những thư viện từ nhiều nguồn khác nhau để bổ sung các tính năng khác. Những thư viện đó có thể bao gồm thư viện vật lý như Box2D,

Chipmunk, thư viện âm thanh như irrKlang, PortAudio và thư viện GUI như AntTweakBar.

Thực tế, Roll-Your-Own engine cho phép những lập trình viên có thể tương tác sâu hơn vào hệ thống, tuỳ biến theo mong muốn một cách dễ dàng. Tuy nhiên, họ cũng sẽ gặp phải rất nhiều lỗi do sự thiếu tương thích giữa các thành phần với nhau.

❖ Most-Ready

Most-Ready engine là những engine tầng trung. Chúng được thiết kế với đầy đủ các tính năng cần thiết, kèm theo đó là rất nhiều công cụ thoả mãn hầu hết các mục đích sử dụng của người dùng. Những engine loại này khá phong phú, bao gồm OGRE, Genesis3D, Torge, Gamebryo và cả Unreal, IdTech.

Những lập trình viên cần lập trình thêm để gắn kết các thành phần trong engine lại với nhau để tạo nên một game hoàn chỉnh. Vì tính chuyên biệt cao hơn, nên với loại engine này các lập trình viên sẽ không còn nhiều khả năng tương tác sâu như các engine loại Roll-Your-Own.

❖ Point-And-Click

Point-And-Click là loại game engine dễ sử dụng nhất. Nói như vậy không có nghĩa là mọi người đều có thể sử dụng chúng để làm ra game, vẫn cần một ít kiến thức lập trình để có thể làm được, tuy nhiên, lượng công việc đó đã được tối giản xuống mức tối thiểu. Các Point-And-Click engine có chứa hầu như tất cả những thứ cần thiết để làm game, người dùng chỉ cần lựa chọn và kéo thả. Các engine loại này bao gồm GameMaker, Torque Game Builder và Unity.

2.1.2. Một số game engine trên thị trường

❖ CryEngine

CryEngine ban đầu được Crytek phát triển như một bản demo công nghệ cho Nvidia. Khi cảm thấy tiềm năng của CryEngine, Crytek đã phát triển nó và tung ra game đầu tiên Far Cry khá thành công. Qua thời gian CryEngine ngày càng phát triển mạnh mẽ hơn với những tựa game nổi tiếng như Far Cry, Crysis, Crysis

Warhead, Crysis 2 và Aion: Tower of Eternity. Phiên bản hiện tại, CryEngine 3 được phát hành vào ngày 4/10/2009. Theo Crytek, CryEngine 3 được tạo ra với tham vọng trở thành công cụ phát triển game tất cả trong một, ứng dụng trên cả Windows, PlayStation 3 và Xbox 360. Riêng đối với nền PC, CryEngine 3 sẽ hỗ trợ tốt cả DirectX 9, 10 và 11.

Điểm nổi trội của CryEngine so với các engine đương thời là tập trung vào mặt xử lý hình ảnh, ngoài ra hiệu ứng âm thanh và chuyển động cũng được mô phỏng xuất sắc.

❖ OGRE

OGRE (Object-Oriented Graphics Rendering Engine) là một engine dựng hình 3D linh hoạt, tập trung vào dựng hình hơn là một công cụ tạo game. Bộ thư viện của OGRE trừu tượng hóa các thư viện ở mức hệ thống như DirectX 3D và OpenGL để cung cấp một giao diện lập trình dựa trên các đối tượng thế giới thực và các lớp cấp cao.

Tuy chỉ tập trung vào dựng hình và không cung cấp các thành phần hỗ trợ âm thanh, vật lý, tuy nhiên đó lại được xem là một ưu điểm của OGRE khi nó cho phép các lập trình viên có thể tùy ý lựa chọn các bộ thư viện để tích hợp vào để đáp ứng yêu cầu của từng dự án.

❖ Panda3D

Panda3D là một game engine mã nguồn mở, một thư viện dựng hình 3D và phát triển game được xây dựng bằng ngôn ngữ Python và C++. Bản thân engine được viết bằng C++ và sử dụng một bộ sinh mã tự động để tạo thành các chức năng hoàn chỉnh cho engine thông qua giao diện của Python. Cách tiếp cận này giúp tận dụng được những ưu điểm của giao diện lập trình Python nhưng vẫn giữ lại được tốc độ của C++ trong nhân của engine. Panda3D thường được dùng trong các dự án ở trường học, hoặc các dự án thương mại cỡ lớn. Những game sử dụng Panda3D như: Toondown Online, Pirates of the Caribbean, A Vampyre Story, Signal Ops.

❖ Unreal Engine

Unreal Engine được phát triển bởi Epic Games, ra mắt vào năm 1998 với game Unreal. Unreal được xem là đối thủ trực tiếp của CryEngine trên thị trường engine làm game FPS. Được xây dựng dựa trên C++ nên Unreal có thể chạy tốt trên hầu hết mọi nền tảng như Windows, Linux, Android, iOS, Mac OS hay các máy chơi game Xbox, PS, Dream Cast.

2.2. Unity

2.2.1. Giới thiệu

Unity là một engine làm game 3D, dựng hình thời gian thực, thuộc loại Point-And-Click, được phát triển bởi Unity Technologies. Unity có thể chạy trên hệ điều hành Windows và Mac OS X. Sản phẩm tạo ra từ Unity có thể chạy trên hầu hết các nền tảng Windows, Mac, Linux, Wii, iOS, Android. Bên cạnh đó, Unity Engine có khả năng phát triển Game nền Web hỗ trợ cả Mac và Windows.

Xét về mặt đồ họa, Unity không phải là engine mạnh nhất, CryEngine vẫn đang dẫn đầu với ưu thế đồ họa 3D cực kỳ chân thực. Tuy nhiên, để có được những trải nghiệm tốt như vậy, CryEngine hay Unreal đều cần những máy cấu hình từ khá đến cao, không giống như Unity, có thể chạy tốt trên những thiết bị cấu hình trung bình. Tuy đồ họa không thể so sánh nhưng chất lượng hình ảnh mà Unity mang lại cũng quá đủ để thỏa mãn những game thủ hiện nay. Hơn nữa, Unity còn có những ưu điểm mà không phải game engine nào cũng có:

❖ Đa nền tảng

Unity được phát triển dựa trên kết quả của Mono Project, một dự án mã nguồn mở đa nền tảng trên ngôn ngữ C#. Chính vì điều đó, Unity có thể chạy tốt trên Windows, Mac OS, Linux, PlayStation 3, Xbox 360, Wii U, iOS, Android, Windows Phone, Blackberry 10, và thậm chí cả web và Flash. Với gói phần mềm của Unity, các studio có thể phát triển các game và phát hành trên nhiều nền tảng một cách nhanh nhất và tiết kiệm nhất.

❖ Hỗ trợ 2D và 3D

Unreal Engine và CryEngine đều chỉ hỗ trợ làm game 3D trong khi Unity hỗ trợ cả game 2D. Đây là một trong những ưu điểm vượt trội của Unity cho phép các studio vừa và nhỏ có thể sử dụng để phát triển các game từ nhỏ đến lớn, tùy vào kinh phí của họ. Phiên bản Unity 5 sắp ra mắt vào năm 2015 hứa hẹn sẽ cung cấp khả năng hỗ trợ game 2D tốt hơn nữa.

❖ Dễ sử dụng

Đối tượng hướng đến của Unity là cả các lập trình viên không chuyên và các studio chuyên nghiệp, do vậy nó được thiết kế để dễ dàng sử dụng nhất. Unity cung cấp một hệ thống toàn diện các công cụ từ thiết kế, soạn thảo mã nguồn đến sửa lỗi. Các tính năng được ưa thích của Unity là cho phép các lập trình viên có thể xây dựng các phần plugin để gắn vào editor của Unity và tính năng xem trước, cho phép nhà phát triển vừa kiểm lỗi, vừa thiết kế.

Hơn nữa Unity là một trong những engine phổ biến nhất thế giới, do đó, các nhà phát triển có thể dễ dàng tìm kiếm sự giúp đỡ từ cộng đồng rất nhanh.

2.2.2. Các tính năng chính

Với mong muốn tạo ra công cụ hỗ trợ tốt nhất, Unity giúp người dùng có thể tập trung phát triển ý tưởng của mình hơn là quan tâm giải quyết các vấn đề kỹ thuật, các khó khăn công nghệ.

Các tính năng chính của Unity bao gồm:

❖ Dụng hình

- DirectX 11 được hỗ trợ làm cho hiệu suất dựng hình được nâng cao đáng kể so với các phiên bản trước
- Kỹ thuật chiếu sáng Pre-Pass và đa luồng cho ra ánh sáng chân thực nhất
- 100 kiểu shader sẵn có, từ đơn giản đến phức tạp phục vụ mọi mục đích của nhà phát triển
- Cho phép can thiệp xuống tầng thấp của quá trình dựng hình để tối ưu hóa

❖ Ánh sáng

- Tích hợp bộ công cụ Beast của Autodesk cho phép thực hiện kỹ thuật Lightmap một cách dễ dàng
- Kỹ thuật Lightmap kép được dùng cho các khung cảnh lớn giúp hạn chế số lượng lightmap được tạo ra
- Hard-shadow và soft-shadow tuỳ vào phần cứng cho trải nghiệm tốt nhất trên từng thiết bị

❖ Kỹ thuật đặc biệt

- Vô số hiệu ứng tiền xử lý toàn màn hình cho phép chất lượng hình ảnh tuyệt vời hơn
- Tăng tính chân thực với các hiệu ứng vẽ lên texture
- Chất lỏng với nhiều trạng thái và màu sắc
- Quản lý nhiều particle cùng lúc hiệu quả với Particle Editor

❖ Âm thanh

- Thao tác đưa âm thanh vào game một cách đơn giản, hiệu quả
- Hỗ trợ streaming với file kích thước lớn
- Cung cấp rất nhiều bộ lọc để tạo ra các hiệu ứng âm thanh khác nhau

❖ Tài nguyên

- Công nghệ Substance của Allegorithmic giúp giảm kích thước tài nguyên và tạo ra các hiệu ứng ấn tượng

❖ Địa hình

- Tạo địa hình đơn giản nhưng hiệu quả hoàn toàn bằng thao tác kéo thả
- Công nghệ Lightmap của Beast cho ánh sáng địa hình được chân thực

❖ Vật lý

- Tích hợp engine vật lý NVIDIA PhysX cho hiệu suất xử lý vật lý cực cao
- Thư viện Box2D được sử dụng cho game 2D giúp giảm chi phí tính toán
- Xây dựng game đua xe đơn giản hơn với Wheel Collider

❖ Trí tuệ nhân tạo

- Bộ thư viện tìm đường đi được tích hợp hỗ trợ phát triển các đối tượng tự động

❖ 2D and 3D

- Unity cho phép sử dụng model từ các phần mềm phổ biến như Maya, 3ds Max, Modo, Cinema 4D, Blender
- Xây dựng các màn chơi đơn giản chỉ với thao tác kéo thả
- Công cụ Profiler cho phép kiểm soát tài nguyên, tối ưu hóa hệ thống tốt hơn
- Hỗ trợ nhiều camera cho phép tích hợp các đối tượng 2D và 3D trong cùng một khung cảnh dễ dàng

❖ Hoạt họa

Từ phiên bản 4.0, Unity đã cung cấp thêm công cụ xây dựng hoạt họa. Các thao tác trực tiếp trên khung xương ở chế độ 2D hay 3D đều đơn giản và dễ dàng.

❖ Hiệu suất

- Giải pháp tiền xử lý Occlusion Culling độc quyền của Unity, phát triển cùng với Umbra Software đảm bảo chỉ những đối tượng được nhìn thấy mới cần phải vẽ
- LODGroups hỗ trợ thực hiện kỹ thuật Level-Of-Detail dễ dàng
- Cơ chế tự động tối ưu bằng cách gắn nhiều khối nhỏ thành một khối lớn
- Cơ chế tự động loại bỏ các tài nguyên không sử dụng
- Bộ tối ưu mã GLSL trên mobile giúp tăng tốc độ lên 2-3 lần

❖ Đa nền tảng

- Tất cả các nền tảng mà Unity hỗ trợ bao gồm: iOS, Android, Windows Phone 8, Blackberry 10, Windows, Windows Store Apps, Mac OS, Linux, Web Player, PS3, PS4, PlayStation Vista, PlayStation Mobile, Xbox One, Xbox 360 và Wii U
- Quá trình build đơn giản với một thao tác

❖ Giao diện người dùng

Bộ thư viện GUI mới của Unity hỗ trợ đầy đủ các loại điều khiển từ cơ bản đến phức tạp, hỗ trợ layout và anchor giúp giao diện phù hợp với nhiều kích thước màn hình.

2.2.3. Ngôn ngữ lập trình

Với Unity, ta có thể sử dụng một hoặc kết hợp ba ngôn ngữ để phát triển là: C#, JavaScript và Boo.

Theo một khảo sát của trang <http://forum.unity3d.com> vào tháng 7/2013 thì ngôn ngữ được sử dụng nhiều nhất trong Unity là C#.

Ngôn ngữ	Tỷ lệ
Boo only	3.28%
C# only	40.10%
JavaScript only	27.92%
Boo & C#	1.97%
Boo & JavaScript	0.52%
JavaScript & C#	25.29%
Boo, C# and JavaScript	0.92%

Bảng 2.1 Tỷ lệ sử dụng các ngôn ngữ trong Unity

Giữa 3 ngôn ngữ này, không có sự khác biệt lớn về tốc độ và hiệu năng, tuy nhiên ta có một vài sự khác biệt giữa chúng.

❖ C#

Ưu điểm:

- Lượng người sử dụng lớn.

- Tốc độ nhanh hơn trong một vài trường hợp.
- Ràng buộc nghiêm ngặt, có thể phát hiện lỗi ngay lúc build.
- .NET Framework hỗ trợ nhiều, có cộng đồng lớn ngoài cộng đồng Unity.

Nhược điểm: Chưa thấy

❖ JavaScript

Ưu điểm:

- Dễ học, dễ hiểu.
- Có lượng người sử dụng lớn.

Nhược điểm:

- Không nghiêm ngặt, chỉ có thể phát hiện lỗi lúc chạy.

❖ Boo

Ưu điểm:

- Đẹp, dễ đọc, dễ hiểu.
- Nghiêm ngặt, có thể phát hiện lỗi ngay lúc build.

Nhược điểm:

- Không có nhiều người sử dụng.

2.2.4. Nền tảng Mono

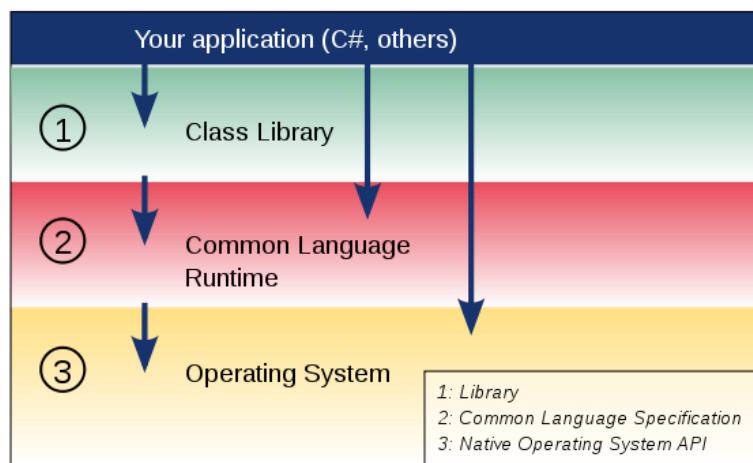


Mono là một phiên bản mã nguồn mở đa nền tảng, hiện thực Microsoft .NET Framework dựa trên tiêu chuẩn ECMA Common Language Runtime. Được tài trợ bởi Novell và hiện tại là Xamarin, dự án Mono có một cộng đồng đóng góp tích cực

và nhiệt tình. Mono bao gồm các công cụ phát triển và cơ sở hạ tầng để có thể chạy được các ứng dụng .NET ở client và server.

Unity được phát triển dựa trên phiên bản Mono 2.6. Hiện tại Unity chỉ hỗ trợ API của .NET 2.0.

2.2.4.1. Kiến trúc



Hình 2.1 Kiến trúc Mono

Ứng dụng được viết bằng ngôn ngữ C# hoặc các ngôn ngữ khác dựa trên các lớp thư viện tiện ích của Mono. Khi được thực thi, các lớp này sẽ gọi đến Common Language Runtime để thực thi các lệnh. Ở đây, tuỳ vào hệ điều hành mà các lệnh khác nhau sẽ được chuyển xuống từng hệ điều hành tương ứng. Do đó, ở khía cạnh nhà phát triển ứng dụng thì các lớp thư viện đều như nhau ở mọi hệ điều hành, họ không cần quan tâm đến chi tiết của mỗi nền tảng trong ứng dụng của mình.

2.2.4.2. Các thành phần

❖ C# Compiler

Mono C# Compiler hỗ trợ đầy đủ tính năng cho C# 1.0, 2.0, 3.0, 4.0 và 5.0.

❖ Mono Runtime

Mono Runtime hiện thực tiêu chuẩn ECMA Common Language Infrastructure (CLI). Máy ảo cung cấp một trình biên dịch Just-In-Time (JIT), một trình biên dịch

Ahead-Of-Time (AOT), một library-loader, một garbage-collector, một threading-system và các thư viện chức năng. Các thư viện đó bao gồm:

- **Base Class Library:** Mono hiện thực tất cả các lớp cơ bản để có thể phát triển hoàn thiện một ứng dụng. Các lớp này hoàn toàn tương thích với các lớp của Microsoft .NET Framework.
- **Mono Class Library:** Mono còn cung cấp các lớp nâng cao, tiện dụng và đặc biệt là tập trung cho các ứng dụng trên nền tảng Linux, ví dụ như các lớp cho Gtk+, Zip file, LDAP, OpenGL, Cairo, POSIX, ...

2.2.4.3. Các tính năng

❖ Đa nền tảng

Mono hỗ trợ cho cả hệ thống 32bit và 64bit của tất cả các kiến trúc máy tính. Các hệ điều hành được hỗ trợ bao gồm: Linux, Mac OSX, iOS, Sun Solaris, OpenBSD, FreeBSD, NetBSD, Microsoft Windows, Nintendo Wii, Sony PlayStation 3

❖ Đa ngôn ngữ

Rất nhiều ngôn ngữ có thể được sử dụng với nền tảng Mono. Bất kỳ ngôn ngữ nào có thể biên dịch được sang ngôn ngữ IL (Intermediate Language) đều có thể chạy được trên nền tảng Mono. Danh sách các ngôn ngữ có thể chạy được, bao gồm: C#, F#, Scala, Java, Boo, Nemerle, Visual Basic .NET, JavaScript, Operon, PHP, Object Pascal, Lua, Cobra và nhiều ngôn ngữ khác.

❖ Tương thích với các API của Microsoft .NET Framework

Mono có thể chạy được các ứng dụng ASP.NET, ADO.NET, SilverLight và Windows.Form mà không cần phải biên dịch lại.

❖ Mã nguồn mở, ứng dụng miễn phí

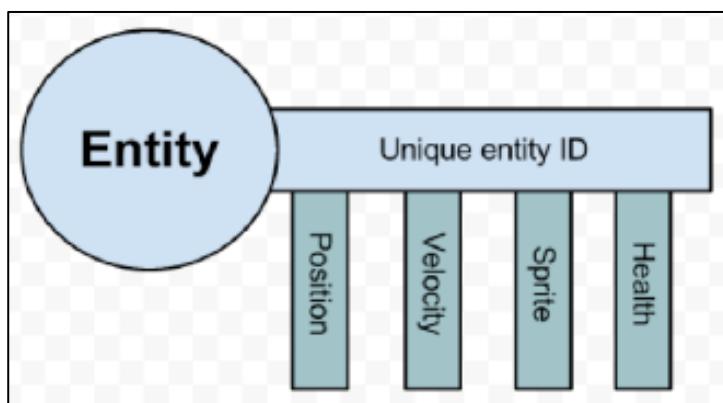
Mono được phân phối dưới các giấy phép sau:

- C# Compiler xuất bản dưới 2 giấy phép MIT/X11 và GPL
- Các công cụ xuất bản dưới giấy phép GPL

- Các thư viện runtime xuất bản dưới giấy phép LGPL 2.0
- Các lớp thư viện xuất bản dưới giấy phép MIT X11
- Các thư viện liên quan với Microsoft được xuất bản dưới giấy phép Microsoft Permissive License, và một vài thư viện có giấy phép Apache2

2.2.5. Kiến trúc Component

Kiến trúc Component là hướng lập trình hướng dữ liệu (Document-Oriented). Trong đó một đối tượng thường được gọi là entity (thực thể). Các thực thể là những đối tượng rõ ràng, không chứa bất kỳ dữ liệu nào riêng biệt và không có bất kỳ sự khác nhau nào giữa chúng. Một thực thể có thể gắn thêm nhiều component khác nhau để thực hiện các mục đích riêng. Mỗi thực thể sẽ mang những thuộc tính và hành vi của component được thêm vào, khi đó các thực thể mới thực sự khác nhau. Trong Unity, các thực thể này gọi là GameObject.



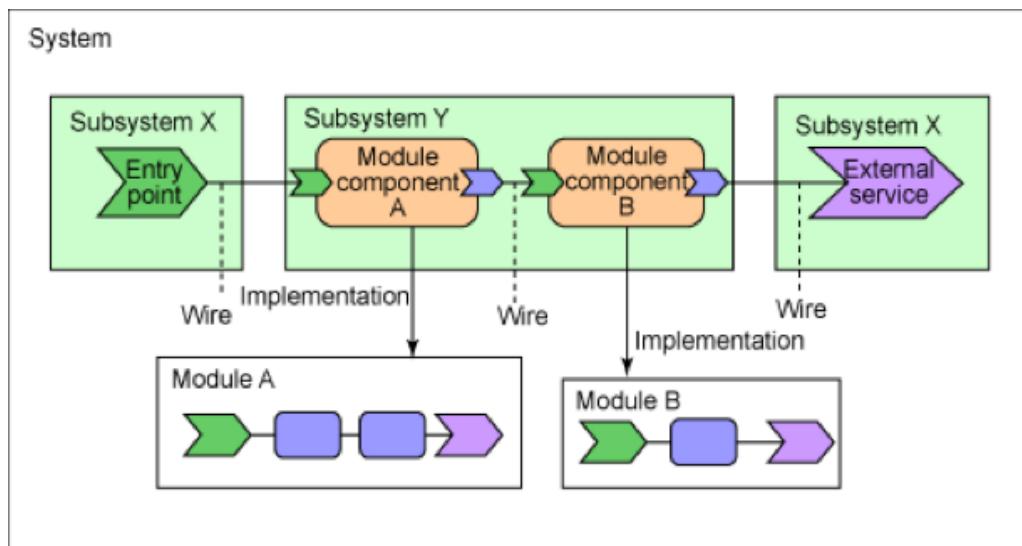
Hình 2.2 Thực thể theo kiến trúc component

Trong thực thể ở trên, có các component là: Position, Velocity, Sprite và Health.

Mỗi component thông thường sẽ có hai thành phần là:

- Attributes: dữ liệu riêng của component.
- Behaviours: các hàm xử lý các dữ liệu.

Mỗi component trong một thực thể sẽ hoàn toàn độc lập với nhau và được thực thi trên những hệ thống riêng biệt, chính vì thế, tốc độ thực thi của kiến trúc Component nhanh hơn kiến trúc cây thừa kế rất nhiều.



Hình 2.3 Các hệ thống xử lý các component

Trong kiến trúc Component hầu như không tạo ra một đối tượng nhất định giống như kiến trúc OOP mà tất cả các đối tượng trong component đều là thực thể. Không có kiến trúc theo cây thừa kế trong kiến trúc Component.

❖ Ưu điểm

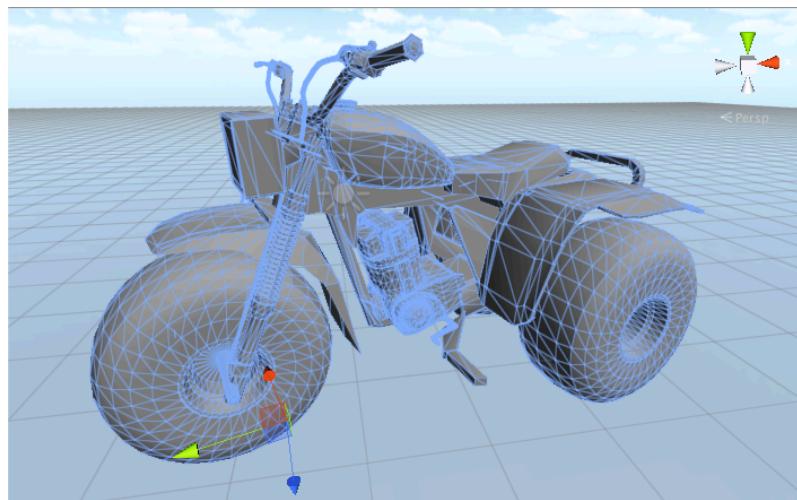
- Tái sử dụng code tốt, giảm độ phức tạp của dự án. Một component có thể được sử dụng cho nhiều thực thể khác nhau có chung một số đặc điểm về component đó.
- Tốc độ xử lý cực nhanh vì mỗi component sẽ chạy trên một hệ thống riêng biệt nhau và không phải trả chi phí cho các hàm thừa kế trừu tượng như trong cấu trúc cây thừa kế.
- Dễ vận hành và bảo trì game: mỗi component gần như hoàn toàn độc lập với nhau nên rất dễ thích nghi với những thay đổi trong game, ít ảnh hưởng tới các component còn lại.
- Dễ dàng cho các dự án có nhiều người cùng tham gia khi mỗi người có thể quản lý mỗi component riêng biệt.

❖ Nhược điểm

- Không thích hợp với các dự án nhỏ.
- Thời gian phát triển ban đầu lớn.

2.2.6. Các Component trong Unity

❖ Nhóm Mesh



Nhóm Mesh dùng để hiển thị các model 3D trong Unity.

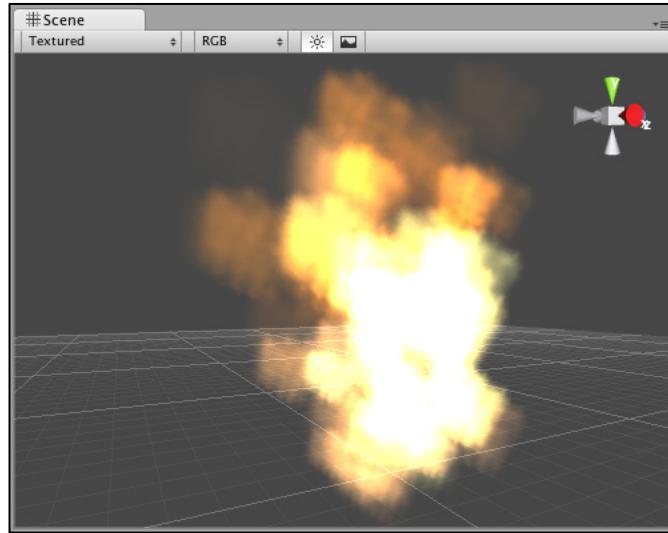
Bao gồm các component:

- Mesh
Lớp dữ liệu, dùng để lưu các model 3D được import từ assets.
- Mesh Filter
Dùng để truyền một Mesh từ assets vào cho Mesh Renderer hiển thị.
- Text Mesh
Một Mesh đặc biệt dùng để hiển thị chữ trong không gian 3D.
- Mesh Render
Dùng để hiển thị Mesh và Text Mesh trong không gian 3D. Một GameObject phải được gắn component Mesh Renderer mới có thể hiển thị được model 3D trong không gian.

Các định dạng model 3D được Unity hỗ trợ gồm 2 loại:

- Các định dạng đã kết xuất: .FBX, .OBJ
- Các định dạng chưa kết xuất: .MAX, .BLEND nhận từ các phần mềm 3D Studio Max hoặc Blender

❖ Nhóm Effect



Tạo ra các hiệu ứng trong game như: khói, hơi nước, hiệu ứng khí quyển... Nhóm này bao gồm các component:

- Particle System

Dùng để hiện thực một hiệu ứng particle nào đó.

- Trail Renderer

Tạo hiệu ứng lưu vết phía sau cho các đối tượng di chuyển trong không gian.

- Line Renderer

Tạo ra các đường vẽ trong không gian, dựa vào các điểm được xác định.

Đường được vẽ không phải là các điểm mà là các texture có độ dày nhất định.

- Lens Flare

Được dùng để thêm vào một nguồn sáng để tạo hiệu ứng cho nguồn sáng đó.

Hiệu ứng được tạo ra giống như khi một camera quay trực tiếp vào một nguồn sáng mạnh.

- Halo

Tạo hiệu ứng vàng sáng xung quanh một đối tượng. Halo được dùng chủ yếu cho hiệu ứng của một điểm sáng (Point Light).

- Projector
Tạo ra hiệu ứng đổ bóng lên một mặt phẳng.

❖ Nhóm Rendering



Tạo các thành phần hiển thị trong không gian 3D, hoặc trong giao diện 2D. Nhóm này bao gồm các component:

- Camera
Dùng để quay lại tất cả các đối tượng có trong không gian đưa lên màn hình. Đây là component không thể thiếu trong game. Nếu không có đối tượng nào chứa component này, sẽ không có gì được thể hiện lên màn hình.
- Flare Layer
Được gắn vào Camera để tạo ra hiệu ứng Lens Flare.
- GUI Layer
Tương tự Flare Layer, cần được gắn vào Camera để thể hiện các GUI Text và GUI Texture.
- GUI Text
Dùng để hiển thị chữ lên màn hình 2D.
- GUI Texture
Dùng để hiển thị Texture lên màn hình 2D.

- Light

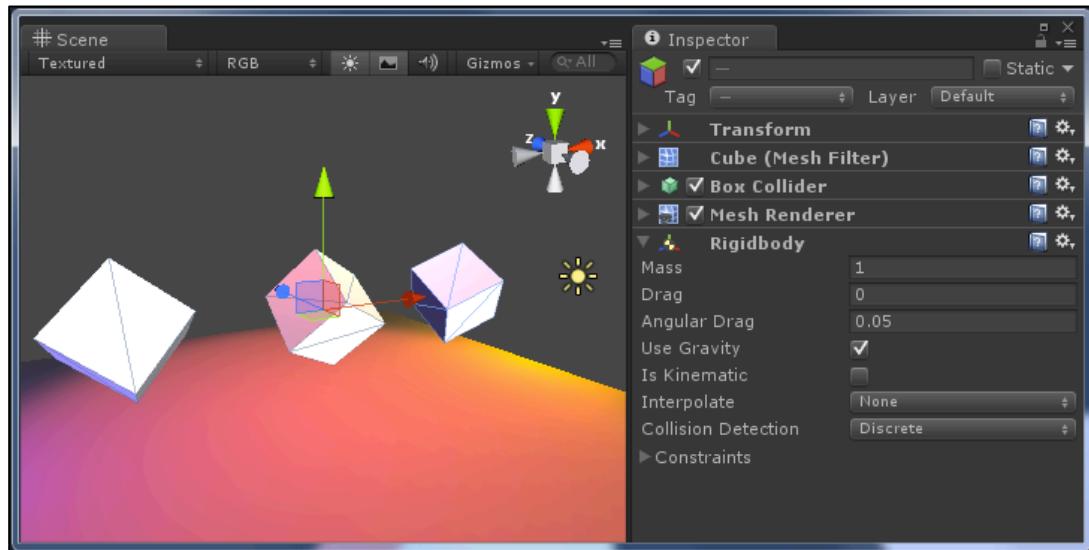
Tạo ra một nguồn sáng, chiếu sáng các thành phần hoặc toàn bộ các đối tượng có trong không gian. Có 4 loại Light:

- o Direction: nguồn sáng song song, chiếu từ một nơi rất xa, có tác dụng với tất cả các đối tượng trong không gian. Ví dụ: ánh sáng mặt trời.
- o Point: điểm sáng, chiếu từ 1 điểm ra tất cả các hướng. Ví dụ: bóng đèn tròn.
- o Spot: nguồn sáng hình chóp. Ví dụ: đèn pin.
- o Area: Chỉ có tác dụng khi tạo light mapping, và không có tác dụng lúc thực thi.

- Skybox

Dùng để giả lập một không gian vô hạn bao quanh toàn bộ không gian. Ví dụ: bầu trời.

❖ Nhóm Physics



Unity sử dụng NVIDIA PhysX để giả lập các tương tác vật lý trong game. Để tạo các chuyển động vật lý, va chạm, phản hồi, ... cần gắn một component physic vào đối tượng. Các component nhóm này bao gồm:

- Rigidbody

Cho phép các đối tượng có thể chịu tác động của lực, bao gồm cả trọng lực trái đất, lực cản không khí... Đối với các đối tượng tĩnh, không cần phản ứng với lực thì không cần tới component này.

- Character Controller

Được dùng để gắn vào các đối tượng mà người chơi điều khiển trong game như: nhân vật góc nhìn thứ nhất, góc nhìn thứ 3. Đối tượng này không chịu sự tác động của các lực vật lý để đảm bảo người chơi có thể di chuyển giống thực, nhưng nó có khả năng gây ra lực lên các đối tượng khác khi va chạm.

- Constant Force

Tạo ra một lực lập tức tác động liên tục lên đối tượng ở mọi khung hình thay vì chỉ một khung hình nếu sử dụng hàm *Rigidbody.AddForce*. Được dùng nhiều cho các loại tên lửa hay viên đạn bay.

- Collider

Cho phép các đối tượng có thể va chạm với nhau. Nếu không có một Collider nào được gắn vào đối tượng, thì chúng sẽ di chuyển xuyên qua nhau. Có các loại Collider sau:

- o Box Collider: Khối lập phương
- o Sphere Collider: Khối cầu
- o Capsule Collider: Khối trụ có 2 đầu tròn.
- o Mesh Collider: Khối được tạo nên từ một Mesh.
- o Wheel Collider: Một dạng đặc biệt, dùng cho các loại xe.

❖ Nhóm Audio

Cung cấp các component liên quan đến âm thanh trong game. Bao gồm các component:

- Audio Listener

Thể hiện một chiếc micro để lắng nghe các âm thanh có trong không gian 3D và thường được gắn vào Camera chính. Trong môi trường 2D thì có thể gắn

ở bất kỳ đâu cũng được. Ở một scene, chỉ được phép tồn tại một Audio Listener.

- **Audio Source**

Đại diện cho một đối tượng phát ra âm thanh trong không gian. Trong môi trường 3D, nếu một Audio Source có vị trí càng gần Audio Listener thì âm thanh mà người dùng nghe được sẽ càng to hơn.

❖ Nhóm Terrain



Tạo ra địa hình cho game như mặt đất, đường, đồi núi, cây cối, biển, sông suối,... Nhóm này chỉ có một component là Terrain. Các thao tác chỉnh Terrain đều được thực hiện trên giao diện đồ họa.

2.2.7. Script

Trong Unity, một file script định nghĩa một class cũng được xem là một component khi class đó thừa kế từ class MonoBehaviour.

Vì script là một phần không thể thiếu và được dùng vô cùng phổ biến, nên MonoBehaviour cũng là một class vô cùng quan trọng. Đây là class sẽ được hầu hết các class của người dùng thừa kế. Với JavaScript, tất cả các file script đều tự động thừa kế từ class này, tuy nhiên, với C# và Boo thì việc thừa kế phải được thực hiện tinh minh.

Các thành phần quan trọng của class này bao gồm.

Thuộc tính:

Tên	Ý nghĩa
enabled	Nếu enabled thì hàm Update của MonoBehaviour sẽ được gọi mỗi frame.
tag	Tag của GameObject, dùng để phân nhóm các GameObject.
name	Tên của GameObject, dùng để định danh các GameObject.

Bảng 2.2 Các thuộc tính của MonoBehaviour

Phương thức:

Tên	Ý nghĩa
GetComponent	Trả về component được gắn vào GameObject.
GetComponentInChildren	Trả về component được gắn vào GameObject hoặc bất kỳ đối tượng con của GameObject.
SendMessage	Gọi phương thức của GameObject theo tên có trong bất kỳ MonoBehaviour nào của đối tượng đó.
Invoke	Gọi phương thức theo tên sau một khoảng thời gian.
InvokeRepeating	Gọi lặp lại phương thức theo tên sau một khoảng thời gian.
CancelInvoke	Hủy hoặc dừng việc thực hiện 2 hàm trên.
IsInvoking	Kiểm tra phương thức có được chờ để gọi.
StartCoroutine	Bắt đầu một coroutine (tương tự thread).
StopCoroutine	Dừng tất cả coroutine chạy trên phương thức có tên này.
StopAllCoroutines	Dừng tất cả coroutine chạy trên đối tượng này.

Bảng 2.3 Các phương thức của MonoBehaviour

Bên cạnh đó, có một vài hàm rất quan trọng, thường được sử dụng để thửa kẽ lại trong các script của người dùng.

Tên	Ý nghĩa
Update	Sẽ được gọi ở mỗi frame, nếu MonoBehaviour được kích hoạt.
LateUpdate	Sẽ được gọi ở mỗi frame, sau hàm Update nếu MonoBehaviour được kích hoạt.
FixedUpdate	Được gọi ở mỗi frame cố định, nếu MonoBehaviour được kích hoạt, được dùng chủ yếu cho các xử lý vật lý, cần thời gian chính xác.
Awake	Được gọi khi đối tượng được load (trước hàm Start) và một lần duy nhất cho một đối tượng, thường được dùng để thực hiện các tiền xử lý.
Start	Được gọi đầu tiên (sau hàm Awake), thường được dùng để khởi tạo đối tượng.
Reset	Thiết lập lại các giá trị mặc định.
OnMouseEnter	Sẽ được gọi một lần khi đưa chuột vào GUIElement hoặc Collider.
OnMouseOver	Sẽ được gọi mỗi frame khi chuột nằm trên GUIElement hoặc Collider.
OnMouseExit	Sẽ được gọi một lần khi đưa chuột ra khỏi GUIElement hoặc Collider.
OnMouseDown	Sẽ được gọi lúc người dùng nhấn chuột và chuột đang nằm trên GUIElement hoặc Collider.
OnMouseUp	Sẽ được gọi khi thả chuột.

OnMouseDown	Được gọi khi click lên GUIElement hoặc Collider và giữ trạng thái đó rồi di chuyển chuột.
OnTriggerEnter	Được gọi một lần khi các collider khác bắt đầu va chạm với trigger.
OnTriggerStay	Được gọi một lần cho từng frame cho tất cả các collider khác đang chạm vào trigger.
OnTriggerExit	Được gọi một lần khi collider khác không chạm vào trigger nữa.
OnCollisionEnter	Được gọi một lần khi collider/rigidbody bắt đầu chạm vào một collider/rigidbody khác.
OnCollisionStay	Được gọi một lần mỗi frame cho tất cả các collider/rigidbody khác va chạm với collider/rigidbody này.
OnCollisionExit	Được gọi một lần khi collider/rigidbody khác không còn va chạm với collider/rigidbody này.

Bảng 2.4 Các phương thức cần thừa kế của MonoBehaviour

2.2.8. Physics

Vật lý trong game là tất cả những thứ liên quan đến sự chuyển động và tác động qua lại của các vật thể nhằm mô phỏng như trong thế giới thật, bao gồm sự chuyển động phức tạp của các vật thể, sự va chạm, phản hồi, ma sát, dòng chảy... dựa trên sự tác động của lực đẩy, trọng lực, gió...

Thông thường, CPU đảm nhiệm vai trò tính toán logic và vật lý trong game, GPU chỉ nhận nhiệm vụ dựng hình cho game. Chỉ những máy tính chơi game chuyên dụng, có card vật lý (PPU) thì các tính toán vật lý mới được thực hiện chuyên nghiệp ở đây.

PhysX	Core 2 Quad	GeForce GTX 280
Cores	4	240
GFLOPS	96	930
Fluids	1	15
Soft Bodies	1	12x
Cloth	1	13x

Bảng 2.5 So sánh tốc độ xử lý vật lý của CPU và GPU

Một lý do chính để chuyển việc tính toán vật lý từ CPU sang GPU là do sức mạnh to lớn trong tính toán của GPU cao hơn CPU rất nhiều lần. Qua bảng so sánh bên trên có thể thấy ưu thế rõ rệt của GPU so với CPU trong việc xử lý các hiệu ứng vật lý.

Hiện nay chỉ duy nhất engine vật lý PhysX của NVIDIA là có thể vừa được xử lý thông qua CPU lại vừa được xử lý thông qua GPU trong khi những engine như Havok, Box2D, Chipmunk lại chỉ có một cách xử lý duy nhất là thông qua CPU.

Unity sử dụng engine vật lý NVIDIA PhysX, điều này cho phép Unity có thể trình diễn những chức năng và hiệu ứng vật lý rất mượt mà và độc đáo.

Các tính năng của PhysX bao gồm:

- Giả lập đối tượng vật lý
- Phát hiện va chạm
- Điều khiển nhân vật
- Hiệu ứng particles
- Phương tiện giao thông
- Các loại trang phục

Với việc tích hợp hệ thống engine NVIDIA PhysX, Unity đã kế thừa được sức mạnh của hệ thống này để hỗ trợ các xử lý vật lý của mình mượt mà nhất.

2.2.8.1. Các Component vật lý trong Unity

❖ Rigidbody

Để làm cho một đối tượng chịu ảnh hưởng của các yếu tố vật lý (trọng lực, va chạm,...) thì ta chỉ cần thêm component Rigidbody cho đối tượng đó.

Rigidbody có các thuộc tính vật lý, được dùng cho những GameObject có khả năng chịu tác động lực và tác động lực lên đối tượng khác.

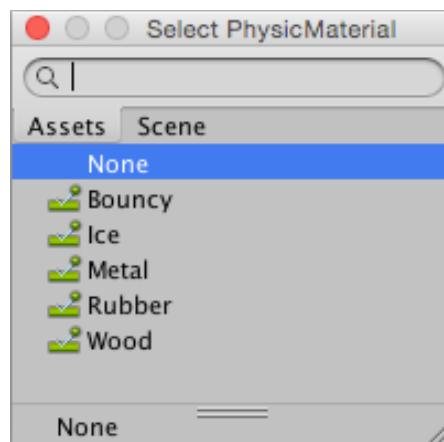
Các thuộc tính bao gồm:

Tên	Ý nghĩa
Mass	Khối lượng của đối tượng.
Drag	Sức cản không khí lên đối tượng khi di chuyển.
Angular Drag	Sức cản không khí lên đối tượng khi quay.
Use Gravity	Nếu được kích hoạt, đối tượng sẽ chịu tác động bởi trọng lực.
Is Kinematic	Nếu được kích hoạt, đối tượng sẽ không chịu tác động bởi hệ thống vật lý, nó chỉ được điều khiển thông qua Transform.
Interpolate	Chế độ làm mượt chuyển động. Gồm các tham số: <ul style="list-style-type: none">- None: Không áp dụng- Interpolate: Transform sẽ mượt mà dựa trên Transform của khung hình trước đó.- Extrapolate: Transform sẽ mượt mà dựa trên dự đoán Transform của khung hình kế tiếp.
Collision Detection	Được dùng để phát hiện va chạm trong các trường hợp vật di chuyển quá nhanh.
Constraints	Giới hạn phạm vi chịu tác động lựu của các Rigidbody, gồm giới hạn di chuyển và giới hạn quay theo 3 trục.

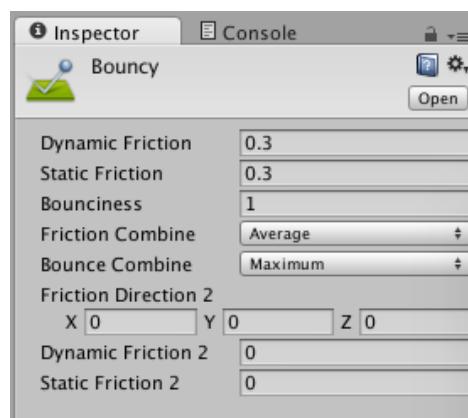
Bảng 2.6 Các thuộc tính của Rigidbody

❖ Physic Material

Các tính chất của vật liệu như độ ma sát, độ đàn hồi, độ cứng... được định nghĩa trong Physics Material. Standard Assets có chứa hầu hết các loại vật liệu vật lý phổ biến. Ta cũng có thể tạo một loại vật liệu riêng bằng cách tùy chỉnh các thông số vật lý cho vật liệu.



Hình 2.4 Các loại Physic Material mặc định trong Unity
Các Physic Material đều có các thuộc tính chung sau:



Hình 2.5 Các thuộc tính chung của các Physic Material

Thuộc tính:

- *Dynamic Friction*: Ma sát động hay ma sát khi đã di chuyển. Có giá trị từ 0 đến 1. Giá trị 0 sẽ làm đối tượng trượt hoàn toàn trên bề mặt, giá trị 1 sẽ làm cho nó dừng nhanh chóng trừ khi có một lực lớn tác động lên vật.

- *Static Friction*: Ma sát nghỉ hay ma sát khi đứng yên. Có giá trị từ 0 đến 1. Giá trị 0 sẽ làm đối tượng trượt hoàn toàn trên bề mặt, giá trị 1 sẽ làm đối tượng khó có thể di chuyển.
- *Bounciness*: Độ đàn hồi. Giá trị 0 là không đàn hồi, giá trị 1 là đàn hồi hoàn toàn mà không mất năng lượng.
- *Friction combine mode*: Công thức xác định giá trị ma sát của 2 đối tượng va chạm khi kết hợp:
 - o Average: giá trị ma sát là trung bình cộng của cả 2.
 - o Min: giá trị nhỏ nhất trong 2 giá trị.
 - o Max: giá trị lớn nhất trong 2 giá trị.
 - o Multiply: tích của 2 giá trị.
- *Bounce combine*: Xác định giá trị đàn hồi của 2 đối tượng va chạm khi kết hợp, giống như Friction Combine mode.
- *Friction Direction 2*: Ma sát không cùng hướng được kích hoạt nếu biến này có giá trị khác Vector Zero. Dynamic Friction 2 và Static Friction 2 sẽ được áp dụng theo Friction Direction 2.
- *Dynamic Friction 2*: Nếu ma sát không cùng hướng được kích hoạt, Dynamic Friction 2 sẽ được áp dụng theo Friction Direction 2.
- *Static Friction 2*: Nếu ma sát không cùng hướng được kích hoạt, Static Friction 2 sẽ áp dụng theo Friction Direction 2.

❖ Các component Collider

Collider là một loại component phải được thêm vào cùng với rigidbody để cho phép va chạm xảy ra. Nếu 2 rigidbody va đập vào nhau, hệ thống vật lý sẽ không tính toán va chạm trừ khi cả hai đối tượng đều có gắn một component Collider.

Trong các trường hợp khác, Collider được sử dụng để xác định vùng không gian xung quanh vật thể để nhận các sự kiện mouse, touch hoặc để hàm raycast xác định được chính xác đối tượng.

Có các loại Collider gồm:

Tên	Miêu tả
Box Collider	Hình dáng cơ bản của một khối hộp.
Sphere Collider	Hình dáng cơ bản của một khối cầu.
Capsule Collider	Hình dáng cơ bản của một khối trụ, hai đầu tròn (hình con nhộng).
Mesh Collider	Một Collider được tạo ra từ một Mesh của GameObject, không thể va chạm với Mesh Collider khác.
Wheel Collider	Một loại Collider đặc biệt sử dụng cho đối tượng xe cộ trong game.

Bảng 2.7 Các loại Collider

❖ Joint

Joint được dùng để kết nối các đối tượng vật lý trong Unity lại với nhau, tạo thành một hệ chuyển động, tương tác với nhau. Một ví dụ như: con lắc đơn, lò xo, ...

Có ba loại Joint cơ bản: Fixed Joint, Spring Joint và Hinge Joint với các mục đích sử dụng khác nhau.

- Fixed Joint: Gắn liền các đối tượng lại với nhau
- Spring Joint: Kết nối các đối tượng dạng lò xo
- Hinge Joint: Kết nối các đối tượng dạng sợi dây

❖ Constant Force

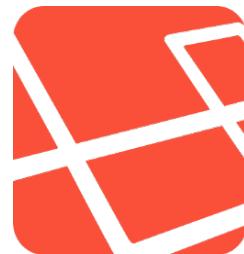
Constant Force giúp ta tạo lực tác động liên tục vào Rigidbody. Điều này hoạt động tuyệt vời cho những đối tượng tăng tốc tức thời như tên lửa, viên đạn, ...

Thuộc tính:

- *Force*: Vector lực tác dụng trong hệ tọa độ thế giới.
- *Relative*: Vector lực tác dụng trong hệ tọa độ địa phương.
- *Torque*: Vector của mô-men xoắn tác dụng trong hệ tọa độ thế giới.

- *Relative Torque*: Vector của mô-men xoắn tác dụng trong hệ tọa độ địa phương.

2.3. Laravel

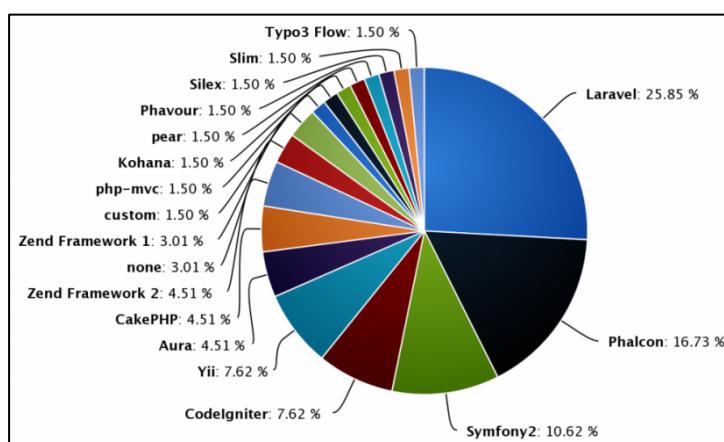


2.3.1. Giới thiệu

Laravel là một framework phát triển web miễn phí, mã nguồn mở, được thiết kế theo mô hình MVC, xuất bản dưới giấy phép MIT.

Laravel ra đời lần đầu vào ngày 22/2/2012 bởi Taylor Otwell. Theo một khảo sát của các lập trình viên, Laravel là framework PHP được sử dụng nhiều nhất trên thế giới trong năm 2013 và tiếp tục trong năm 2014, đứng sau nó là Phalcon, Symfony2, CodeIgniter và Yii.

Laravel 4 là phiên bản mới nhất của Laravel, ra đời rất muộn, vào 28/5/2013, chính vì vậy mà framework này đã kế thừa lại được những thế mạnh của các framework khác. Laravel đã nhanh chóng có được một cộng đồng rất lớn trong thế giới các framework của ngôn ngữ lập trình PHP.



Hình 2.6 Tỷ lệ sử dụng các framework PHP năm 2013

Sự thành công của Laravel trước hết là nhờ sự tinh tế nằm ở chỗ bắt kịp được xu hướng công nghệ mà điểm nhấn ở đây là các tính năng mới trong các phiên bản PHP 5.3 trở lên. Điều đó được thể hiện qua khái niệm namespace, composer, closure và rất nhiều những tiêu chuẩn trong design pattern được áp dụng trên nền tảng framework này. Đồng thời, với cách hướng dẫn đơn giản đã khiến người dùng dễ dàng tiếp cận và sử dụng.

Laravel 4 cũng có sự tích hợp một phần của thư viện Symfony và áp dụng triệt để mô hình ORM với khái niệm liên quan đến Eloquent class. Đồng thời, nó cũng giải quyết được những vấn đề mà các framework khác đang mắc phải, chẳng hạn như master layout, mô hình xử lý với ORM, event model,....

2.3.2. Một số ưu điểm

- Route trong Laravel thật sự khác biệt, mới mẻ và đầy mạnh mẽ.
- Master layout được tích hợp sẵn cùng Blade template giúp code trở nên gọn gàng và tiện dụng.
- Migration quản lý database thật dễ dàng khi làm việc đội nhóm.
- Eloquent class đầy mạnh mẽ nổi bật khi xử lý cơ sở dữ liệu quan hệ 1-N và N-N.
- Composer quản lý và tích hợp các thư viện khác nhanh chóng, đơn giản và không lo lắng khi thư viện đó bị thay đổi.
- Document dễ đọc và dễ hiểu.
- Eloquent ORM: đây là một ORM tuyệt vời với khả năng migration data và làm việc tốt với MySQL, Postgres, SQL Server và SQLite.

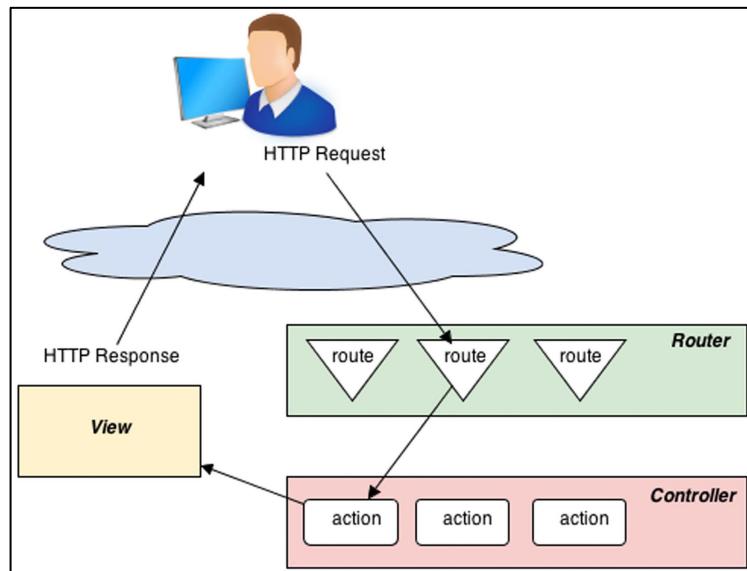
Và còn nhiều chức năng tuyệt vời khác...

2.3.3. Request LifeCycle

Vòng đời tiêu chuẩn của một request khi gửi đến Laravel như sau:

- Yêu cầu từ phía người dùng gửi tới Route
- Yêu cầu HTTP từ Route tới một Controller
- Controller sẽ thực hiện những action và gửi kết quả tới View

- View sẽ thể hiện những kiểu dữ liệu phù hợp và gửi lại thông qua HTTP Response



Hình 2.7 Vòng đời một Request trong Laravel

Tuy nhiên, có rất nhiều ngoại lệ xảy ra trong vòng đời tiêu chuẩn này.

Ví dụ:

- Route có thể trả về trực tiếp Views hay Responses, bỏ qua Controller.
- Một hoặc nhiều Filter có thể xảy ra trước hoặc sau Route để thực hiện việc kiểm tra các ràng buộc.

2.3.4. Route

Đây là một định nghĩa khá quen thuộc khi làm việc với các framework PHP, tuy nhiên, ở Laravel, Route thật sự hơn hẳn. Mục đích của Route là định tuyến request của người sử dụng đến những Controller tương ứng. Tuy nhiên, ta có thể không cần thiết phải tạo ra Controller mà thực thi những công việc mong muốn 1 cách trực tiếp tại Route dễ dàng theo phương pháp closure quen thuộc của JavaScript.

Để làm việc được với Route trước hết ta tiếp xúc với công thức cơ bản sau:

```
Route::method ('Tên định danh', Tham số);
```

Trong đó:

- *Method*: xây dựng dựa trên RESTful, có 5 method sau:
 - o *POST*: dành cho các thao tác lấy dữ liệu từ form như thêm bản ghi
 - o *GET*: dành cho các thao tác truy cập thông thường
 - o *PUT*: dành cho các thao tác lấy dữ liệu từ form nhưng là cập nhật bản ghi
 - o *DELETE*: dành cho các thao tác thực thi hành động xóa bỏ
 - o *ANY*: là sự tổng hợp của các thao tác ở trên
- *Tên định danh*: là đường dẫn mong muốn trong ứng dụng.
- *Tham số*: là những tuỳ biến mà chúng ta mong muốn với các định danh trên. Tham số có thể là một hàm xử lý, có thể là một array() chứa các thông tin xử lý khác, có thể là một action trong một Controller sẽ xử lý,...

Có nhiều cách truyền tham số, ta hoàn toàn có thể kết hợp chúng theo những mong muốn tùy thích, trong đó ta quy về 2 cách phổ biến là:

- Truyền vào dạng closure xử lý trực tiếp, không cần qua Controller
- Truyền vào tên một action và một Controller sẽ xử lý

2.3.5. View

View là các file .php trong thư mục *app/views*, là chữ V trong mô hình MVC dùng để tách biệt những đoạn mã xử lý logic với những đoạn mã hiển thị. Để sử dụng view, rất đơn giản:

```
View::make("Tên", Đối số nếu có);
```

Trong đó *Tên* là tên file (không tính phần mở rộng “.php”) trong thư mục *app/views*, sau đó là phần đối số được truyền vào cho view.

Có nhiều cách truyền đối số vào cho view.

- Sử dụng đối số truyền trực tiếp
- Sử dụng phương thức *with*
- Sử dụng phương thức *compact* trong PHP

2.3.6. Controller

Controller là chữ C trong mô hình MVC của Laravel, Controller là phần trung gian giao tiếp với Model (M) và View (V). Laravel cho phép vận hành trực tiếp trong Route thông qua cách viết closure, nhưng khi phải xử lý các thao tác phức tạp và nhiều thì tốt hơn hết nên thao tác chúng trên Controller. Có như thế thì ứng dụng mới trở nên linh hoạt và dễ mở rộng sau này.

Để thao tác với Controller, trước hết cần tạo ra trong thư mục *app/controllers* một file theo định dạng: *TênController.php*. Ví dụ: *DemoController.php*

Trong file trên ta xây dựng những đoạn lệnh mở đầu như sau:

```
<?php  
class DemoController extends BaseController {  
    public function index() {  
        return View::make("demo.index")->with("title", "Hello");  
    }  
}
```

Mã lệnh ở trên tạo ra *DemoController* kế thừa lớp chủ đạo là *BaseController* và phương thức đầu tiên ta xây dựng là *index()*. Trong phương thức này ta thực hiện việc gọi một View từ thư mục *views/demo*, nạp tập tin *index.php*. Đồng thời ta truyền sang View một biến tên *title* với giá trị là "Hello".

Kế tới, ta tạo ra file *index.php* trong thư mục *views/demo* với mã lệnh sau:

```
<?php  
echo $title;  
?>
```

Sau cùng, ta cần khai báo trong file *routes.php* như sau:

```
<?php  
Route::get("demo", "DemoController@index");  
?>
```

Chạy đường dẫn như sau: *localhost/laravel/public/demo* ta sẽ thấy thông báo hiển thị như mong đợi.

2.3.7. Filter

Filter có thể hiểu một cách đơn giản là bộ lọc, nhiệm vụ của nó là lọc dữ liệu từ phía người dùng trước hoặc sau khi nó chạm tới Route. Vì thế, nếu tận dụng tốt được Filter sẽ tạo ra bộ ứng dụng phòng thủ chuyên nghiệp trong website. Điều này sẽ giúp xử lý dữ liệu tối ưu hơn.

Với Filter của Laravel thì mọi chuyện thật dễ dàng, ta có thể làm điều đó một cách độc lập trên Filter, sau đó lồng ghép một cách khéo léo vào Route mong muốn. Thì ngay khi request tới Route sẽ bị Filter chặn lại và xử lý. Nếu hợp lệ thì mới tiếp tục đi tới Route, còn nếu không thì sẽ thực thi việc gì đó dễ dàng. Với Filter ta có thể tối giản code ở mức cao nhất và tái sử dụng vô cùng đơn giản.

Để làm việc với filter, ta thiết lập một filter như sau:

```
Route::filter('Tên', Hành xử lý);
```

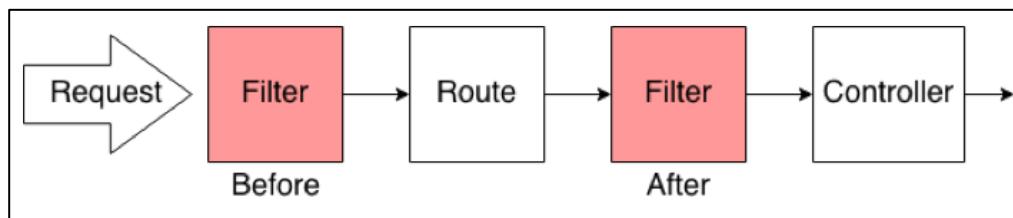
Trong đó, *Tên* là tên của filter, *Hành xử lý* là hàm sẽ được gọi để xử lý dữ liệu trong filter.

Sau đó, filter được lồng ghép vào Route như sau:

```
Route::method('Tên định danh', array('before'=>'Tên', Tham số));
```

Trong đó, *Tên* là tên của filter vừa tạo, *Tên định danh* và *Tham số* giống như trong Route.

Filter có thể đặt trước Route hoặc sau Route đều được, tùy vào mục đích sử dụng.



Hình 2.8 Request Filter trong Laravel

2.4. MongoDB

2.4.1. NoSQL

NoSQL là 1 dạng CSDL mã nguồn mở không sử dụng T-SQL để truy vấn thông tin. NoSQL viết tắt bởi: None-Relational SQL, hay có nơi thường gọi là Not-Only SQL.

NoSQL được phát triển trên JavaScript framework với kiểu dữ liệu JSON và dạng dữ liệu theo kiểu key-value. NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu sót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS về tốc độ, tính năng, khả năng mở rộng, memory cache,...

❖ Ưu điểm

Khi sử dụng SQL Server hoặc MySQL thì giữa các bảng trong CSDL sẽ có các mối quan hệ khóa chính-khóa ngoại vì vậy sẽ gặp nhiều khó khăn trong việc mở rộng dữ liệu sau này. Nhưng trong NoSQL thì khác, ta có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc, ...

❖ Nhược điểm

Vì NoSQL không hạn chế việc mở rộng dữ liệu nên tồn tại nhiều nhược điểm như: sự phụ thuộc của từng bản ghi, tính nhất quán, tính toàn vẹn dữ liệu,... nhưng chúng ta có thể chấp nhận những nhược điểm đó để khiến ứng dụng có được hiệu suất cao hơn khi giải quyết những bài toán lớn về hệ thống thông tin, phân tán hay lưu trữ dữ liệu.

❖ Sử dụng

NoSQL được sử dụng ở đâu?

NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ Facebook sử dụng Cassandra do Facebook phát triển, Google phát triển và sử dụng BigTable,... và rất nhiều công ty lớn khác.

2.4.2. MongoDB

MongoDB là 1 hệ thống CSDL mã nguồn mở được phát triển và hỗ trợ bởi 10gen (hiện tại là MongoDB Inc), là CSDL NoSQL hàng đầu được hàng triệu người sử dụng. Vì được viết bởi C++ nên nó có khả năng tính toán với tốc độ cao, hơn hẳn các hệ quản trị CSDL hiện nay.

Thay vì lưu trữ dữ liệu dưới dạng bảng và các tuple như trong các CSDL quan hệ thì nó lưu trữ dữ liệu dưới dạng JSON (trong MongoDB được gọi là dạng BSON vì nó lưu trữ dưới dạng binary từ 1 JSON document).

Hiện nay MongoDB đang được sử dụng tại một số công ty lớn như: MTV Networks, Craigslist, Foursquare, Forbes, Expedia, Bosch, và nhiều công ty lớn khác.

2.4.2.1. Các khái niệm

❖ Collection

Collection trong MongoDB là nhóm các tài liệu (document), nó tương đương với một bảng (table) trong CSDL thông thường nên mỗi Collection sẽ thuộc về một database duy nhất. Tuy nhiên, có một sự khác biệt đó là không có ràng buộc giữa các Collection như trong các hệ quản trị CSDL khác nên việc truy xuất rất nhanh. Đồng thời mỗi Collection có thể chứa nhiều loại dữ liệu khác nhau không giống như các field cố định trong table của MySQL.

❖ Document

Document trong MongoDB có cấu trúc tương tự như kiểu dữ liệu JSON, nghĩa là sẽ có các cặp key-value nên nó có tính năng động rất lớn. Ta có thể hiểu Document giống như các record dữ liệu trong MySQL, tuy nhiên, có một sự khác biệt là các cặp key-value có thể không giống nhau ở mỗi Document.

Để rõ hơn, ta có bảng so sánh các thuật ngữ giữa Cơ sở dữ liệu quan hệ (RDBMS) và MongoDB:

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
Column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (mặc định là <code>_id</code>)

Bảng 2.8 So sánh các thuật ngữ trong RDBMS và MongoDB

2.4.2.2. Tính năng chính

❖ Document-Oriented

Thay vì chia nhỏ các thuộc tính của một đối tượng để lưu trong các bảng khác nhau như trong SQL thì MongoDB lưu toàn bộ đối tượng trong một Document của một Collection. Ví dụ: Thay vì lưu *TênSách* và *TácGiả* ở hai bảng khác nhau như trong MySQL, thì MongoDB lưu toàn bộ đối tượng *Sách* vào một Document của một Collection, trong đó chứa toàn bộ các thông tin của đối tượng *Sách*.

❖ Server-side JavaScript Execution

JavaScript có thể được sử dụng ở phía server để thao tác trực tiếp với MongoDB.

❖ Indexing

Tất cả các key-value trong MongoDB đều có thể được dùng để index. Việc index một key-value giúp tăng tốc độ truy vấn đối với key-value đó lên rất nhiều lần.

❖ Ad hoc queries

MongoDB cho phép tìm kiếm theo key, theo miền giá trị, sử dụng regular expression để tìm kiếm. Truy vấn có thể trả về một vài key-value của dữ liệu, và thậm chí có thể trả về một hàm được người dùng định nghĩa.

❖ Replication

MongoDB cung cấp cơ chế tạo bản sao hiệu quả, qua đó tăng cường khả năng phân tán dữ liệu, một trong những tính năng được sử dụng rất nhiều trong những hệ thống lớn hiện nay.

❖ Load Balancing

MongoDB cho phép mở rộng cơ sở dữ liệu theo chiều dọc dựa trên cơ chế phân mảnh (Shard). Người dùng có thể chọn một *Shard Key* để xác định cách mà dữ liệu được phân mảnh. Dữ liệu sẽ được cắt nhỏ thành những miền giá trị và phân ra nhiều mảnh. MongoDB cho phép chạy trên nhiều server, cho phép tạo bản sao để phòng ngừa các trường hợp một server trong hệ thống gặp lỗi.

❖ File Storage

MongoDB cho phép phân tán các file nhằm tăng cường khả năng Load Balancing và Replication dữ liệu.

❖ Capped Collections

MongoDB cho phép tạo ra các Collection có kích thước cố định, và khi dữ liệu đầy, nó sẽ được ghi đè ở đầu bên kia của Collection. Cơ chế giống như hàng đợi vòng tròn (circular queue).

2.4.2.3. Sử dụng

MongoDB thật sự rất tốt nhưng không phải lúc nào ta cũng sử dụng, những trường hợp nên sử dụng:

- Nếu website có tính chất INSERT cao, bởi vì mặc định MongoDB có sẵn cơ chế ghi với tốc độ cao và an toàn.
- Hệ thống rộng lớn và không đáng tin cậy. MongoDB cung cấp cơ chế làm việc trên nhiều server, replication và recover dữ liệu tự động, an toàn cho phép hệ thống có thể thoả mãn điều kiện này.

- Dữ liệu dựa trên khu vực. MongoDB đã xây dựng sẵn các chức năng không gian, vì vậy việc tìm kiếm dữ liệu có liên quan đến các địa điểm cụ thể rất nhanh chóng và chính xác.
- Website ở dạng thời gian thực nhiều, nghĩa là nhiều người thao tác với ứng dụng. Nếu trong quá trình load bị lỗi tại một điểm nào đó thì nó sẽ bỏ qua phần đó nên sẽ an toàn.
- Website có lượng dữ liệu lớn. Giả sử website có đến 10 triệu record thì đó là cơn ác mộng với MySQL. Bởi vì MongoDB có khả năng tìm kiếm thông tin liên quan khá nhanh nên trường hợp này nên dùng nó.

Những trường hợp không nên sử dụng:

- Các ứng dụng cần sử dụng nhiều transaction như ngân hàng
- Các ứng dụng cần ràng buộc chắc chắn

Chương 3. PHÂN TÍCH THIẾT KẾ

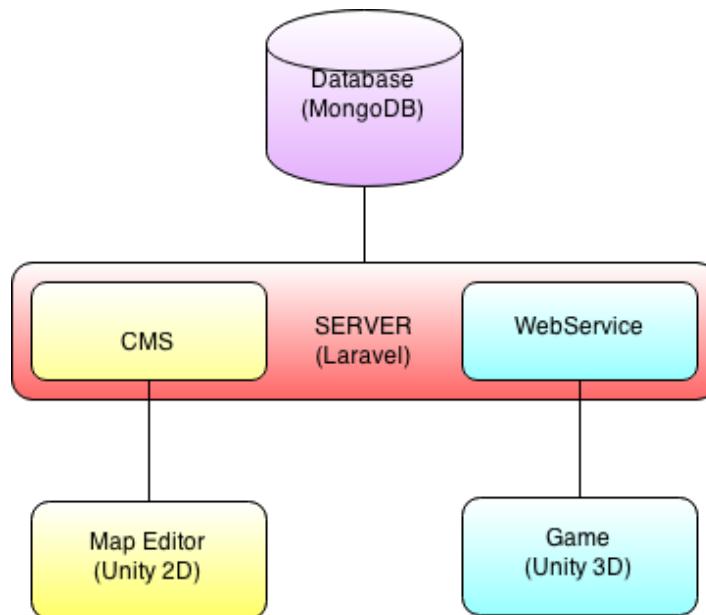
3.1. Tổng quan chức năng

Hệ thống gồm 3 thành phần: Game, CMS và Map Editor. Trong đó, chức năng chi tiết của từng phần như sau:

- Game
 - o Đăng nhập / Đăng xuất
 - o Chơi game
 - Kiểm tra các lỗi vi phạm
 - Xem thông tin về lỗi
 - Xem mức phạt
 - o Xem lịch sử các lần chơi
- CMS
 - o Đăng nhập / Đăng xuất
 - o Quản lý tài khoản
 - o Quản lý điểm
 - o Quản lý nội dung game
 - o Lập báo cáo
- Map Editor
 - o Tạo map mới
 - o Mở map cũ
 - o Chính sửa map với các thành phần
 - Các thành phần đường
 - Làn đường
 - Giao lộ
 - Điểm dừng xe buýt
 - Biển báo
 - Biển báo cấm
 - Biển báo hiệu lệnh

- Phong cảnh
 - Toà nhà
 - Cây cối
- Khác
 - Đèn đỏ
 - Điểm xuất phát / Điểm kết thúc
 - Các điểm checkpoint
 - Xe tự động

3.2. Kiến trúc tổng quát



Hình 3.1 Kiến trúc tổng quát hệ thống

Trong đó:

Server sử dụng Laravel, làm việc trực tiếp với database MongoDB. Server bao gồm hai thành phần là CMS và Web Service.

❖ CMS – Map Editor

Làm việc trực tiếp với Map Editor, khi nhà quản trị sử dụng Map Editor để tạo ra map mới hoặc chỉnh sửa map, nhà quản trị sẽ sử dụng CMS để upload dữ liệu file map và đưa vào database. Kèm theo đó là các thông tin của map, bao gồm:

- Tên map
- Level
- Thời gian ảo trong map
- Thumnail

❖ **Web Service – Game**

Cung cấp các API để Game thao tác với dữ liệu trên database, bao gồm:

- Đăng ký
- Quên mật khẩu
- Đăng nhập - Đăng xuất
- Lấy thông tin user
- Lấy lịch sử chơi của user
- Lấy danh sách các map khả dụng
- Lấy dữ liệu map
- Đăng kết quả chơi

3.3. Game

3.3.1. Yêu cầu

Mục đích chính của game là nhằm mô phỏng chân thực hoạt động tham gia giao thông hàng ngày của người chơi, do đó, game cần thoả mãn được những yêu cầu sau:

- Giả lập được góc nhìn người chơi
- Giả lập đầy đủ các điều khiển cơ bản của các xe máy phổ thông như:
 - o Bản hiển thị tốc độ
 - o Bật / Tắt đèn
 - o Đèn chiếu xa / Đèn chiếu gần
 - o Đèn chuyển hướng trái / phải
 - o Còi
 - o Mũ bảo hiểm

- Xây dựng được các thành phần chính trong cơ sở hạ tầng giao thông, bao gồm:
 - o Đường, làn đường, giao lộ, vỉa hè, điểm dừng xe buýt
 - o Vạch kẻ đường, dải phân cách
 - o Biển báo giao thông
 - o Đèn giao thông
 - o Những xe tham gia giao thông cùng người chơi
- Hỗ trợ kiểm tra các ràng buộc trên đường, bao gồm:
 - o Tốc độ tối thiểu – tối đa
 - o Các hướng được phép rẽ
 - o Các loại xe được chạy, được dừng
- Hỗ trợ phát hiện các lỗi vi phạm của người chơi khi tham gia giao thông

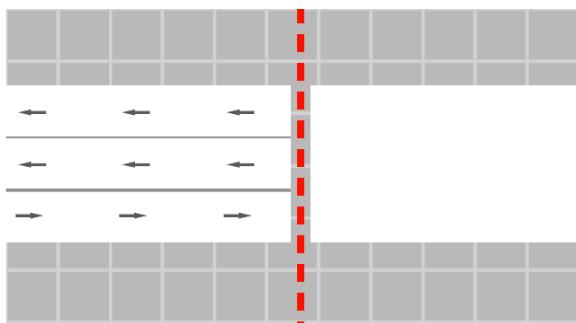
3.3.2. Phân tích

Để thiết kế Game, trước hết ta cần nhận diện và giải quyết các vấn đề của bài toán.

❖ Vấn đề 1

Đơn vị cơ sở của bản đồ là gì?

Có hai lựa chọn cho vấn đề này: Đường và làn đường.



Hình 3.2 Làn đường (bên trái) và đường (bên phải)

Đường trong thành phố thường có tên định danh và bắt đầu ở một giao lộ, kết thúc ở một giao lộ khác. Làn đường là một phần của đường, có độ rộng đủ để xe chạy an toàn. Một đường có thể có một hoặc nhiều làn đường.

Trên thực tế, một con đường thường rất dài, và xuyên suốt con đường đó có rất nhiều các biển báo giao thông khác nhau tạo ra những ràng buộc khác nhau trên con đường. Đồng thời cấu trúc của đường cũng có những thay đổi, có những đoạn là một chiều, có đoạn lại hai chiều, có đoạn 2 làn đường, có đoạn được chia ra 4 làn đường. Điều này sẽ gây rất nhiều khó khăn nếu sử dụng đường làm đơn vị cơ sở của bản đồ.

Trong khi đó, trên một làn đường sẽ chỉ có một chiều duy nhất. Các ràng buộc trên một làn đường cũng thống nhất xuyên suốt làn đường đó. Ví dụ: làn đường chỉ dành cho một loại phương tiện, tốc độ tối đa khi chạy trên làn đường, làn đường cấm đỗ xe...

Với các làn đường, ta hoàn toàn có thể xây dựng các đường với nhiều làn đường khác nhau và thay đổi ở những đoạn khác nhau. Nếu có nhiều đoạn đường có những ràng buộc phức tạp, ta hoàn toàn có thể chia nhỏ một làn đường thành nhiều làn đường nhỏ. Điều này tạo ra sự linh hoạt cho việc thiết kế bản đồ.

Vậy, làn đường sẽ là đơn vị cơ sở để xây dựng bản đồ.

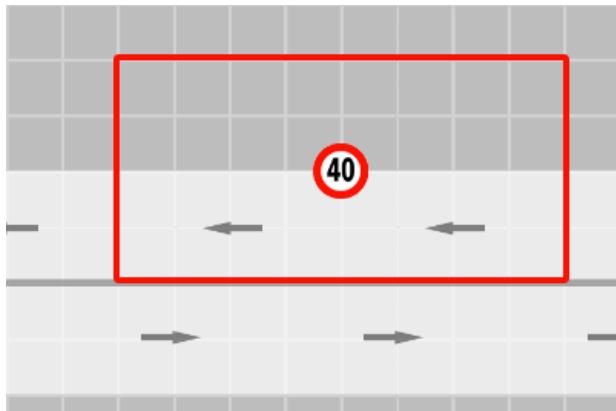
❖ **Vấn đề 2**

Làm cách nào để biển báo giao thông có thể tạo ra các ràng buộc trên làn đường?

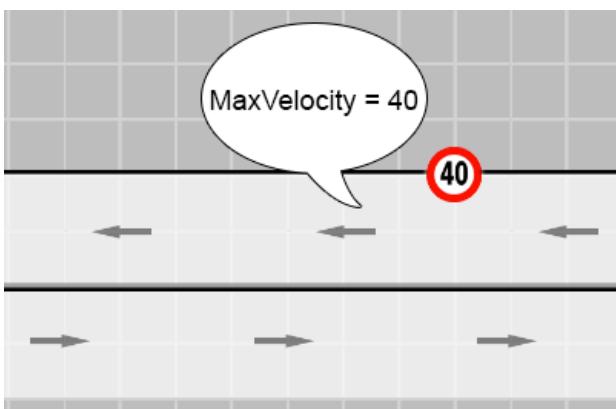
Ta nhận thấy rằng, việc kiểm tra xe của người chơi có đang vi phạm các ràng buộc mà các biển báo giao thông gần đó quy định hay không là một việc khó. Trước hết, ta cần xác định làn đường mà xe chạy, sau đó, tìm ra các biển báo giao thông đang có hiệu lực trên làn đường đó, từ đó mới xác định người chơi có vi phạm quy định nào hay không. Với quy trình như vậy, đòi hỏi biển báo giao thông phải được đặt chính xác tại một vị trí theo một tiêu chuẩn nào đó mới có thể xác định được làn đường mà nó có tác dụng. Hơn nữa, cần phải giới hạn phạm vi ảnh hưởng của biển báo bắt đầu từ đâu và kết thúc ở đâu.

Với những khó khăn đó, thay vì ta sử dụng biển báo giao thông thành đối tượng chứa thông tin ràng buộc, thì ta sử dụng làn đường.

Một làn đường luôn có phạm vi giới hạn của nó. Với việc làn đường chứa thông tin, biển báo giao thông sẽ chỉ là đối tượng hiển thị, do đó không cần phải yêu cầu đặt tại một vị trí chính xác.



Hình 3.3 Biển báo giao thông là nơi chứa thông tin ràng buộc



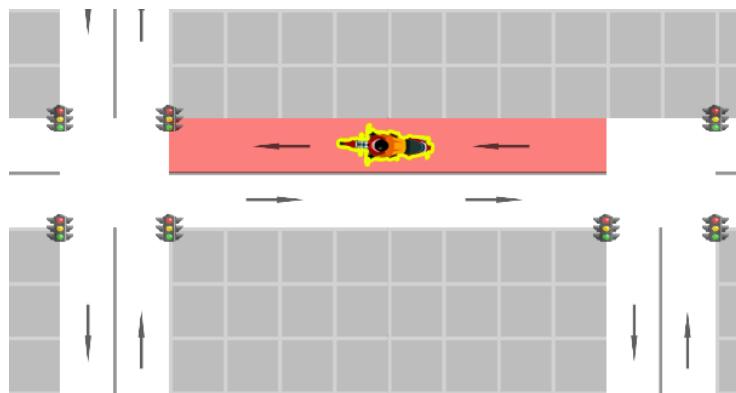
Hình 3.4 Làn đường là nơi chứa thông tin ràng buộc

Hình ảnh so sánh trên cho thấy, giải pháp sử dụng làn đường trở thành đối tượng lưu thông tin sẽ dễ dàng hiện thực hơn là sử dụng biển báo giao thông.

❖ Vấn đề 3

Làm cách nào để kiểm tra lỗi của người chơi?

Khi di chuyển trên bản đồ, người chơi luôn nằm trên một làn đường nào đó, và không bao giờ vượt ra ngoài. Đồng thời, các ràng buộc đều được gắn trên làn đường, do đó, nếu muốn kiểm tra các lỗi, thì chỉ cần xem xét thông tin trên làn đường mà người chơi đang chạy.



Hình 3.5 Vấn đề kiểm tra lỗi người chơi

Ví dụ:

- Để kiểm tra xe vượt quá tốc độ, ta so sánh tốc độ của xe và tốc độ giới hạn của làn đường.
- Để kiểm tra xe chạy ngược chiều, ta so sánh hướng đi của xe so với hướng của làn đường.

❖ Vấn đề 4

Các xe tự động cần được di chuyển như thế nào?

Các xe tự động cần đảm bảo chạy đúng luật. Khi xuất phát, xe sẽ được đặt trên một làn đường hợp lệ và hướng chạy hợp lệ. Xe sẽ chạy đều, khi đến các giao lộ, xe sẽ lựa chọn ngẫu nhiên 1 hướng chạy và tiếp tục chạy theo hướng đó. Nếu không tìm ra hướng chạy tiếp theo, xe sẽ dừng lại ở đó. Ngoài ra, trong quá trình chạy, xe sẽ luôn kiểm tra để ngăn chặn va chạm với các vật thể khác.

3.3.3. Cấu trúc dữ liệu

❖ Road

Đây là đối tượng cơ bản trên bản đồ, chứa tất cả thông tin ràng buộc của các biển báo giao thông đặt trên đường đó. Các thông số ràng buộc của làn đường bao gồm:

- Tốc độ tối thiểu – tốc độ tối đa
- Được phép rẽ trái – rẽ phải – đi thẳng – quay đầu
- Danh sách loại xe được đi

- Danh sách loại xe được dùng
- Các vỉa hè (trên, dưới, trái, phải)
- Các dải phân cách (trên, dưới, trái, phải)
- Trạng thái đèn giao thông

❖ Player

Lớp quản lý xe của người chơi và thực hiện kiểm tra các lỗi mà người chơi mắc phải.

Các thông số của Player:

- Trạng thái mũ bảo hiểm (có/không)
- Trạng thái đèn (bật/tắt)
- Trạng thái đèn chiếu (xa/gần)
- Trạng thái đèn rẽ hướng (tắt/trái/phải)
- Hướng xe chạy (trái, phải, lên, xuống)
- Tốc độ (km/h)
- Làn đường đang chạy

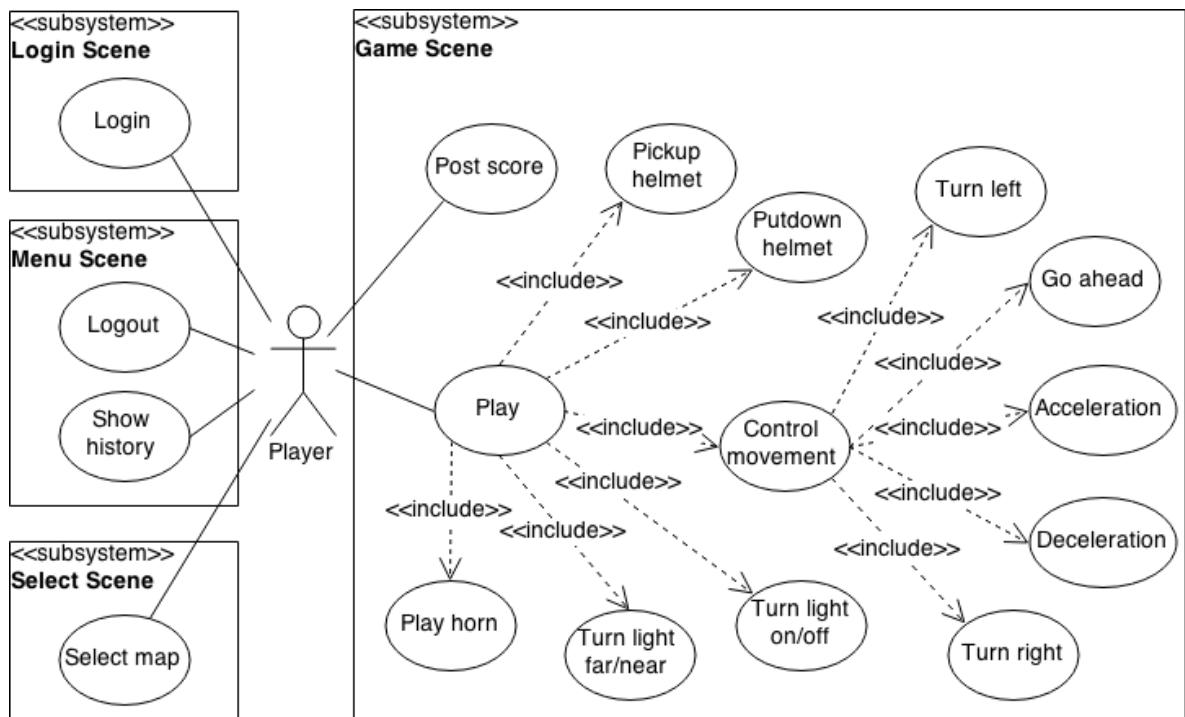
❖ Map

Là lớp dữ liệu, chứa tất cả thông tin của bản đồ.

Map được xây dựng bằng Map Editor và được xuất ra file dưới định dạng JSON. Sau đó được upload lên server, khi game chạy sẽ load về dưới dạng text để chuyển về lại định dạng ban đầu.

3.3.4. Sơ đồ use case

Game có 4 màn hình chính, do đó, sơ đồ use case được chia thành 4 subsystem tương ứng với các màn hình: Login Scene, Menu Scene, Select Scene và Game Scene.



Hình 3.6 Sơ đồ use case của Game

Danh sách Actor:

STT	Tên Actor	Ý nghĩa
1	Player	Người chơi chính

Bảng 3.1 Danh sách Actor trong sơ đồ use case của Game

Danh sách use case:

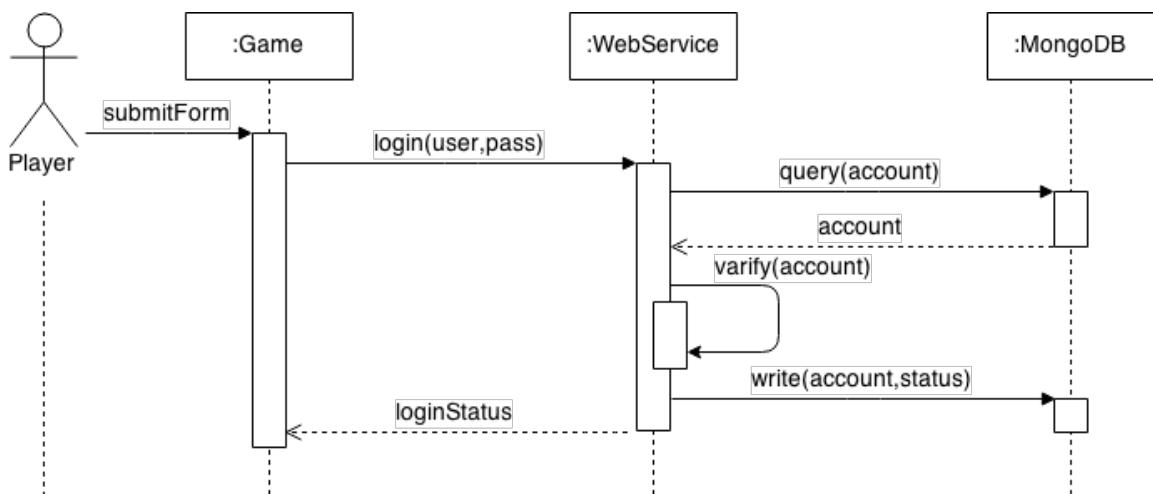
STT	Tên use case	Ý nghĩa
1	Login	Đăng nhập vào hệ thống
2	Logout	Đăng xuất khỏi hệ thống
3	Show history	Xem lịch sử các lần chơi trước đó
4	Select map	Chọn một map để chơi
5	Play	Các thao tác trong màn hình chơi chính
6	Pickup helmet	Đội mũ bảo hiểm

7	Putdown helmet	Bỏ mũ bảo hiểm
8	Play horn	Bấm còi
9	Turn light on/off	Bật đèn / tắt đèn
10	Turn lighth near/far	Bật đèn chiếu gần / đèn chiếu xa
11	Control movement	Các thao tác điều khiển xe
12	Turn left	Cho xe rẽ trái
13	Turn right	Cho xe rẽ phải
14	Go ahead	Cho xe chạy thẳng và tăng tốc
15	Acceleration	Tăng tốc cho xe
16	Deceleration	Giảm tốc cho xe
17	Post score	Lưu điểm lên server

Bảng 3.2 Danh sách use case trong Game

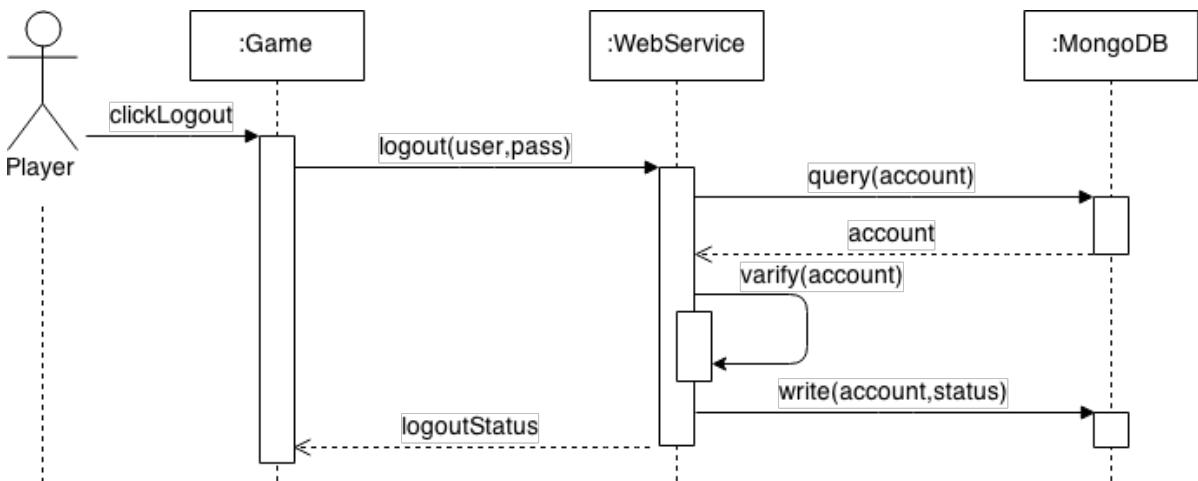
3.3.5. Sơ đồ tuần tự

❖ Sơ đồ tuần tự use case Login



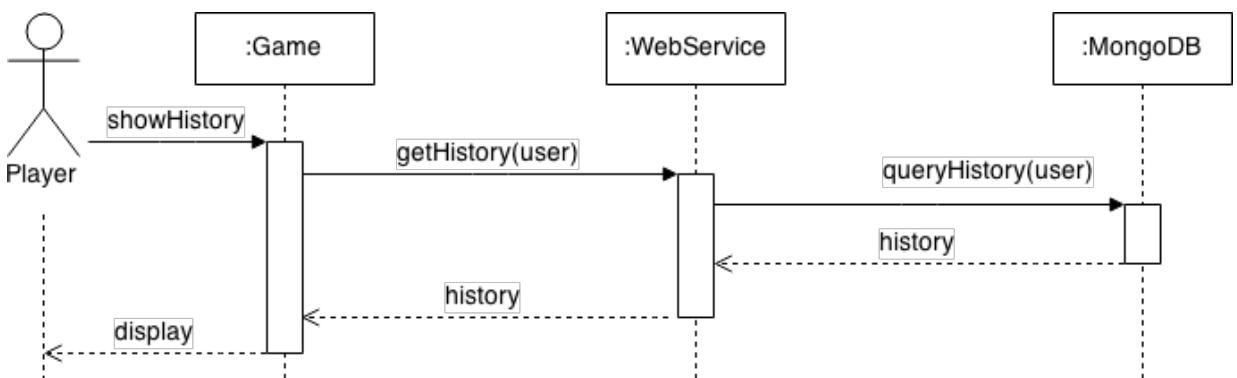
Hình 3.7 Sơ đồ tuần tự use case Login

❖ Sơ đồ tuần tự use case Logout



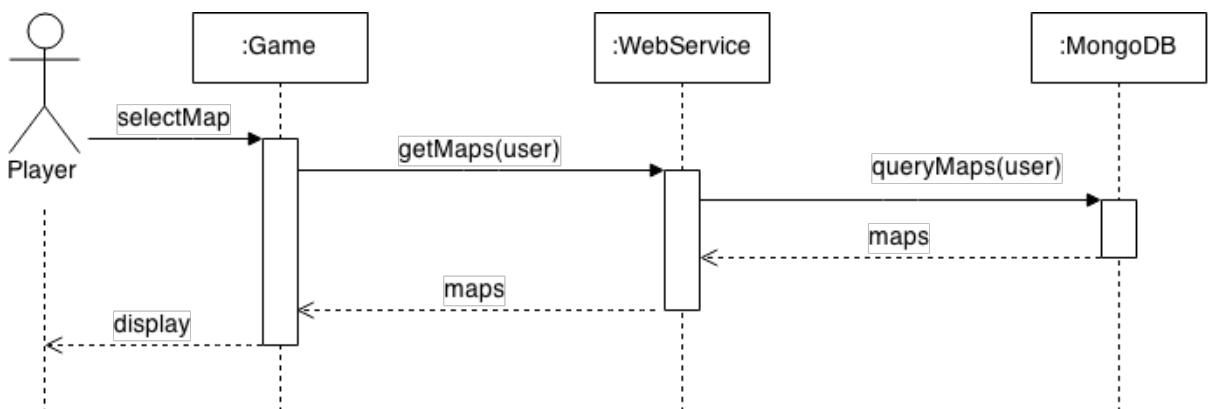
Hình 3.8 Sơ đồ tuần tự use case Logout

❖ Sơ đồ tuần tự use case Show history



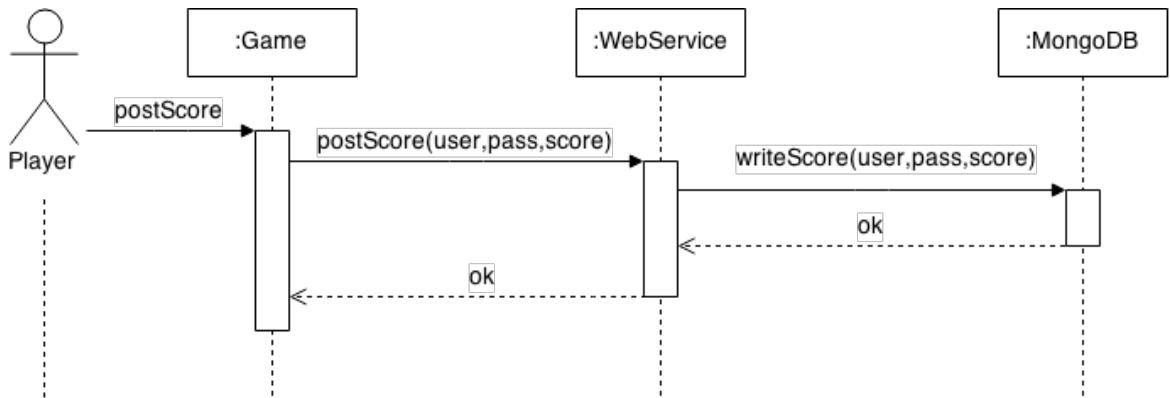
Hình 3.9 Sơ đồ tuần tự use case Show history

❖ Sơ đồ tuần tự use case Select map



Hình 3.10 Sơ đồ tuần tự use case Select map

❖ Sơ đồ tuần tự use case Post score



Hình 3.11 Sơ đồ tuần tự use case Post score

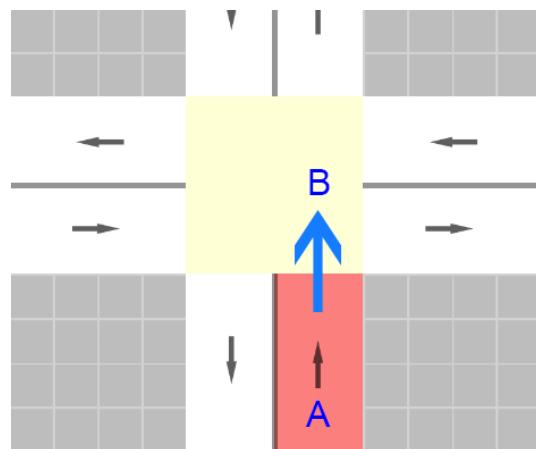
3.3.6. Giải thuật

Các trường hợp bắt lỗi vi phạm của người chơi đều dựa trên các thông tin của Player và Road.

❖ Lỗi đi ngược chiều

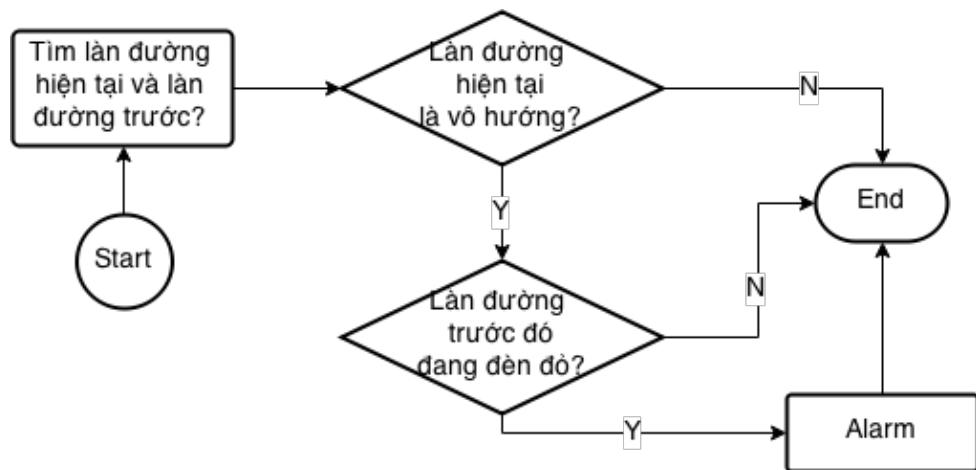
Khi người chơi di chuyển theo chiều ngược lại với chiều của làn đường thì người chơi mắc lỗi đi ngược chiều.

❖ Vượt đèn đỏ



Hình 3.12 Vượt đèn đỏ

Làn đường ở giữa giao lộ còn được gọi là làn đường vô hướng (màu vàng). Làn đường hiện tại của người chơi (A) đang ở trạng thái đèn đỏ. Khi đó, nếu người chơi di chuyển sang làn đường vô hướng B thì người chơi đã vi phạm lỗi vượt đèn đỏ.



Hình 3.13 Sơ đồ logic phát hiện lỗi vượt đèn đỏ

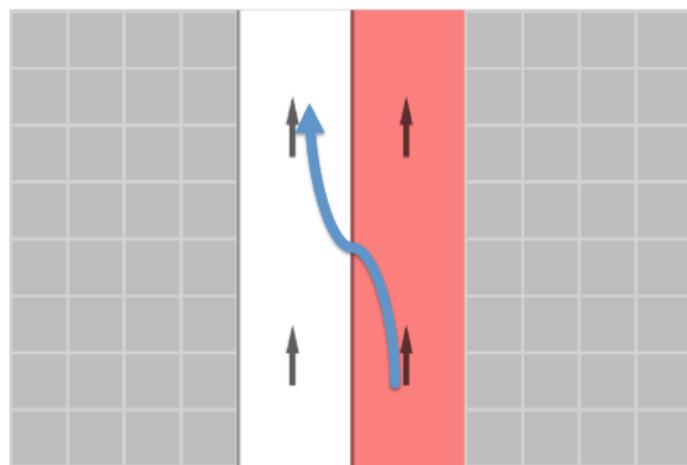
❖ **Dừng không đúng chỗ**

Có các vị trí dừng không đúng bao gồm:

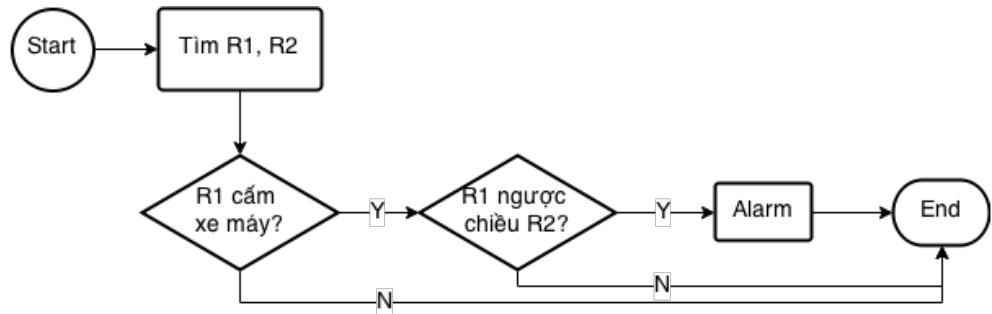
- Dừng trên làn đường xe chạy (không sát mép đường)
- Dừng tại trạm xe buýt
- Dừng giữa giao lộ
- Dừng trên vạch người đi bộ qua đường

❖ **Lấn tuyến**

Lấn tuyến là lỗi khi người chơi di chuyển từ làn đường hợp lệ, sang 1 làn đường khác cùng chiều, nhưng không giành cho xe máy chạy.



Hình 3.14 Lấn tuyến

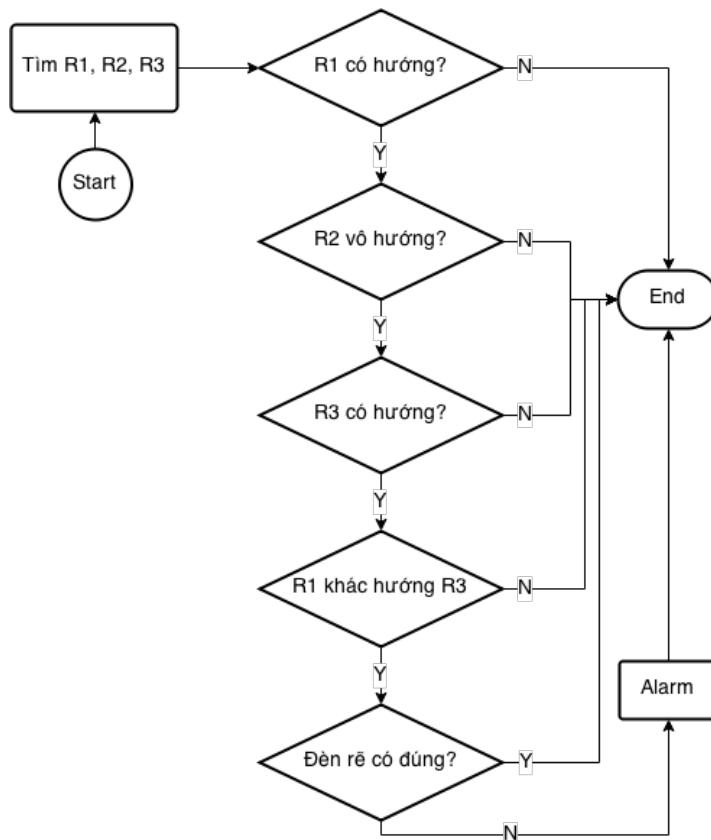


Hình 3.15 Sơ đồ logic phát hiện lỗi lần tuyến

Trong đó:

- R1, R2 lần lượt là làn đường hiện tại và làn đường trước đó
- ❖ **Chuyển hướng không có tín hiệu**

Chuyển hướng là rẽ trái, rẽ phải, hoặc quay đầu xe. Khi người chơi chuyển hướng mà trạng thái đèn chuyển hướng không hợp lý thì sẽ mắc lỗi chuyển hướng không có tín hiệu.



Hình 3.16 Sơ đồ logic lỗi chuyển hướng không tín hiệu

Trong đó:

- R1, R2, R3 lần lượt là làn đường hiện tại, làn đường trước đó, làn đường trước của R2

❖ Dừng không đúng chỗ

Có các vị trí dừng không đúng bao gồm:

- Dừng trên làn đường xe chạy (không sát mép đường)
- Dừng tại trạm xe buýt
- Dừng giữa giao lộ
- Dừng trên vạch người đi bộ qua đường

3.3.7. Danh sách các lỗi

Danh sách các lỗi vi phạm mà hệ thống có thể phát hiện:

STT	Lỗi
1	Lấn tuyến
2	Vượt đèn đỏ
3	Đi ngược chiều
4	Va chạm giao thông
5	Chuyển hướng không có tín hiệu báo trước
6	Không đội mũ bảo hiểm
7	Không bật đèn khi về ban đêm, sương mù hoặc thời tiết xấu
8	Chạy quá tốc độ từ 5 km/h đến 10 km/h
9	Chạy quá tốc độ từ 10 km/h đến 20 km/h
10	Chạy quá tốc độ trên 20 km/h

11	Dừng xe trên làn đường xe chạy
12	Bấm còi từ 22h đến 5h trong khu đô thị
13	Sử dụng đèn chiếu xa trong khu đô thị
14	Dừng xe tại trạm xe buýt
15	Lạng lách, đánh võng
16	Quay đầu xe không đúng nơi quy định
17	Vượt bên phải xe khác
18	Chuyển hướng ko giảm tốc độ
19	Không có tín hiệu xin vượt trước khi vượt
20	Dừng xe tại làn đường người đi bộ qua đường
21	Đi vào làn đường cấm
22	Chạy dưới tốc độ tối thiểu
23	Dừng xe không có tín hiệu báo trước
24	Chuyển làn đường không đúng nơi quy định
25	Chuyển làn đường không có tín hiệu báo trước
26	Rẽ phải tại nơi không được phép
27	Rẽ trái tại nơi không được phép
28	Vượt đèn vàng

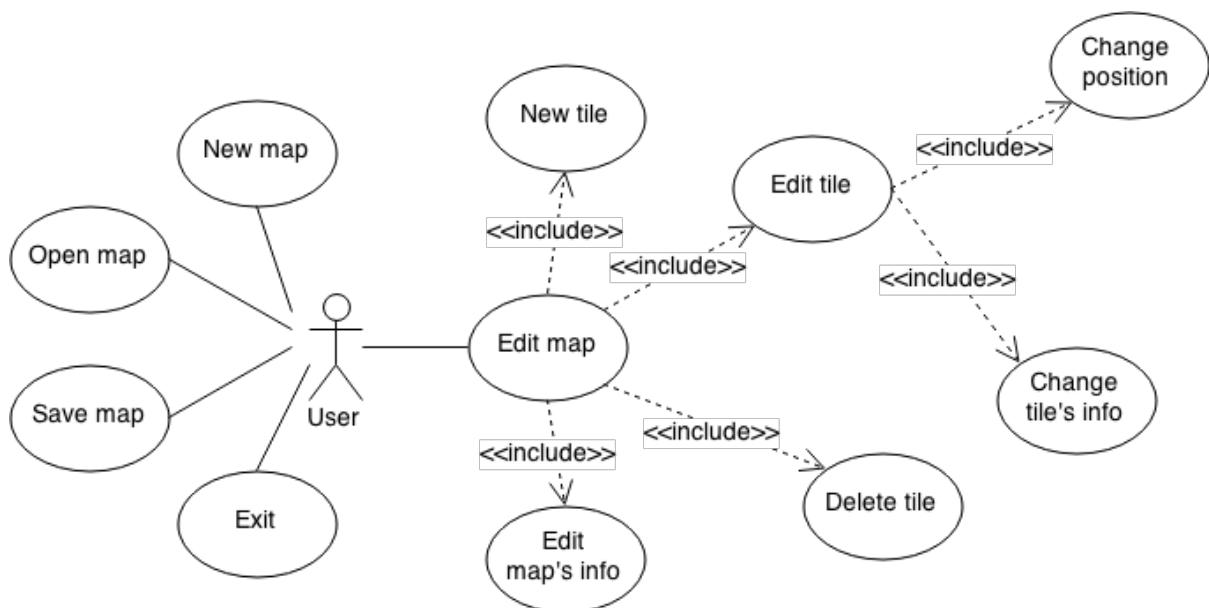
Bảng 3.3 Danh sách các lỗi vi phạm

3.4. Map Editor

Map Editor là phần mềm độc lập, chạy trên Mac OS, Windows hoặc trên Web. Map Editor được dùng để thiết kế bản đồ thành phố, các tuyến đường giao thông, đặt các biển báo, vị trí người chơi, các xe tự động, ... Sau đó, phần mềm sẽ xuất ra file theo định dạng JSON để có thể dùng trong Game. Map Editor sẽ được nhà quản trị nội dung sử dụng để tạo bản đồ, sau đó upload file JSON lên trang CMS và được Game download về để sử dụng.

Do yêu cầu không cần cao về các loại đối tượng, các loại địa hình trong bản đồ, do đó nhóm đã lựa chọn xây dựng Map Editor dạng 2D thay vì 3D. Tuy không cho phép người chơi xem trước được kết quả khi hiển thị trong Game, nhưng cũng đủ để đáp ứng được những yêu cầu cơ bản của hệ thống.

3.4.1. Sơ đồ use case



Hình 3.17 Sơ đồ use case Map Editor

Danh sách Actor:

STT	Tên Actor	Ý nghĩa
1	User	Người dùng

Bảng 3.4 Danh sách Actor trong Map Editor

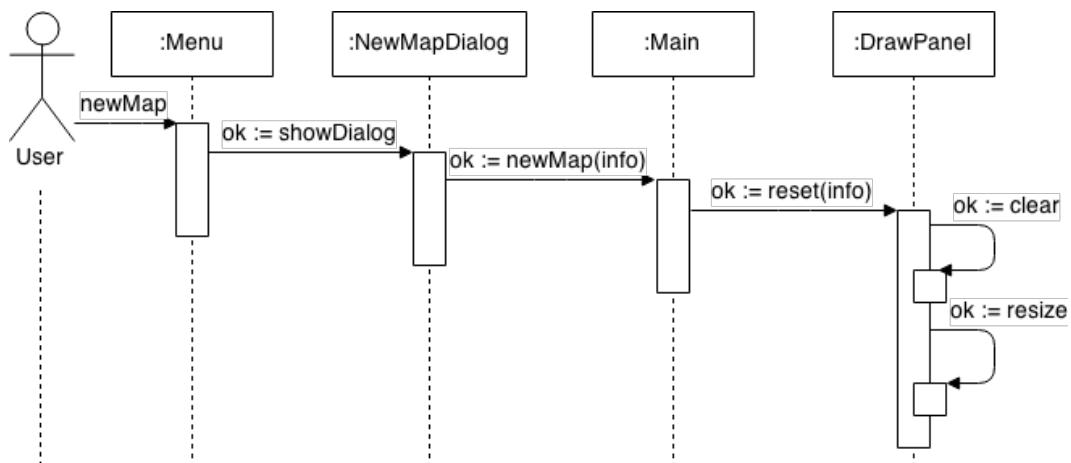
Danh sách use case:

STT	Tên use case	Ý nghĩa
1	New map	Tạo map mới
2	Open map	Mở file map đã lưu
3	Save map	Lưu map thành file
4	Exit	Thoát ứng dụng
5	Edit map	Chỉnh sửa map
6	Edit map's info	Chỉnh sửa thông tin map
7	New tile	Tạo một đối tượng mới trên map
8	Delete tile	Xoá một đối tượng trên map
9	Edit tile	Chỉnh sửa đối tượng trên map
10	Change position	Thay đổi vị trí đối tượng trên map
11	Change tile's info	Chỉnh sửa thông tin của đối tượng trên map

Bảng 3.5 Danh sách use case của Map Editor

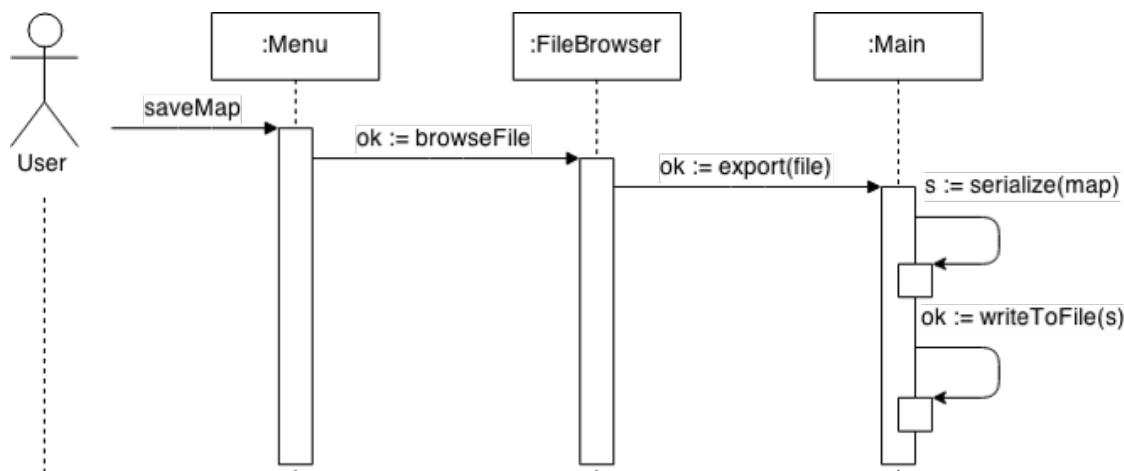
3.4.2. Sơ đồ tuần tự

❖ Sơ đồ tuần tự use case New map



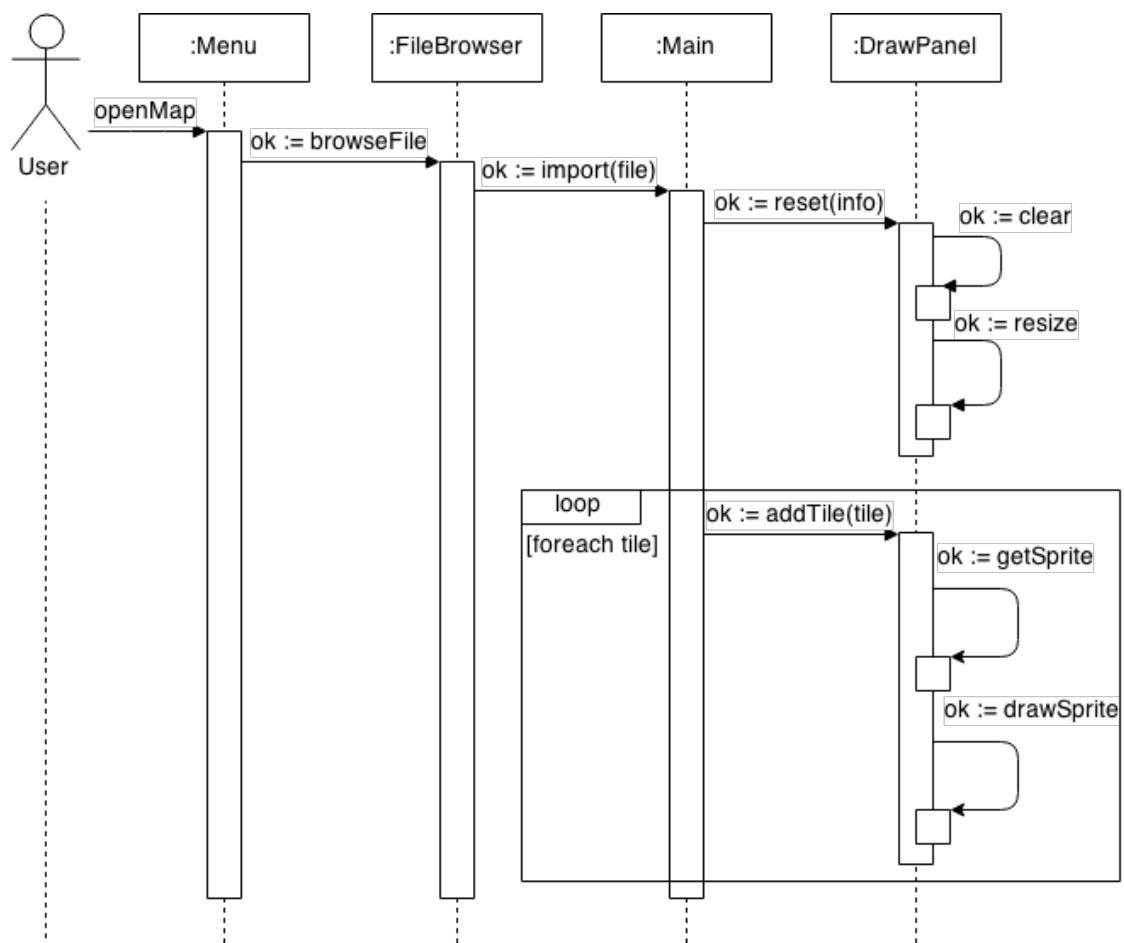
Hình 3.18 Sơ đồ tuần tự use case New map

❖ Sơ đồ tuần tự use case Save map



Hình 3.19 Sơ đồ tuần tự use case Save map

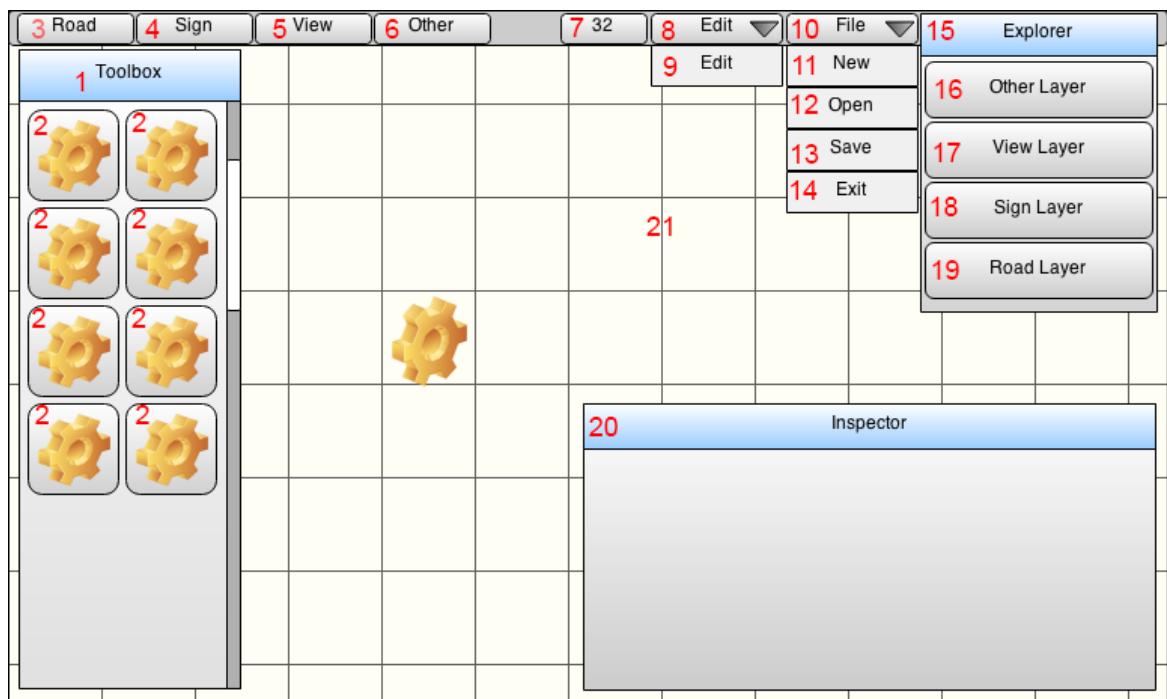
❖ Sơ đồ tuần tự use case Open map



Hình 3.20 Sơ đồ tuần tự use case Open map

3.4.3. Thiết kế giao diện

❖ Màn hình chính



Hình 3.21 Giao diện chính Map Editor

Chú giải giao diện:

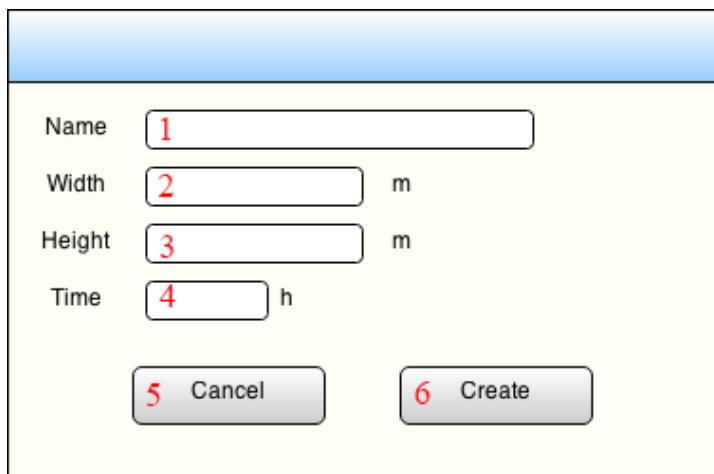
STT	Tên	Chức năng	Ghi chú
1	Toolbox	Hiển thị những đối tượng được dùng để tạo bản đồ	Có thanh cuộn
2	Tile	Các đối tượng được dùng để tạo bản đồ	
3	RoadTab	Tab hiển thị các đối tượng liên quan tới làn đường, và các dấu hiệu trên mặt đường	
4	SignTab	Tab hiển thị các đối tượng biển báo giao thông	
5	ViewTab	Tab hiển thị các đối tượng phong cảnh	

6	OtherTab	Tab hiển thị các đối tượng khác, ngoài các kiểu ở trên	
7	Step	Chỉnh sửa giá trị step, dùng trong quá trình di chuyển các đối tượng trên bản đồ	
8	MenuEdit	Menu	
9	Edit	Chỉnh sửa thông số bản đồ, bao gồm: tên, chiều dài, chiều rộng, thời gian.	
10	File	Menu	
11	New	Tạo bản đồ mới	
12	Open	Mở bản đồ đã tạo	
13	Save	Lưu bản đồ thành file JSON	
14	Exit	Thoát ứng dụng	
15	Explorer	Chuyển qua lại giữa các layer đối tượng trên bản đồ	
16	Other Layer	Chuyển sang layer Other	
17	View Layer	Chuyển sang layer View	
18	Sign Layer	Chuyển sang layer Sign	
19	Road Layer	Chuyển sang layer Road	
20	Inspector	Thông tin chi tiết của đối tượng được chọn	
21	Grid	Khu vực tạo bản đồ	

Bảng 3.6 Chú giải giao diện chính Map Editor

❖ Màn hình New map và Edit map

Màn hình giao diện New map và Edit map hoàn toàn giống nhau và có thiết kế như sau:



Hình 3.22 Giao diện New map và Edit map

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	Name	Tên bản đồ	
2	Width	Chiều rộng bản đồ theo chiều ngang	Đơn vị m
3	Height	Chiều dài bản đồ theo chiều dọc	Đơn vị m
4	Time	Thời gian ảo trong hệ thống	
5	Cancel	Hủy	
6	Create	Tạo bản đồ mới	

Bảng 3.7 Chú giải giao diện New map

3.4.4. Cấu trúc dữ liệu

Map sau khi thiết kế sẽ được lưu lại dưới dạng file JSON. File chứa đầy đủ thông tin về thiết kế, trạng thái sửa đổi của file.

Cấu trúc JSON:

```
{
  "info": {
    //thông tin chung của bản đồ
  },
  "layer": {
    //dữ liệu bản đồ
  },
  "state": {
    //trạng thái sửa đổi
  }
}
```

Trong đó

❖ Info

Phần info chứa thông tin chung về bản đồ, bao gồm:

```
{
  "name": "Map",      //tên bản đồ
  "width": 120,       //chiều rộng, đơn vị m
  "height": 120,      //chiều cao, đơn vị m
  "tile": 32,         //kích thước 1 ô trong lưới (pixel)
  "simulateTime": 8 //thời gian giả lập
}
```

❖ State

Phần state chứa thông tin trạng thái sửa đổi của bản đồ, dùng để lưu các trạng thái của lần chỉnh sửa trước đó, ví dụ: vị trí sửa, layer được chọn, độ zoom màn hình, ...

Mục đích là để việc chỉnh sửa các bản đồ lớn đơn giản hơn.

```
{
  "x": 0,           //vị trí
  "y": 0,
  "scale": 1,        //độ zoom
  "step": 32,        //step để di chuyển đối tượng
  "current_layer": 1, //layer đang được chọn
  "cam_x": 0,        //vị trí camera
  "cam_y": 0,
  "cam_z": 0,
  "cam_size": 2.100 //kích thước camera
}
```

❖ Layer

Phần layer chứa dữ liệu của bản đồ, các đối tượng trong bản đồ được lưu ở đây.
Phần Layer chứa trực tiếp các layer có trong bản đồ.

```
{  
    "1": {  
        "id": 1,           //id của layer  
        "name": "Other 1", //tên layer  
        "type": "Other",  //loại layer  
        "tile": {          //danh sách các đối tượng  
        }  
    }  
    //, ...  
    //, ...  
}
```

Trong đó, các đối tượng tile chứa bên trong 1 layer sẽ có cấu trúc sau:

```
{  
    "objId": 4,           //id của đối tượng  
    "typeId": 310,        //loại đối tượng  
    "layerType": "Other", //đối tượng thuộc loại layer nào  
    "x": -96,            //tọa độ  
    "y": -192,  
    "w": 32,              //kích thước  
    "h": 46,  
    "properties": {       //tất cả các thuộc tính của đối tượng  
        "key": "value"    //lưu dạng key-value  
    }  
}
```

3.5. CMS

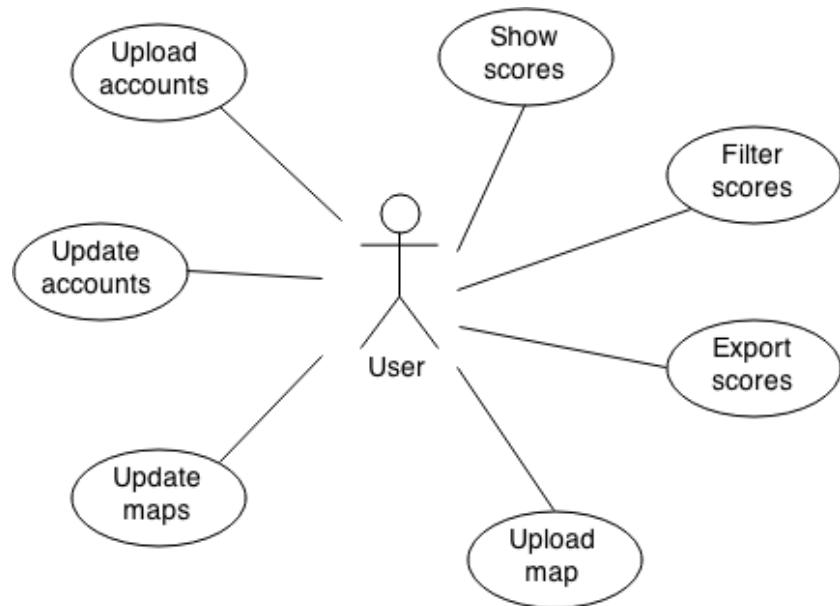
CMS là hệ thống quản lý nội dung (Content Management System), được quản trị viên sử dụng để quản lý nội dung trong Game, quản lý tài khoản người dùng, quản lý điểm số, lập báo cáo, ...

Các chức năng chính bao gồm:

- Quản lý tài khoản người chơi
- Quản lý nội dung bản đồ

- Quản lý điểm số người chơi
- Xuất báo cáo

3.5.1. Sơ đồ use case



Hình 3.23 Sơ đồ use case CMS

Danh sách Actor:

STT	Tên Actor	Ý nghĩa
1	User	Người dùng

Bảng 3.8 Danh sách Actor trong CMS

Danh sách use case:

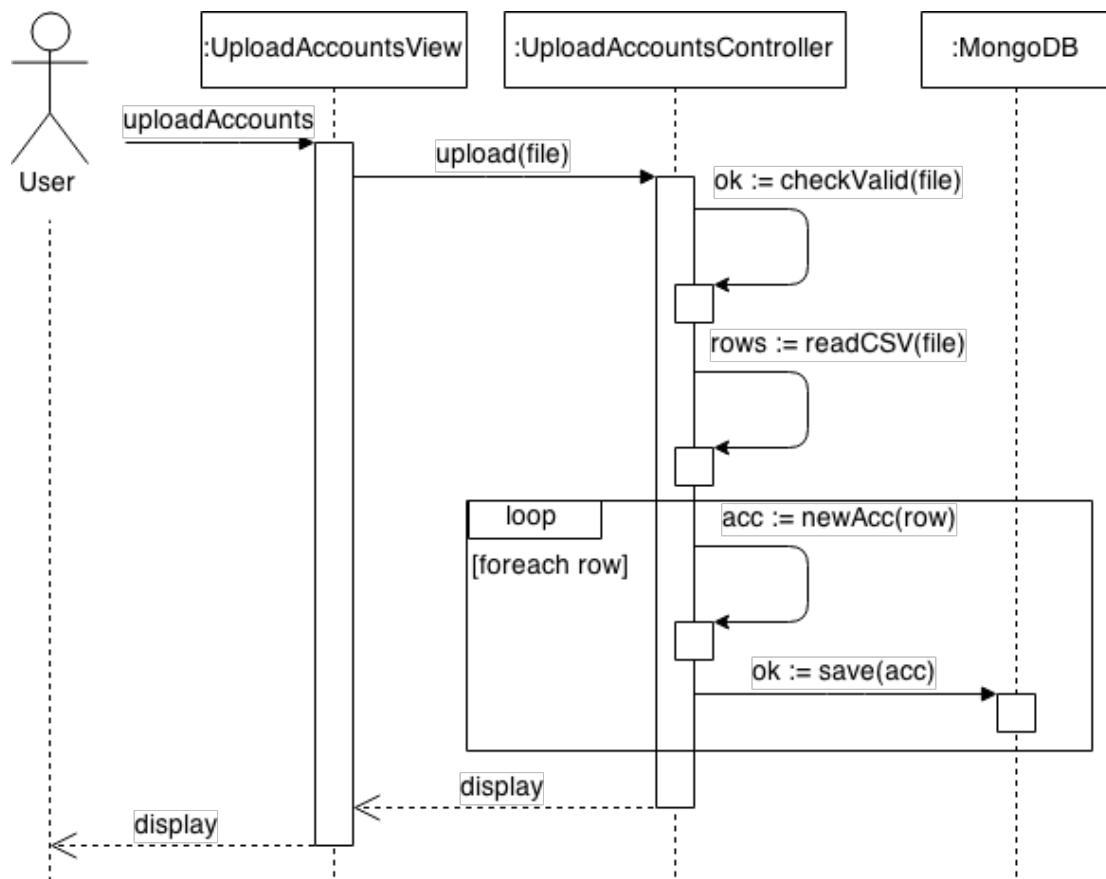
STT	Tên use case	Ý nghĩa
1	Upload accounts	Upload danh sách tài khoản người chơi lên server
2	Update accounts	Sửa đổi thông tin tài khoản người chơi
3	Upload map	Upload file map mới lên server
4	Update maps	Sửa đổi thông tin map đã có trên server

5	Show scores	Xem danh sách điểm của người chơi
6	Filter scores	Lọc kết quả người chơi theo thời gian
7	Export scores	Export kết quả điểm số người chơi thành file để download

Bảng 3.9 Danh sách use case CMS

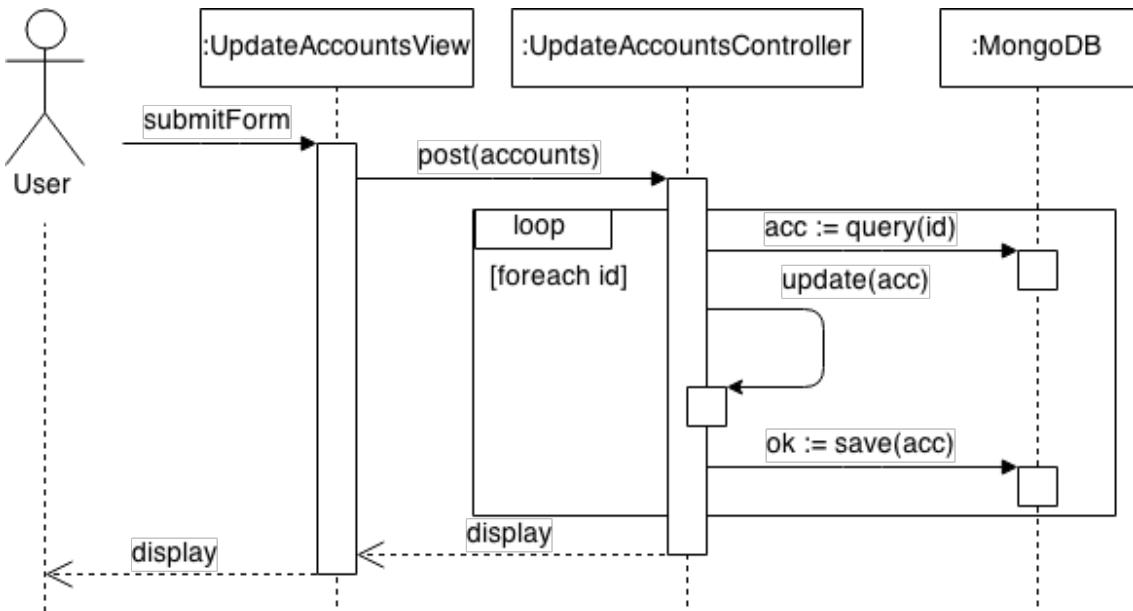
3.5.2. Sơ đồ tuần tự

❖ Sơ đồ tuần tự use case Upload accounts



Hình 3.24 Sơ đồ tuần tự use case Upload accounts

❖ Sơ đồ tuần tự use case Update accounts



Hình 3.25 Sơ đồ tuần tự use case Update accounts

3.5.3. Thiết kế cơ sở dữ liệu

Tất cả các Collection trong MongoDB đều được tạo ra với các trường mặc định sau:

STT	Thuộc tính	Kiểu dữ liệu	Ghi chú
1	<code>_id</code>	String	Khoá chính
2	<code>created_at</code>	DateTime	Thời gian lúc tạo ra
3	<code>updated_at</code>	DateTime	Thời gian lần cuối sửa dữ liệu

Bảng 3.10 Các thuộc tính mặc định trong MongoDB

Tất cả các bảng dữ liệu sau đây sẽ không đề cập tới các trường mặc định ở trên.

Danh sách các Collection có trong CMS bao gồm:

❖ Collection accounts

Collection chứa thông tin các tài khoản của người chơi. Với hệ thống CMS hiện tại, dữ liệu ở collection này sẽ được quản trị viên upload lên dưới dạng file CSV. Tên đăng nhập và mật khẩu là các giá trị mặc định.

STT	Thuộc tính	Kiểu dữ liệu	Ghi chú
1	username	String	Tên đăng nhập
2	password	String	Mã MD5 của password
3	name	String	Tên hiển thị
4	level	Number	Level người chơi
5	info	String	Thông tin ghi chú

Bảng 3.11 Collection accounts

❖ Collection maps

Collection chứa thông tin các map. Dữ liệu trong collection này được tạo ra khi quản trị viên upload các file map xuất ra từ Map Editor lên CMS.

STT	Thuộc tính	Kiểu dữ liệu	Ghi chú
1	name	String	Tên bản đồ
2	enable	Boolean	Được kích hoạt hay không
3	level	Number	Level của bản đồ
4	url	String	Đường dẫn tới file JSON
5	thumnail	String	Đường dẫn tới file thumbnail của map
6	time	Number	Thời gian để chơi, đơn vị phút
7	info	String	Thông tin ghi chú

Bảng 3.12 Collection maps

❖ Collection scores

Collection chứa thông tin các lần chơi của từng người chơi. Dữ liệu được tạo ra khi người chơi hoàn thành các màn chơi và Game sẽ tự động upload kết quả lên server.

STT	Thuộc tính	Kiểu dữ liệu	Ghi chú
1	username	String	Tên đăng nhập của người chơi
2	map	String	_id của bản đồ trong bảng maps
3	score	String	Điểm
4	detail	Number	Danh sách ID các lỗi của người chơi
5	time	DateTime	Thời gian lúc gửi kết quả

Bảng 3.13 Collection scores

3.5.4. Thiết kế giao diện

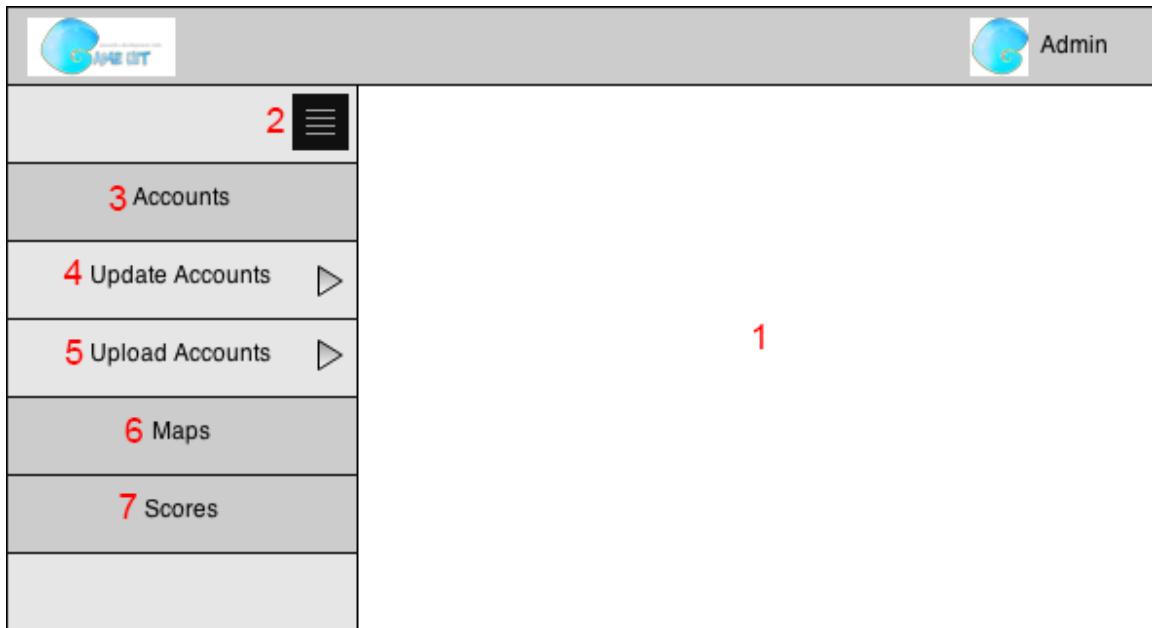
Giao diện của CMS gồm có 6 trang thực hiện các chức năng chính.

STT	Tên	Chức năng	Ghi chú
1	Đăng nhập	Quản trị viên đăng nhập	
2	Upload accounts	Upload file danh sách thông tin tài khoản người chơi	Định dạng CSV
3	Update accounts	Update thông tin tài khoản người chơi	
4	Upload maps	Upload file map mới	File JSON được xuất ra từ Map Editor
5	Update maps	Thay đổi các thông tin các map	
6	Score	Xem danh sách điểm số người chơi và xuất file excel	

Bảng 3.14 Các giao diện trong CMS

❖ Giao diện chung của CMS

Tất cả các trang trong CMS (trừ trang đăng nhập) đều có cấu trúc giao diện chung như sau:



Hình 3.26 Giao diện chung các trang của CMS

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	Content	Nội dung của mỗi trang khác nhau	
2	ShowHideMenu	Ẩn hiện menusidebar	
3	MenuAccounts	Các chức năng liên quan tới accounts	
4	UpdateAccounts	Update thông tin accounts	
5	UploadAccounts	Upload file danh sách account	
6	MenuMaps	Các chức năng liên quan tới bản đồ	
7	MenuScores	Các chức năng liên quan tới điểm số người chơi	

Bảng 3.15 Chú giải giao diện chung các trang CMS

Mỗi trang khác nhau trong CMS sẽ có phần Content khác nhau.

❖ Đăng nhập

The image shows a login form titled "Login to your account". It contains four numbered fields: 1. A text input field for the username containing "admin" with a red "1" above it. 2. A password input field containing "*****" with a red "2" above it. 3. A "Remember me" checkbox labeled "3" with a red "O" to its left. 4. A "Login" button labeled "4" with a red "L" inside it.

Hình 3.27 Giao diện Đăng nhập

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	Username	Tên đăng nhập	
2	Password	Mật khẩu đăng nhập	
3	Remember	Lưu trạng thái đăng nhập	
4	Login	Đăng nhập	

Bảng 3.16 Chú giải giao diện Đăng nhập

❖ Update accounts

Update Accounts

No.	Username	Name	Level	Info
1	1 10520100	2 Nguyen Van A	3 1	4

5 Save
<< 1 2 >>

6 7 7 8

Hình 3.28 Giao diện Update accounts

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	Username	Tên đăng nhập	
2	Name	Tên hiển thị	
3	Level	Level người chơi	
4	Info	Ghi chú thông tin	
5	Save	Lưu các thay đổi	
6	Back	Trở về phân trang trước đó	Mỗi phân trang có 10 record
7	Page	Đến phân trang bao nhiêu	
8	Next	Tới phân trang tiếp theo	

Bảng 3.17 Chú giải giao diện Update accounts

❖ Update maps

Trang thay đổi thông tin các map

Update Maps					
No.	Name	Enable	Level	Time	Info
1	1 Map So 1	2 <input checked="" type="checkbox"/>	3 1	4 5	5
6Save		<input type="button" value="<<"/> <input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value=">>"/> <input type="button" value="7"/> <input type="button" value="8"/> <input type="button" value="8"/> <input type="button" value="9"/>			

Hình 3.29 Giao diện Update maps

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	Name	Tên bản đồ	
2	Enable	Trạng thái kích hoạt	Kích hoạt mới được sử dụng
3	Level	Level của bản đồ	
4	Time	Thời gian để chơi	Đơn vị là phút
5	Info	Thông tin ghi chú	
6	Save	Lưu thay đổi	
7	Back	Trở về phân trang trước đó	Danh sách được phân trang, mỗi trang có 10 record
8	Page	Chuyển tới phân trang số mấy	
9	Next	Tới phân trang tiếp theo	

Bảng 3.18 Chú giải giao diện Update maps

❖ Scores

Xem kết quả người chơi và xuất dữ liệu ra file excel.

The screenshot shows a user interface titled "Scores". At the top, there are three input fields: "From" with value 1 1/12/2014, "To" with value 2 1/12/2014, and a "Filter" button. Below these is a "Export" button labeled "4 Export". The main area displays a table with columns: No., Username, Time, Score, Detail, and Map. One row is visible, showing: 1, 5 10520100, 6 2014-12-06 15:32:59, 71024, 8 1,4,5,8,2, and 9 Map So 1.

No.	Username	Time	Score	Detail	Map
1	5 10520100	6 2014-12-06 15:32:59	71024	8 1,4,5,8,2	9 Map So 1

Hình 3.30 Giao diện Scores

Chú giải giao diện:

STT	Tên	Chức năng	Ghi chú
1	TimeFrom	Thời gian bắt đầu để chọn lọc	
2	TimeTo	Thời gian kết thúc để chọn lọc	
3	Filter	Lọc các kết quả theo thời gian	
4	Export	Xuất dữ liệu ra file excel	
5	Username	Tên đăng nhập người chơi	
6	Time	Thời gian chơi	
7	Score	Điểm	
8	Detail	Danh sách các lỗi vi phạm	
9	Map	Tên bản đồ	

Bảng 3.19 Chú giải giao diện Scores

Chương 4. CÀI ĐẶT MINH HOA

4.1. Game

Game khi cài đặt thực tế có thể chạy tốt trên tất cả các trình duyệt phổ biến có cài đặt Unity Web Player, bao gồm: Chrome, Safari, Firefox, Opera, Internet Explorer.

Game có thể phát hiện ra: 28 tình huống vi phạm giao thông, chủ yếu là các tình huống vi phạm khi đang di chuyển trên đường.

Hiện tại, Game đã được cài đặt tại địa chỉ:

<http://widocom.com/projects/trafficgame/game>

Tài khoản khả dụng:

- username: user
- password: user

4.1.1. Giao diện

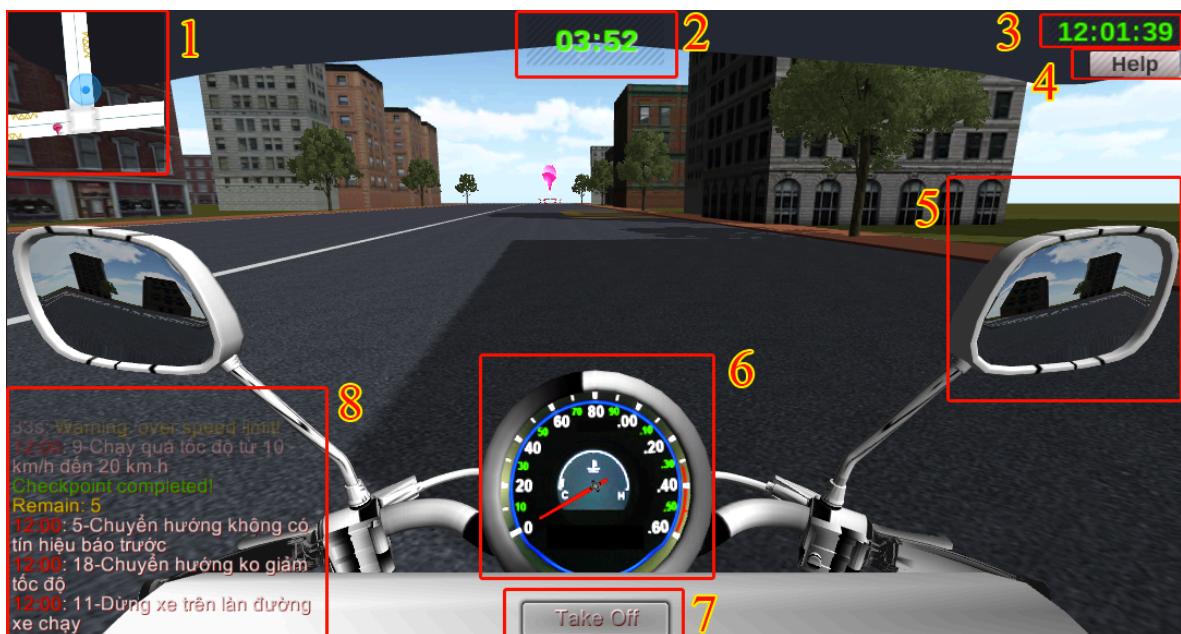
Giao diện khi cài đặt thực tế của Game như sau:



Hình 4.1 Giao diện Game

4.1.2. Hướng dẫn sử dụng

❖ Hiển thị



Hình 4.2 Các thành phần giao diện Game

Danh sách các thành phần:

STT	Tên	Ý nghĩa
1	Mini Map	Bản đồ thu nhỏ
2	Timer	Thời gian còn lại cho màn chơi.
3	Clock	Đồng hồ hiển thị thời gian trong hệ thống.
4	Helper	Hướng dẫn chơi.
5	Mirror	Gương chiếu hậu.
6	Velocity Clock	Đồng hồ tốc độ của xe.
7	Helmet	Nút gõ bỏ mũ bảo hiểm.
8	Notifier	Bảng thông báo

Bảng 4.1 Các thành phần giao diện Game

❖ Điều khiển

Điều khiển trong Game bao gồm các phím sau:

Tên	Ý nghĩa
Left/A	Rẽ trái
Right/D	Rẽ phải
Up/W	Đi thẳng, tăng tốc
Down/S/Space	Phanh xe
Q	Bật đèn rẽ trái (tắt đèn rẽ phải)
E	Bật đèn rẽ phải (tắt đèn rẽ trái)
L	Bật/tắt đèn chiếu trước
F	Bật đèn chiếu xa
N	Bật đèn chiếu gần
B	Bấm còi

Bảng 4.2 Các phím điều khiển trong Game

Trước khi xuất phát, cần đội mũ bảo hiểm bằng cách click chọn vào mũ bảo hiểm.
Có thể bỏ mũ bảo hiểm xuống bằng nút [Take Off]

❖ Luật chơi

Mỗi bản đồ sẽ có tổng thời gian khác nhau để hoàn thành. Nếu trong thời gian đó, người chơi không hoàn thành thì cuộc chơi sẽ dừng tại đó.

Trong mỗi bản đồ, sẽ có các trạm dừng mà người chơi phải đi qua và điểm đến cuối cùng là trạm dừng quan trọng nhất.

Hãy hoàn thành bản đồ và đi đúng luật giao thông. Tất cả lỗi vi phạm sẽ được ghi lại và hiển thị cuối màn chơi.

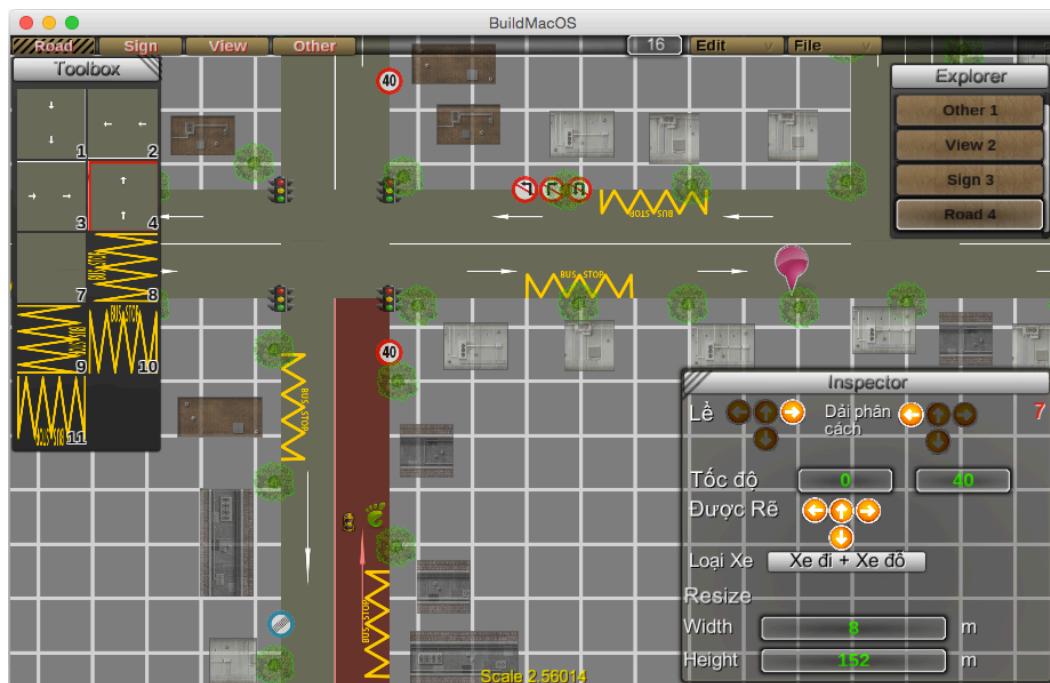
4.2. Map Editor

Map Editor có chạy trên nhiều nền tảng khác nhau, bao gồm Windows, Mac OSX, Linux và Web thông qua Unity Web Player, điều này cho phép nhà quản trị có thể sử dụng ở mọi lúc, mọi nơi.

Hiện tại, Map Editor đã được cài đặt tại địa chỉ:

<http://widocom.com/projects/trafficgame/editor>

4.2.1. Giao diện



Hình 4.3 Giao diện Map Editor

4.2.2. Hướng dẫn sử dụng

- Tạo map mới bằng menu File -> New map
- Lựa chọn layer phù hợp để chỉnh sửa ở Explorer
- Lựa chọn đối tượng cần tạo ở Toolbox
- Kéo thả đối tượng ở Grid
- Click vào đối tượng để chọn
- Chỉnh sửa các thuộc tính của đối tượng ở Inspector
- Lưu file ở menu File -> Save map

4.3. CMS

4.3.1. Yêu cầu phần cứng

Yêu cầu về máy chủ web để có thể cài đặt CMS:

- Hệ điều hành: Linux
- Web server: Apache
- MongoDB 2.6
- PHP 5.3 trở lên, có cài đặt MongoDB driver.
- FTP Server

Hiện tại, hệ thống CMS đã được cài đặt tại địa chỉ:

<http://widocom.com/projects/trafficgame/backend>

Tài khoản khả dụng:

- username: admin
- password: 123456

4.3.2. Giao diện

❖ Home / Scores

The screenshot shows the CMS interface for managing scores. On the left, there's a sidebar with navigation links: Accounts, Maps, and Scores. The main area is titled "Show Score (01/12/2014 -> 30/12/2014)". Below this, there's a search bar with "From: 01/12/2014" and "To: 30/12/2014", a "Filter" button, and an "Export" link. A table lists six entries, each with a user ID, date, score, and a detailed score breakdown. The last entry has a very long score string: "9,9,10,10,10,10,10,10,10,10,10,10,10,10,10,10,11,16,2".

No.	Username	Time	Score	Detail	Map
1	10520163	2014-12-06 15:32:59	111	1,2,3	Map So 5
2	10520163	2014-12-06 15:47:00	0		Map So 1
3	10520163	2014-12-06 15:49:02	0	2	Map So 1
4	10520163	2014-12-06 18:32:53	0		Map So 1
5	10520163	2014-12-06 18:41:10	0		Map So 1
6	10520163	2014-12-06 18:46:06	0	9,9,10,10,10,10,10,10,10,10,10,10,10,10,10,10,11,16,2	Map So 1

Hình 4.4 Giao diện Home / Scores trong CMS

❖ Update accounts

The screenshot shows the CMS interface with a dark theme. On the left, there is a sidebar with navigation links: 'Accounts' (selected), 'Maps' (selected), and 'Scores'. Under 'Accounts', there are 'Update Accounts' and 'Upload Accounts'. Under 'Maps', there are 'Update Maps' and 'Upload Maps'. The main content area has a title 'Update Accounts' and a sub-section 'Update Accounts'. It displays a table with 6 rows of account data:

No.	Username	Name	Level	Info
1	Username	Phạm Tân Long	1	
2	10520101	User001	2	
3	10520102	User002	3	
4	10520101	User1	1	
5	10520102	User2	2	
6	10520103	User3	3	

Below the table are a 'Save' button and a page navigation bar with pages 1, 2, and 3.

Hình 4.5 Giao diện Update accounts trong CMS

❖ Upload map

The screenshot shows the CMS interface with a dark theme. On the left, there is a sidebar with navigation links: 'Accounts' (selected), 'Maps' (selected), and 'Scores'. Under 'Maps', there are 'Update Maps' and 'Upload Maps'. The main content area has a title 'Upload Map' and a sub-section 'Upload Map'. It contains form fields for 'Name' (Map So 1), 'Level' (3), 'Time (minute)' (5), and an 'Information' text area (Map về khu vực ngã tư Thủ Đức.). Below these are file upload fields for 'Map file (.json)' (Choose File: map1.json) and 'Thumbnail file (image)' (Choose File: map1.png), followed by an 'Upload' button.

Hình 4.6 Giao diện Upload map trong CMS

❖ Update maps

No.	Name	Enable	Level	Time	Information
1	Map So 1	<input checked="" type="checkbox"/>	1	0.25 min	báu giáp ở đâu đó trong thành phố. Map số 1 Haha, cũng không khó lắm, nhưng cũng không dễ tí nào.
2	Map So 2	<input checked="" type="checkbox"/>	2	1 min	MAP SO 2 Trong thời gian 20p, người chơi phải hoàn thành đường đi từ điểm xuất phát đến đích. Trong quá trình di, có những điểm checkpoint mà người chơi cần phải
3	Map So 3	<input checked="" type="checkbox"/>	3	17 min	Map So 3 Trong thời gian 20p, người chơi phải hoàn thành đường đi từ điểm xuất phát đến đích. Trong quá trình di, có những

Hình 4.7 Giao diện Update maps trong CMS

Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Từ những cấu trúc, mô hình và phương pháp tiếp cận nêu trên, nhóm đã xây dựng thành công hệ thống ứng dụng học luật giao thông. Trong đó, Game đã có thể phát hiện được 28 tình huống vi phạm giao thông thường gặp và cung cấp thông tin về các lỗi vi phạm đó.

Nhóm đã cài đặt hệ thống tại địa chỉ: <http://widocom.com/projects/trafficgame> với giao diện đơn giản nhằm mục đích thử nghiệm hoạt động của hệ thống.

5.2. Hướng phát triển

Trong thời gian tới, hệ thống sẽ có thể cung cấp thêm các tính năng:

- Hỗ trợ lựa chọn nhiều phương tiện giao thông
- Hỗ trợ thêm nhiều kiến trúc hạ tầng phức tạp hơn
- Xây dựng hệ thống trí tuệ nhân tạo tốt hơn cho các xe tự động
- Nâng cấp giao diện

TÀI LIỆU THAM KHẢO

- [1.] A. Sahni and D. Segleau, NoSQL and SQL Introspective. California: Oracle, 2013
- [2.] K. Chodorow and M. Dirolf, MongoDB The Definitive Guide. California: O'Reilly, 2010.
- [3.] R. Saunier, Getting Started with Laravel 4. Birmingham, UK: Packt, 2014.
- [4.] S. Blackman, Beginning 3D Game Development with Unity 4, 2nd ed. New York: Apress, 2013.
- [5.] [Online]. Available: <http://antoangiaothong.gov.vn/den-xanh/nam-2014-ca-nuoc-giam-hon-4-ngan-vu-tngt-62911.html>
- [6.] [Online]. Available:
http://www.moj.gov.vn/vbpq/_layouts/print.aspx?id=12345
- [7.] [Online]. Available:
http://www.moj.gov.vn/vbpq/_layouts/print.aspx?id=25264