MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Car Rental Portal

Group 3	
Group members	Trần Hữu Đức – SE61448 Huỳnh Công Thành – SE61297 Lê Vũ Đăng Khoa – SE61238 Nguyễn Tường Tâm – SE61384
Supervisor	Lâm Hữu Khánh Phương
Ext. Supervisor	N/A
Capstone Project Code	CRP

- Ho Chi Minh City, 01 September 2016 -

Fpt University

TRƯỜNG ĐẠI HỌC FPT

CAPSTONE PROJECT REGISTER

Class: Duration time: from ..……../20…. To ..……/20…..

(*) Profession: <Software Engineer> Specialty: <ES> ☐ <JS> ☒

(*) Kinds of person make registers: Lecturer ☒ Students ☐

1. Register information for supervisor (if have)

	Full name	Phone	E-Mail	Title
Supervisor 1	Lâm Hữu Khánh Phương	0915353001	phuonglhk@fpt.edu.vn	Mr.

2. Register information for students (if have)

	Full name	Student code	Phone	E-mail	Role in Group
Student 1	Trần Hữu Đức	SE61448	0944006278	ducthse61448@fpt.edu.vn	Leader
Student 2	Huỳnh Công Thành	SE61297	01646560468	thanhhcse61297@fpt.edu.vn	Member
Student 3	Lê Vũ Đăng Khoa	SE61238	0968368595	khoalvdse61238@fpt.edu.vn	Member
Student 4	Nguyễn Tường Tâm	SE61384	01687548624	tamntse61384@fpt.edu.vn	Member

3. Register content of Capstone Project

(*) 3.1. Capstone Project name:

- English: Car Rental Portal
- Vietnamese: Cổng thông tin cho thuê xe
- Abbreviation:CRP

(*) 3.2. Main proposal content (including result and product)

a) Theory and practice (document):

- Student should apply the software development process and UML 2.0 in modelling system.
- The documents include User Requirement, Software Requirement Specification, Architecture Design, Detail Design, System Implementation and Testing Document, Installation Guide, sources code, and deployable software packages
- Server side technologies:
  - Web Server: .NET, Java or Javascript (Node.js…) , Windows Azure
  - Database Design: SQL Server or MySQL.
- Client side technologies:
  - Cross platform applications (Phone Gap, Cordova) or native apps (Android , iOS)
  - Web Client: HTML5, CSS3, Javascript.

b) Program:

Nowadays, car is popular in Vietnam. Moving with car can help customer don't care too much about the weather or safety. But car's price is a barrier to allow people to own their private car. So, car rental is one of solutions for solving the problem besides taxi, bus or Uber, Grab..

In this project, you will build a portal B2C and C2C for both sides: car rental service providers and customers.

System includes:

- Website: Show list of car rental service providers. In detail of each provider, list of cars with detail information will be shown.
-
- Website have 3 roles:
  - Customers:
    - Customer can search cars and book cars

        - Customer can payment for each order
        - View booking history
    - Providers:
        - Manage car
        - Manage schedule for each cars.
        - Manage booking for their own cars
        - View notification
    - Administrator
        - Manage users
        - View report

- Mobile Apps: (on iOs and Android) is used by user for make a booking.
- Web service API: middle tiers for user's app submit data to system.

4. Other comment (propose all relative thing if have)

N/A

HCM, date 20/8 /2016

Supervisor (If have)	**On behalf of Registers**
(Sign and full name)	*(Sign and full name)*

ACKNOWLEDGEMENTS

We wish to express our greatest gratitude to FPT University for granting us the chance to take on this wonderful project, to our supervisor, Mr. Lâm Hữu Khánh Phương, for his guidance, advices, and many research resources, which contributed greatly to the completion of this project, and to Mr. Nguyễn Trọng Tài, who has helped to review this very document.

We would like to express many thanks to our parents who have given us many mental supports to help us stand strong during this difficult challenge and to our wonderful friends from other capstone projects, especially from group 2 and 5, for their indispensable cooperation.

Table of Contents

List of Tables

List of Figures

Definitions, Acronyms, and Abbreviations

Name	Definition
CRP	Car Rental Portal

Table 1: Definitions, Acronyms, and Abbreviations

A. Introduction

1. Project Information

- Project name: **Car Rental Portal**
- Project Code: **CRP**
- Product Type: **Website Application**
- Start Date: **September 1st, 2016**
- End Date: **December 1st, 2016**

2. Introduction

In this document, we introduce a car rental portal solution – a website application to help car rental service provider to bring their business online. The solution allows providers to define their own rental policy and price. It also simplifies the providers' rental management process. Our solution also allows customer to easily find and book the most suitable car for their trip.

3. Current Situation

Nowadays, there are many car rental providers in Vietnam. However, only a small percent of them have utilized internet to promote their business. Part of them (Large rental companies, taxi companies) have their own website or mobile application, while some choose to become driver for online rental portal solution like Uber and Grab.

Many small and middle-sized providers cannot afford their own website or mobile application, and also find contract with Uber and Grab too rigid because they are not allowed to define their own rental price and policy. Uber's and Grab's solutions also do not apply well for long-period rental.

A new solution that has just appeared recently is Aleka, which they themselves claimed to be inspired by Uber and Airbnb. Their approach take a lot after Airbnb. They allow providers to post their vehicles on their website, and then helps customer to find and book the most suitable vehicle in their system. Their solution however does not offer any way for customer to rate and comment about the service of provider, as well as to read other customers' ratings and reviews before deciding to book the vehicle.

4. Problem Definition

The current situation has several problems:

- **Cost barrier:** The cost to develop an online rental application is too high for small rental providers.
- **Ill-suited rental portal solutions:** There are great rental services like Uber, Grab, or Aleka on the market. However, their solutions have yet to completely satisfy the user. Their shortcomings include limiting provider's ability in deciding their own policies, cannot handle long-period rental very well, or lack of a mean for rating and reviewing provider's service.

5. Proposed Solution

Our proposed solution, named Car Rental Portal, is a car rental portal website application that takes after Airbnb. We provide a flexible mean for provider to promote their rental service online while still allows them to define their own rental policy. Our solution also allows customer to rate and comment about their past renting, as well as read others' review on vehicles. Our solution also offers a searcher integrated with recommender engine that can help customer to find the best matched vehicle quickly and easily.

5.1. Featured functions

- **Manage online rental service:** Provider can manage their vehicles' and bookings' information, as well as declare their own rental policies, including price contrainst, time constraint, travel distance constraint…
- **Advanced vehicle searcher with recommender engine:** System can help customers to find the most suitable vehicle fast and easily based on their booking history.
- **Review vehicle and read vehicle's reviews:** Customer can comment and rate rental service after renting vehicle, as well as read reviews from other customer for a vehicle.
- **Book vehicle and cancel booking:** Customer can book and pay booking fee through NganLuong payment portal, as well as cancel their booking easily.

5.2. Advantages

- Offer a cheap and easy online rental management solution.
- Allow provider to define their own rental policies.
- Support both hourly and daily rental.
- Offer vehicle searching with recommender engine.
- Offer rental service rating/reviewing function.
- Allow paying through NganLuong payment portal.

5.3. Disadvantages:

- Since recommender works depending on customer's booking history, customer must have booked at least once in the past to utilize it.
- Depends on NganLuong and has no alternative payment method in case NganLuong goes down.

6. Functional Requirements

- Manage vehicles:
  - Provider can create, update, and delete vehicle.
- Manage vehicles' bookings:
  - Provider can keep track of their vehicle's booking schedule.
  - Provider can create self-booking to reserve their vehicle.
  - Provider can cancel self-booking.
- Manage garage:
  - Provider can create, update, and delete garage.

    - Provider can specify their garage's rental policy.
    - Provider can group vehicle into garage.
    - Provider can close their garage temporary.
- Manage price group:
    - Provider can create, update, and delete price group.
    - Provider can group vehicle that he want to apply specific price options.
    - Provider can disable the group and every vehicle inside the group.
- Manage booking:
    - Customer can book vehicle.
    - Customer can cancel their booking.
    - Customer can keep track of their booking history.
    - Customer can rate and comment about their booking.
- Manage user account:
    - Administrator can lock and unlock user account.
- Search and view vehicle's information.
- View report:
    - Administrator can view system's business report in dashboard.
    - Provider can view personal business report in dashboard.
- Receive notification email:
    - Customer will receive notification email when he has booked a vehicle.
    - Provider will receive notification email when their vehicle has been booked.
    - Customer will receive notification email when he has canceled a booking.
    - Provider will receive notification email when their vehicle's booking has been canceled.

7. Roles and Responsibility

No	Full Name	Role	Position	Contact
1	Lâm Hữu Khánh Phương	Project Manager	Supervisor	phuonglhk@fpt.edu.vn
2	Trần Hữu Đức	Developer	Leader	ducthse61448@fpt.edu.vn
3	Huỳnh Công Thành	Developer	Member	thanhchse61297@fpt.edu.vn
4	Lê Vũ Đăng Khoa	Developer	Member	khoalvdse61238@fpt.edu.vn
5	Nguyễn Tường Tâm	Developer	Member	tamntse61384@fpt.edu.vn

Table 2: Roles and Responsibility

B. Software Project Management Plan

1. Problem Definition

1.1. Name of this Capstone Project

- **Official name**: Car Rental Portal
- **Vietnamese name**: Cổng thông tin cho thuê xe
- **Abbreviation**: CRP

1.2. Problem Abstract

Nowadays, there are many car rental providers in Vietnam. However, only a small percent of them have utilized internet to bring their business online. Aside from creating their own online rental applications, providers can also utilize rental solution services like Grab, Uber, or Aleka. These new services however still contain many flaws that keep a large number of providers away.

1.3. Project Overview

1.3.1. Current Situation

Our project has the following problems:

- **Recommender engine is dependent on customer's booking history:** The recommender engine cannot work on customer that has not placed any booking in the past.
- **Has no alternative payment method:** Our system depends completely on NganLuong for payment processing. There is currently no alternative payment method.

1.3.2. The Proposed System

The proposed system is a rental portal website application that can help car rental service provider to bring and manage their rental service online and allow customer to find and book vehicle easily. The system utilize NganLuongApi to process payment and has a recommender engine to improve vehicle searching.

Separated based on user's role, the system offers the following functions:

- For guests:
  - Search vehicle.
  - View vehicle's rental information.
- For customers:
  - Search vehicle.
  - View vehicle's rental information.
  - Book vehicle and cancel booking.
  - Track booking history.
  - Register providership.
  - Receive notification email upon booking or canceling booking.
- For providers:
  - Manage vehicles.
  - Manage vehicles' bookings.

  - Manage garages.
  - Manage price groups.
  - View personal business report.
  - Extend providership period.
  - Receive notification email upon receiving booking or booking was canceled.
- Administrator:
  - Manage user accounts.
  - View system's business report.

1.3.3. Boundaries of the System

- The system targets mainly small and mid-sized car rental service providers, but can also handle the needs of large providers with large number of vehicles and multiple garages.
- The system supports both daily and hourly booking.
- The language of the system is Vietnamese.
- The completed product includes:
  - Booking system for customer.
  - Rental service managing system for provider.

1.3.4. Future Plans

- Offer pricing scheduler so that price group can change price during special events.
- Support self-driving rental.
- Support motorbike rental.
- Offer driver management functions.
- Support internationalization.
- Develop mobile version.

1.3.5. Development Environment

1.3.5.1. Hardware requirement

For web server:

Hardware	Minimum Requirements	Recommended
Network Bandwidth	4 Mbps	8 Mbps
Operating System	Window Server 2008	Window Server 2008
Computer Processor	Intel® Xeon Dual Core® 1.4GHz (4MB Cache)	Intel® Xeon® Quad Core (8M Cache, 2.40 GHz)
Computer Memory	2GB RAM	4GB or more
Hard disk	40GB	80GB

Table 3: Hardware requirement for Web Server

For web development:

Hardware	Minimum Requirements	Recommended
Internet Connection	512 Kbps	8 Mbps
Operating System	Window Vista, 7, 8	Window 7, 8
Computer Processor	1 GHz	Intel® Core™ i5 CPU 2.53 GHz
Computer Memory	2GB RAM	4GB or more

Table 4: Hardware requirement for Web Server

1.3.5.2. Software requirement

Software	Name / Version
Operating system	Windows 7 or above
Modeling tool	StarUML version 2.7.0 (UML2.0 standard [1])
IDE	Microsoft Visual Studio 2015 Community
DBMS	SQL Server 2014
Source control	Git 2.9.0.windows.1
Web browser	Chrome (v.42 and above), Firefox (v.38 and above), and any equivalent web browser

Table 5 : Software requirement

2. Project organization

2.1. Software Process Model

Modified waterfall model (Sashimi [2]) is applied in this project. This model was chosen because:

- The requirements of the project are well-defined and well-understood by all team members.
- The deliverables in each phase of the model can be corresponded with the reports that have to be submitted in each process of the capstone project.

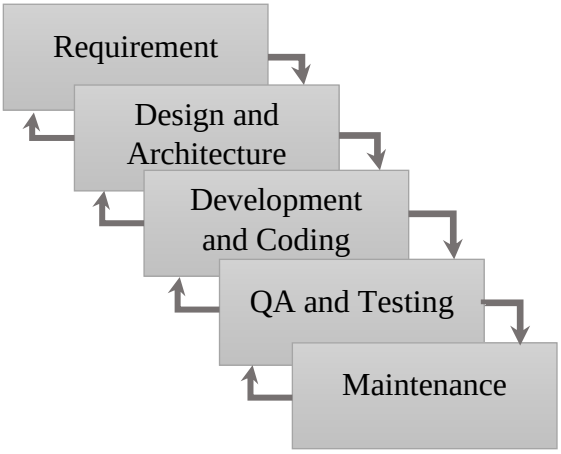

Figure 1: Modified Waterfall Development Model

2.2. Roles and responsibilities

No	Full name	Role in Group	Responsibilities
1	Lâm Hữu Khánh Phương	Supervisor / Project Manager	- Clarify user requirement - Give technical support and business analysis - Track development process - Review document and product
2	Trần Hữu Đức	Team leader, Business Analyst, Developer, Tester	- Manage development process - Analyze requirement - Design database - Prepare documents - Design GUI - Code - Test - Deploy product
3	Huỳnh Công Thành	Team member, Business Analyst, Developer, Tester	- Analyze requirement - Design database - Prepare documents - Design GUI - Code - Test
4	Lê Vũ Đăng Khoa	Team member, Business Analyst, Developer, Tester	- Analyze requirement - Design database - Prepare documents - Design GUI - Code - Test
5	Nguyễn Tường Tâm	Team member, Business Analyst, Developer, Tester	- Analyze requirement - Design database - Prepare documents - Design GUI - Code - Test

Table 6: Roles and responsibilities

2.3. Tools and Techniques

Tool / Technique	Name / version
Frontend	**HTML5, CSS3, JavaScript, jQuery, Bootstrap**
Backend	**ASP.NET MVC 5, Entity Framework, NganLuong, Cloudinary**
Web server	**IIS 8**
Development tool	**Microsoft Visual Studio 2015 Community**
DBMS	**MSSQL Server 2014**
Source control	**Git 2.9.0.windows.1**
Modeling tool	**StarUML 2.7.0**
Document tool	**Microsoft Word 2013**

Table 7: Tools and Techniques

3. Project Management Plan

3.1. Tasks

3.1.1. Initiating

Task name	Initiating
Description	- Clarify user requirements on the rental portal. - Research business logic of car rental service. - Research popular car rental solutions (Grab, Uber, Aleka) as well as other similar websites (Carmudi, Airbnb). - Offer team's solution, declare its pros and cons, introduce some core function of the new system.
Deliverables	Report No.1 – Introduction
Resource needed	20 man-days
Dependencies and constraints	No
Risk	- Missing requirement - Unclear project scope - Lacking teamwork in new formed team

3.1.2. Planning

Task name	Planning
Description	- Review team's solution - Verify project's scope - Build project management plan
Deliverables	Report No.2 – Software Project Management Plan
Resource needed	16 man-days
Dependencies and constraints	- Base on Report No.1 – Introduction. - Planned project must be completed in the following 12 weeks
Risk	Impractical plan

3.1.3. Specifying requirements

Task name	Specifying requirements
Description	Identify and clarify software requirements.
Deliverables	Report No. 3 – Software Requirement Specification
Resource needed	20 man-days
Dependencies and constraints	Base on Report No.2 – Software Project Management Plan.
Risk	Misunderstood or unclear system's requirement

3.1.4. Designing database

Task name	Designing database
Description	- Create conceptual, logical and physical database designs - Implement in SQL Server 2014
Deliverables	Physical database and SQL script.
Resource needed	12 man-days
Dependencies and constraints	Base on Report No. 3 Software Requirement Specification
Risk	Unreasonable database design

3.1.5. Create Software Design Description

Task name	Create Software Design Description
Description	Decide software architect and clarify software detail design.
Deliverables	Report No. 4 – Software Design Description
Resource needed	40 man-days
Dependencies and constraints	Base on Report No. 3 Software Requirement Specification and designed database
Risk	Unreasonable software design

3.1.6. Implementing

Task name	Implementing
Description	Implements all functions of system.
Deliverables	Software package.
Resource needed	80 man-days
Dependencies and constraints	- Base on Software Requirement Specification and Software Design Description. - Follow coding convention.
Risk	Incorrect implementation

3.1.7. Testing

Task name	Testing
Description	- Create test plan - Perform tests. - Fix bugs
Deliverables	Report No. 5 – System Implementation & Test
Resource needed	40 man-days
Dependencies and constraints	Implementation is finished
Risk	- Incompleted test plan - Lacking time for iterative test and bug fixing

3.1.8. Creating User's Manual

Task name	Creating User's Manual
Description	Create user's manuals
Deliverables	Report No.6 – Software User's Manual
Resource needed	12 man-days
Dependencies and constraints	Product has passed acceptance test.
Risk	Bad and unclear documenting

4. Coding Convention

This project follows "C# coding conventions".

- Naming Convention :
  - For variable's name, use camel case (numOfVehicle, currentDate…)
  - For function name, class name, use pascal case (SearchVehicle, VehicleInfoViewModel…)
- Layout Convention :
  - Use the default Code Editor Settings (smart indenting, four-character indents, tabs saved as spaces).
  - Write only one statement/declaration per line.
  - Add at least one blank line between method definitions and property definitions.
  - Use parentheses to make clauses in an expression apparent
- Commenting Convention :
  - Place the comment on a separate line, not at the end of a line of code.
  - Begin comment text with an uppercase letter.
  - End comment text with a period.
  - Insert one space between the comment delimiter (//) and the comment text.
- Language Guidelines :

https://msdn.microsoft.com/en-us/library/vstudio/ff926074.aspx

C. Software Requirement Specification

1. User Requirement Specification

1.1. Guest requirements:

A guest is a person who has not been authenticated by the system. A guest has access to a limited number of system's functions, which are:

- Log in
- Sign up
- Recover account's password
- Search vehicle
- View rental information of vehicle

1.2. Customer requirements:

A customer is an authenticated user that can utilize the system to find and book vehicle. A customer is able to:

- Log out
- Change personal information
- Change account's password
- Search vehicle
- View rental information of vehicle
- Book vehicle
- Cancel booking
- Track booking history
- Comment and rate the car provider's service after:
    - The booking has been canceled
    - The rental period has started
- Receive notification email when:
    - A booking has been placed successfully using this account
    - A booking has been canceled using this account

1.3. Provider requirements:

A provider is an authenticated user that can use the system to manage their car rental business. A provider can:

- Log out
- Change personal information
- Change account's password
- Search vehicle
- View rental information of vehicle
- Manage vehicles:
    - Register vehicle
    - Deregister vehicle

    - Update vehicle's information
    - Group vehicles into garage
    - Group vehicles into vehicle group for price management
- Manage vehicles' bookings:
    - Review booking schedule
    - Create self-booking to reserve vehicle
    - Cancel self-booking
- Manage garages:
    - Register new garage
    - Deregister garage
    - Edit garages' information
    - Close garage
    - Reopen garage
    - Add vehicle to garage
    - Move vehicle to another garage
- Manage vehicle grouping for easier price management:
    - Register new group
    - Deregister group
    - Change group's information and pricing
    - Deactivate group
    - Reactivate group
    - Add vehicle to group
    - Remove vehicle from group
- Extend providership period
- Receive notification email when
    - A vehicle has been booked.
    - A customer has canceled a booking.
- View personal business report

1.4. Administrator requirements:

An administrator is an authenticated user that is tasked with managing the system and its users. Administrator is able to:

- Log out
- Change personal information
- Change account's password
- Search vehicle
- View rental information of vehicle
- Deactivate customer/provider account
- Reactivate customer/provider account
- View system's business report

2. System Requirement Specification

2.1. System Overview Use Case

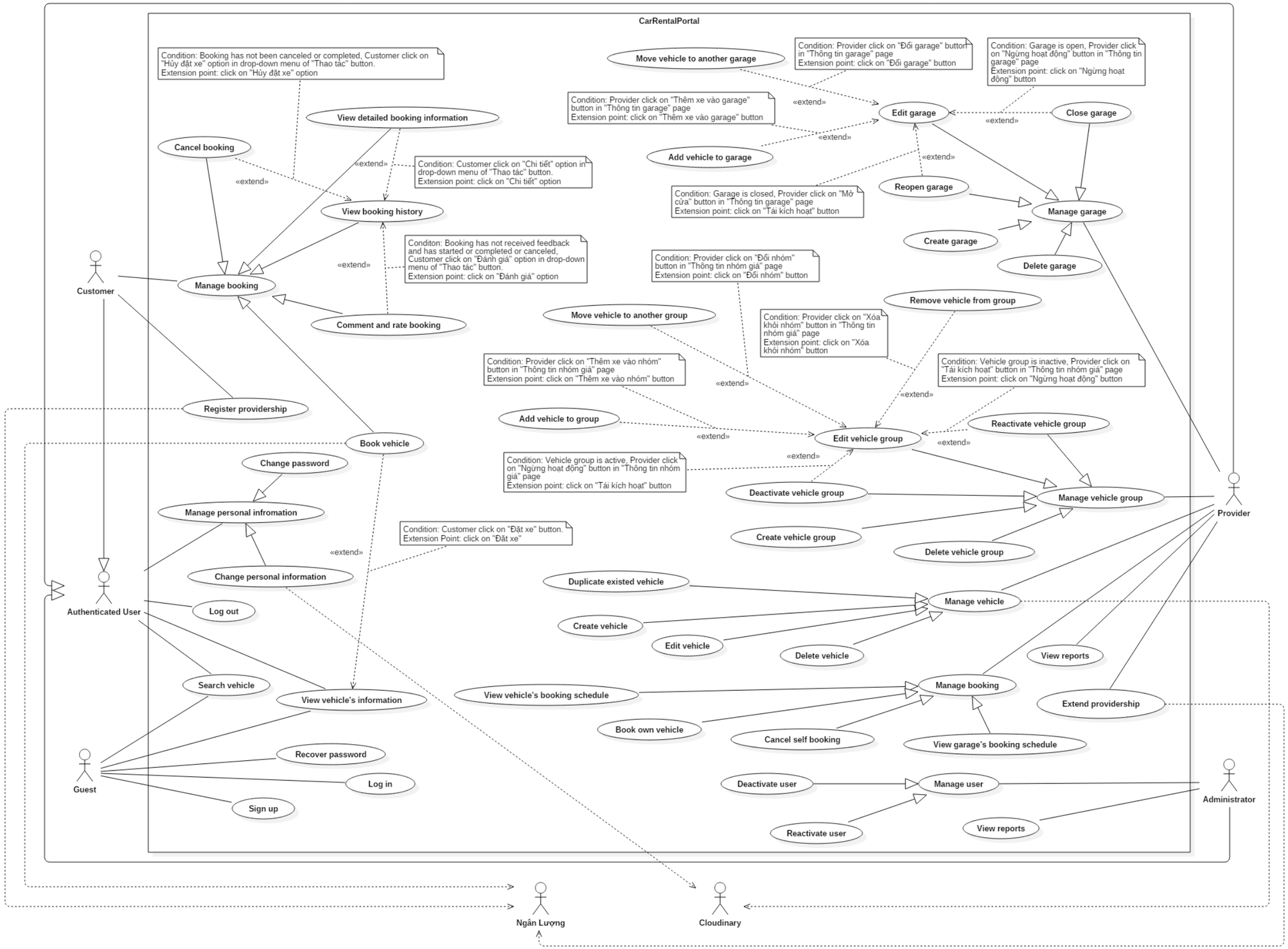

Figure 2: System Overview Use Case

3. Conceptual Diagram

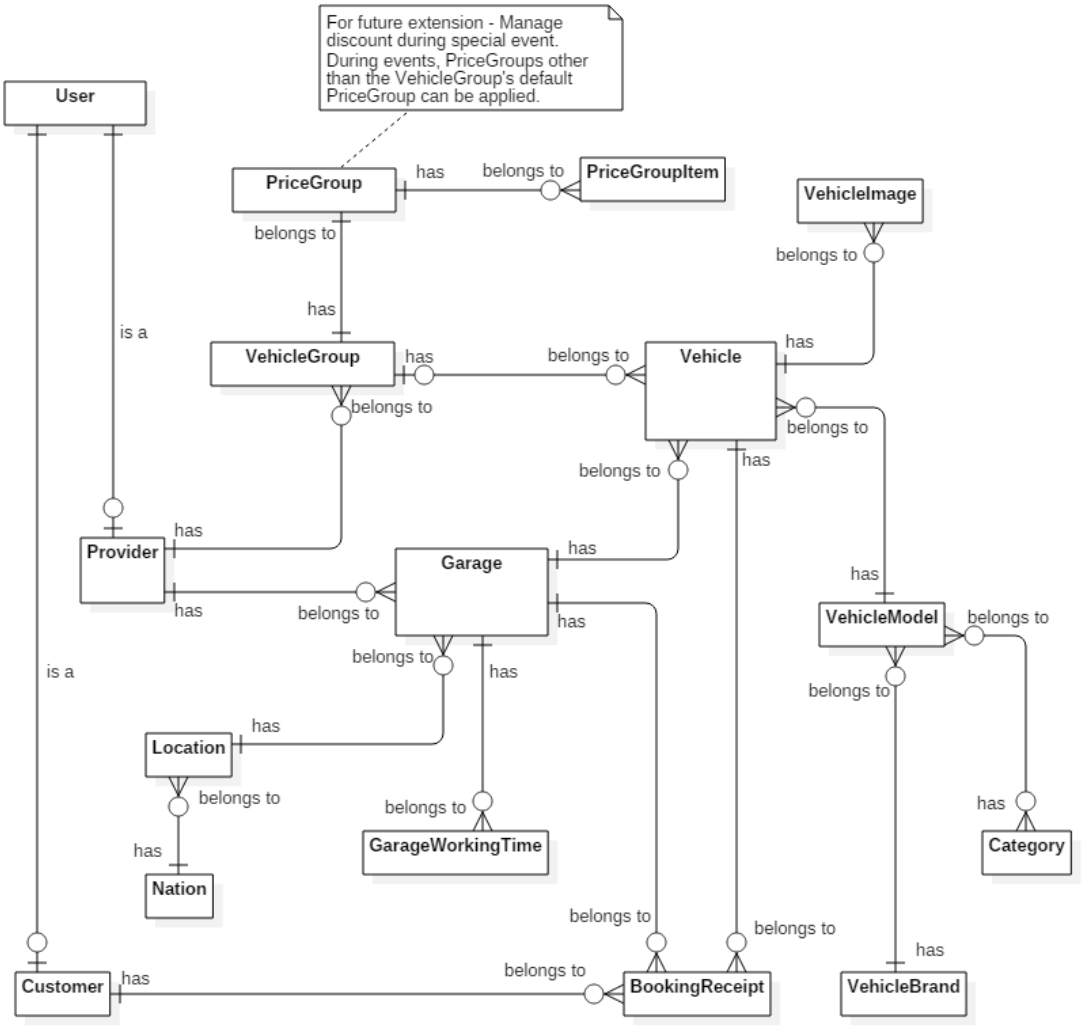

Figure 3: Conceptual Diagram

Entity Data dictionary	
Entity Name	**Description**
User	Abstract entity, represent user in system
Provider	User with provider role
Customer	User with customer role
Vehicle	Represent vehicle in system
VehicleImage	Represent image of vehicle
VehicleModel	Represent vehicle model
VehicleBrand	Represent vehicle brand
Category	Represent vehicle category
Garage	Represent garage
GarageWorkingTime	Describe working time constraints of garage each day of week
Location	Represent location inside a country
Nation	Represent nation
VehicleGroup	Represent group of vehicles with the same rental constraints
PriceGroup	Describe rental constraints of a vehicle group
PriceGroupItem	Describe rental constraints of each daily rental option of price group
BookingReceipt	Represent booking receipt in system

Table 8: Data dictionary

D. Software Design Description

1. Design Overview

This document describes the technical and user interface design of CRP System. The document includes architectural design, detailed design of core functions and business functions, and database design.

The architectural design describes the overall architecture of the system and relations between components and sub-systems inside.

The detailed design describes static and dynamic structure of each component and function. The design includes class diagram, class explanations, and sequence diagrams of each of the important use case.

The database design describes the relationships between system entities and details about attributes of each entity.

Document overview:

- Section 2: System architectural design.
- Section 3: Component diagrams describes the connections and integration of components inside the system.
- Section 4: Detailed design description including class diagram, class explanations, and sequence diagrams.
- Section 5: User interface design.
- Section 6: Database design.
- Section 7: Algorithms.

2. System Architecture Design

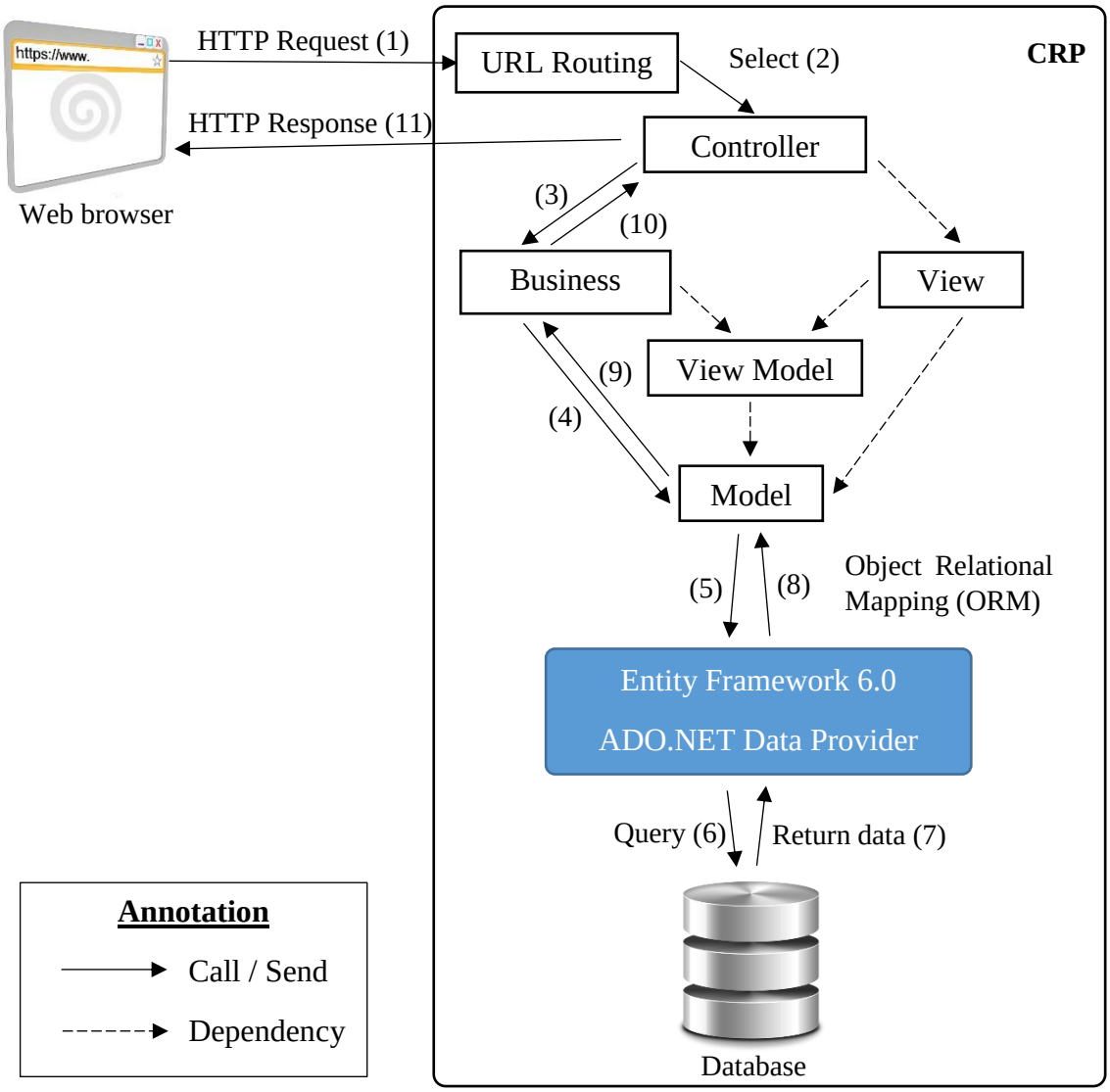

Figure 4: System architecture design

2.1. Web Application architecture description

The system is developed using ASP.NET MVC5 and Entity Framework 6.0 (EF6). The system has 5 important parts:

- **Model** is the part that acts as a data transfer object between the system and database. Models are generated using Object relational mapping (ORM) offered by EF6.
- **View** is the part that handles the display of data. A view renders the display using data from the model or view model that is passed down by controller.
- **Controller** is the part that acts as event handler to handle the user interaction. Typically, a controller reads data from the request, calls the appropriate Business's method, then selects view to render display and then send response back to user.
- **Business** is the part that contains all business handling methods.

- **View Model** is the part that acts like value object. Typically controller passes view model to view and view uses view model as data source for display rendering.

NganLuong's web-service is utilized for payment processing while Cloudinary's web-service is used for image management.

3. Component Diagram

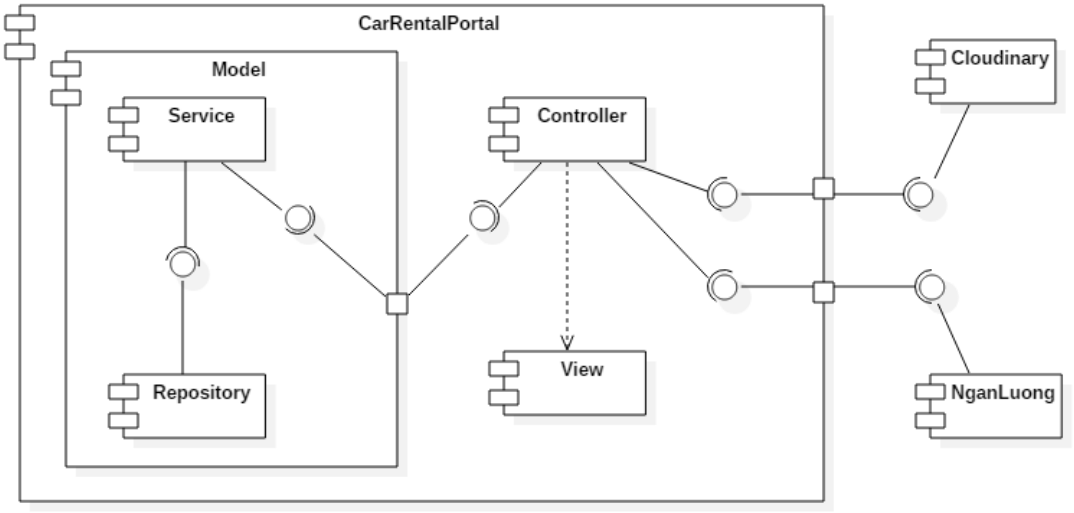

Figure 5 : Component diagram

Component	Description
Controller	Represent all controllers in the system.
View	Represent all views in the system.
Model	Represent all models mapped by Entity Framework.
Repository	Are components that provide simple CRUD operations to interact with database.
Service	Are components that provide more complex, business-related operations, implemented using repository's simple functionality.
Clodinary	Online file management solution
NganLuong	Online payment gateway

Table 9: Component dictionary

4. Detailed Description

4.1. Class Diagram

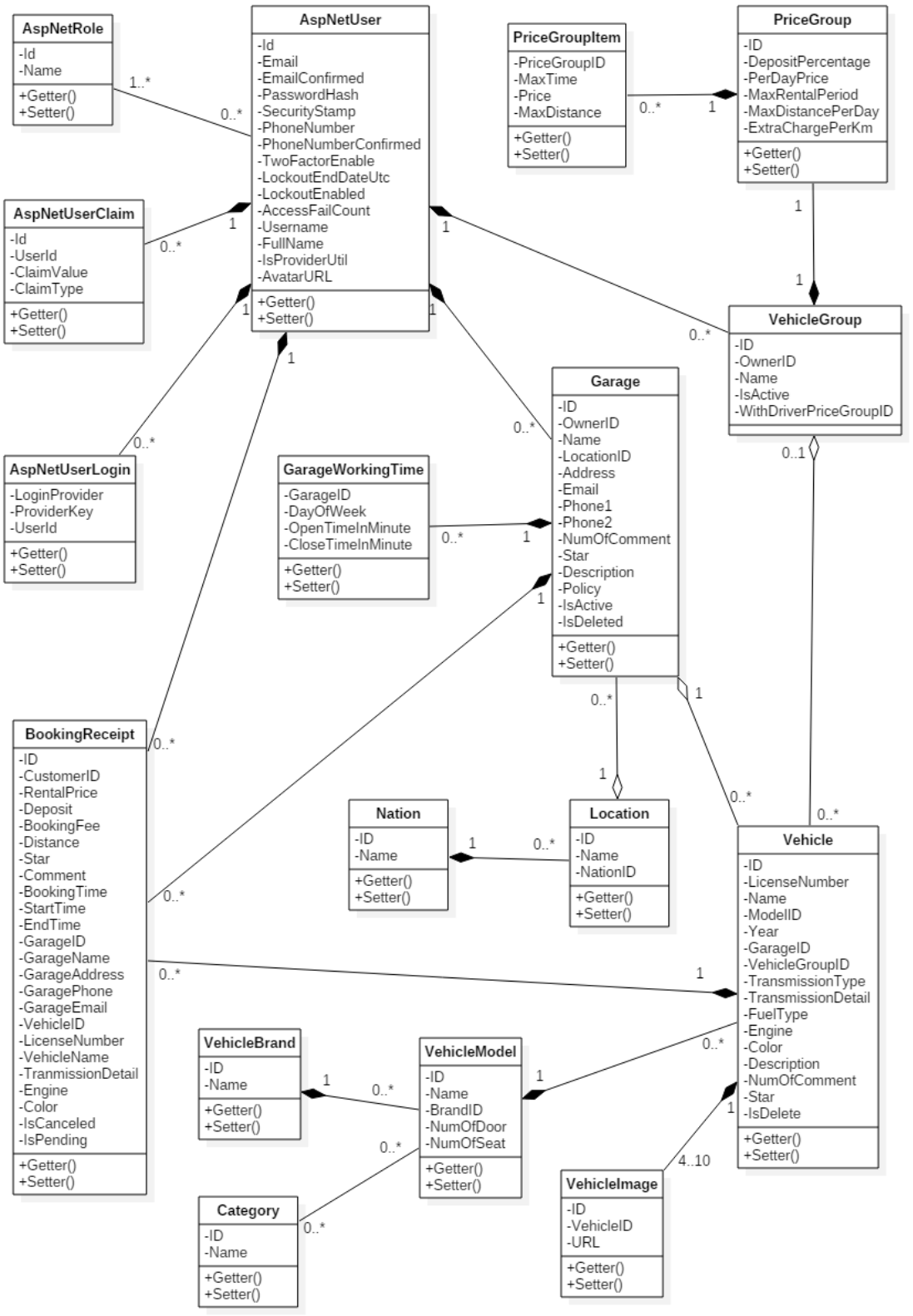

Figure 6: Class Diagram

Class dictionary		
Class Name	**Mapping column with Conceptual diagram**	**Description**
AspNetUser	User	Generated by ASP.NET MVC5. Describe all user in system
AspNetRole	N/A	Generated by ASP.NET MVC5. Describe existing user's roles in system.
AspNetUserClaim	N/A	Generated by ASP.NET MVC5. Describe all stored claims (Special attributes) of AspNetUsers.
AspNetUserLogin	N/A	Generated by ASP.NET MVC5. Describe information about 3rd party/external logins allowed for each AspNetUsers.
Vehicle	Vehicle	Describe all vehicles in system.
VehicleImage	VehicleImage	Describe all images of each vehicle.
VehicleModel	VehicleModel	Describe all vehicle models in system.
VehicleBrand	VehicleBrand	Describe all vehicle brands in system.
Category	Category	Describe all vehicle categories in system.
VehicleGroup	VehicleGroup	Describe all vehicle groups in system
PriceGroup	PriceGroup	Describe rental constraints (Pricing, travel distance…) of each VehicleGroup.
PriceGroupItem	PriceGroupItem	Describe detailed constraints for hourly rental options of each PriceGroup in system.
Garage	Garage	Describe all garages in system
GarageWorkingTime	GarageWorkingTime	Describe working time constraints of each Garage in system.
Location	Location	Describe all locations in system.
Nation	Nation	Describe all nations in system.
BookingReceipt	BookingReceipt	Describe all booking receipts in system.

Table 10: Class dictionary

5. Database Design

6.1 Entity relationship diagram

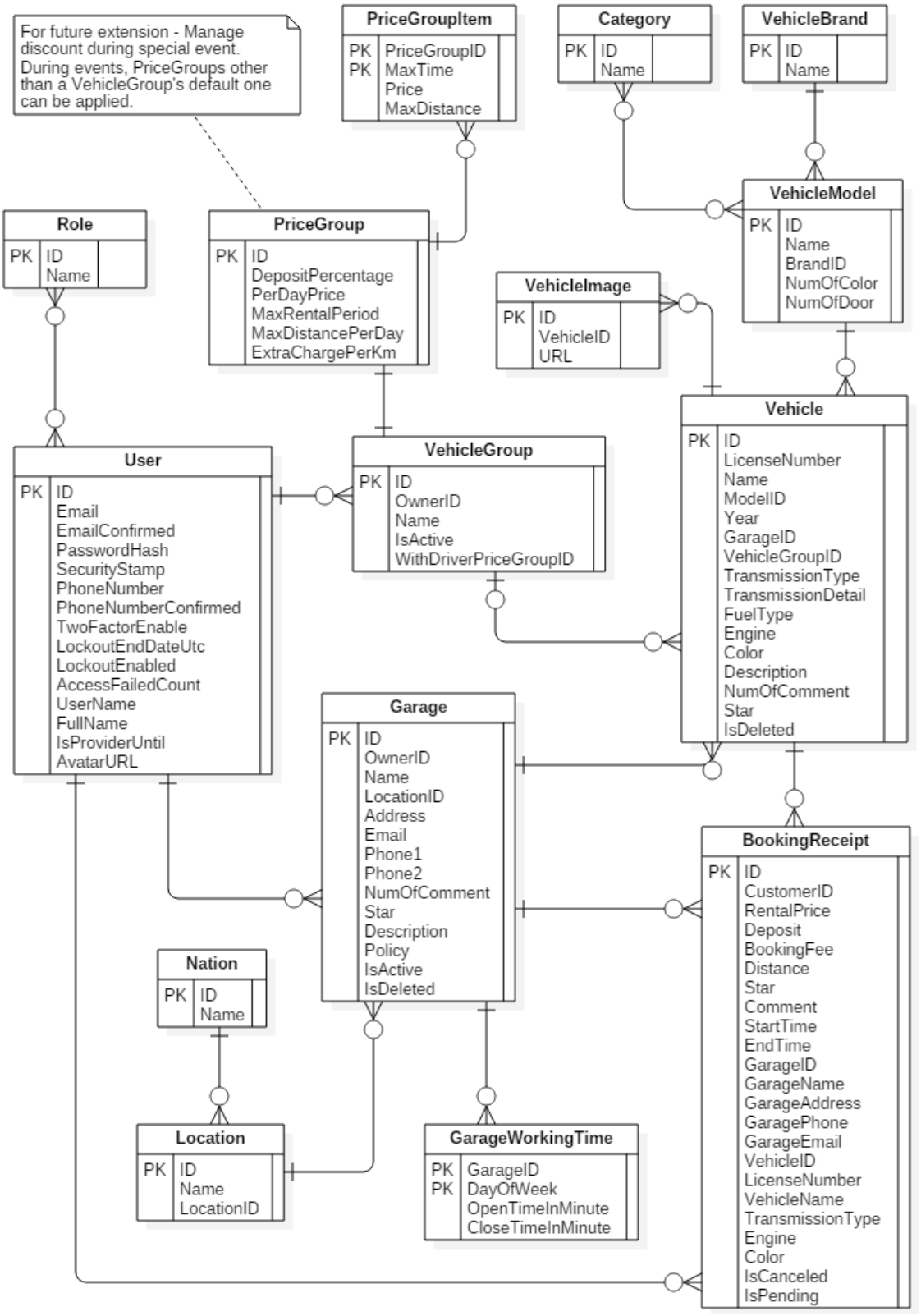

Figure 7 : Entity Relationship Diagram

Entity dictionary	
Entity Name	**Description**
User	Abstract entity, represent user in system
Provider	User with provider role
Customer	User with customer role
Vehicle	Represent vehicle in system
VehicleImage	Represent image of vehicle
VehicleModel	Represent vehicle model
VehicleBrand	Represent vehicle brand
Category	Represent vehicle category
Garage	Represent garage
GarageWorkingTime	Describe working time constraints of garage each day of week
Location	Represent location inside a country
Nation	Represent nation
VehicleGroup	Represent group of vehicles with the same rental constraints
PriceGroup	Describe rental constraints of a vehicle group
PriceGroupItem	Describe rental constraints of each daily rental option of price group
BookingReceipt	Represent booking receipt in system

Table 11 : Entity dictionary

6. Algorithms

6.1. Recommender Engine

6.1.1. Introduction

As a car rental solution, CRP has to be able to introduce the most suitable vehicles to customers. This will helps improving not only the customers' but also the providers' satisfaction, and ultimately gaining more transactions, as well as interest for our application. Consequently, a recommender engine is necessary.

6.1.2. Common approaches

Two common approaches on designing recommender solution are *Collaborative filtering* and *Content-based filtering*.

Collaborative filtering methods are based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences and predicting what users will like based on their similarity to other users. [3]

Content-based filtering methods are based on a description of the item and a profile of the user's preference. In a content-based recommender system, keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past (or is examining in the present). [4]

Collaborative approach does not require analyzing the content of item that it recommend since it is based entirely on the user's information. However, exactly because it depends solely on the data generated by users, it suffers the three common problem of computing, namely *cold start* (Lack of user's interaction in the beginning), *scalability* (Scale badly when the number of user and their interaction increase), and *sparsity* (The number of item is much bigger than the number of user). Considering the low-number-of-transaction nature of car rental service when compares to other kind of services, scalability and sparsity issues can be evaded.

Content-based approach works well even under scarce user's interaction environment since its recommendations are based on the items' description. Its issue of not being able to recommend items with different content type (For instance, car and phone) is also not a problem, since CRP only has one type of item.

6.1.3. Solution's approaches

CRP's recommender design take a hybrid approach between content-based and collaborative. We score vehicles base on how each of them is similar to the vehicles the customer has booked in the past and how many other customers that have booked this vehicle also have booked a similar vehicle as this customer in the past. Vehicles with higher score will have higher priority in recommendation.

Since our system's recommended targets, namely vehicles, have many attributes that we can take advantage of (Brand, number of seat, color…), we initially tackles the problem using content-based methods. These includes *Vector space model* and *tf-idf.*

6.1.3.1. Represent items using vector space model

We first abstract the vehicles and their attributes by applying *Vector space model* [5], an item presentation algorithm.

Each vehicle is modeled as a vector (Refer to as *master vector* from now) in a multi-dimension space (Refer to as *vector space* from now), with each dimension corresponds to an attribute. If a vehicle has an attribute, the component vector (Refer to as *attribute vector* from now) corresponding to that attribute will has non-zero length.

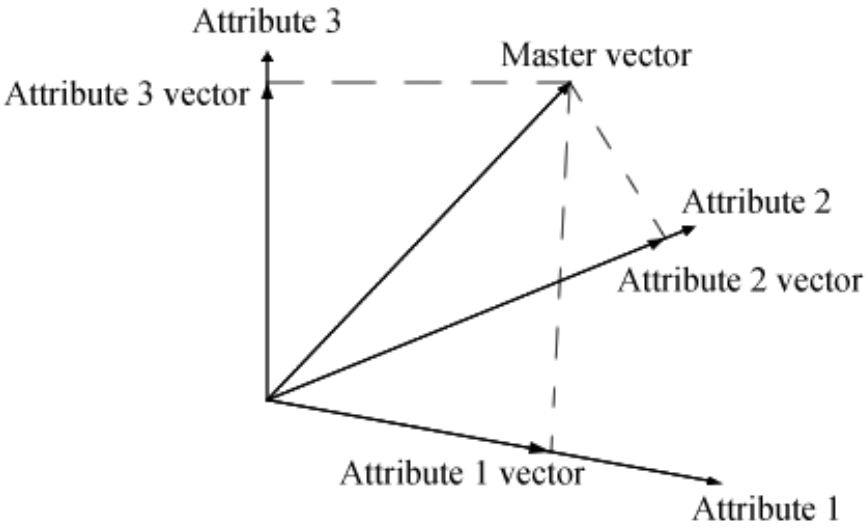

Figure 8 – Example for 3-dimensional vector space

There are several weighting scheme that can be used to calculate the attribute vector's length. Amongst them, the most popular one is *tf-idf* (Term frequency – Inverse document frequency) [6], which. In our solution, tf-idf is applied with *binary scheme* (Further explanation will be given in *solution's design* section).

6.1.3.2. Customer profile

A customer profile is another vector in the *vector space* which indicate a particular customer's interest in vehicle, like which color or which fuel type that he has more affinity with.

In vector space model, the angle between 2 vectors determine the similarity between them. This means the smaller the angle between a vehicle's *master vector* and a customer profile, the more similar that vehicle is to the customer's reference.

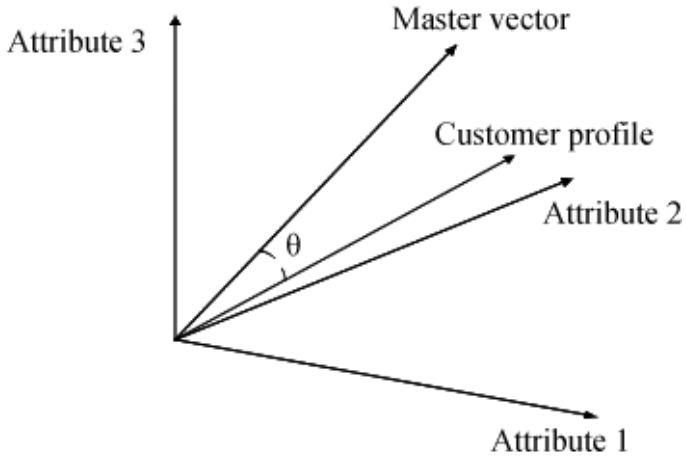

Figure 9 – Example for customer profile vector

It is common to use the cosine of this angle to represent their similarity, since their values are between 1 and -1. This form allows us to tell at how many percent a customer will like a vehicle (Positive value), or dislike it (Negative value). Our recommender engine will find the cosine between each vehicle's *master vector* and a customer profile, then recommend vehicles with the highest cosine value to that customer.

6.1.3.3. Mix in collaborative element

To further improve the diversity of our solution's recommendation, we apply collaborative methods into our content-based solution. Our collaborative approach work under the assumption that other customers that has booked the same vehicle with this customer (Refer to as neighbor from now) will have similar vehicle reference; and the more *neighbors* a vehicle has, the more similar it is to this customer's reference.

Our collaborative approach introduces new attributes into the *vector space*. Each of these new attribute represent a neighbor and whether the item has been booked by this neighbor before.

	Neighbor 1	Neighbor 2	Neighbor 3	Neighbor 4
Vehicle 1	Yes	Yes	No	No
Vehicle 2	No	Yes	No	No
Vehicle 3	Yes	Yes	Yes	No

Table 12 – Example of vehicle with 4 neighbor attributes

6.1.4. Solution's design

6.1.4.1. How to calculate cosine of angle between 2 vectors

Cosine of the angle θ between 2 vectors $a = [a_1, a_2, \dots, a_n]$ and $b = [b_1, b_2, \dots, b_n]$ in n-dimension *vector space* can be calculated as follow:

$$\cos\theta = \frac{a \cdot b}{\|a\|\|b\|}$$

where $a \cdot b$ is dot product of 2 vectors, $\|a\|$ and $\|b\|$ is the norm of each vector.

The dot product can be calculated as follow:

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

The norm of a vector $v = [v_1, v, \dots, v_n]$ can be calculated as follow:

$$\|v\| = \sqrt{\sum_{i=1}^{n} v_i^2}$$

6.1.4.2. Apply tf-idf weighting scheme

Applying tf-idf, length, or 'weight' of each *attribute vector* of $v = [v_1, v, \dots, v_n]$ can be calculated as follow:

$$v_i = tf_i \cdot idf_i$$

idf (Inverse document frequency) can be calculated as follow:

$$idf_i = \log \frac{|D|}{|df_i| + 1}$$

with $|D|$ being the total number of item and $|df_i|$ being the total number of item that has attribute i.

The scheme to calculate tf will be discussed in the next section.

6.1.4.3. Binary representation of attributes

In our approach, raw attributes can only either appear or does not appear in an item. This leads to representing them as binary values. Under this form, we can apply *binary tf weighting scheme* of *tf-idf*, where tf weight equals the raw binary value.

	4-seat	7-seat	Gasoline	Diesel	Neighbor 1	Neighbor 2
Vehicle 1	1	0	1	0	1	1
Vehicle 2	1	0	0	1	0	1
Vehicle 3	0	1	1	0	1	1
Vehicle 4	0	1	1	0	1	0
Vehicle 5	1	0	0	1	0	0

Table 13 – Example tf values of vehicles with 6 binary attributes

With this approach, there is also no need to apply normalization to eliminate item's size bias (For document-like item, the total number of 'word' in them varies, meaning bigger document will more likely to have more 'hit'(tf) for each search term/attribute. However, with binary attributes, an item either 'has' or 'does not have' an attribute. In other words, tf is limited to 1 and 0, and therefore has no such bias).

attribute		4-seat	7-seat	Gasoline	Diesel	Neighbor 1	Neighbor 2
tf	Vehicle 1	1	0	1	0	1	1
	Vehicle 2	1	0	0	1	0	1
	Vehicle 3	0	1	1	0	1	1
	Vehicle 4	0	1	1	0	1	0
	Vehicle 5	1	0	0	1	0	0
df		3	2	3	2	3	3
idf (With D = 5)		0.097	0.222	0.097	0.222	0.097	0.097
weight	Vehicle 1	0.097	0	0.097	0	0.097	0.097
	Vehicle 2	0.097	0	0	0.222	0	0.097
	Vehicle 3	0	0.222	0.097	0	0.097	0.097
	Vehicle 4	0	0.222	0.097	0	0.097	0
	Vehicle 5	0.097	0	0	0.222	0	0

Table 14 – Example weight of 5 vehicles with 6 binary attributes

6.1.4.4. Build the customer profile

We can build a user profile by using that user's booking history as reference. Every booking has all the necessary data to construct an item vector similar to vehicle vector.

#	Vehicle	4-seat	7-seat	Neighbor 1	Neighbor 2	Star
1	Vehicle 1	1	0	1	1	2
2	Vehicle 5	1	0	0	0	1
3	Vehicle 3	0	1	1	1	4
4	Vehicle 2	1	0	0	1	5
5	Vehicle 3	0	1	1	1	-

Table 15 – Example of 5 bookings with 4 attributes

The bookings also have star-rating, which we can utilize to determine whether the customer liked or dislike the booking. In our approach, we assume that a rating lower than 3-star indicates *dislike* (-1), higher than 3-star indicates *like* (1) and equals 3 or empty rating means *neutral* (0).

#	Vehicle	4-seat	7-seat	Neighbor 1	Neighbor 2	Star	Like
1	Vehicle 1	-1	0	-1	-1	2	-1
2	Vehicle 5	1	0	0	0	1	-1
3	Vehicle 3	0	1	1	1	4	1
4	Vehicle 2	1	0	0	1	5	1
5	Vehicle 3	0	0	0	0	-	0

Table 16 – Example tf of 5 bookings with 4 attributes applying like-dislike scheme

tf of each of the customer profile's *attribute vector* can be calculated as tf of sum of tf of every booking's attribute vector of the same dimension. Since these sum are no longer binary, we calculate customer profile's tf_i of dimension i using *log normalization* scheme instead:

$$tf_i = tf\left(\sum_j^m tf_j\right) = \begin{cases} 1 + log\sum_j^m tf_j, & if \sum_j^m tf_j > 0 \\ -\left(1 + log\left(-\sum_j^m tf_j\right)\right), & if \sum_j^m tf_j < 0 \\ 0, & otherwise \end{cases}$$

In this equation, m is the total of booking. This equation has been modified from the original scheme to accomplice negative value. The customer profile's vector can be calculated using our default tf-idf weighting scheme.

Attribute	#	Vehicle	4-seat	7-seat	Neighbor 1	Neighbor 2
Booking's tf	1	Vehicle 1	-1	0	-1	-1
	2	Vehicle 5	-1	0	0	0
	3	Vehicle 3	0	1	1	1
	4	Vehicle 2	1	0	0	1
	5	Vehicle 3	0	0	0	0
df			3	1	2	3
idf (D=5)			0.097	0.398	0.222	0.097
Customer profile	**tf**		-1	1	0	1
	weight		-0.097	0.398	0	0.097

Table 17 – Example weight for attribute vectors of a customer profile

6.1.4.5. Calculate vehicle's score

As mentioned, the score used to recommend vehicle will be the cosine between the vehicle master vector and the customer profile.

$$cos\,\theta = \frac{a \cdot b}{\|a\|\|b\|} = \frac{\sum_{i=1}^n a_i b_i}{\sqrt{\sum_{i=1}^n a_i^2} \cdot \sqrt{\sum_{i=1}^n b_i^2}}$$

Attribute			4-seat	7-seat	Neighbor 1	Neighbor 2
Customer profile			-0.097	0.398	0	0.097
Vehicle	**tf**	Vehicle 1	1	0	1	1
		Vehicle 2	1	0	0	1
		Vehicle 3	0	1	1	1
		Vehicle 4	0	1	1	0
		Vehicle 5	1	0	0	0
	df		3	2	3	3
	idf (D=5)		0.097	0.222	0.097	0.097
	weight	Vehicle 1	0.097	0	0.097	0.097
		Vehicle 2	0.097	0	0	0.097
		Vehicle 3	0	0.222	0.097	0.097
		Vehicle 4	0	0.222	0.097	0
		Vehicle 5	0.097	0	0	0
	Score	Vehicle 1				0
		Vehicle 2				0
		Vehicle 3				0.889907
		Vehicle 4				0.866331
		Vehicle 5				-0.23042

Table 18 – Example score for 5 vehicles with 4 attributes

A positive value shows us the probability the customer will like the vehicle, while a negative one shows us the probability the customer will dislike the vehicle.

6.1.5. Algorithm's time complexity

Considering a *vector space* with n attributes, k vehicles, and a customer with m bookings in her booking history; assuming all n attributes of an item (either a vehicle or a booking) has been ready before-hand, C#'s Math.Log10(double) and Math.Sqrt(double) have O(1) time complexity, we can estimate the time complexity of this recommender algorithm as follow.

6.1.5.1. Build customer profile

Step	**Complexity**
Calculate each attribute vector a_i of the profile a	O(m)
• Calculate idf_i	O(m)
❖ Calculate $\lvert df_i \rvert$	O(m)
❖ Calculate $idf_i = \log \frac{\lvert D \rvert}{\lvert df_i \rvert + 1}$	O(1)
• Calculate tf_i	O(m)
❖ Calculate $\sum_j^m tf_j$	O(m)
❖ Calculate $tf_i = \begin{cases} 1 + log \sum_j^m tf_j, & if \ \sum_j^m tf_j > 0 \\ -\left(1 + log\left(-\sum_j^m tf_j\right)\right), & if \ \sum_j^m tf_j < 0 \\ 0, & otherwise \end{cases}$	O(1)
• Calculate $a_i = tf_i \cdot idf_i$	O(1)
Total	**O(nm)**

*Table 19 – Time complexity of **Build customer profile** step*

6.1.5.2. Build vehicle vectors

Step	Complexity
Calculate each vehicle vector v	O(nk)
• Calculate each attribute vector v_i of v	O(k)
❖ Calculate idf_i	O(k)
Calculate $\lvert df_i \rvert$	O(k)
Calculate $idf_i = \log \frac{\lvert D \rvert}{\lvert df_i \rvert + 1}$	O(1)
❖ Calculate $v_i = tf_i \cdot idf_i$	O(1)
Total	**O(nk²)**

*Table 20 – Time complexity of **Build vehicle vectors** step*

6.1.5.3. Score vehicles

Step	Complexity
Calculate norm $\lVert a \rVert = \sqrt{\sum_{i=1}^{n} a_i^2}$	O(n)
Calculate each vehicle's score	O(n)
• Calculate norm $\lVert v \rVert = \sqrt{\sum_{i=1}^{n} v_i^2}$	O(n)
• Calculate dot product $a \cdot v = \sum_{i=1}^{n} a_i v_i$	O(n)
• Calculate vehicle's score $\cos\theta = \frac{a \cdot b}{\lVert a \rVert \lVert b \rVert}$	O(1)
Total	**O(nk)**

*Table 21 – Time complexity of **Score vehicles** step*

6.1.5.4. Overall time complexity

The overall time complexity of our algorithm is **O(n(k² + k + m))**. This complexity has 2 pain points. The first being k^2, which can somehow be reduced by applying normal filtering first to lessen the number of vehicle in vector space. The second pain point is the number of collaborative attributes in n.

n is the sum between the content-based attributes presented naturally on every vehicle and the collaborative attributes which are the neighbors a vehicle has. The more booking the customer makes, the more neighbors he may have, and the bigger n will become. However, as a car rental solution, our transactions tend to be big in value but small in quantity. The average number of booking a single customer makes may just stop at a few dozen. Further conclusion can only be drawn with concrete statistics, however.

Conclusively, this algorithm's scalability is quite poor, but this is common and is still very acceptable as a recommendation solution, especially those follow collaborative approach. In the future that the system will increase in size, there will be a need for either an upgrade of hardware and software to increase computing power, or the development of a new engine.

6.2. Rental payment procedure

CRP not only helps customer to place booking but also helps provider to take advance payment for each booking.

To place a booking, customer must first pay an *AdvanceFee* to the system. This includes the booking *Deposit*, which is advance payment part of total *RentalPrice*, and *BookingFee*, which is the fee paid for our service.

AdvanceFee = Deposit + BookingFee

Provider can specify how many percent of the total *RentalPrice* customer has to pay in advance using VehicleGroup's *DepositPercentage*.

*Deposit = RentalPrice * DepositPercentage*

The rest of the *RentalPrice* can be paid later either at the start or the end of rental period, depending on provider's rental policy.

The *BookingFee* is calculated base on *RentalPrice*. As out solution is new, we propose that the *BookingFee* should be low (1% of *RentalPrice*). We can raise *BookingFee* later as our service becomes more mature and successful.

E. システムの展開とテスト

1. データベース関係ダイアグラム

1.1 ウェブアプリケーション物理的なダイアグラム

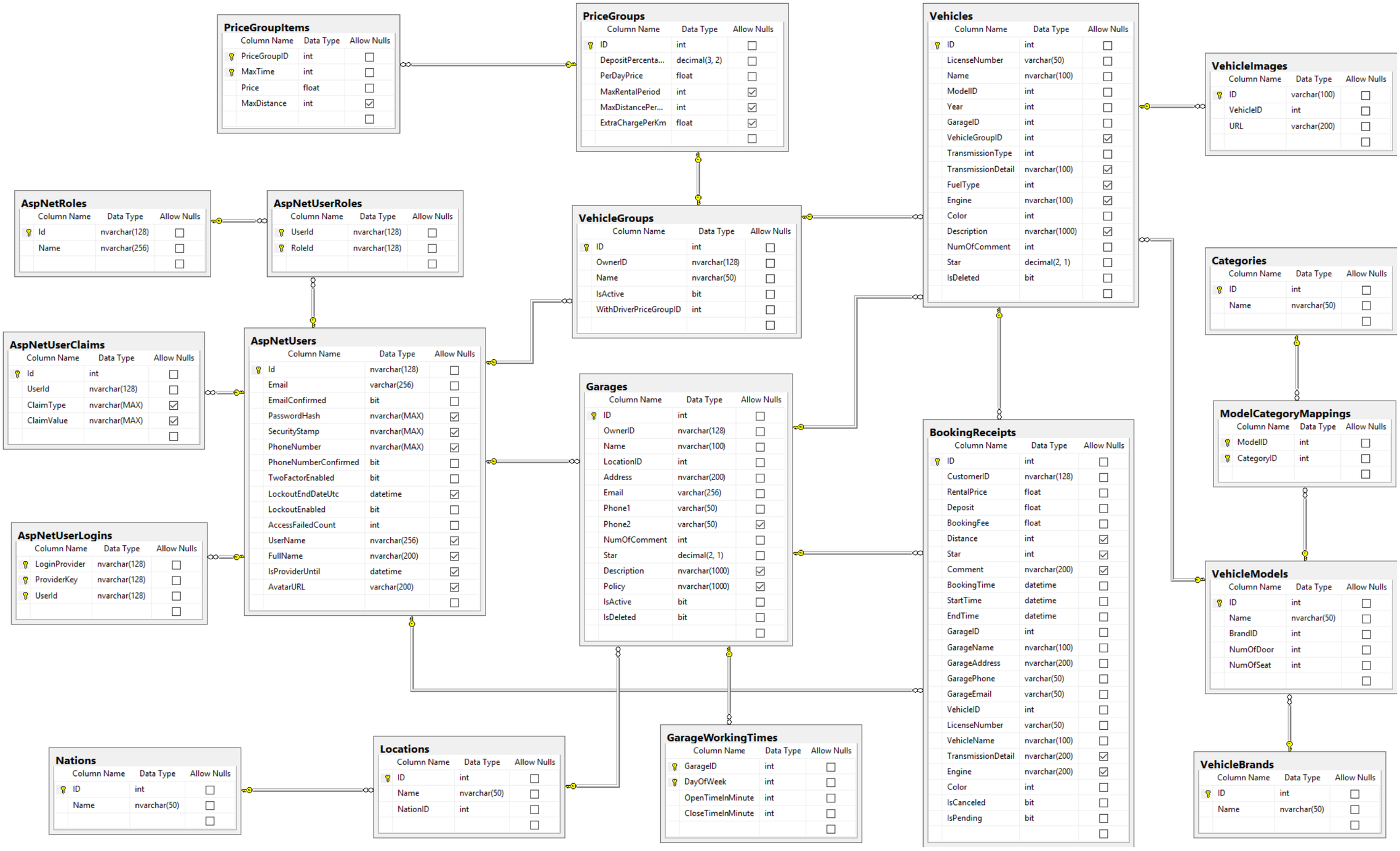

Figure 10: ウェブサイトアプリケーション物理的なダイアグラム

F. Tasksheet

No.	Product Deliverables	Task	DucTH	ThanhHC	KhoaLVD	TamNT	Man-day
1	**Report1 - Introduction**	Project introduction		**O**			1
		Current situation		**O**			1
		Problem definition				**O**	1
		Proposed solution				**O**	1
		Feature functions	**O**	**O**			2
		Advantages and disadvantages			**O**		1
		Functional requirement	**O**				2
		Review and merge document	**O**				2
2	**Report2- Software Project Management Plan**	Problem definition				**O**	1
		Project organization	**O**				1
		Project management plan	**O**				2
		Coding Convention			**O**		1
		Review and merge document	**O**				2
3	**Report 3- Software Requirement Specification**	User Requirement Specification		**O**			2
		System Requirement Specification					
		External interface requirements			**O**		1
		System overview use case		**O**			3
		List of use cases					
		<Guest> Log in/Log out				**O**	0.25
		<Guest> Sign up				**O**	0.25

		<Guest> Recover password				**O**	0.25
		<Guest> Search vehicle				**O**	0.75
		<Guest> View vehicle's information				**O**	0.25
		<Authenticated user> Log out				**O**	0.25
		<Authenticated user> Change password				**O**	0.25
		<Authenticated user> Change personal information				**O**	0.25
		<Authenticated user> Search vehicle				**O**	0.75
		<Authenticated user> View vehicle's information				**O**	0.25
		<Customer> Book vehicle	**O**				0.5
		<Customer> View booking history	**O**				0.25
		<Customer> View detailed booking information	**O**				0.25
		<Customer> Cancel booking	**O**				0.25
		<Customer> Comment and rate booking	**O**				0.25
		<Customer> Register providership	**O**				0.5
		<Provider> Create vehicle			**O**		0.5
		<Provider> Duplicate existed vehicle			**O**		0.5
		<Provider> Edit vehicle			**O**		0.5
		<Provider> Delete vehicle			**O**		0.25
		<Provider> View vehicle's booking schedule			**O**		0.25
		<Provider> View garage's booking schedule			**O**		0.25
		<Provider> Book own vehicle			**O**		0.25
		<Provider> Cancle self-booking			**O**		0.5
		<Provider> Create garage		**O**			0.5
		<Provider> Edit garage		**O**			0.5
		<Provider> Close garage		**O**			0.25

		<Provider> Reopen garage		**O**			0.25
		<Provider> Add vehicle to garage		**O**			0.25
		<Provider> Move vehicle to another garage		**O**			0.25
		<Provider> Create vehicle group		**O**			0.5
		<Provider> Delete vehicle group				**O**	0.25
		<Provider> Edit vehicle group		**O**			0.5
		<Provider> Deactive vehicle group				**O**	0.25
		<Provider> Reactive vehicle group				**O**	0.25
		<Provider> Add vehicle to group				**O**	0.25
		<Provider> Remove vehicle from group				**O**	0.25
		<Provider> Move vehicle from group				**O**	0.25
		<Provider> View reports			**O**		0.5
		<Provider> Extend providership			**O**		0.5
		<Admin> View reports				**O**	0.5
		<Admin> Deactive user				**O**	0.25
		<Admin> Reactive user				**O**	0.25
		Software System Attribute		**O**			1
		Conceptual Diagram		**O**	**O**		2
		Review and merge document		**O**			2
4	**Report 4- Software Design Description**	System architecture design	**O**				1
		Component diagram		**O**	**O**		1
		Class diagram		**O**			2
		Interactive diagram					
		Sequence diagram	**O**	**O**	**O**	**O**	2
		Activity diagram	**O**	**O**	**O**	**O**	2

		User interface design	**O**	**O**	**O**	**O**	2
		Database design	**O**	**O**	**O**	**O**	2
		Review and merge document	**O**				2
5	**Report 5 - Software Implementation and Test Document**	Database relationship diagram				**O**	2
		Test plan	**O**				1
		Test cases		**O**			1
		Review and merge document	**O**				2
6	**Report 6 - Software User's Manual**	Installation Guide	**O**				1
		User's Guide		**O**			1
		Review and merge document	**O**				1
7	**Database**	Design database	**O**	**O**	**O**	**O**	11
		Review database	**O**	**O**	**O**	**O**	1
8	**Architecture**	Design project structure	**O**				2
		Apply framework & design pattern	**O**		**O**		2
9	**Implementation**	Implement database	**O**			**O**	5
		Guest functions	**O**			**O**	20
		Authenticated user functions	**O**				10
		Customer functions	**O**				30
		Provider functions		**O**	**O**	**O**	25
		Admin functions				**O**	10

G. Appendix

1. UML 2.0 standard
   http://www.omg.org/spec/UML/2.0

2. Sashimi – Modified Waterfall Model:
   http://www.waterfall-model.com/sashimi-waterfall-model

3. Collaborative filtering approach
   https://en.wikipedia.org/wiki/Recommender_system#Collaborative_filtering

4. Content-based approach
   https://en.wikipedia.org/wiki/Recommender_system#Content-based_filtering

5. Vector space model
   https://en.wikipedia.org/wiki/Vector_space_model

6. tf-idf
   https://en.wikipedia.org/wiki/Tf%E2%80%93idf