MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Insurance Card

Group 2	
Group members	Đinh Quang Trung – SE60994 Nguyễn Hữu Phúc – SE60749 Phùng Quang Minh Trí – SE60746 Nguyễn Chí Kha – 60351
Supervisor	Kiều Trọng Khánh
Ext. Supervisor	N/A
Capstone Project Code	MIC

- Ho Chi Minh City, 12 May 2015 -

This page is intentionally left blank

Table of Contents

List of Tables

List of Figures

Definitions, Acronyms, and Abbreviations

Name	Definition
MIC	Motor Insurance Card
NFC	Near field communication

Table 1: Definitions, Acronyms, and Abbreviations

A. Introduction

1. Project Information

- Project name: **Insurance Card**
- Project Code: **MIC**
- Product Type: **Website & Android Application**
- Start Date: **May 11th, 2015**
- End Date: **August 24th,2015**

2. Introduction

In this document, we introduce a solution for motorbike insurance company. Current insurance company systems have some problems like delayed in renew contracts for customer or inconvenient in checking insurance card validation process. Based on our researches and analysis, we proposed a solution for insurance company in Vietnam and other developed countries.

We build a system, which help the insurance companies to solve current problems. In the process of analysis, we believe the NFC cards is capable to resolve the problem by using NFC card to save information about insurance contract. NFC cards are convenient to manage the contract information and checking, validating process. Beside of that we also provide an information system to manage NFC cards so that insurance companies will manage the contracts easier.

This document also describes our working process in 4 months includes our perspective in the system, component designs and detailed core workflows. We hope the system and our solution will help resolve the problems from insurance companies in Vietnam and other developed countries.

3. Current Situation

When participating in traffic, vehicle owners are required to have compulsory insurance (according to Article 6, Decree on compulsory insurance for civil liability of motor vehicle owners, Decree No. 103/2008/ND-CP by Vietnam Government). Therefore, vehicle owners buy insurance from insurance companies or its agents. They pay insurance premium by cash or in online website and receive an insurance certificate with a term of one year, the term can be shorter in some specific situation. When their insurance out of date, they must buy a new insurance, old certificate will be useless. Traffic police will read insurance certificate to check traffic participants.

4. Problem Definition

Below are disadvantages of current situation:

- **Forget insurance's expired date**: Vehicle owners usually keeps their insurance certificate in wallet or somewhere on their vehicle. However, except in cases of necessity, people are not often check their insurance so they could forget its expired date. An expired insurance is not good while it be revealed by traffic officers and could get worse in case of traffic accident.

- **Hard for traffic officers to check and verify insurance**: Traffic officers must read insurance certificate to check and verify vehicle owner's information. It can be difficult and hinder their work in some cases as at dark or handwriting illegible on insurance certificate.
- **No mechanism to renew old contract**: customers have to handy register new contract when the old one is expired, this is inconvenient for customers.
- **Insurance certificate made of paper:** It could be torn, wet, smudged and especially is counterfeited.
- **Claim/compensation process is ineffective** between customer and insurance company.
- **Difficult to track and manage number of traffic violations and collisions:** In current scenario, insurance companies almost impossible knows vehicle owner's history to adjust their insurance policy.

According to Vietnam's laws, motor vehicle owners must have insurance contract with fixed term and fixed fee for each type of vehicle.

5. Proposed Solution

Our proposed solution is to build an insurance NFC card system named "MIC system" to resolve the current situations and compatible with current laws, we also design the system to be scalable so we can deploy this system to a multiple insurance services company in future plan.

MIC system includes a web application and two mobile applications with following functions:

5.1. Feature functions

- Web application:
  - **Register insurance**: user can register a new insurance card with on website using online payment. A staff will contact the user to create contract and sends an insurance NFC card to him/her. If users already have a NFC card, they can use the website to renew current contract.
  - **Check card information**: user can login into the website and check for their card's information.
  - **Request compensation**: user can fill data into the sample fields and sends compensation request to the company.
  - **Make/manage contracts**: staff can make and manage contracts.
  - **Resolve compensation**: staff can receive and resolve compensation requests.
  - **Notify contract state**: system will send an email to notify the insured one when their insurance is expired.
  - **Notify compensation state**: system will send an email to info the insured one when their compensation were accepted or rejected.
- Insurance card printer (mobile app):
  - **Simulating NFC card printer**: staff can print NFC card.
- Insurance card checker (mobile app):

- **Check card:** traffic police and Police Department can check specified motor insurance card expired or not.
- **Update the punishment of violator:** traffic police and Police Department can update the punishment of violator to the card information.

5.2. Advantages and disadvantages

- Advantages:
  - The interaction between the insured one and the insurance company: the insured one and the company now are easier to communicate through the website when each person has an account.
  - Reduce risk of insurance card made of paper: the NFC insurance card will not be torn, wet or smudged. Moreover, it is difficult to be counterfeit than insurance card made of paper.
  - Support police to check valid insurance card easier.
- Disadvantages:
  - Currently not consistent with the law of Vietnam about insurance card issues.
  - Checking the valid of card can take a long time when the internet is slow.

6. Functional Requirements

Function requirements of the system are listed as below:

- **User component:**
  - New contract request
  - Check card information.
  - Renew contract.
  - Request compensation.
  - New card request
  - Cancel contract
- **Staff component**
  - Create new contracts
  - Manage contracts.
  - Resolve compensation requests.
  - Resolve new card request
- **System component**
  - Manage contract states
- **Payment system**
  - Process payments
- **Notify component**
  - Notify contract expiration.
  - Notify compensation states (approved / rejected).
- **Checker mobile application**
  - Check card validation.
  - Update punishment information.
  - Retrieve card information.

- **Printer mobile application**
  - Get contract information from server.
  - Print NFC insurance card.

7. Roles and Responsibility

No	Full Name	Role	Position	Contact
1	Kiều Trọng Khánh	Project Manager	Supervisor	khanhkt@fpt.edu.vn
2	Đinh Quang Trung	Developer	Leader	trungdqse60994@fpt.edu.vn
3	Nguyễn Hữu Phúc	Developer	Member	phucnhse60749@fpt.edu.vn
4	Phùng Quang Minh Trí	Developer	Member	tripqmse60746@fpt.edu.vn
5	Nguyễn Chí Kha	Developer	Member	khanc60351@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**: Insurance Card
- **Vietnamese name**: Thẻ bảo hiểm
- **Abbreviation**: MIC

1.2. Problem Abstract

As current in Viet Nam customer use Motor Insurance Certificate Paper when they get problems with their motor. Using the Motor Insurance Certificate Paper is inconvenient, for example, it can be wet or to insert or update the information in to insurance certificate paper is complicate. So we use the NFC card we call it is insurance card to handle it. Insurance company supplies the NFC card when the customer buy insurance. The card contains the information of customer, if the customer joins with many insurance service they just use only one card.

We provide a software to check the validation of card, the expired date of card and insurance services that customer joined. We also provide other advantages that can help save time and costs in some process of company. For example, the software can automatic extend the insurance service, update the information about accidents of motor. In addition, we also provide a system software to manage the information of customer via some insurance card we bought, this software will deploy at insurance company.

1.3. Project Overview

1.3.1. Current Situation

Below are the problems encountered in this project:

- **Security**: currently, there is few possible problems encountered with NFC tags, as NFC tags can be counterfeited, attacked during data transmission caused data loss, data corruption.
- **Server crash**: all the needed data is stored in the server. So if server crash, all the devices cannot get card information.
- **Absence of team members**: team members can get sick or unexpected problems.
- **Currently not consistent with the law** of Vietnam about insurance card issues.

1.3.2. The Proposed System

According to the technology researches, we found out that the NFC technology is very capable of resolve the current situations in insurance companies. We can use a feature of NFC tag to resolve the security problem from NFC card. The basic idea is to use a NFC tag (or NFC "card") which contains a unique card ID as an insurance card instead of paper card currently.

We also build a high available webserver to maintain the main system to work 24/7 to make sure that if mobile applications need access to the information there will be always available.

We assign responsibility in vertical to make sure if any member in this problem cannot continue to work in our team there will be the least harmful to the project processes.

To resolve problem from Vietnam laws of insurance for motorbike, we support the insurance companies to propose new law sections about using technology devices to work with insurance certificate paper to make our system work legally in current situation.

Our system includes three main subsystems: an online website for company's staffs, a mobile application for police officers and a mobile application to simulate the card printer.

1.3.2.1.Website

Website is a common communication portal for insurance company's staffs and users (customers). Website provide following features:

- For users (customers):
  - Users can register new insurance card with online payment.
  - Users can look up information about their insurance card: compensation history, punishment history, expired date…
  - Users can renew current insurance contract with online payment.
  - Users can request compensations to insurance company when an accident occurs.
  - Users will be notified by emails when insurance card is nearly expired or a compensation request is approved/rejected.
- For staffs:
  - Staffs can create new contract for customer.
  - Staffs can manage contracts, see all insurance cards published and see statistics
  - Staffs can update compensation requests, resolve a compensation request when the case is done.

Beside above, website system also provides an API interface for two mobile applications to retrieve, update data from mobile applications.

1.3.2.2.Printer Mobile Application

This is a simulating application to simulate the work of Card Printer. In reality, the company who deploy this system need to have a NFC Card Printer to write information about the insurance company and customer information into an NFC card. However, our system currently

only support this as a simulating application. This application is used by company's staffs and do followings:

- Retrieves insurance contract information and write data to a physical NFC card.

1.3.2.3.Checker Mobile Application

This is a simulating application to simulate the work of Card Checker Device for traffic police officers. In practical situation this application will connect to the center database from police department to get information about vehicle. In this project we simulate this device as an application run on an Android smartphone.

Traffic police officer uses this mobile application. This application does followings:

- Check if an insurance card (NFC card) is valid or not.
- Send punishment if the customer has law violations. Punishment information will be updated in server.

1.3.3. Boundaries of the System

This section supposes that the government laws in local area supports the method of using NFC cards as insurance cards, and accept NFC insurance cards are legal.

- Every company who has Information System infrastructure can deploy this system.
- Companies who deployed this system has to equip enough devices for the system to run, includes:
  - Computer system with internet connection.
  - Smartphone devices with built-in NFC technology.
- The language of this system is Vietnamese
- The complete product includes:
  - Website application for staffs and users
  - Printer mobile application for staffs.
  - Checker mobile application for traffic police officers

1.3.4. Future plans

Current system only can deploy to a company which provides single service: motor insurance card, we call this is the **Isolated Single Service Model**. We design the system to make it easy to scale to 2 bigger models:

- **Isolated Multiple Service Model:** system can be deployed to one company which provide multiple insurance services such as motor insurance, health insurance, assets insurance…
- **Distributed Multiple Service Model:** system can be deployed to multiple companies; each company provides multiple insurance services. The companies all share information through a center system to maintain the synchronization between companies.

1.3.5. Development Environment

1.3.5.1.Hardware requirement

- For continuous integrating server:

Hardware	Minimum Requirements	Recommended
Internet Connection	512Kbps	8 Mbps
Operating System	Ubuntu Server 12 LTS	Ubuntu Server 14.04.2 LTS
Computer Processor	Intel® Pentium II	Intel® Core(TM) i5 CPU , M 460 @ 2.53GHz
Computer Memory	128MB of RAM	3GB of RAM or more

Table 3 Hardware requirement for continuous integrating server

- For web development:

Hardware	Minimum Requirements	Recommended
Internet Connection	512Kbps	8 Mbps
Operating System	Windows Vista, 7, 8	Windows 7, 8
Computer Processor	1 GHz	Intel® Core(TM) i5 CPU , M 460 @ 2.53GHz
Computer Memory	1GB of RAM	3GB of RAM or more

Table 4 Hardware requirement for web development

- For mobile development:

Hardware	Minimum Requirements	Recommended
Internet Connection	512Kbps	Wi-Fi Connection 12MB
Operating System	Android 4.0	Android 4.0
Hardware	NFC supported	NFC supported
Memory	128MB of RAM	1GB of RAM or more

Table 5 Hardware requirement for mobile development

1.3.5.2.Software requirement

Software	Name / Version
Operating system	Windows 7 or above
Environment	Java EE 6
Modeling tool	Microsoft Visio 2013
IDE	Netbeans 7.2.1, Intellij IDEA 14.1
DBMS	MySQL 5.6
Source control	TortoiseSVN 1.8.11
Web browser	Chrome 42 or above

Table 6 Software requirement

2. Project organization

2.1. Software Process Model

This project is developed under waterfall model. We apply customized waterfall model to capable with current situation in our team. We choose this model because the following reasons:

- This project is 4 months long due to the FPT University Capstone Project timeline, which can be consider a short project.
- Based on researches and clarify Vietnam laws of insurance for motorbike and current system in insurance companies, the requirements of this project are stable, clear, fixed and well understood by all team members.
- This project use NFC technology, which we have strong background knowledge and well practice skills. We also have experience in designing, building web and mobile application system.

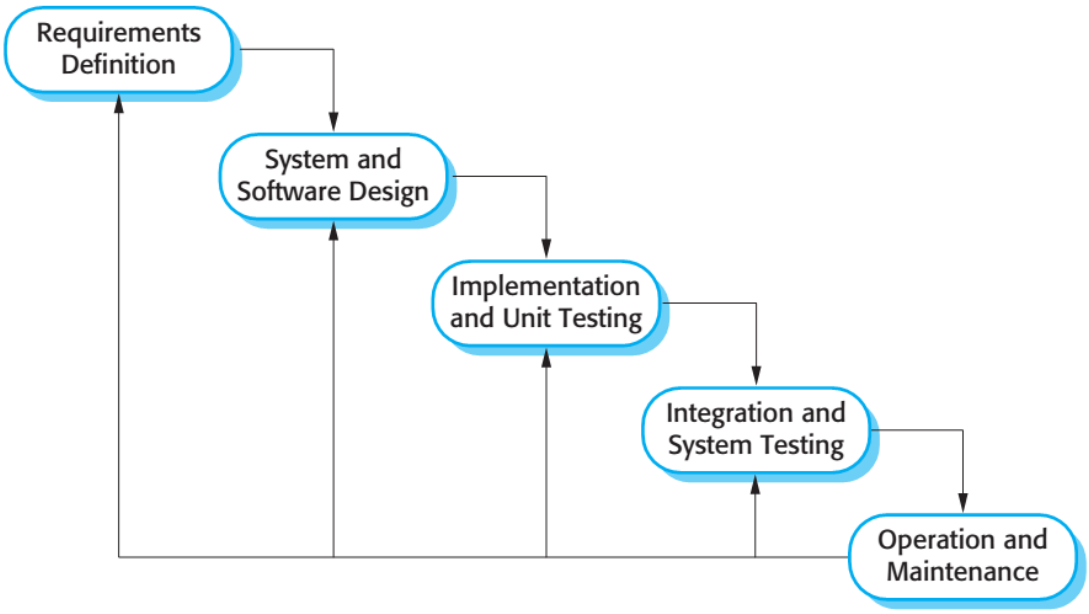


Figure 1: Waterfall model

Reference: Page 30, chapter 2, Software process model, SOFTWARE ENGINEERING 9th Edition, by Ian Sommerville.

We customize the waterfall model from the reference to make the process more capable with current situation of our team.

2.2. Roles and responsibilities

No	Full name	Role in Group	Responsibilities
1	Kiều Trọng Khánh	Supervisor / Project Manager	- Clarify user requirement. - Technical support and business analysis. - Tracking development process. - Review document and product.

2	Đinh Quang Trung	Team leader, BA, Developer, Tester	- Tracking process. - Planning project, distribute tasks. - Requirement analysis. - Database design. - Documentation. - GUI Design. - Coding. - Testing. - Deploy product.
3	Nguyễn Hữu Phúc	BA, Developer, Tester	- Requirement analysis. - Database design. - Documentation. - GUI Design. - Coding. - Testing.
4	Phùng Quang Minh Trí	BA, Developer, Tester	- Requirement analysis. - Database design. - Documentation. - GUI Design. - Coding. - Testing.
5	Nguyễn Chí Kha	BA, Developer, Tester	- Requirement analysis. - Database design. - Documentation. - GUI Design. - Coding. - Testing.

Table 7 Roles and responsibilities

2.3. Tools and Techniques

Tool / Technique	Name / version
Frontend	HTML, CSS, JavaScript, jQuery, Bootstrap
Backend	JavaEE, Servlet, JSP, Hibernate
Web server	Apache Tomcat 7
Development tool	NetBeans 7.2.1, IntelliJ IDEA 14
DBMS	MySQL 5.6
Source control	TortoiseSVN 1.8.11
Modeling tool	StarUML 5.0, Lucid Chart
Document tool	Microsoft Word 2013

Table 8 Tools and Techniques

3. Project Management Plan

3.1. Software development life cycle

Below are all the major tasks that need to be performed sequentially during the development of the system.

Phase	Description	Deliverables	Resource needed	Dependencies and Constrains	Risk
Requirements Definition	Identify and clarify system requirements.	Report No.1 Introduction.	20 man-days	N/A	- Missing requirement. - Project's scope can be unclear. - Lack of member share and understand.
System and Software Design	- Identify hardware and software requirements. - Decide software architect and clarify software detail design. - Design database.	Report No.2 Software Project Management Plan, Report No. 3 Software Requirement Specification and Report No. 4 Software Design Description.	50 man-days	Depend on Requirements Definition.	- Misunderstood or unclear system's requirement. - Lack of practical experience leading to unreasonable design.
Implementation and Unit Testing	- Implements all functions of system. - Create test plan. - Perform Unit testing.	Software package.	120 man-days	- Base on Software Requirement Specification and Software Design Description. - Coding try to follow coding convention.	- Member does not performs unit test. - Lack of practical experience.
Integration and System Testing	- Perform integration test and system test.	Report No. 5 System Implementation & Test	35 man-days	Implementation and Unit Testing are finished.	- Lack of testing experience leading to lack of test cases. - Not enough time for performing test.

Operation and Maintenance	- Deploy the system - Create the user's manuals. - Do routine maintenance activities.	Report No.6 Software User's Manual	15 man-days	Integration and System Testing are finished.	User's manual may be difficult for user to understand and confuse.

Table 9 Software development life cycle

3.2. Phase Detail

3.2.1. Phase 1: Requirements Definition

Task	Description	Author
Identify and clarify system requirements.	Research current systems to collect requirements. Define main and needed functions the system must include.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha

3.2.2. Phase 2: System and Software Design

Task	Description	Author
Identify hardware and software requirements.	Find out the suitable hardware and software for the system, as well as its minimum and recommended requirements.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Decide software architect and clarify software detail design.	- Define the major software components and interfaces. - Draw core flow diagram, use case diagram, prototype … - Group meeting to review and modify.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Design database.	- Design database for the system.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha

3.2.3. Phase 3: Implementation and Unit Testing

Task	Description	Author
Implements all functions of system.	Coding all the components.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Create test plan.	Planning for testing.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Perform Unit testing.	- Write Unit test cases. - Implement Unit tests.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha

3.2.4. Phase 4: Integration and System Testing

Task	Description	Author
Perform integration test and system test.	- Test groups of modules and test whole the system.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha

3.2.5. Phase 5: Operation and Maintenance

Task	Description	Author
Deploy the system	Deploy the system in client environment.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Create the user's manuals.	Create a guideline to instruct users using system.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha
Do routine maintenance activities.	Do routine maintenance activities for client system.	Đinh Quang Trung Nguyễn Hữu Phúc Phùng Quang Minh Trí Nguyễn Chí Kha

3.3. Task sheet

Refer to "Task sheet" folder.

3.4. All Meeting Minutes

Refer to "Meeting minutes" folder.

4. Coding Convention

This project follows "Code Conventions for the Java TM Programming Language, by Sun Microsystems, rev April 20, 1999".

http://www.oracle.com/technetwork/java/codeconventions-150003.pdf

We use followings naming convention from the reference to capable with current situation in our team:

- Naming:
  - Class names must be in Pascal case.
  - Variable names must be in Camel case.
  - Each Java class belongs to a single file.
- Intentions:
  - Use four spaces intentions.
  - Avoid lines with more than 80 characters
- Declaration:

- One declaration per line is recommended since it encourages commenting.
- In absolutely no case should variables and functions be declared on the same line.
- Do not put different types on the same line.

C. Software Requirement Specification

1. User Requirement Specification

1.1. Customer requirement

Customer is user who uses service of system. The customer can use some following functions:

- View history information include:
  - View payment history
  - View accident history
  - View punishment history
- View compensation history
- View contract information
- View personal information
- Create new contract
- New card request
- Renew contract
- Cancel contract
- Request compensation

1.2. Staff requirement

Staff is people who works directly with system to track the information of customer or manages customer, staff can handle directly some problems if it happens from customer. Staff can use some following functions:

- View profile: they can change password of customer
- View customer information
- View card information
- Resolve new card request
- Resolve compensation requests
- Update contract type information
- Manage customer includes:
  - View customer information
  - Edit customer information
  - Create new customer
- Manage contracts includes:
  - Create new contract
  - Update contract information
  - Renew contract
  - Cancel contract
  - Update compensation history
  - Update punishment history
  - Update accident history
- Print NFC card for customer

1.3. Police requirement

Police is people who is interactive with system for checking information about customer's NFC card and handling in case the customer violates the traffic rule or make accident. Police can use some following functions:

- Verify card validation
- Get contract information
- Update punishment information

1.4. Admin requirement

Admin is people who manages staff. Administrator can use some following functions:

- Manage staff includes:
  - Remove staff
  - Add staff

2. System Requirement Specification

2.1. External Interface Requirement

2.1.1. User interface

- The user interface uses Vietnamese language.
- Use consistent palette of colors between the text and the background.
- The user interface displays best on 1024x768-screen size.

2.1.2. Hardware Interface

- Smartphone with NFC support.

2.1.3. Software Interface

- Web application: work with Firefox (v30 or above), Chromes (v14 or above), Internet Explorer (v10 or above) browse.
- Mobile application: Android operating system (v 4.0 or above).

2.1.4. Communication Protocol

- Use HTTP protocol 1.1 for communication between the web browser and the web server.
- Use HTTP protocol 1.1 for communication between the mobile application and the web service.

2.2. System Overview Use Case

2.2.1. Web Application

This page is intentionally left blank

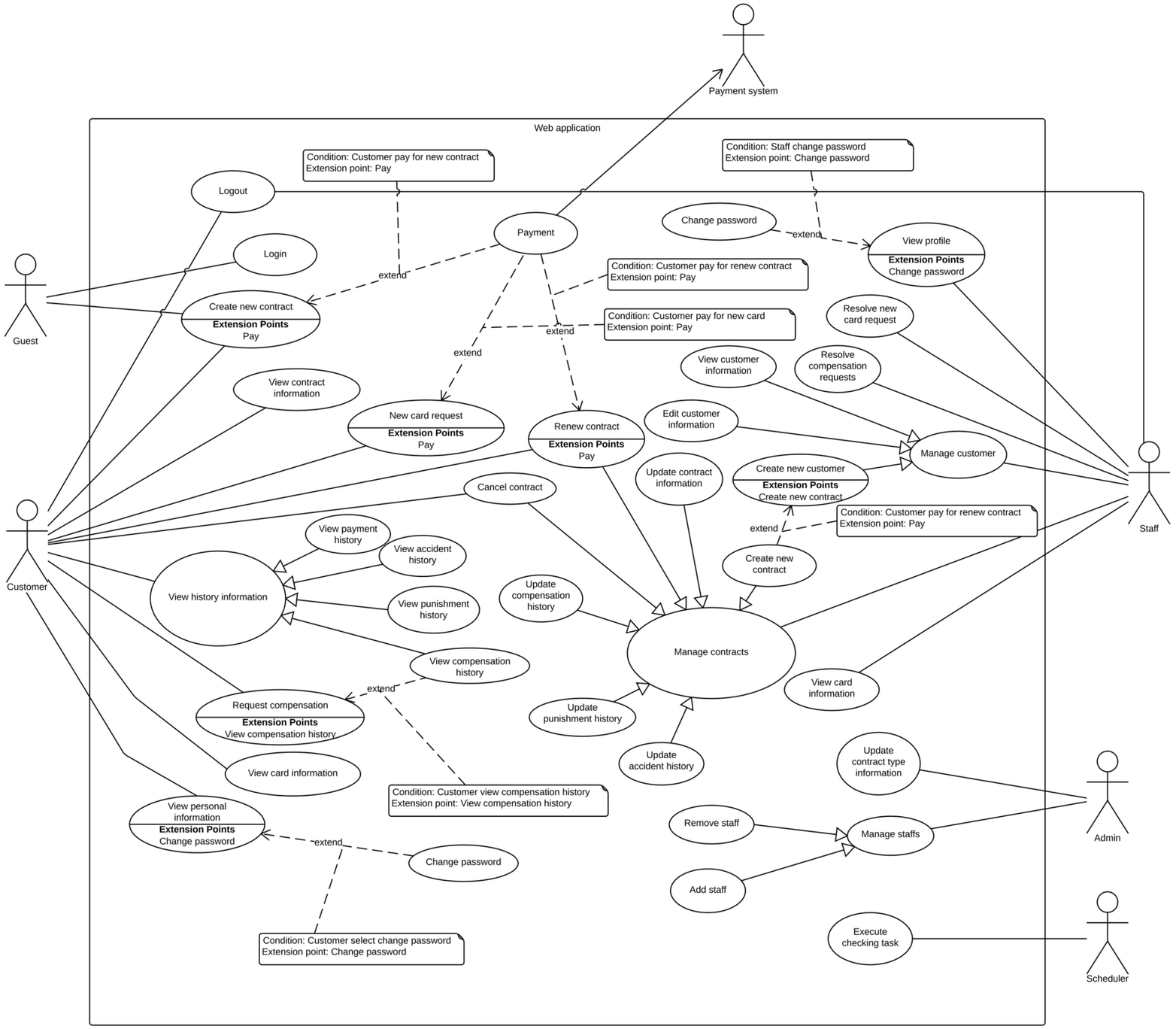


Figure 2: Web Application Overview Use Case

This page is intentionally left blank

2.2.2. Checker Mobile Application

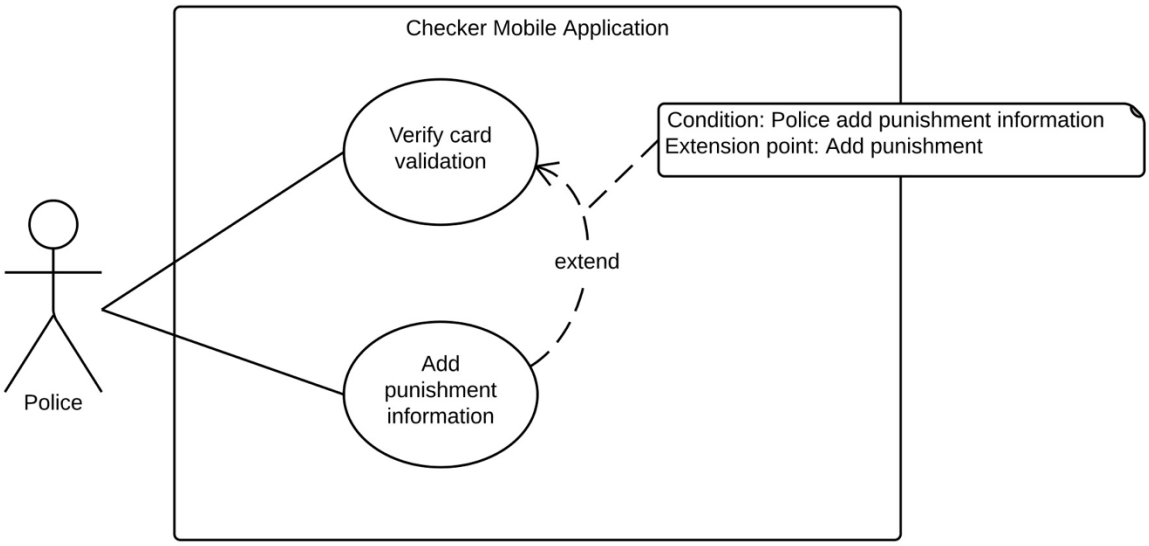


Figure 3: Checker Mobile Application

2.2.3. Printer Mobile Application

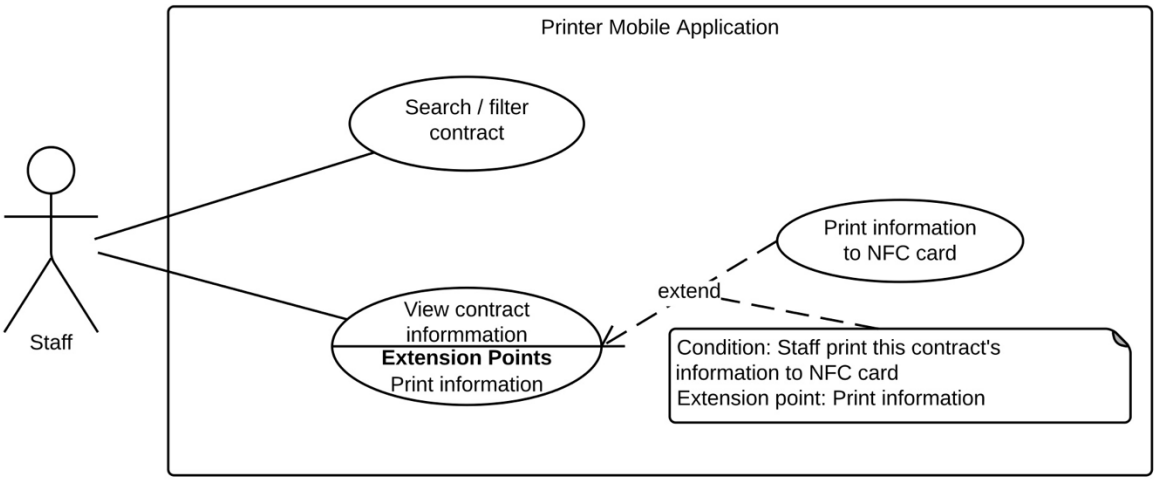


Figure 4: Printer Mobile Application

2.3. List of Use Case

2.3.1. Web Application

2.3.1.1.<Guest> Overview Use Case

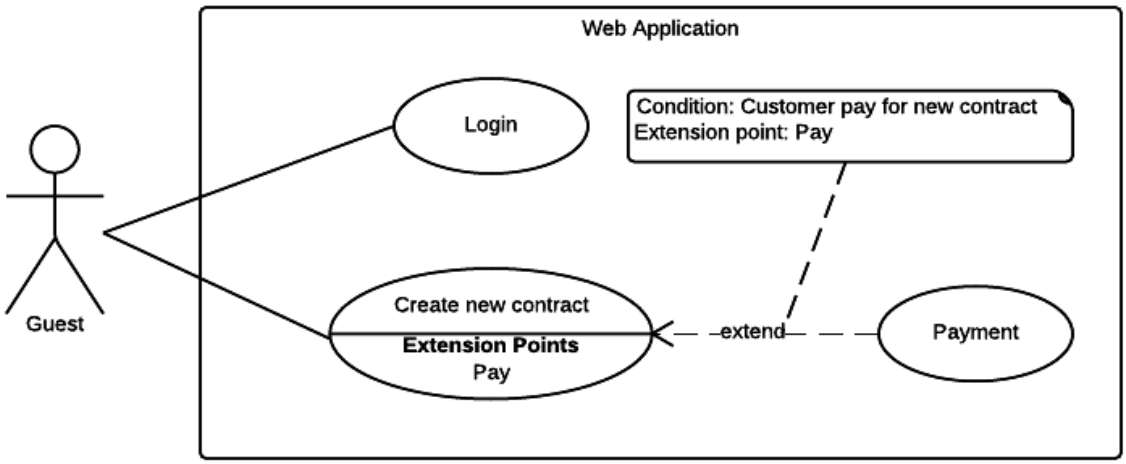


Figure 5 <Guest> Overview Use Case

2.3.1.1.1. <Guest> Create new contract request

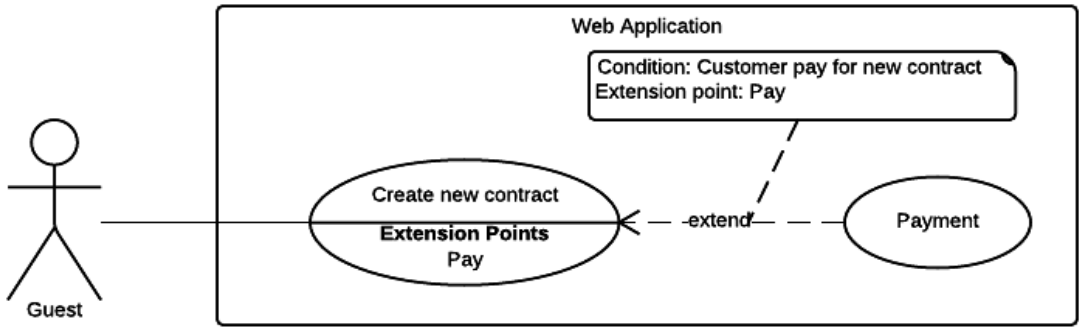


Figure 6 <Guest> Create new contract request

USE CASE – WG02			
Use Case No.	WG02	**Use Case Version**	2.0
Use Case Name	Create new contract request		
Author	TrungDQ		
Date	27/05/2015	**Priority**	Normal

Actor:
- Guest

Summary:
- This use case allows guest to create new contract request.

Goal:
- Guest can create new contract request.

Triggers:
- Guest sends command to create contract request.

Preconditions:
- N/A

Post Conditions:

- **Success:** New account and new contract will be created for guest.
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Guest goes to new contract view.	System requires information from guest: **Personal information** - Name: free text input, required, length 3 – 80. - Address: free text input, required, length 3 – 250. - Email: free text input, required, length 3 – 250. - Phone number: free text input, required, length 8 – 15. - Personal ID: free text input, length 8 – 15. **Contract information** (all information below are required) - Contract's type: select one of the options. - Start date: date time input, required. - Contract term: text - Contract's fee: text **Vehicle information** - Plate: free text input, required, length 4 – 15. - Brand: free text input, required, length 2 – 20. - Model code: free text input, length 2 – 20. - Vehicle type: free text input, length 2 – 20. - Color: free text input, length 2 – 20. - Engine: free text input, required, length 2 – 20. - Chassis: free text input, required, length 2 – 20. - Capacity: free text input, required, length 2 – 20. - Year of manufacture: number text input, value from 1900 to current year. - Weight: free text input, value from 1 – 1000, unit: kilogram - Seat capacity: free text input, value from 1 – 100. **Security question** - Answer: free text input, required, length 1 - 10
2	Guest inputs information.	
3	Guest sends command to create	System validate information, display contract details and request for confirmation. [Exception 1, 2, 3]

	new contract request.	
4	Guest sends command to create new contract request.	Add new account and new contract information to the system. Show successful message and ask user to process payment.
5	Guest sends command to process payment	Display new view let user select one of following payment gateways: - PayPal payment gateway. - Direct payment. And show guest the fee: Contract's fee: text.
6	If user chooses PayPal gateway and sends confirm command. [Alternative 1]	Forward to PayPal payment view to process the payment.
7	User process the PayPal payment	If payment succeed: Show message created successful. [Exception 4]

Alternative Scenario:

No	Actor Action	System Response
1	If user chooses direct payment method	Show company address map.

Exceptions:

No	Actor Action	System Response
1	Guest sends command to create new contract request	System shows error message to ask user input missing required fields.
2	Guest's email is existed in the system	Show message to notify guest that their email is existed in the system.
3	Guest's vehicle plate is existed in the system	Show message to notify guest that their vehicle is existed in the system.
4	If payment failed	Show message to notify user that payment failed and the renew request has been aborted.

Relationships: Payment

Business Rules:

- New customer account and new contract will be created in the system with inputted information.
- The initial status of contract will be set to "Pending".
- When customer completed payment process:
  + if the contract's start date has come, contract's status would change from "Pending" to "No Card".
  + If start date is not come yet, the contract status is not changed.
- Staff will receive a notification about new contract request, they verify contract's information and issue a card for this contract, in this case, contract's status would change from "No Card" to "Ready".
- System must ensure has no duplicate customer or vehicle.

- An email contains customer code and password will be sent to user, user can use this information to login to the system later. - Start date must not be earlier than the current date. - Contract term is specified by the system. - Contract types are loaded from system, contract type can be managed by system administrator. - Contract price would be calculated from contract type and contract term.

Table 10 Use case WG02 - <Guest> Create new contract request

2.3.1.2.<Customer> Overview Use Case

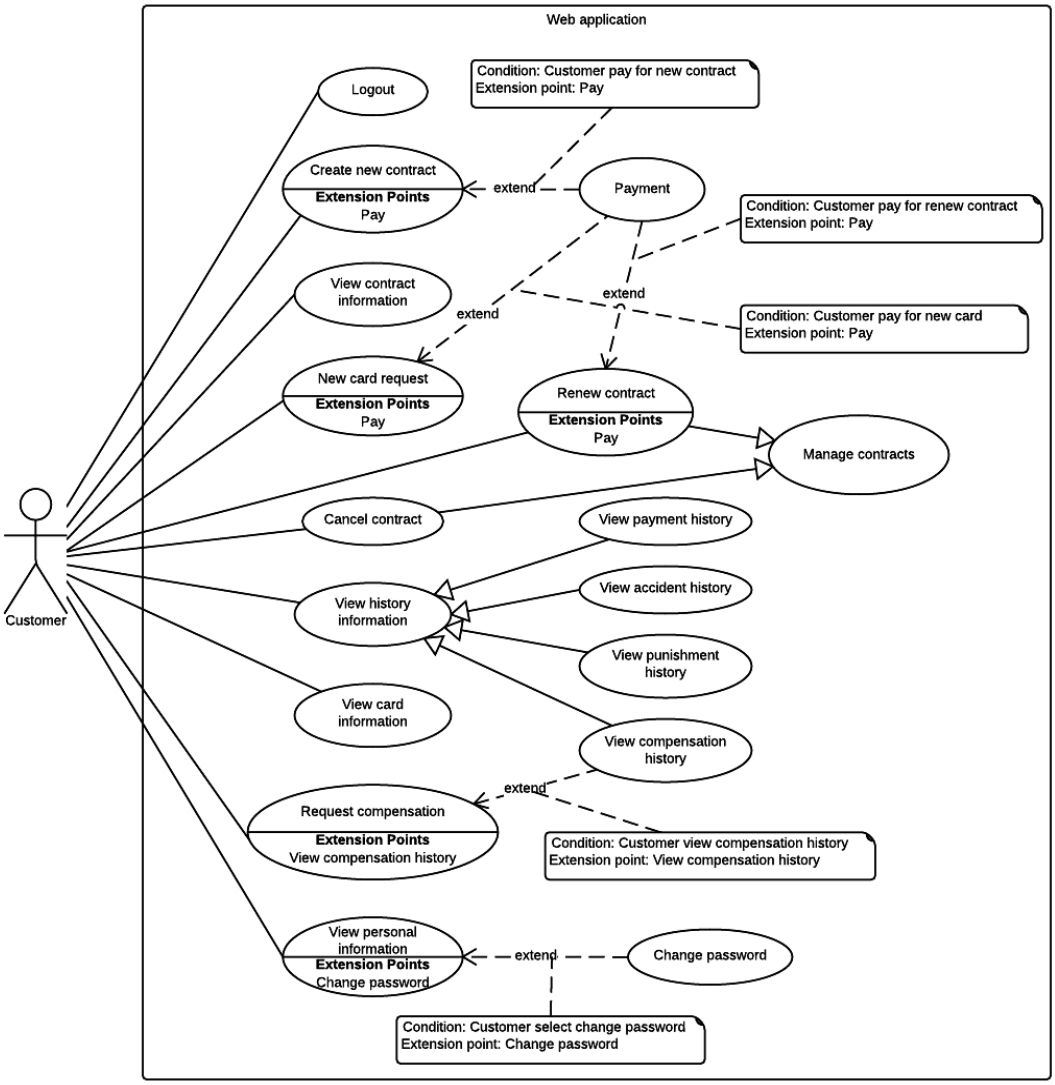


Figure 7 <Customer> Overview Use Case

2.3.1.2.1. <Customer> Renew contract

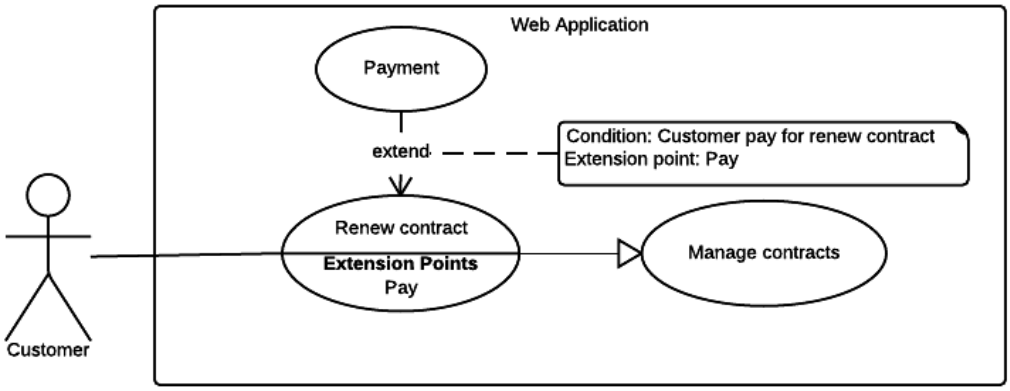


Figure 8 <Customer> Renew contract

USE CASE – WC02			
Use Case No.	WC02	**Use Case Version**	2.0
Use Case Name	Renew contract by user		
Author	TriPQM		
Date	27/05/2015	**Priority**	High

Actor:
- Customer.

Summary:
- This use case helps user to renew their contract.

Goal:
- User can renew their insurance contract.

Triggers:
- User sends renew contract command.

Preconditions:
- User must login into the system with role Customer.
- Contract's status must be "No Card", "Ready" or "Expired".
- The contract type belongs to this contract is NOT deactivated yet.
- Contract remaining days must NOT exceed the limit in administrator's configuration.

Post Conditions:
- **Success**: Customer's contract is renewed or a renew contract request would be sent to the Staff.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User goes to renew contract view.	Display renew contract information: - Type of contract: text - Start date: text - New expired date: text - Renewal fee: text.
2	User sends renew contract command.	Display new view with following information: - Payment content: text

		- Renewal fee: text And let user select one of following payment gateways: - PayPal payment gateway. - Direct payment.
3	If user chooses PayPal gateway and sends confirm command. [Alternative 1]	Forward to PayPal payment view to process the payment.
4	User process the PayPal payment	If payment succeed: Update information to the system. Renew user insurance contract. Show message renew successful. [Exception 1]

Alternative Scenario:

No	Actor Action	System Response
1	If user chooses direct payment method and sends the confirm command.	Show list of company brands address.

Exceptions:

No	Actor Action	System Response
1	If payment failed	Show message to notify user that payment failed and the renew request has been aborted.

Relationships: Extend "Payment".

Business Rules:

- If contract status is "Expired" and user paid the renew fee through PayPal, system automatically change the contract status to "Ready".
- If user paid the renew fee directly, contract's status is "Pending" and Staff will update the payment for that contract and change contract status to "Ready".
- If contract status as "No Card", it will remain in despite of renew process is completed successfully.
- Contract type cannot be changed when renew contract.
- If contract type of contract is not active the customer will not be able renew contract.
- A notification will be sent to staff after the process is completed.

Table 11 Use case WC02 - <Customer> Renew contract by user

2.3.1.2.2. <Customer> Cancel contract

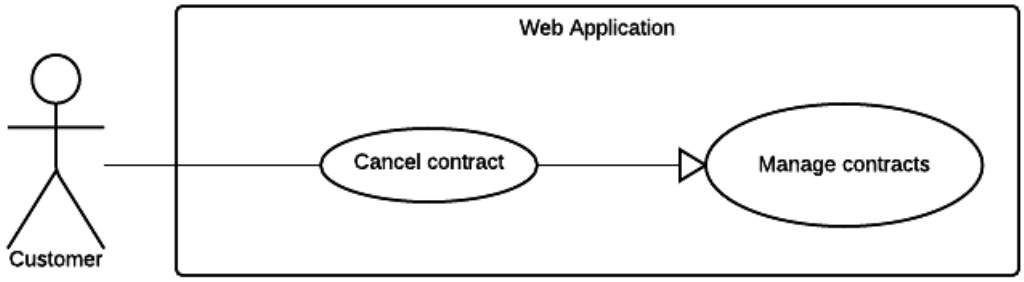


Figure 9 <Customer> Cancel contract

USE CASE – WC03			
Use Case No.	WC03	**Use Case Version**	2.0
Use Case Name	Cancel contract		
Author	TriPQM		
Date	27/05/2015	**Priority**	High

Actor:
- Customer.

Summary:
- This use case helps user cancel their contract.

Goal:
- Customer can cancel the contract.

Triggers:
- Customer sends cancel contract request.

Preconditions:
- User must login into the system with role Customer.
- User's contract has not expired.
- Customer's contract status must not be "Expired", "Cancelled" or "Request cancel".

Post Conditions:
- **Success**: Send to the staff the cancel contract request.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User goes to cancel contract view.	Display new view require user input some information: - Reason to cancel the contract: can be optional selected from these values: o "Xe cơ giới bị thu hồi đăng ký và biển số theo quy định của pháp luật" o "Xe cơ giới hết niên hạn sử dụng theo quy định của pháp luật" o "Xe cơ giới bị mất được cơ quan công an xác nhận" o "Xe cơ giới hỏng không sử dụng được hoặc bị phá huỷ do tai nạn giao thông được cơ quan công an xác nhận" o Other reason: free text input, required, length 1-250.
2	User inputs information	
3	User sends cancel contract request command.	- Change contract status. - Send request to the Staff. [Exception 1]

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	If user didn't check any reason to cancel contract	Show message to notify user that they have to choose the reason for cancel contract.

Relationships: N/A

Business Rules:
- Cancel contract request will be sent to the system with inputted information. - System update status of the contract from "Pending", "No Card" or "Ready" to "Request cancel". - A notification will be sent to staff after the process is completed.

Table 12 Use case WC03 - <Customer> Cancel contract

2.3.1.2.3. <Customer> Request compensation

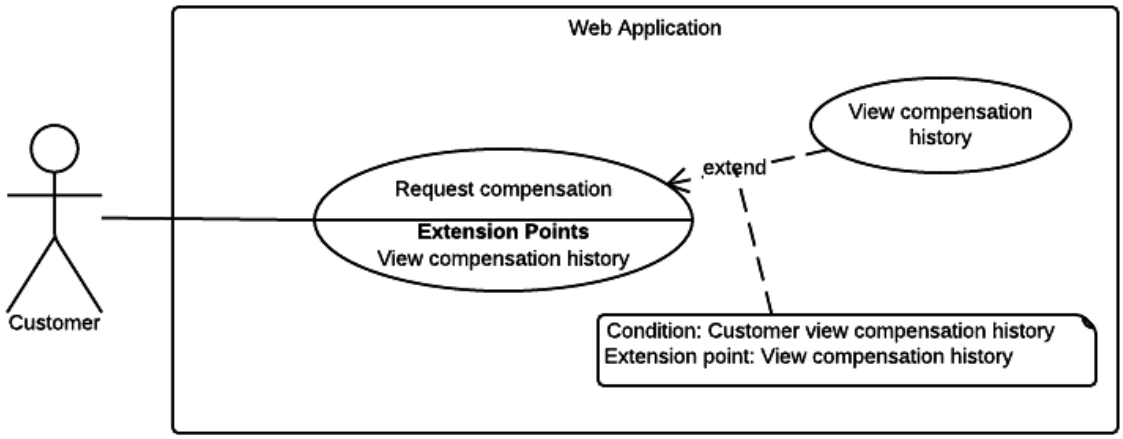


Figure 10 <Customer> Request compensation

USE CASE – WC07			
Use Case No.	WC07	**Use Case Version**	2.0
Use Case Name	Request compensation		
Author	TriPQM		
Date	27/05/2015	**Priority**	High

Actor:
- Customer.

Summary:
- This use case helps user to request compensation.

Goal:
- User can request compensation.

Triggers:
- User sends request compensation command.

Preconditions:
- User must login into the system with role Customer.
- Contract's status must not be "Pending".
- Contract's status must not be "Cancelled" or "Expired" has expired date is more than 30 days with current date.

Post Conditions:
- **Success**: Store the compensation request to into the system.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	User goes to request compensation view.	Display new view ask user to input required information includes:

		- Driver name: free text input, required, length 3-80. - License number: free text input, required, length 10-15. - License type: free text input, required, length 1-10. - Driver phone: free text input, required, length 8-15. - Vehicle capacity: free text input, required, length 1-20. - Driver address: free text input, required, length 3-250. - Plate number of accident motor: free text input, required, length 4-15. - Date of accident: date time input, required. - Place of accident: free text input, required, length 3-250. - Control Police Department: free text input, required, length 3-250. - Description: free text input, required, length 1-2000. - Human damage: free text input, required, length 1-2000. - Asset damage: free text input, required, length 1-2000. - Observer: free text input, required, length 3-80. - Compensation note: free text input, required, length 1-2000. - Attachment file: file upload input.
2	User fill required information and attach the minutes of the accident (if any).	
3	User sends request compensation command	Show message to notify that request punishment succeed. [Exception 1, 2]

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User input missed one of requirement information.	Show message to notify user what required information is missing.
2	The inputted information length is out of range.	Show message to notify user what information is out of range.

Relationships: N/A

Business Rules:

- Compensation request will be sent to the system with inputted information.
- Only permit user chooses the accident date before or at the request date.

- New compensation request status will be set to “Đang xử lý”. - Compensation decision will be set to “Chưa quyết định”. - A notification will be sent to staff after the process is completed.

Table 13 Use case WC07 - <Customer> Request compensation

2.3.1.2.4. <Customer> New card request

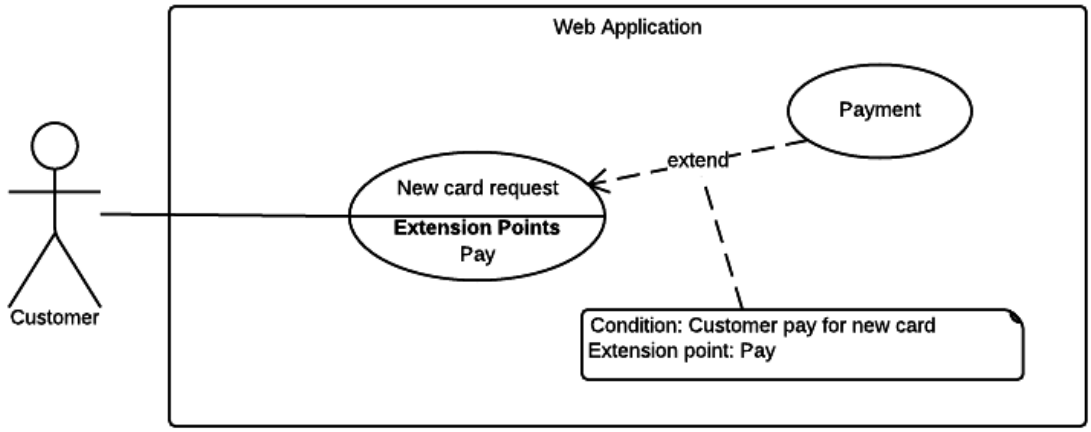


Figure 11 <Customer> New card request

USE CASE – WC09			
Use Case No.	WC09	**Use Case Version**	2.0
Use Case Name	New card request		
Author	TriPQM		
Date	27/05/2015	**Priority**	High

Actor:
- Customer.

Summary:
- This use case helps user to request a new card.

Goal:
- User can request a new card.

Triggers:
- User sends new card request command.

Preconditions:
- User must login into the system with role Customer.
- User must has a card in the system.
- Contract status must be “Ready”

Post Conditions:
- **Success**: New card request will be sent to the system.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response

1	User sends new card request command.	Display a new view shows to user: - A text box to confirm by password: free text input, required, length 6-32. - A text box to enter note: : free text input, length 0-2000. - Payment gateways: can be optional selected from these selections: o PayPal payment gateway. o Direct payment. - Check boxes for customer to choose deactivate the old card immediately and delivery card request. - The new card fee: text. - The delivery fee: text. - The total fee: text.
2	User enter password and choose the PayPal payment gateway. Then sends confirm command. [Alternative 1]	Forward to PayPal payment process view. [Alternative 2]
3	User process the PayPal payment.	If payment succeed, Show message to notify that the new card request and payment is succeed. [Exception 1]

Alternative Scenario:

No	Actor Action	System Response
1	User choose the direct payment method.	Show message to notify that the new card is sent. [Alternative 2]
2	If user enter wrong password	Show message to notify that user has entered wrong password.

Exceptions:

No	Actor Action	System Response
1	If payment failed	Show message to notify that the payment failed. The request is aborted.

Relationships: <Payment system> Payment (WP01)

Business Rules:

- The new card fee and delivery fee is specified by the system.
- At a time, user can only has one new card request for each contract.
- A notification will be sent to staff after the process is completed.

Table 14 Use case WC09 - <Customer> New card request

2.3.1.3.<Staff> Overview Use Case

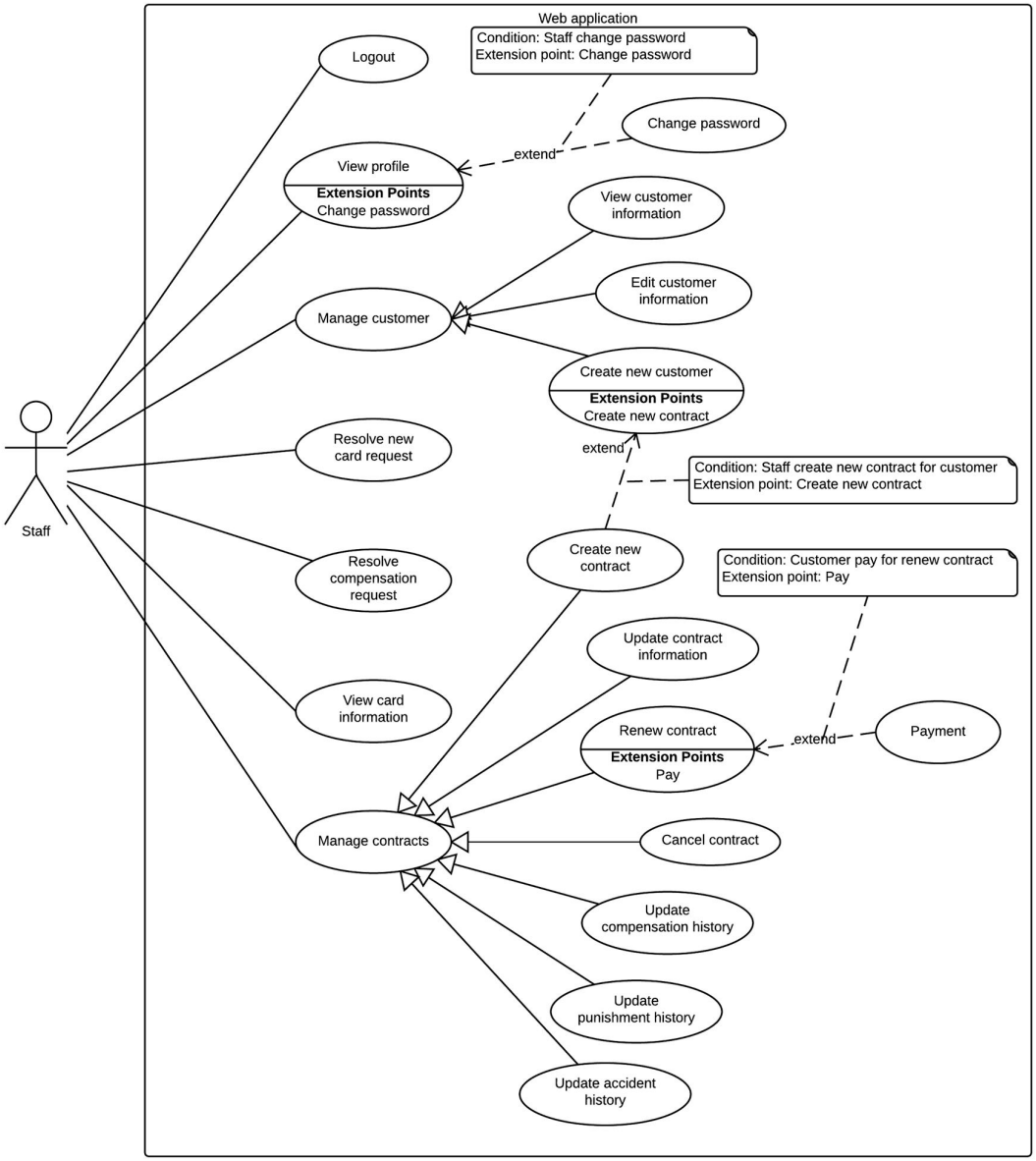


Figure 12 <Staff> Overview Use Case

2.3.1.3.1. <Staff> Resolve new card request

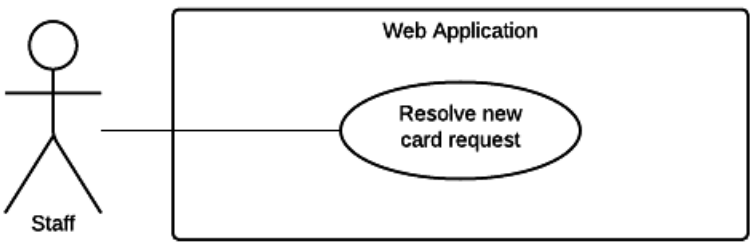


Figure 13 <Staff> Resolve new card request

USE CASE – WS03

Use Case No.	WS03	**Use Case Version**	2.1
Use Case Name	Resolve new card request		
Author	KhaNC		
Date	27/05/2015	**Priority**	Medium

Actor:
- Staff.

Summary:
- This use case allows staff view and solve new card request.

Goal:
- Staff can view and solve request for new card from customers.

Triggers:
- Staff sends view and solve new card request command.

Preconditions:
- User must login into the system with role Staff.
- There is at least 1 unsolved request for new card from customer.

Post Conditions:
- **Success**: A customer's new card request is solved, a new card issued and the date request solved updated.
- **Fail**: Request is unsolved.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to view all new card request view.	New card request view will be shown with following information: - Date of request: text - Note of customer: text - ID of old card: link to card details. - Name of card owner: link to detailed information of the customer who own this card. - Date a new card be issued for this customer: text - ID of new card: link to card details.
2	Staff add new card for this customer	Update new card information to customer's request and sends notification. - Date new card issued. - ID of new card

Alternative Scenario:

No	Actor Action	System Response
1	There are no unsolved new card request	Show message to notify staff there are no unsolved new card request.

Exceptions: N/A

Relationships: N/A

Business Rules:
- List of new card request is always loaded from the system.
- List of new card request is sorted by resolved date in ascending order.
- The old card will change status from "Ready" to "Deactivated".
- The new card will has status is "Ready".
- Search bar on the top help user finding card faster by its id.

- Pagination must be display if number of requests larger than 10. - A solved request must have code of new card and the date this card issued.

Table 15 Use case WS03 - <Staff> Resolve new card request

2.3.1.3.2. <Staff> Resolve compensation request

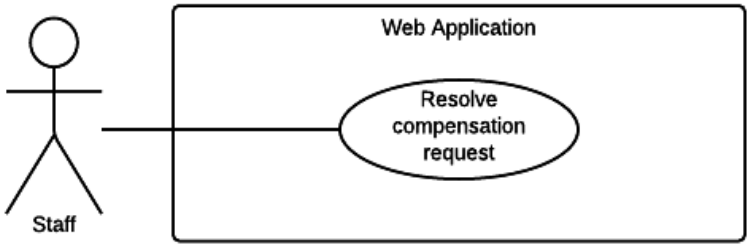


Figure 14 <Staff> Resolve compensation request

USE CASE – WS04			
Use Case No.	WS04	**Use Case Version**	2.0
Use Case Name	Resolve compensation request		
Author	KhaNC		
Date	27/05/2015	**Priority**	High

Actor:
- Staff.

Summary:
- This use case allows staff view and solves compensation requests.

Goal:
- Staff can view and solve request for compensation from customers.

Triggers:
- Staff sends view and solve compensation request command.

Preconditions:
- User must login into the system with role Staff.
- There is at least 1 unsolved request for compensation from customer.
- The compensation's decision must be negotiated with the customer, approved by the insurance company.

Post Conditions:
- **Success**: A customer's compensation request solved, decision for this compensation has been made and the date request solved is updated.
- **Fail**: Request is unsolved.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to view list of request for compensation.	List of compensation request will be shown with following information: - Compensation code: link to request detail - Request created date: text - Contract's code: link to contract detail - Contract's owner full name: link to customer detail

		- Request status: text
2	Staff select an unsolved request to view.	Compensation request detail will be shown with following information: - Contract's code: link to contract details. - Contract's owner full name: link to customer details. - Compensation created date: text - Compensation status: text - Driver's full name: text - Driver's phone number: text - Driver's address: text - Driver's license number: text - Driver's license type: text - Accident vehicle plate number: text - Vehicle's capacity or seat capacity: text - Time of the accident: text - Place of the accident: text - Police department solved accident: text - Description of the accident: text - Human damaged in the accident: text - Asset damaged in the accident: text - Name of the observer: text - Address of the observer: text - Detail of compensation request from customer: text - Compensation attachment: link to attachment detail
3	Staff send resolve this compensation request command.	Resolve compensation request view is shown with following labels and fields - Resolve date: date time input, required - Note for compensation: free text input, length 1 – 2000
4	Staff input resolve compensation request information and select one from two decision	
5	Staff sends save changes command	Update compensation status and notify to customer

Alternative Scenario:

No	Actor Action	System Response
1	There are no compensation request	Show message to notify staff there are no compensation request.

Exceptions: N/A
Relationships: N/A

Business Rules: - List of compensation request is always loaded from the system. - List of compensation request is sorted by compensation request resolved date and created date in ascending order. - With unsolved request, its solved date does not exist. - Compensation resolved date must be restricted in the limit by configuration of the administrator. - A compensation request could have one of following decision: o "Chấp nhận bồi thường" o "Từ chối bồi thường" - A compensation request could have one of following status: o "Đã giải quyết" o "Chưa giải quyết" - Compensation information input in this screen is the final information after company and customer has done physical business about dealing the compensation fee. - Search bar on the top help user finding contract or customer faster. - Pagination must be display if number of requests larger than 10. - A notification will be sent to request customer after the process is completed.

Table 16 Use case WS04 - <Staff> Resolve compensation request

2.3.1.3.3. <Staff> Create new customer

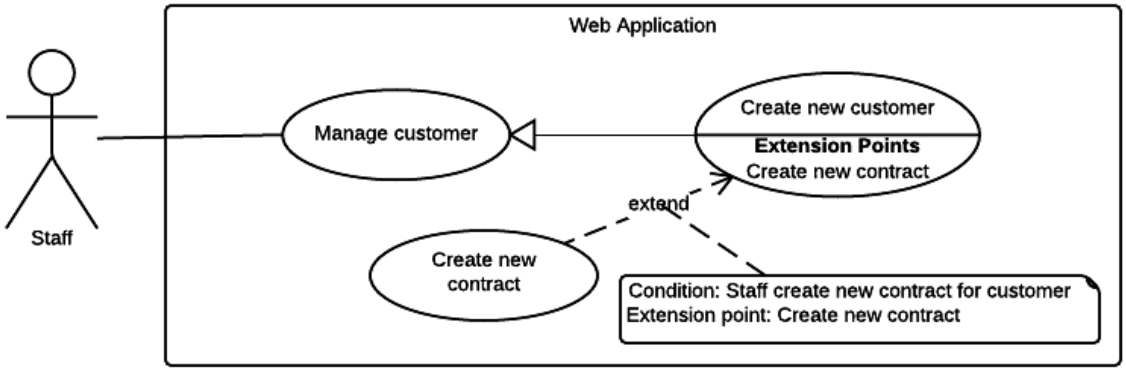


Figure 15 <Staff> Create new customer

USE CASE – WS07			
Use Case No.	WS07	**Use Case Version**	2.0
Use Case Name	Create new customer		
Author	KhaNC		
Date	20/05/2015	**Priority**	Medium
Actor: - Staff. **Summary:** - This use case allows staff create new customer. **Goal:** - A new customer is added to the system. **Triggers:** - Staff sends create new customer command.			

Preconditions:
- User must login into the system with role Staff.
- This customer is not existed in the system yet.

Post Conditions:
- **Success**: New customer is added to the system.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to create customer view.	Create customer view is shown with following labels and fields: - Customer's full name: free text input, required, length 3 – 80 - Customer's address: free text input, required, length 3 – 250 - Customer's email address: free text input, required, length 3 – 250 - Customer's phone number: free text input, required, length 8 – 15 - Customer's personal ID: free text input, length 8 – 15
2	Staff fills out the form.	
3	Staff sends create new customer command.	- Validate data [Exception 1, 2, 3, 4] - Add new customer to the system. - Display create customer success message.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	Missing of required fields	Show message notify staff input missed fields
2	Length of field's value is out of range	Show message notify staff which field's value is out of range
3	Entered email address is not a valid email	Show message notify entered email is not valid
4	Entered email is existed in the system	Show message notify entered email is existed

Relationships: <Staff> Create new contract (WS08)

Business Rules:
- In case of success scenarios, a new customer would be added to the system.
- An email address must be validated by this regular expression: /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/
- System will suggest staff create new contract for the customer has just created.

Table 17 Use case WS07 - <Staff> Create new customer

2.3.1.3.4. <Staff> Create contract

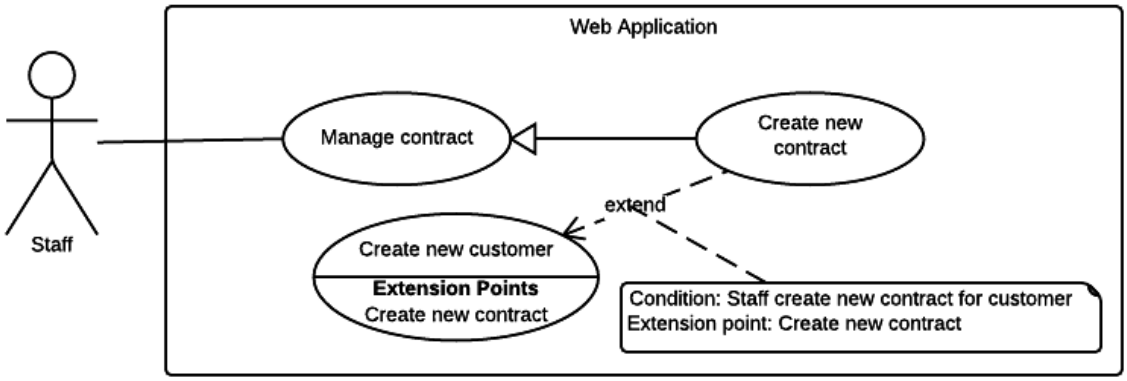


Figure 16 <Staff> Create contract

USE CASE – WS08			
Use Case No.	WS08	**Use Case Version**	2.0
Use Case Name	Create new contract		
Author	KhaNC		
Date	20/05/2015	**Priority**	Medium

Actor:
- Staff.

Summary:
- This use case allows staff create new contract.

Goal:
- A new contract is added to the system.

Triggers:
- Staff sends create new contract command.

Preconditions:
- User must login into the system with role Staff.
- There is at least 1 customer in the system.

Post Conditions:
- **Success**: New contract is added to the system
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to create new contract view.	Contract detail is shown with following label and fields: - Customer code: free text input, required, length 6 – 10 Contract's information - Contract' type: select one of the options. - Start date: date time input, required - Expired date: date time input, required - Contract's fee: text Vehicle's information:

		- Vehicle's plate number: free text input, required, length 4 – 15 - Vehicle's brand: free text input, required, length 2 – 20 - Vehicle's model code: free text input, length 2 – 20 - Vehicle's type: free text input, length 2 – 20 - Vehicle's color: free text input, length 2 – 20 - Vehicle's engine: free text input, required, length 2 – 20 - Vehicle's chassis: free text input, required, length 2 – 20 - Vehicle's capacity: free text input, required, length 2 – 20 - Vehicle's year of manufacture: free text input, length: 4 - Vehicle's empty weight: free text input, length 1 – 4 - Vehicle's seat capacity: free text input, length 1 – 3 Payment information: - Paid date: date time input, required
2	Staff fills out the form.	
3	Staff sends create new contract command.	Display contract's information to review and request for confirmation
4	Staff confirms create new contract command.	- Validate data [Exception 1, 2, 3] - Add new contract's information to the system. - Notify to staff created contract successfully.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	Missing of required fields	Show message notify staff input missed fields
2	Length of field's value is out of range	Show message notify staff which field's value is out of range
3	Entered vehicle is existed in the system	Show message notify staff this vehicle is existed.

Relationships: <Staff> Create new customer (WS07)

Business Rules:

- In case of success scenarios, a new contract would be added to the system.
- Staff can search and select a contract owner from available customers.
- List of contract's type must be loaded from the system.
- A new contract created successfully will has the initial status is "No card".

- Contract's start date, expired date, paid date must be restricted in the limit by configuration of the administrator.
- Default contract's term is set up by configuration of the administrator.
- Contract's term must not exceed contract's default term.
- Contract's expired date must not be earlier than contract's start date.
- Contract's fee is calculated by the following formula, with price per year is loaded from the system belongs to selected contract type:

$$\boldsymbol{contract\ fee} = \frac{\boldsymbol{price\ per\ year}}{\mathbf{12}} \times \boldsymbol{contract\ term}$$

- Contract's term will be rounded up to nearest month, for example: 2 months 18 days is 3 months

Table 18 Use case WS08 - <Staff> Create new contract

2.3.1.3.5. <Staff> Renew contract

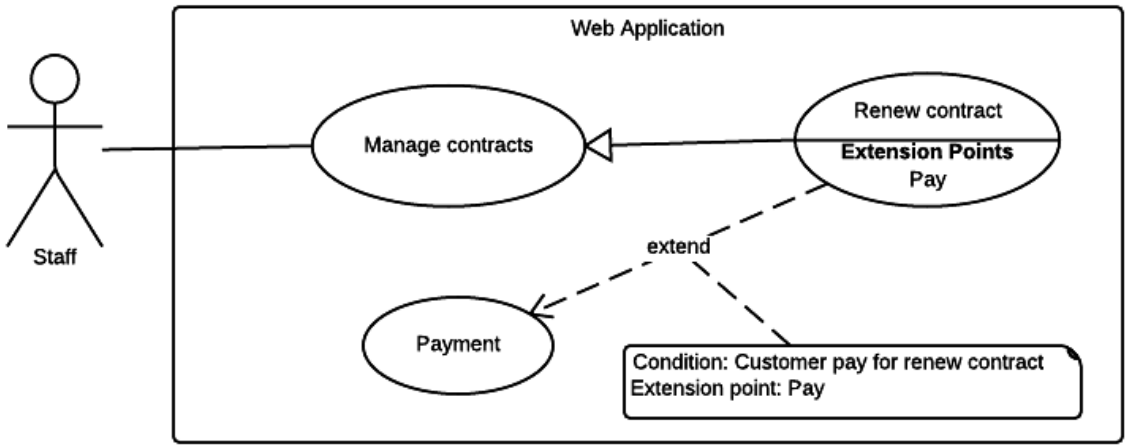


Figure 17 <Staff> Renew contract

USE CASE – WS09			
Use Case No.	WS09	**Use Case Version**	2.0
Use Case Name	Renew contract		
Author	KhaNC		
Date	27/05/2015	**Priority**	Medium

Actor:
- Staff.

Summary:
- This use case allows staff renew contract.

Goal:
- Contract's new expired date will be updated to the system.

Triggers:
- Staff sends renew contract command.

Preconditions:
- User must login into the system with role Staff.
- The contract type belongs to this contract is NOT deactivated yet.
- Contract remaining days must NOT exceed the limit in administrator's configuration.
- Contract's status must be "No card", "Ready" or "Expired".

Post Conditions:

- **Success**: Contract is renewed
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to renew contract view.	Renew contract detail is shown with following renew contract information: - Contract's code: text - Contract's status: text - Contract's type: text - Contract's status: text - Contract's start date: text - Contract's expired date: text - Contract's new expired date: date time input, required - Renew contract fee: text - Request for new card: free text input - New card fee: text - Delivery new card: free text input - Delivery new card fee: text - Paid date: date time input, required
2	Staff fills out the form.	
3	Staff sends renew this contract command	Display confirm renew contract view.
4	Staff confirms renew this contract command.	- Validate data [Exception 1, 2] - Update contract's information to the system. - Reload contract detail.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	Missing of required fields	Show message notify staff input missed fields
2	Length of field's value is out of range	Show message notify staff which field's value is out of range

Relationships: <Payment system> Payment (WP01)

Business Rules:

- In case of success scenarios, a new expired date of contract would be updated to the system.
- A contract renewed successfully will has status is "Ready":
  - If contract's status is "No card" or "Ready", there are no change.
  - In case of the contract has status is "Expired", its will be changed to "Ready".
- Contract's new expired date, paid date must be restricted in the limit by configuration of the administrator.
- Default contract's term is set up by configuration of the administrator.
- Contract's term must not exceed contract's default term.
- Contract's expired date must not be earlier than contract's start date.

- Contract's fee is calculated by the following formula, with price per year is loaded from the system belongs to current contract type: $$\boldsymbol{contract\ fee} = \frac{\boldsymbol{price\ per\ year}}{\boldsymbol{12}} \times \boldsymbol{contract\ term}$$ - Contract's term will be rounded up to nearest month, for example: 2 months 18 days is 3 months.

Table 19 Use case WS09 - <Staff> Renew contract

2.3.1.3.6. <Staff> Cancel contract

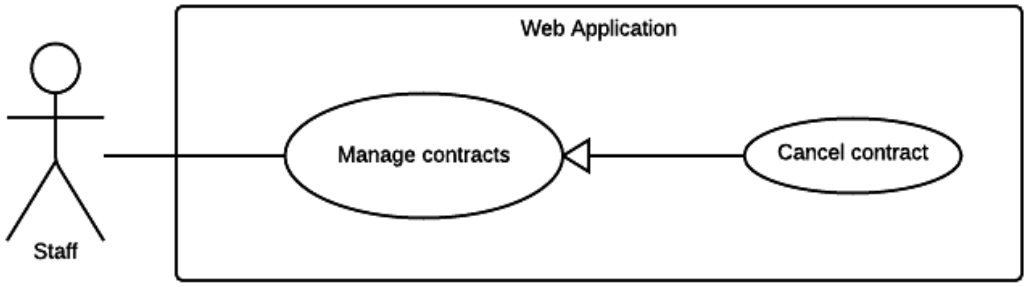


Figure 18 <Staff> Cancel contract

USE CASE – WS10			
Use Case No.	WS10	**Use Case Version**	2.0
Use Case Name	Cancel contract		
Author	KhaNC		
Date	27/05/2015	**Priority**	Medium

Actor:
- Staff.

Summary:
- This use case allows staff cancel a contract.

Goal:
- Contract's new status is updated to the system.

Triggers:
- Staff sends cancel contract command.

Preconditions:
- User must login into the system with role Staff.
- Contract's status must be "Pending", "No card" or "Ready".

Post Conditions:
- **Success**: Contract is cancelled.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff goes to cancel contract view.	Cancel contract detail is shown with following information: - Cancel date: date time input, required - Cancel reason: free text input, required, length 1 – 250 - Description: free text input, length 1 – 2000

2	Staff fills out the form.	
3	Staff sends cancel contract command.	Display confirm cancel contract view.
4	Staff confirms cancel contract command.	- Validate data [Exception 1, 2] - Update contract's information to the system. - Reload contract detail.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	Missing of required fields	Show message notify staff input missed fields
2	Length of field's value is out of range	Show message notify staff which field's value is out of range

Relationships: N/A

Business Rules:

- When contract cancelled successfully, there are no notification would be sent to the customer who own this contract.
- A contract cancelled successfully will change status from "No card", "Ready" or "Pending" to "Cancelled".
- Contract's cancel date must be restricted in the limit by configuration of the administrator.

Table 20 Use case WS10 - <Staff> Cancel contract

Reference to full document for complete list of use case specs.

2.3.1.4.<Admin> Overview Use Case

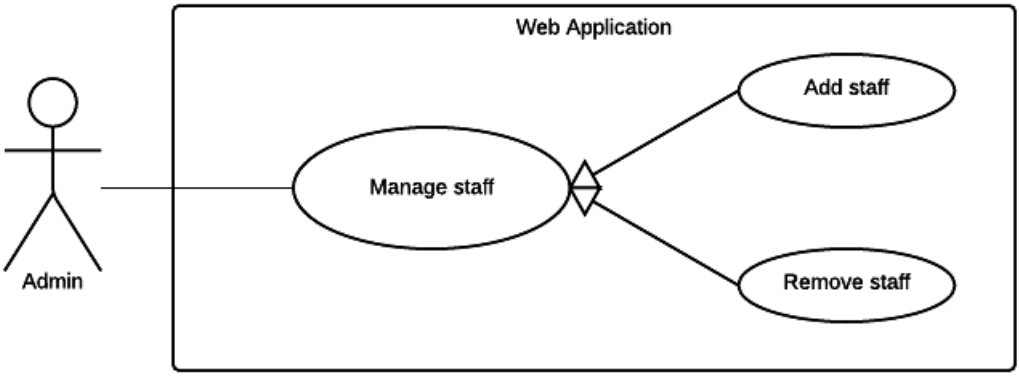


Figure 19 <Admin> Overview Use Case

2.3.1.4.1. <Admin> Update contract type information

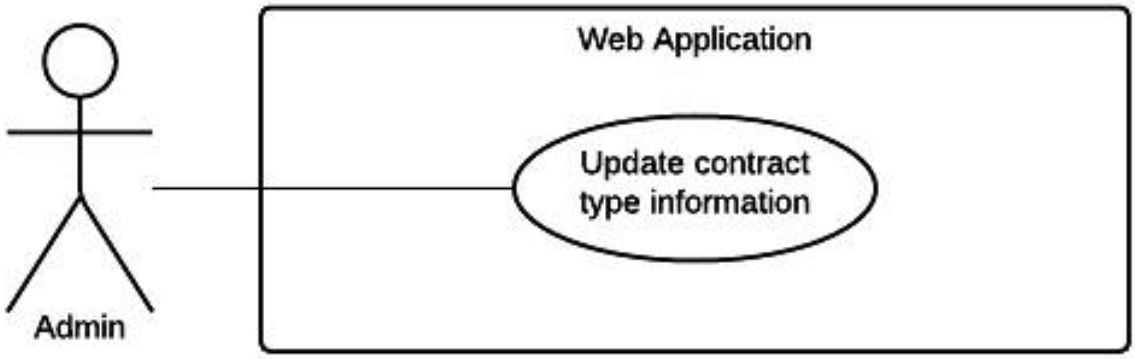


Figure 20 <Staff> Update contract type information

USE CASE – WS16			
Use Case No.	WS16	**Use Case Version**	2.0
Use Case Name	Update contract type information		
Author	TrungDQ		
Date	27/05/2015	**Priority**	High

Actor:
- Admin

Summary:
- This use case allows admin to update contract type.

Goal:
- Admin can update contract type.

Triggers:
- Admin sends command to update contract type.

Preconditions:
- User has to logged in to the system as Admin role
- Contract type is existed in the system

Post Conditions:
- **Success:** Contract type information will be updated
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Admin goes to view contract type information.	System list out information of contract type: - Name: free text input, required, length 1 – 250. - Description: free text input, required, length 1 – 2000. - Price per year: free number input, required, value from 0 to 1 billion, unit: VNĐ.
2	Admin inputs information	
3	Admin sends command to save new information.	System shows message notify contract type information is updated successfully. [Exception 1,2]

Alternative Scenario: N/A

Exceptions:

Step	Actor Action	System Response
1	Admin sends command to save new information.	System shows message notify missing required fields.

Relationships: N/A

Business Rules:

- Contract type information will be updated to system.
- Contract type price per year must be greater than 0 and less than 1 billion.
- When a contract type is deactivated, all the contract which having the contract type is still remain working but will not be able to renew.
- When a contract type is deactivated, it will not be shown at the create contract page / contract type options.

Table 21 Use case WS16 - <Staff> Update contract type information

2.3.1.4.2. < Admin> Remove staff

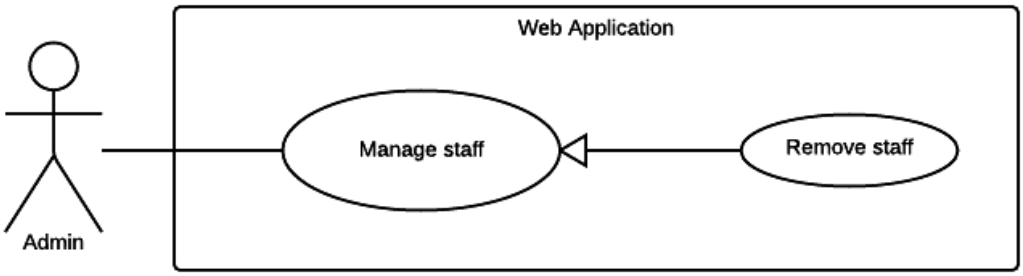


Figure 21 <Admin> Remove staff

USE CASE – WA01			
Use Case No.	WA01	**Use Case Version**	2.0
Use Case Name	Remove staff		
Author	TrungDQ		
Date	27/05/2015	**Priority**	High

Actor:

- Admin

Summary:

- This use case allows admin to remove other staff from the system.

Goal:

- Admin can remove staff from the system.

Triggers:

- Admin sends command to remove staff from the system.

Preconditions:

- User has to logged in to the system as Admin role

Post Conditions:

- **Success:** Admin be able to remove staff from the system
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response

1	Admin goes to manage members	System list out information of staffs in system: - Staff code: text - Name: text - Email: text - Phone number: text
2	Admin select the staff to delete	
3	Admin sends command to delete staff.	System shows message staff has been removed successfully. [Exception 1]

Alternative Scenario: N/A

Exceptions:

Step	Actor Action	System Response
1	Admin sends command to delete staff.	System shows error message user cannot remove themselves from the system.

Relationships: N/A

Business Rules:

- Their must be at least 01 staff in the system.
- Staff account will be deactivated.
- All information about staff still remain in the system.

Table 22 Use case WA01 - <Admin> Remove staff

2.3.1.4.3. < Admin> Add staff

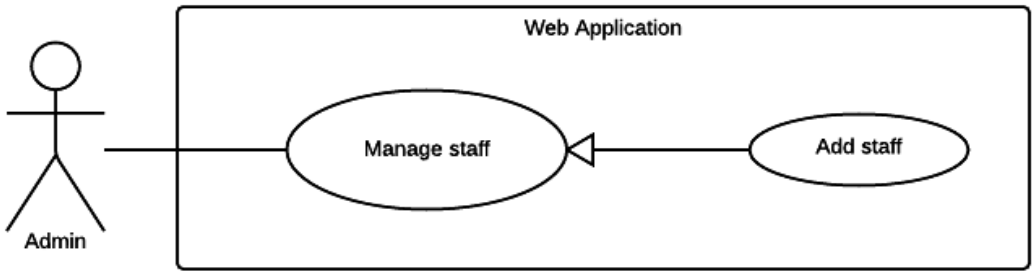


Figure 22 <Admin> Add staff

USE CASE – WA02			
Use Case No.	WA02	**Use Case Version**	2.0
Use Case Name	Add staff		
Author	TrungDQ		
Date	27/05/2015	**Priority**	High

Actor:

- Admin

Summary:

- This use case allows admin to add new staff to the system.

Goal:
- Admin can add new staff to the system.

Triggers:
- Admin sends command to add new staff to the system.

Preconditions:
- User has to logged in to the system as Admin role

Post Conditions:
- **Success:** New staff is added into the system
- **Fail:** Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Admin goes to add staff page.	System require information of staff: - Email: text, required, length 3 – 250. - Password: text, required, length 2 - 32. - Name: text, required, length 3 – 80. - Phone number: text, required, length 8 – 15.
2	Admin inputs information	
3	Admin sends command to add new staff.	System shows message staff has been added successfully. [Exception 1]

Alternative Scenario: N/A

Exceptions:

Step	Actor Action	System Response
1	Admin sends command to add other staff.	System shows error message to ask user to enter missing required field.

Relationships: N/A

Business Rules:
- Staff code is automatic initialized by the system.
- Staff email must not be duplicated.

Table 23 Use case WA02 - <Admin> Add staff

2.3.1.5.<Scheduler> Overview Use Case

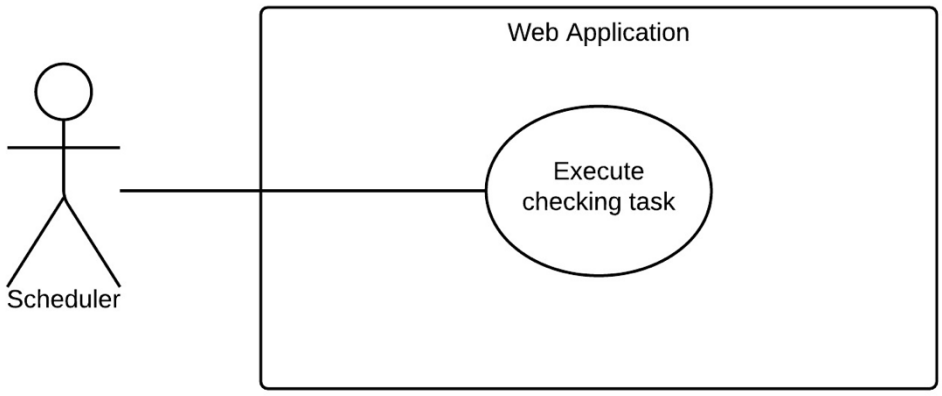


Figure 23 <System> Overview Use Case

2.3.1.5.1. <Scheduler> Execute checking task

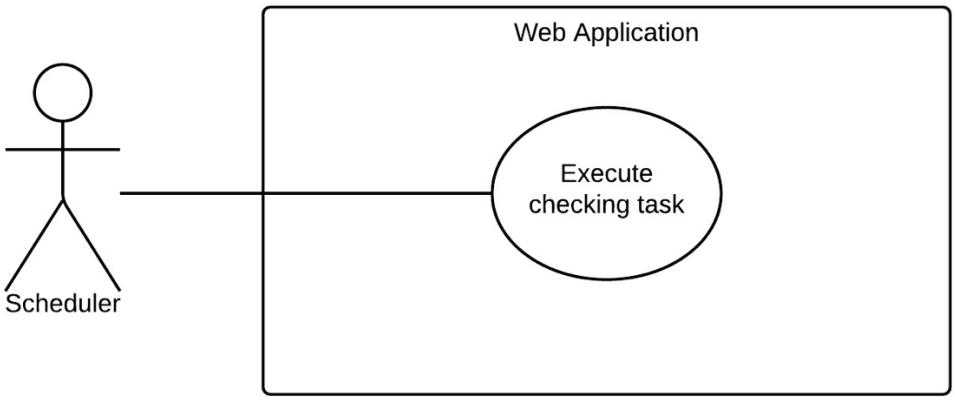


Figure 24 <System> Notify schedule

USE CASE – WY01			
Use Case No.	WY01	**Use Case Version**	2.0
Use Case Name	Execute checking task		
Author	TrungDQ		
Date	26/05/2015	**Priority**	Medium

Actor:
- Scheduler

Summary:
- This use case describes how the system run a schedule task to notify to users when necessary.

Goal:
- System will able to update status of contract, compensation and send notification to users.

Triggers:
- System run a timer task that trigger check event.

Preconditions:
- System time is at 00:00 AM.
- Their must be at least one contract in the system.

Post Conditions:
- **Success**: Show the status of contract and notify to user.
- **Fail**: System display error message.

Main Success Scenario:

Step	Actor Action	System Response
1	System run timer task to check contract state and sends notify	System response: - List of contract status that need to be change. - List of compensation status that need to be change. - Notifications that need to be sent to users. [Exception 1]

Exceptions:

No	Actor Action	System Response

1	System timer task is interrupted	No notification will be sent. Error detail will be tracked in a log file.

Relationships: N/A

Business rules:

- System timer will send check event at 00:00 AM everyday.
- If a contract has expired date remaining is less than a number of days specified by admin, system will send notification to users to ask users to renew their contract. This notification will repeat 2 more times in the specified days if the user do not renew the contract.
- If a contract exceeded expired date, the contract status will be set to "Expired".
- If a contract is not paid within a number of days specified by the system, the contract status will be set to "Cancelled".
- If a pending contract's start date arrived, the contract will be started. And the contract status will be set to "No Card" or "Ready".
- If there is new contract, cancel contract request, new card request or new compensation, notifications will be sent to staffs. And if staff resolved the requests, notifications will be sent to customers.

Table 24 Use case WY01 - <Scheduler> Execute checking task

2.3.1.6.<Payment system> Overview Use Case

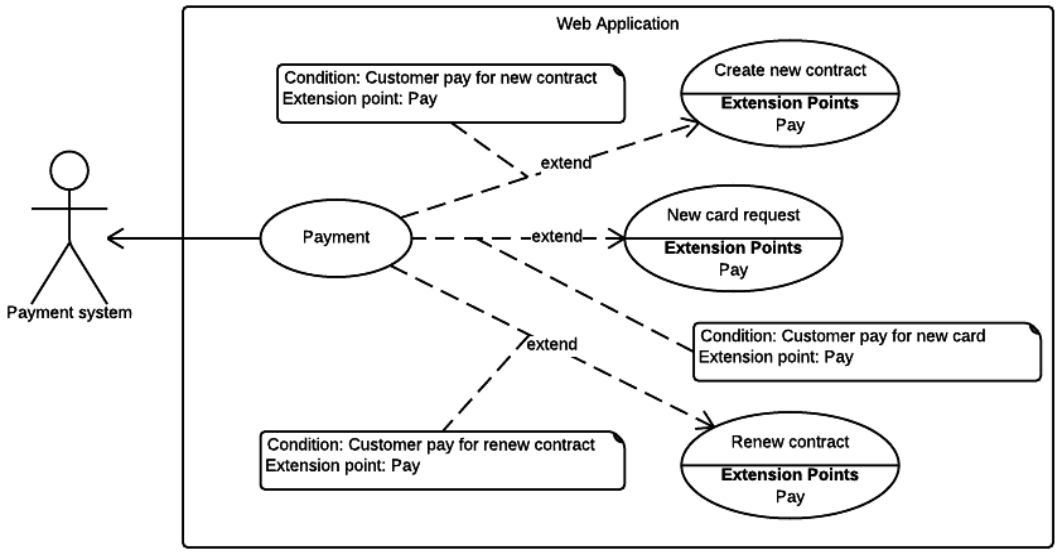


Figure 25 <Payment> Payment

2.3.1.6.1. <Payment system> Payment

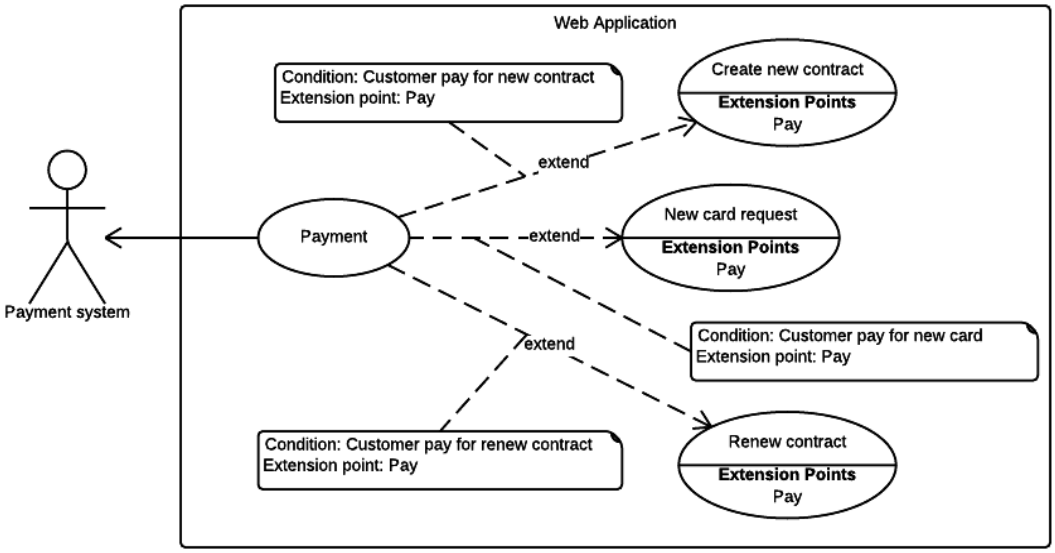


Figure 26 <Payment> Payment

USE CASE – WP01			
Use Case No.	WP01	**Use Case Version**	2.0
Use Case Name	Payment		
Author	KhaNC		
Date	03/06/2015	**Priority**	High

Actor:

- Payment system.

Summary:

- This use case allows payment system process the payment transaction.

Goal:

- Payment process is completed.

Triggers:

- Customer sends pay for charge by payment system command.

Preconditions:

- User must login into the system with role Customer.

Post Conditions:

- **Success**: Payment is completed successfully.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Customer select payment via PayPal.	Summary of customer payment: - Description: text. - Amount: text. - Total: text. Payment view is shown with following labels and fields: - Customer's PayPal account: free text input, required, length 3 – 250

		- Customer's PayPal password: free text input, required, length 6 – 32
2	Staff fills out the form.	
3	Staff sends pay command.	- Validate data [Alternative 1] [Exception 1, 2, 3] - Process payment in PayPal

Alternative Scenario:

No	Actor Action	System Response
1	Identity information is not match.	Show message notify customer input wrong email or password.

Exceptions:

No	Actor Action	System Response
1	Missing of required fields	Show message notify staff input missed fields
2	Length of field's value is out of range	Show message notify staff which field's value is out of range
3	Entered email address is not a valid email	Show message notify entered email is not valid

Relationships:

- <Customer> Create new contract
- <Customer> New card request
- <Customer> Renew contract

Business Rules:

- Use PayPal API to validate customer email and password.
- PayPal will send notification to alert transaction is failed or succeeded.
- An email address must be validated by this regular expression: /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/

Table 25 Use case WP01 - <Payment> Payment

2.3.2. Checker Mobile Application

2.3.2.1.<Police> Overview Use Case

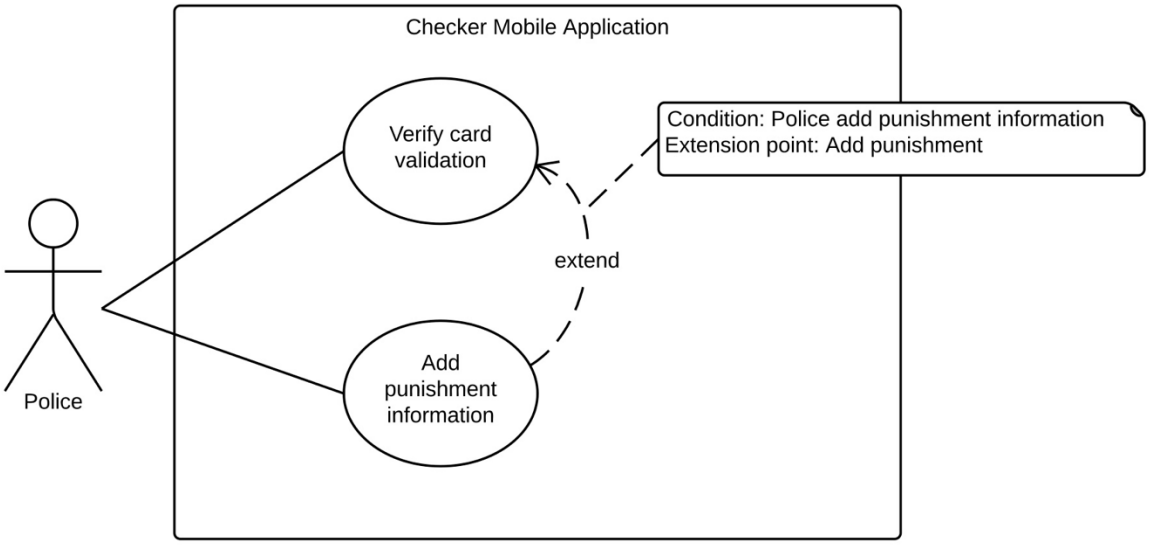


Figure 27 <Police> Overview Use Case

2.3.2.1.1. <Police> Verify card information

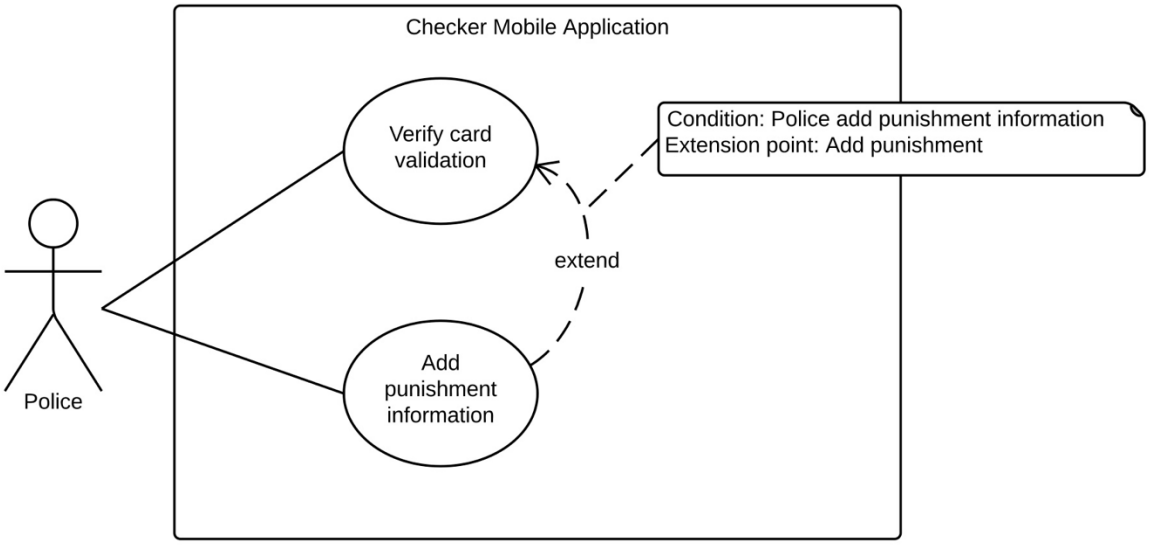


Figure 28 <Police> Verify card information

USE CASE – CP01			
Use Case No.	CP01	**Use Case Version**	2.0
Use Case Name	Verify card information		
Author	PhucNH		
Date	20/06/2015	**Priority**	High
Actor: - Police **Summary:**			

- Traffic Police and Police Department can use the device to view card's information and check if the card is valid or not.

Goal:

- Check if the insurance card is valid or not.

Triggers:

- Police put the NFC card near the device to read card information.

Preconditions:

- Device NFC is turned on
- Device is connected to the internet

Post Conditions:

- **Success**: Show the insurance card information.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Police put the NFC card close to the device.	Show the insurance contract and the motor information: - Motor owner: text. - Owner address: text. - Phone number: text. - Plate number: text. - Engine: text. - Chassis: text. - Brand: text. - Model Code: text. - Type: text. - Color: text. - Capacity: text. - Year of manufacture: text. - Weight: text. - Seat capacity: text. - Contract fee: text. - Start date: text - Expired date: text - Card Status: text. [Alternative 1] [Exception 1, 2]

Alternative Scenario:

No	Actor Action	System Response
1	If unable to read the card or the card is invalid	Mobile app shows error to notify police that mobile cannot read this card

Exceptions:

No	Actor Action	System Response
1	Cannot read card	Mobile app notifies that cannot read card
2	Cannot connect to server.	Mobile app shows error connect to server is fail

Relationships: <Police> Add punishment (CP02)

Business Rules:

- Mobile application sends request about card information to server and receives information about contract.
- A card is "Valid" if all of the following conditions are true:
  o The card ID is in activated status.
  o Contract has status "Ready" or "Request cancel"
- A card is "Nearly expired" if the card is "Valid" and the contract expired day remaining is less than a number of days specified by the system.
- Highlight the status of the card include:
  o Valid card
  o Expired card
  o Card is nearly expired, show remaining days
- Verify card activity and device ID will be logged to card access log in the system.

Table 26 Use case CP01 - <Police> Verify card information

2.3.2.1.2. <Police> Add punishment information

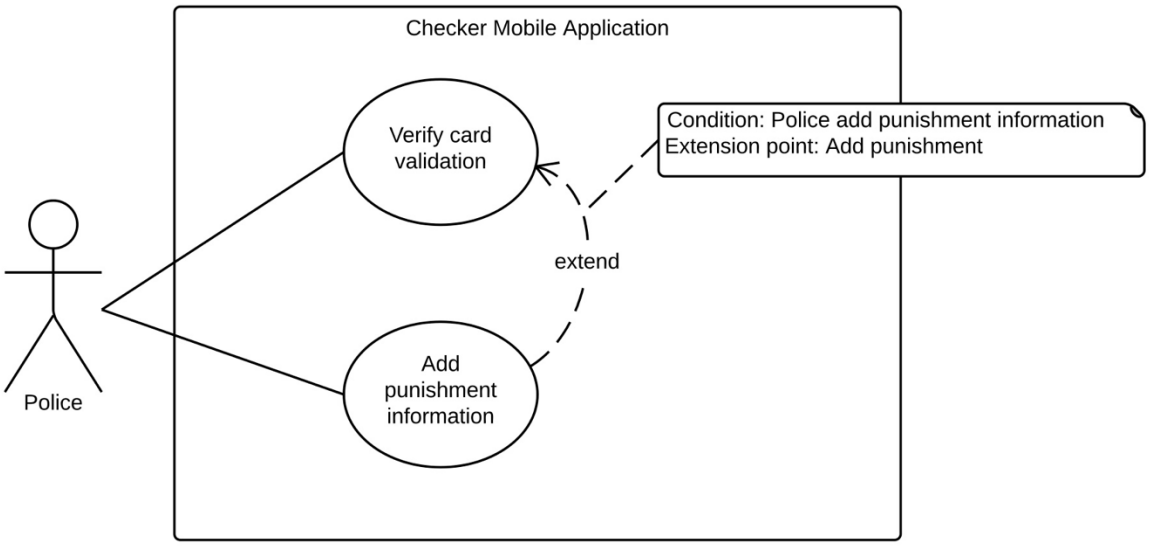


Figure 29 <Police> Add punishment information

USE CASE – CP02			
Use Case No.	CP02	**Use Case Version**	2.0
Use Case Name	Add punishment information		
Author	PhucNH		
Date	26/05/2015	**Priority**	High

Actor:
- Police

Summary:
- Traffic Police and Police Department can use the device to add punishment information of the driver.

Goal:
- Add punishment information of the driver.

Triggers:
- Device NFC is turned on

- Device is connected to the internet
- Reads card successful.

Preconditions:

- The card is verified.

Post Conditions:

- **Success**: Add punishment information of the driver.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Police chooses new punishment	Mobile required police input description about punishment and picture about it. - Description: free text input, length 1 - 200 Picture: file upload input, required
2	Police fill the punishment information and sends add punishment command	Mobile sends punishment from police and notify this action is success [Exception 1, 2]

Exceptions:

No	Actor Action	System Response
1	Police input missed one of punishment description and picture of punishment record.	Mobile reminds input required information.
2	Cannot connect to server.	Show error message to notify that can not connect to server.

Relationships: <Police> Verify card information (CP01)

Business Rules:

- Information about punishment will be sent in server
- Punishment information is updated to contract of user.
- Picture will be resized to 1000 width to minimize the data sent to server and better performance.
- Add punishment activity and device ID will be logged to card access log in the system.

Table 27 Use case CP02 - <Police> Add punishment information

2.3.3. Printer Mobile Application

2.3.3.1.<Staff> Overview Use Case

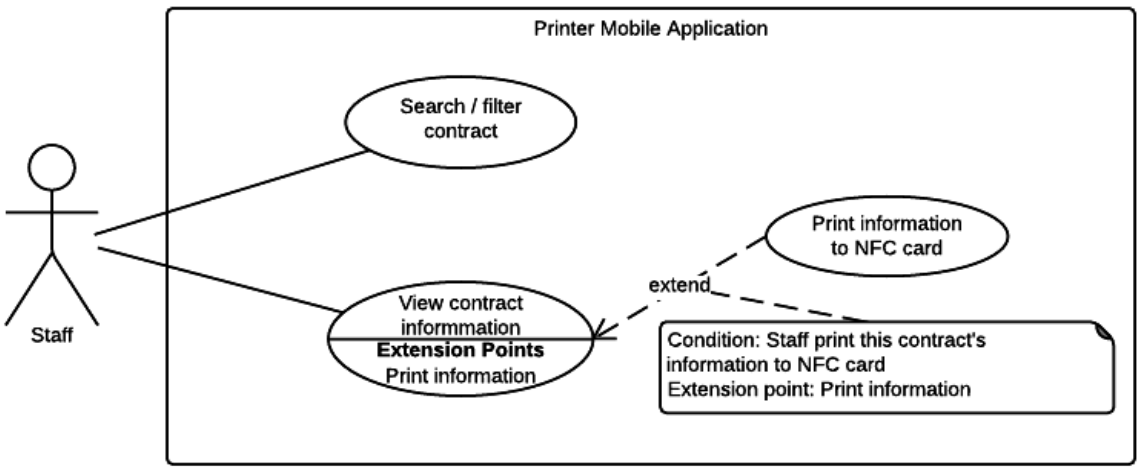


Figure 30 <Staff> Overview Use Case

2.3.3.1.1. <Staff> Search / filter contract

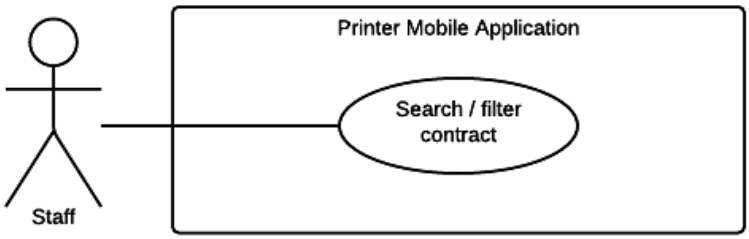


Figure 31 <Staff> Search / filter contract

USE CASE – PS01			
Use Case No.	PS01	**Use Case Version**	2.0
Use Case Name	Search / filter contract		
Author	PhucNH		
Date	26/05/2015	**Priority**	Medium

Actor:
- Staff

Summary:
- This use case helps staff can search or filter the information of contract by contract code or customer name.

Goal:
- The information of contract is shown following the request search of staff.

Triggers:
- Staff sends command to search/ filter contract.

Preconditions:
- Device is connected to the internet
- There must be at least 1 contract in the system

Post Conditions:
- **Success**: The information of request search is shown.
- **Fail**: System display error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff enter keyword	
2	Staff sends command to search/ filter contract Staff chooses contract	System shows list result: Information about request search. System shows information about contract: - Contract code: text - Customer name: text - Status: color [Alternative 1] [Exception 1]

Alternative Scenario:

No	Actor Action	System Response
1	System cannot finds the result from request	System notifies cannot find the request search.

Exceptions:

No	Actor Action	System Response
1	The request search is not valid, includes: - Invalid the value of textbox - Fill some especial character	Website show message to notify.

Relationships: N/A

Business Rules:

- Staff can search by customer name
- Staff can search by contract code
- Search algorithm: return all contract that have customer name or contract code contains the keyword.
- If keyword is empty, system returns 10 latest contracts.
- In the result list, contracts having stats “No card” will be listed first.
- Each contract in the list has an identify color to display contract status.
    - Gray: Pending
    - Blue: No card
    - Green: Ready
    - Red: Expired
    - Pink: Request cancel
    - Black: Cancelled

Table 28 Use case PS01 - <Staff> Search / filter contract

2.3.3.1.2. <Staff> View contract information

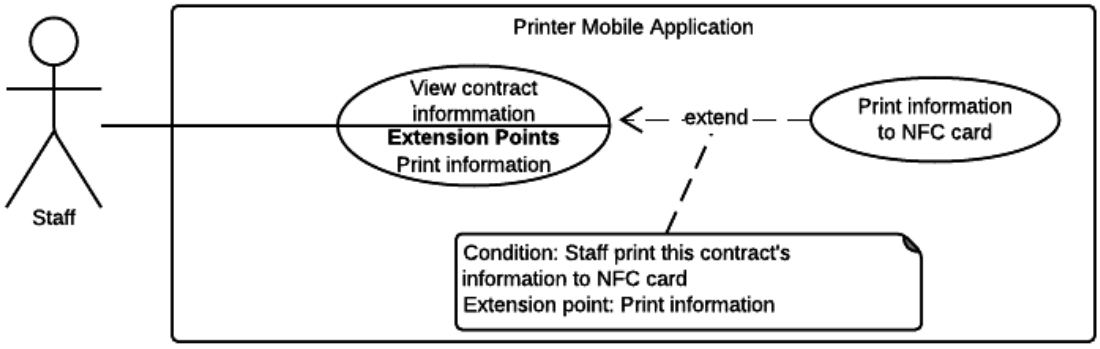


Figure 32 <Staff> View contract information

USE CASE – PS02			
Use Case No.	PS02	**Use Case Version**	2.0
Use Case Name	View contract information		
Author	PhucNH		
Date	21/05/2015	**Priority**	Medium

Actor:

- Staff

Summary:

- This use case helps staff view contract's information.

Goal:

- Staff can view the contract's information.

Triggers:

- Staff send view contract information command.

Preconditions:

- Device is connected to the internet

Post Conditions:

- **Success**: Show contract's information.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff send view contract information command.	Show information about the contract: - Motor owner name: text. - Owner address: text. - Phone number: text - Owner ID/passport number. - Contract code: text - Plate number: text. - Engine: text. - Chassis: text. - Brand: text. - Model Code: text. - Type: text. - Color: text. - Capacity: text.

		- Year of manufacture: text - Weight: text. - Seat capacity: text. - Contract fee: text. - Start date: text - Expired date: text

Exceptions:

No	Actor Action	System Response
1	Cannot connect to server.	System shows error that cannot connect to server.

Relationships: <Police> Print information to NFC card (PS03)

Business Rules:

- The status and the remaining day (if any) of the contract will be highlighted.
- Contract has an identify color to display contract status.
  - Gray: Pending
  - Blue: No card
  - Green: Ready
  - Red: Expired
  - Pink: Request cancel
  - Black: Cancelled
- Contract information must exactly same with contract information in the system.
- Contract start date and expired date is formatted as dd/MM/yyyy
- If information is too long that cannot fit the screen, there is "…" text at the end of the line.

Table 29 Use case PS02 - <Staff> View contract information

2.3.3.1.3. <Staff> Print information to NFC card

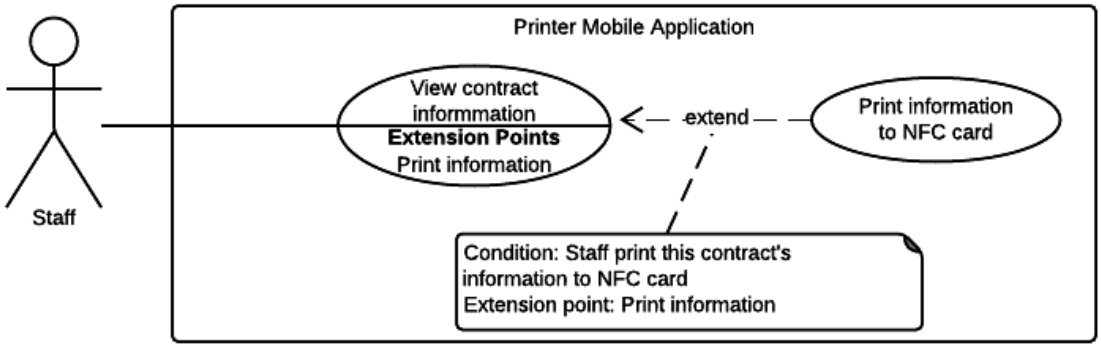


Figure 33 <Staff> Print information to NFC card

USE CASE – PS03			
Use Case No.	PS03	**Use Case Version**	2.0
Use Case Name	Print information to NFC card		
Author	PhucNH		
Date	26/05/2015	**Priority**	High

Actor:
- Staff

Summary:
- This use case helps staff can print information of contract into NFC card.

Goal:
- The information of NFC can be printed from application on mobile

Triggers:
- Staff sends print information into card command.

Preconditions:
- Device must be connected to the internet.
- View contract information succeed.
- Device NFC must be turned on.

Post Conditions:
- **Success**: Print information to NFC card.
- **Fail**: Show error message.

Main Success Scenario:

Step	Actor Action	System Response
1	Staff put the NFC card nearly the device and sends print information into card command	Show message to notify print successful. [Alternative 1,2,3] [Exception 1]

Alternative Scenario:

No	Actor Action	System Response
1	If the contract is cancelled, expired or not paid.	Show message to notify that can't print card for that contract.
2	The NFC card is already issued for another contract.	Show message to notify that the NFC card is already issued for another contract.
3	Can't find NFC card	Show message to notify that can not find NFC card.

Exceptions:

No	Actor Action	System Response
1	Can not connect to server.	Show error message to notify that can not connect to server.

Relationships: <Staff> View contract information (PS02)

Business Rules:
- If the card is already existed in system, it can only be printed when the card is recycled by staff before.
- If there is a new card request for current contract, the new card request status will be set to done.
- Following contracts will not be able to print card:
  - Pending (no payment)
  - Cancelled
- If the contract has no card before, the contract status will be changed from "No card" to "Ready".
- If the contract has a card before, the old card will be deactivated.
- Card access log and history will be logged separately for new card and old card.

Table 30 Use case PS03 - <Staff> Print information to NFC card

3. Software System Attribute

3.1. Usability

3.1.1. Graphic User Interface

All the texts, labels and alerts will be written in Vietnamese.

3.1.2. Usability

- The system usability is easy to use that will need less than 3 days of training for company staffs to use the system.
- Police officers need less than 1 hours of training to use the mobile app.

3.1.3. Installation

- User can follow installation and manual guide for installation. If there are any problems, user cans contacts developer for help.

3.2. Reliability

- System notification success rate is less than 2 failed notifications per 1000 sent.
- System email notification deliver success rate is less than 2 failed deliveries per 1000 sent.
- Scheduler task run at 00:00 everyday with 100% execution rate.
- Web service API response success rate is less than 2 failed requests per 10,000 requests.

3.3. Availability

- N/A

3.4. Security

- All input data are validated before saving to database.
- All privacy information such as password is encrypted to ensure security.
- Users is authentication/authorization for all users when they login to the system.

3.5. Maintainability

- The system is separated into modules.

3.6. Portability

- Admin, staff and customer can use application on every OS supported web browser.
- Police officers and staffs can use mobile application on every Android smartphone that have version greater than 4.0.

3.7. Performance

- Requests from web application are responded in less than 10 seconds at 8 Mbps bandwidth speed.
- Mobile checker application should return card information in less than 1 minute at 8 Mbps bandwidth speed.
- Mobile printer application should write data to NFC card successfully in less than 1 minute at 8 Mbps bandwidth speed.

4. Conceptual Diagram

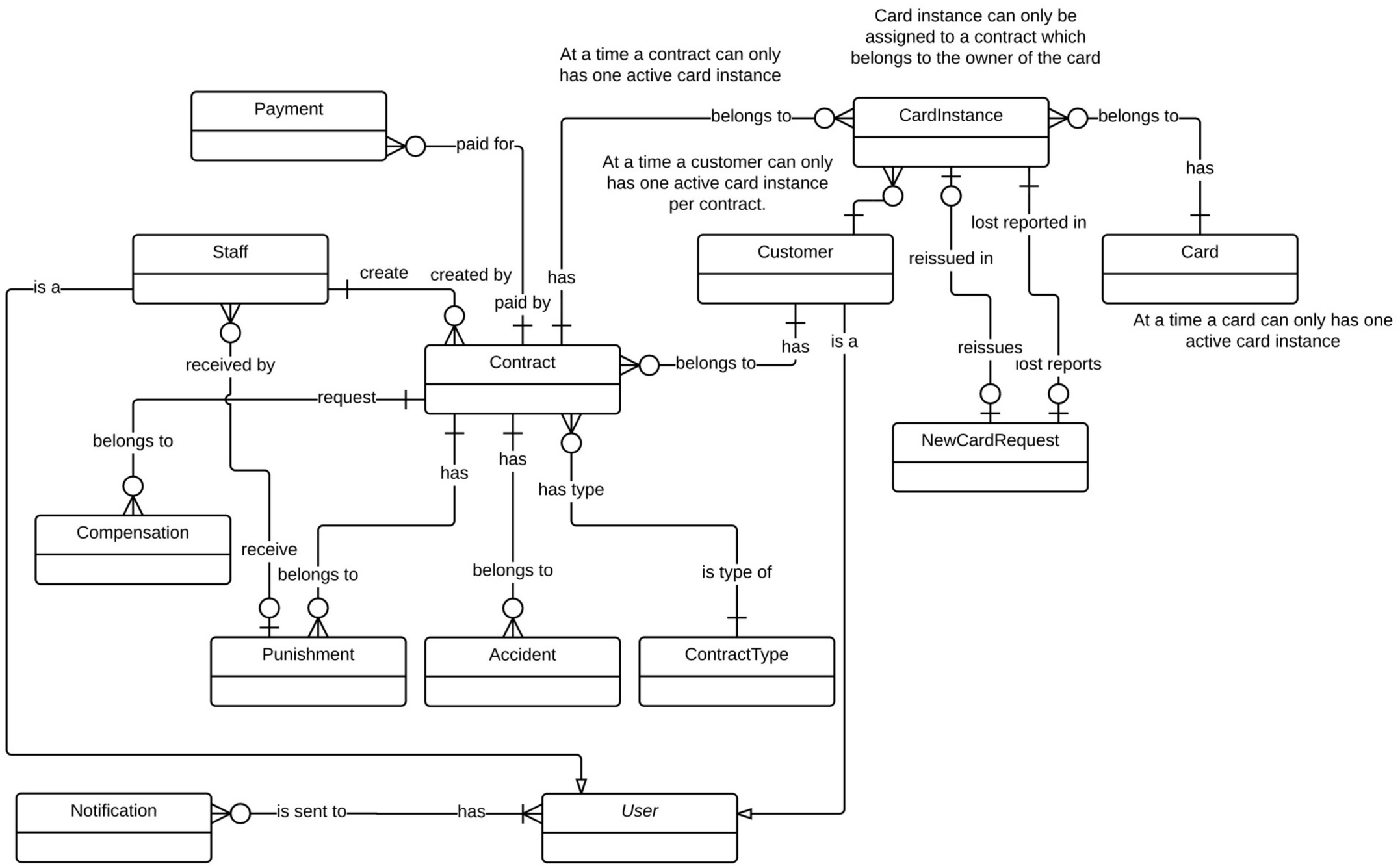


Figure 34 Conceptual diagram

This page is intentionally left blank

Data Dictionary

Entity Data dictionary: describe all content of all entities	
Entity Name	**Description**
User	Abstract entity describes a user in system
Customer	Contain the customer information.
Contract	Contain the contract information.
Card	Contain the card information
CardInstance	Represent a card assigned to a contract
Payment	Contain the payment information.
Staff	Contain the staff information.
Compensation	Contain the compensation information.
Punishment	Contain the punishment information.
Accident	Contain the accident information.
ContractType	Contain the contract type information.
NewCardRequest	Contain the new card request information.
Notification	Contain the notification information

Table 31 Conceptual Diagram Data Dictionary

D. Software Design Description

1. Design Overview

- This document describes the technical and user interface design of MIC system. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.
- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.
- The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.
- The database design describes the relationships between entities and details of each entity.
- Document overview:
    - Section 2: gives an overall description of the system architecture design.
    - Section 3: gives component diagrams that describe the connection and integration of the system.
    - Section 4: gives the detail design description, which includes class diagram, class explanation, and sequence diagram to details the application functions.
    - Section 5: describe a fully attributed Entity Relationship Diagram.

2. System Architecture Design

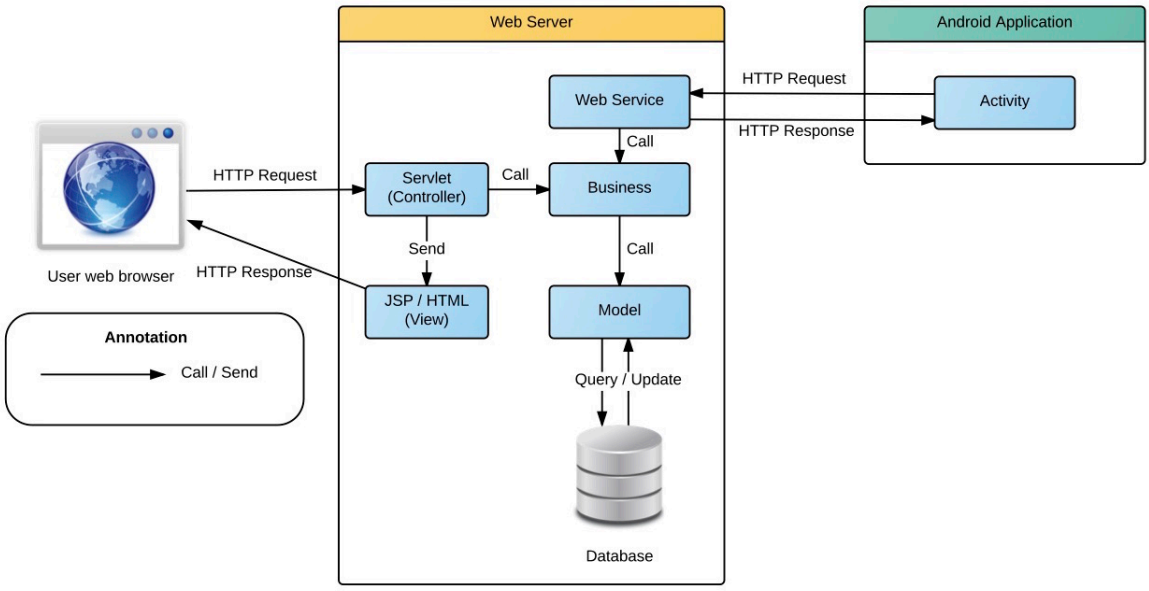


Figure 35 System architecture design

This diagram is referenced and modified from an original concept from: Chapter 6 Architecture Design, SOFTWARE ENGINEERING 9th Edition, by Ian Sommerville.

2.1. Web Application architecture description

In Web Application, the system is developed under J2EE MVC architecture style. We choose this architecture for Web application because of following advantages:

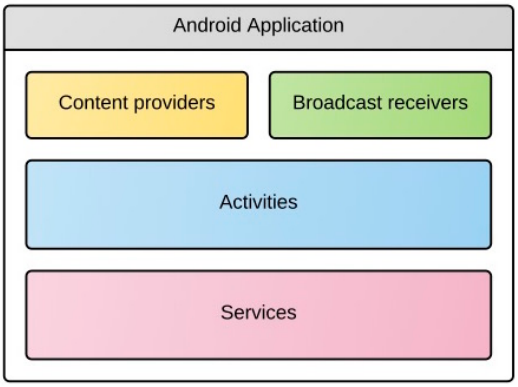
- Web app contains a Web service (public API for mobile app), with MVC architecture, we can separate business code with Controller and View, so we can use the business code in web service without repeat the code.
- Current system only supports motor insurance card, with MVC architecture, we can organize the code better for maintainability, extensibility, reusability so we can expand the scope to other kind of insurance services such as Heath insurance or Asset insurance…
- In scope of 4-member team, MVC architecture make it easier to split the big project into small modules and make it easier to assign each module for members in our team.

This project follows MVC architecture with following components:

- **Servlet (Controller)** is the parts of the application that acts like event handler to handles user interaction. Typically, controller read data from a request and calls appropriate Business's method then selects view to return to user.
- **JSP/HTML (View)** is the parts of the application that handles the display of the data. The selection of View is under control of Controller.
- **Business** is the parts of the application that do business processing to solve domain problems.
- **Model** is the parts of the application that acts like a data transfer object between the system and database.
- **Web Service** is the parts of the application that acts like event handler for web and mobile communication via REST method.

2.2. Mobile Application architecture description

The application is developed as an Android native application. In general, the application architecture conforms to Android architecture.



Reference: Android Developer Guide - Application Fundamentals

http://developer.android.com/guide/components/fundamentals.html

This project follows Android application architecture with following component:

- **Activity** is the basic core of an android application that handles user input,
  create thread to run asynchronous tasks, send request and receive data from server via web services ...

3. Component Diagram

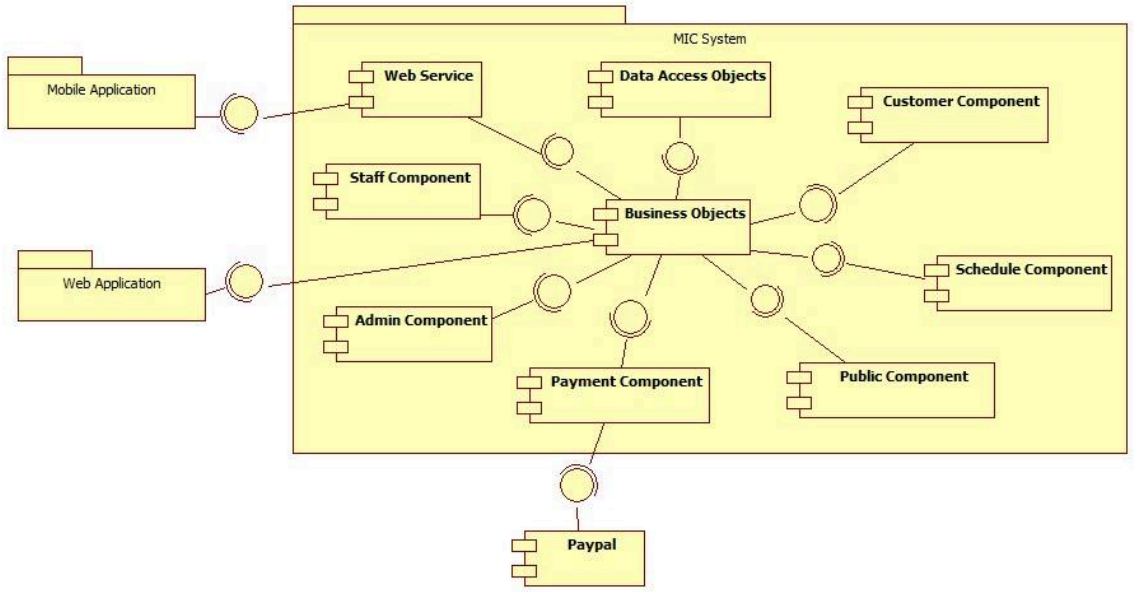


Figure 36 Component Diagram

Component Dictionary: Describes components	
Web Application	Web application package: View, Controller
Mobile Application	Mobile application package
PayPal	Handle payment process with PayPal API
Payment Component	Component to handle payment process
Web Service	Provide API for mobile applications to interact with the system.
Staff Component	Component to handle staff activities in the system
Customer Component	Component to handle customer activities in the system
Public Component	Component to handle guest activities in the system
Admin Component	Component to handle admin activities in the system
Schedule Component	Component to handle scheduler in the system
Business Objects	Common objects to handle domain business operations for each components
Data Access Objects	Component to handle interaction between the system and database

Table 32 Component Dictionary

4. Detailed Description

4.1. Class Diagram

This page is intentionally left blank

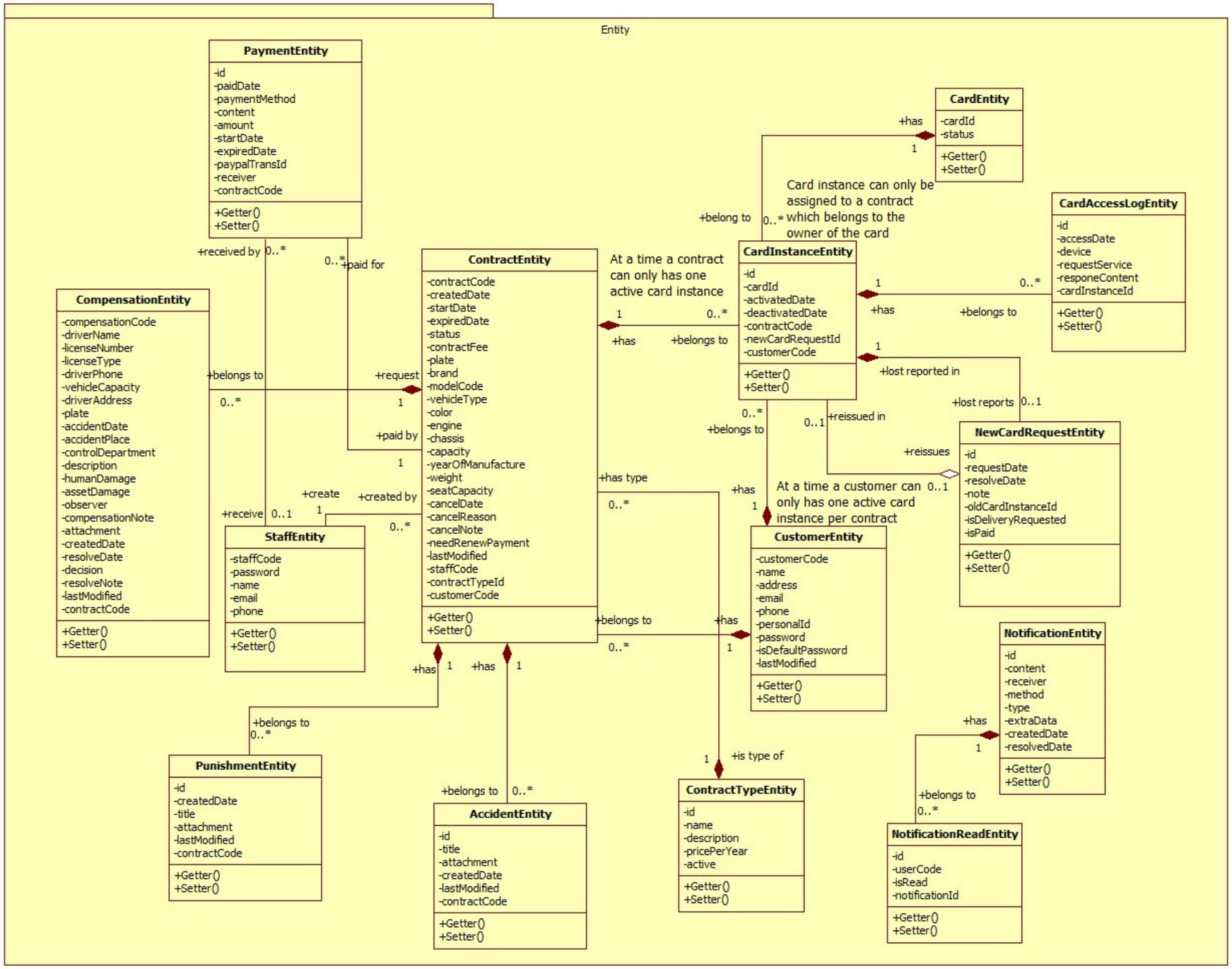


Figure 37 Class Diagram

This page is intentionally left blank

Class dictionary: describe Class		
Class Name	**Mapping column with Conceptual diagram**	**Description**
PaymentEntity	Payment	Contain the payment information.
CardEntity	Card	Contain the card information.
CardInstanceEntity	CardInstance	Contain the card instance information
CustomerEntity	Customer	Contain the customer information.
ContractEntity	Contract	Contain the contract information.
StaffEntity	Staff	Contain the staff information.
CompensationEntity	Compensation	Contain the compensation information.
PunishmentEntity	Punishment	Contain the punishment information.
AccidentEntity	Accident	Contain the accident information.
ContractTypeEntity	ContractType	Contain the contract type information.
NewCardRequestEntity	NewCardRequest	Contain the new card request information.
CardAccessLogEntity	N/A	Not exist in conceptual diagram. But needed in class diagram to contain the card access log information.
NotificationEntity	N/A	Not exist in conceptual diagram. But needed in class diagram to contain the notification information.
NotificationReadEntity	N/A	Not exist in conceptual diagram. But needed in class diagram to know what notifications is read.

Table 33 Class dictionary

4.2. Class Diagram Explanation

4.2.1. PaymentEntity

Attribute	Type	Visibility	Description
Id	int	Private	Unique identifier of an payment
paidDate	Timestamp	Private	The day process the payment
paymentMethod	String	Private	The payment method
content	String	Private	The content of the payment
amount	Double	Private	Total amount of the payment
receiver	String	Private	The payment receiver
paypalTransId	String	Private	The PayPal transaction ID
contractCode	String	Private	The contract code.

Table 34 PaymentEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 35 PaymentEntity Method

4.2.2. CardEntity

Attribute	Type	Visibility	Description
CardId	String	Private	Unique identifier of a card
Status	Integer	Private	Status of the card

Table 36 CardEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 37 CardEntity Method

4.2.3. CardInstanceEntity

Attribute	Type	Visibility	Description
Id	Integer	Private	Unique identifier of a card instance
cardId	String	Private	Card id of the instance
activatedDate	Timestamp	Private	The date the card was activated.
deactivatedDate	Timestamp	Private	The date the card was deactivated.
contractCode	String	Private	The contract code
newCardRequestId	String	Private	The new card request ID.

Table 38 CardInstanceEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 39 CardInstanceEntity Method

4.2.4. CustomerEntity

Attribute	Type	Visibility	Description
customerCode	String	Private	Unique identifier of an customer
name	String	Private	Customer's name.
address	String	Private	Customer's address.
email	String	Private	Customer's email.
phone	String	Private	Customer's phone number.
personalId	String	Private	The customer's personal ID.
password	String	Private	Customer's password.
customerCode	String	Private	The customer's code.

Table 40 CustomerEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 41 CustomerEntity Method

4.2.5. ContractEntity

Attribute	Type	Visibility	Description
contractCode	String	Private	Unique identifier of a contract
startDate	Timestamp	Private	The start date of the contract.
expiredDate	Timestamp	Private	The expired date of the contract.
status	String	Private	The status of the contract.
contactFee	Double	Private	The contract fee.
plate	String	Private	The plate number of the motor.
brand	String	Private	The brand of the motor.
modelCode	String	Private	The model code of the motor.
vehicleType	String	Private	The vehicle type.
color	String	Private	The motor's color.
engine	String	Private	The motor's engine.
chassis	String	Private	The motor's chassis.
capacity	String	Private	The motor's capacity.
yearOfManufacture	int	Private	The year of manufacture.

weight	int	Private	The motor's weight.
seatCapacity	int	Private	The seat capacity.
cancelDate	Timestamp	Private	The cancel date of the contract.
cancelReason	String	Private	The cancel reason.
cancelNote	String	Private	The cancel note.
contractTypeId	int	Private	The contract type ID.
customerCode	String	Private	The customer code.

Table 42 ContractEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 43 ContractEmtity Method

4.2.6. StaffEntity

Attribute	Type	Visibility	Description
staffCode	String	Private	Unique identifier of a staff
password	String	Private	The staff's password
name	String	Private	The staff's name.
email	String	Private	The staff's email.
phone	String	Private	The staff's phone number.

Table 44 StaffEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 45 StaffEntity Method

4.2.7. CompensationEntity

Attribute	Type	Visibility	Description
compensationCode	String	Private	Unique identifier of a compensation
driverName	String	Private	The driver name.
licenseNumber	String	Private	The license number of the motor.
licenseType	String	Private	The license type of the motor.

driverPhone	String	Private	The driver's phone number.
vehicleCapacity	String	Private	The vehicle capacity.
driverAddress	String	Private	The driver's address.
plate	String	Private	The plate number.
accidentDate	Timestamp	Private	The accident date.
accidentPlace	String	Private	The accident place.
controlDepartment	String	Private	The police control department.
description	String	Private	The compensation's description.
humanDamage	String	Private	The human damages.
assetDamage	String	Private	The asset damages.
observer	String	Private	The observer.
compensationNote	String	Private	The compensation note.
attachment	String	Private	The attachment.
createdDate	Timestamp	Private	The created date of the compensation.
resolveDate	Timestamp	Private	The resolve date.
decision	String	Private	The decision.
resolveNote	String	Private	The resolve note.
contractCode	String	private	The contract code.

Table 46 CompensationEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 47 CompensationEntity Method

4.2.8. PunishmentEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of a punishment
createdDate	Timestamp	Private	The created date.
title	String	Private	The title of the punishment.
attachment	String	Private	The attachment.
contractCode	String	Private	The contract code.

Table 48 PunishmentEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 49 PunishmentEntity Method

4.2.9. AccidentEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of an accident
title	String	Private	The title of the accident.
attachment	String	Private	The attachment.
createdDate	Timestamp	Private	The created date.
contractCode	String	Private	The contract code.

Table 50 AccidentEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 51 AccidentEntity Method

4.2.10. ContractTypeEntity

Attribute	Type	Visibility	Description
id	int	Private	unique identifier of a contract type
name	String	Private	The contract's name.
description	String	Private	The description.
pricePerYear	Double	Private	The price per year of the contract.

Table 52 ContractTypeEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 53 ContractTypeEntity Method

4.2.11. NewCardRequestEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of a new card request
requestDate	Timestamp	Private	The request date.

resolveDate	Timestamp	Private	The resolve date.
note	String	Private	The new card request note.
oldCardId	String	Private	The ID of the old card.
customerCode	String	Private	The customer code.

Table 54 NewCardRequestEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 55 NewCardRequestEntity Method

4.2.12. CardAccessLogEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of an card access log
accessDate	Timestamp	Private	The access date.
device	String	Private	The device name.
requestService	String	Private	The request service.
responseContent	String	Private	The response content.

Table 56 CardAccessLogEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 57 CardAccessLogEntity Method

4.2.13. NotificationEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of the notification.
content	String	Private	Content of the notification.
receiver	String	Private	A regex pattern that match with staffCode or customerCode.

method	int	Private	Notification method such as email or web notification.
type	int	Private	Type of the action were notified.
extraData	String	Private	The extra data such as contractCode, compensationCode,etc.
createdDate	Timestamp	Private	The created date.
resolvedDate	Timestamp	Private	The resolved date.

Table 58 NotificationEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 59 NotificationEntity Method

4.2.14. NotificationReadEntity

Attribute	Type	Visibility	Description
id	int	Private	Unique identifier of the notification read.
userCode	String	Private	Customer code or staff code.
isRead	int	Private	Check if the notification is read.
notificationId	int	Private	Notification's Id.

Table 60 NotificationReadEntity Attribute

Method	Return Type	Visibility	Description
Getter	Attribute type	Public	Get attribute value.
Setter	Void	Public	Set value of attribute.

Table 61 NotificationReadEntity Method

4.3. Interactive Diagram

4.3.1. Web Application

4.3.1.1.Staff

4.3.1.1.1. Create new contract

Summary: this diagram show process of staff creates new contract

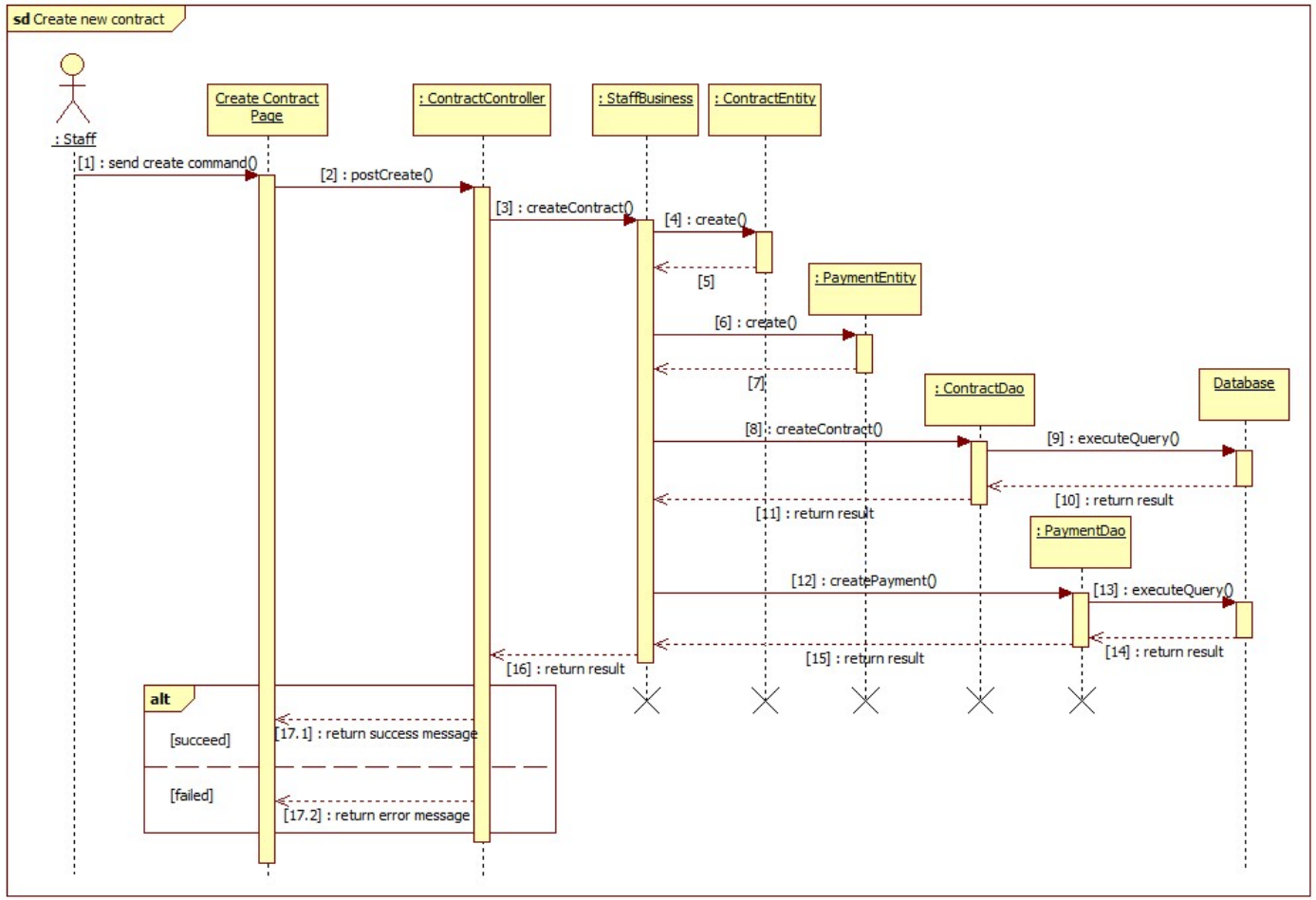


Figure 38 Sequence diagram - <Staff> Create new contract

4.3.1.1.2. Renew contract

Summary: this diagram show process of staff renews contract

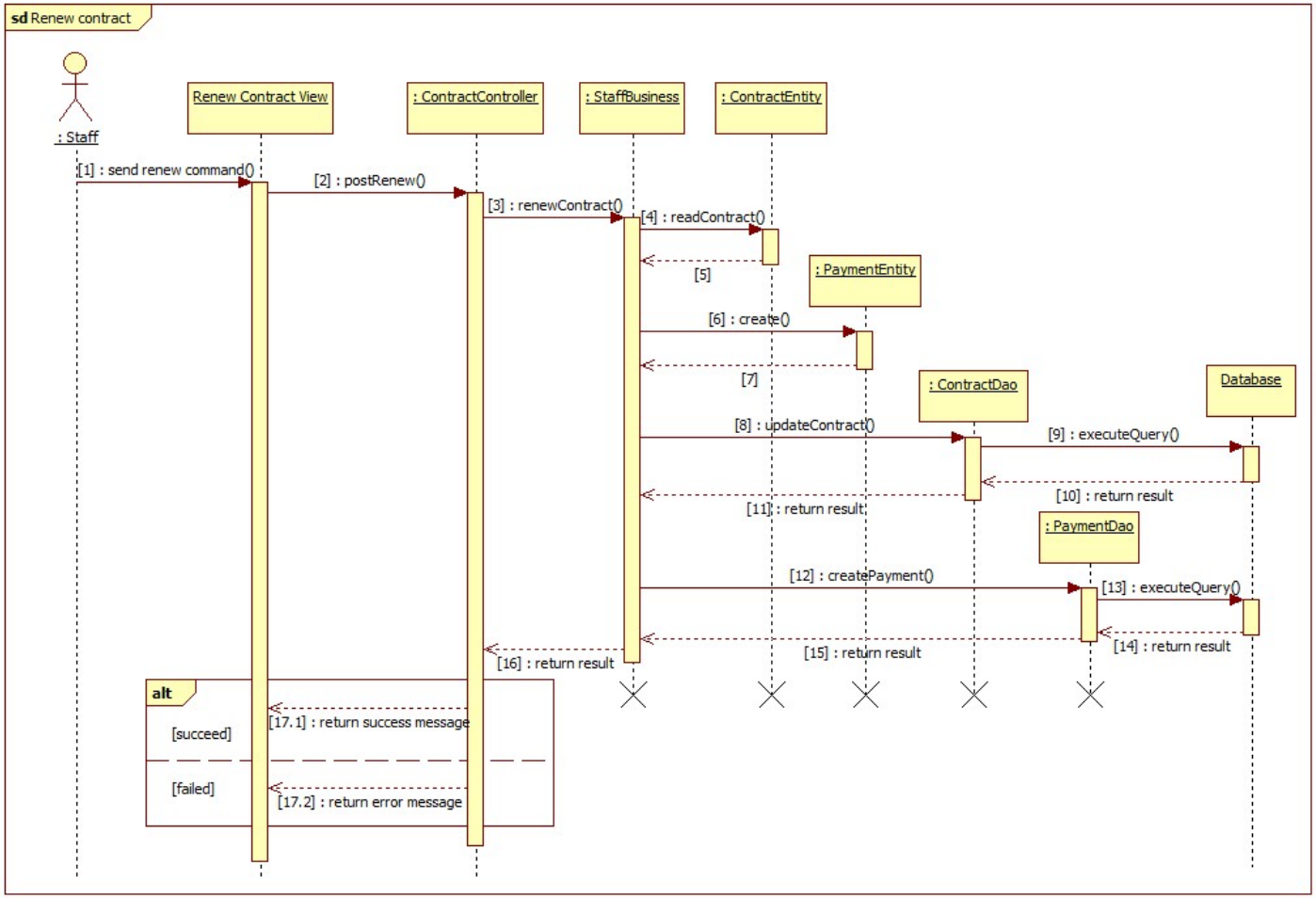


Figure 39 Sequence diagram - <Staff> Renew contract

4.3.1.1.3. Cancel contract

Summary: this diagram show process of staff cancels contract

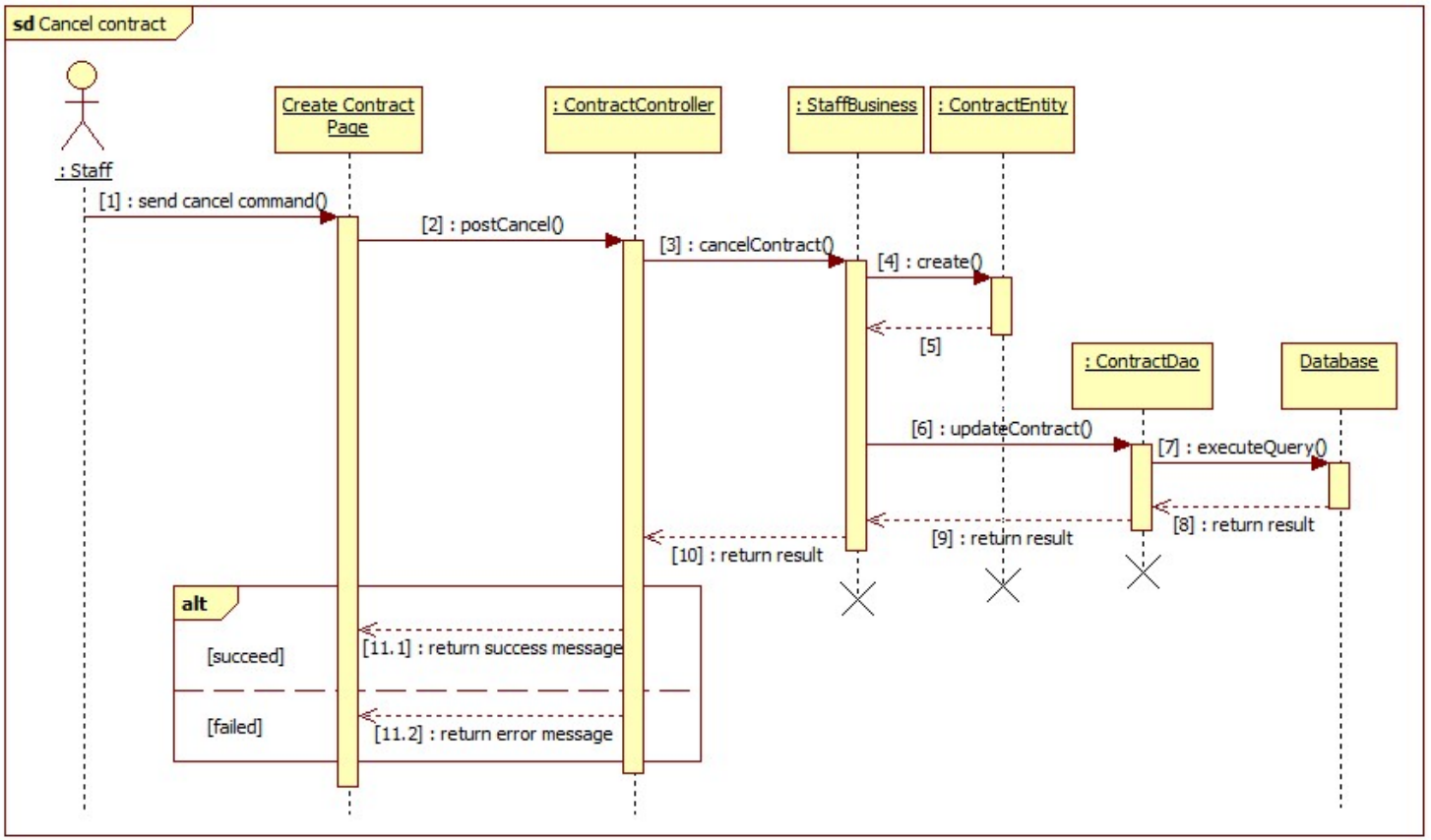


Figure 40 Sequence diagram - <Staff> Cancel contract

4.3.1.2.Customer

4.3.1.2.1. Cancel contract

Summary: this diagram show process of customer cancels contract

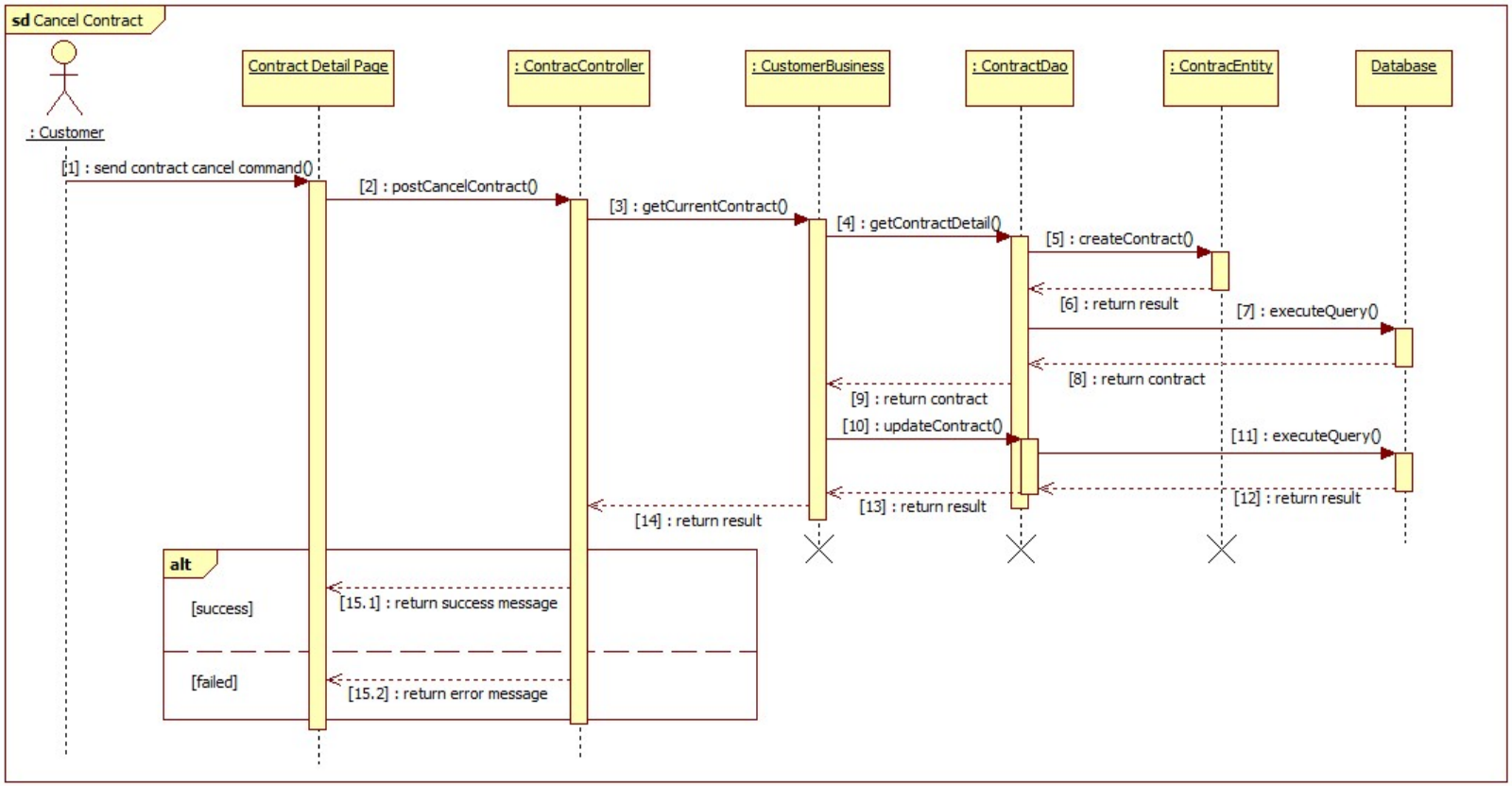


Figure 41 Sequence diagram - <Customer> Cancel contract

4.3.1.2.2. Renew contract

Summary: this diagram show process of customer renews contract

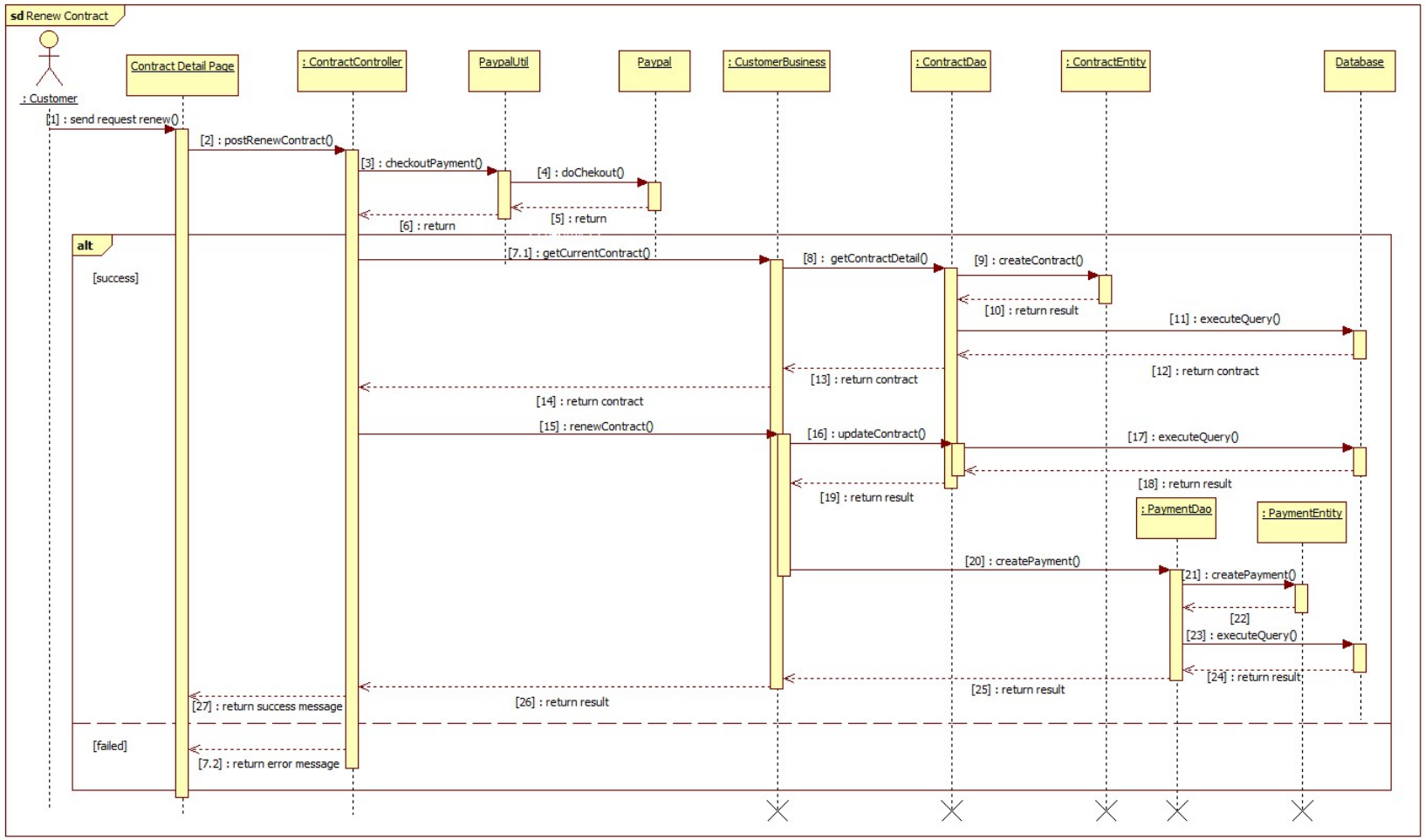


Figure 42 Sequence diagram - <Customer> Renew contract

4.3.1.3.Guest

4.3.1.3.1. Register new contract

Summary: this diagram show process of guest registers new contract

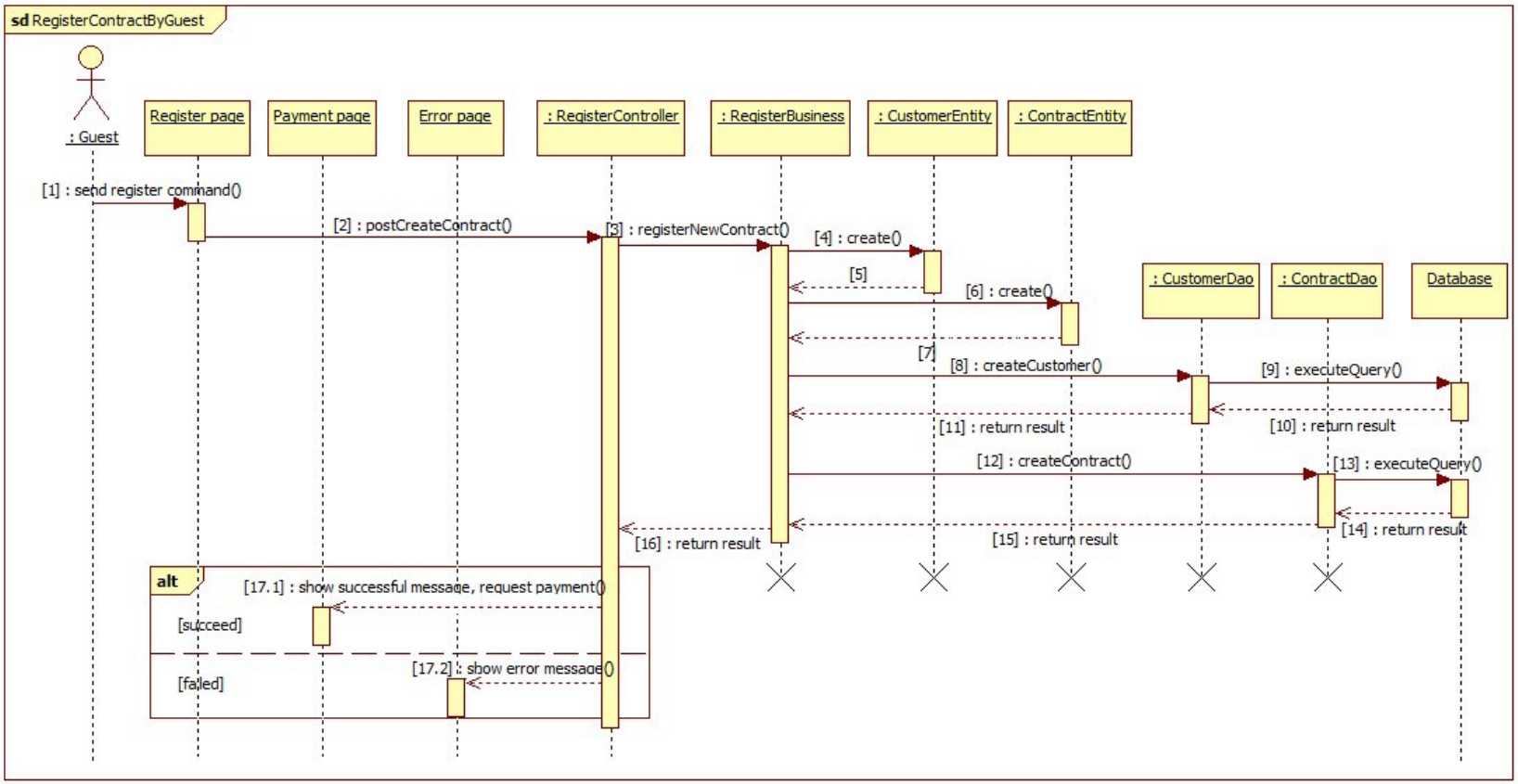


Figure 43 Sequence diagram - <Guest> Register new contract

4.3.1.3.2. PayPal payment

Summary: this diagram show process of guest does payment

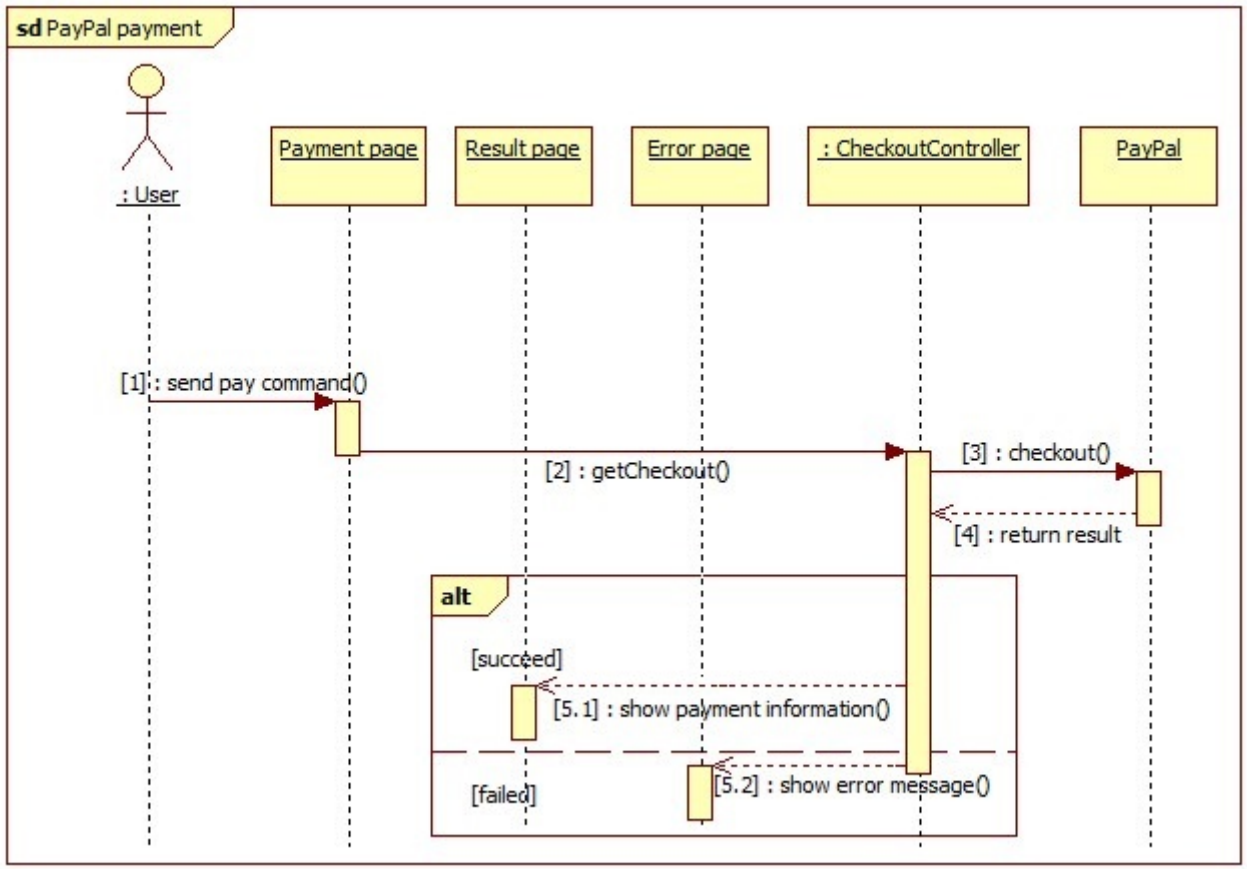


Figure 44 Sequence diagram - <Guest> PayPal payment

4.3.2. Mobile Application

4.3.2.1.Checker Mobile Application

4.3.2.1.1. <Police> Verify card validation

Summary: this diagram show process of police checks card validation.

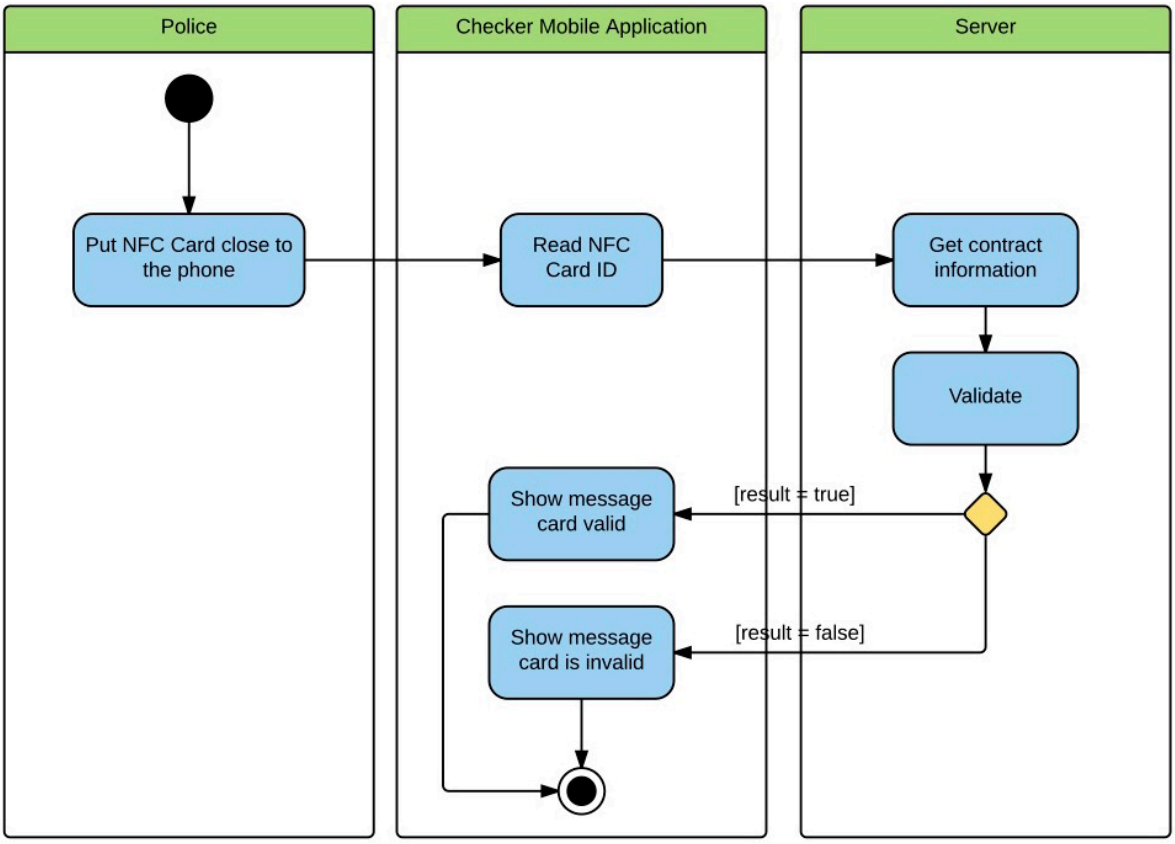


Figure 45 <Police> Verify card validation

4.3.2.1.2. <Police> Add punishment information

Summary: this diagram show process of police adds punishment information.

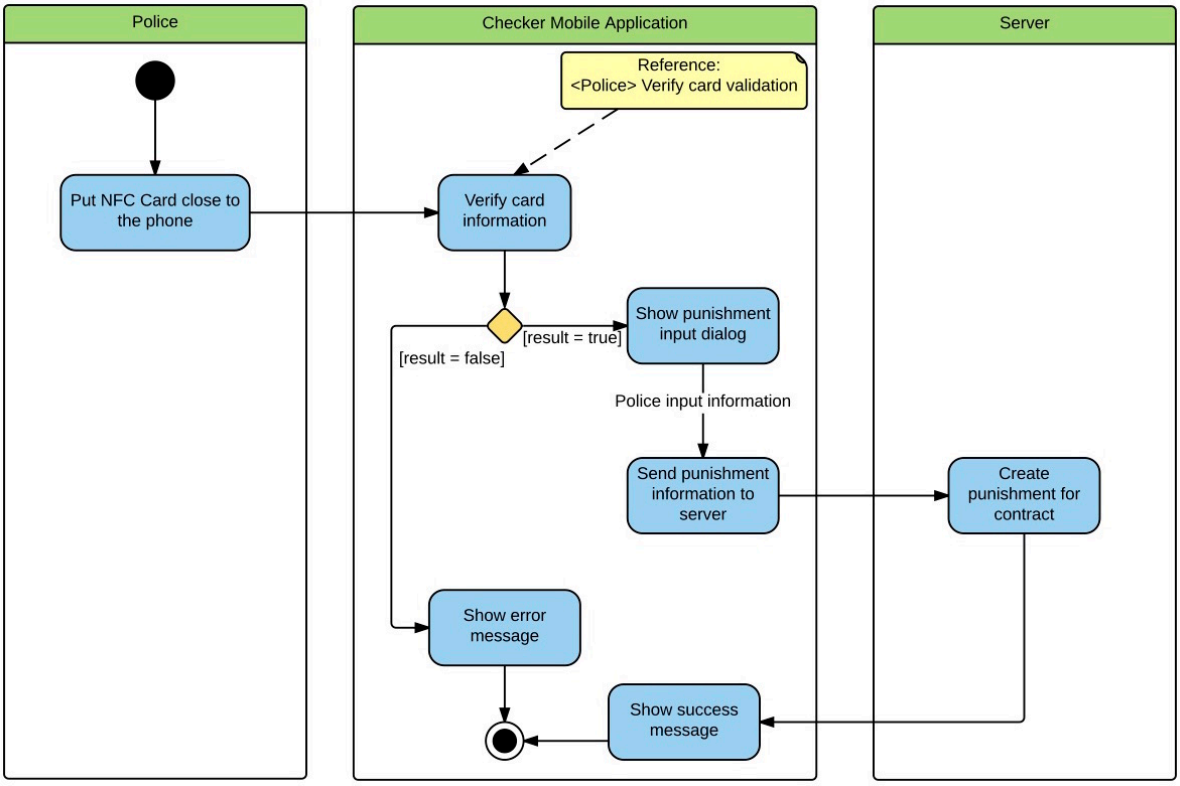


Figure 46 <Police> Add punishment information

4.3.2.2.Printer Mobile Application

4.3.2.2.1. <Staff> Search contract

Summary: this diagram show process of staff searchs for contracts.

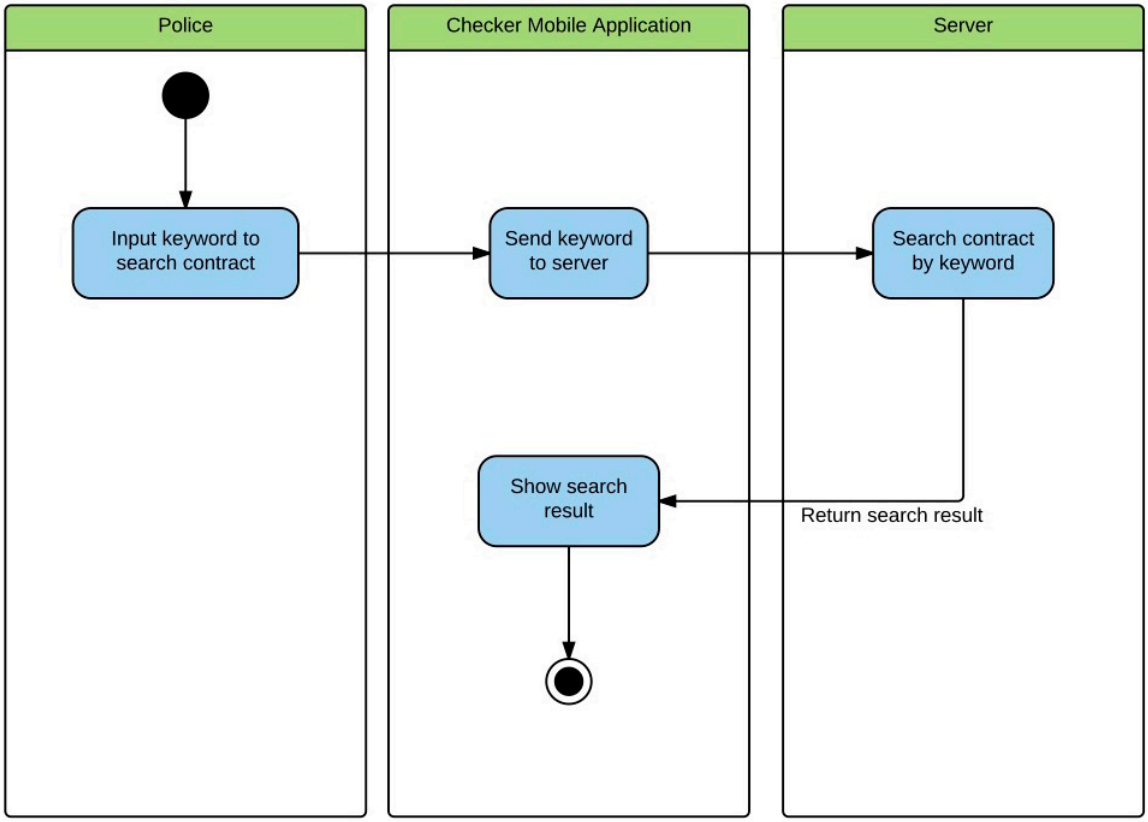


Figure 47 <Staff> Search contract

4.3.2.2.2. <Staff> View contract information

Summary: this diagram show process of staff views contract information.

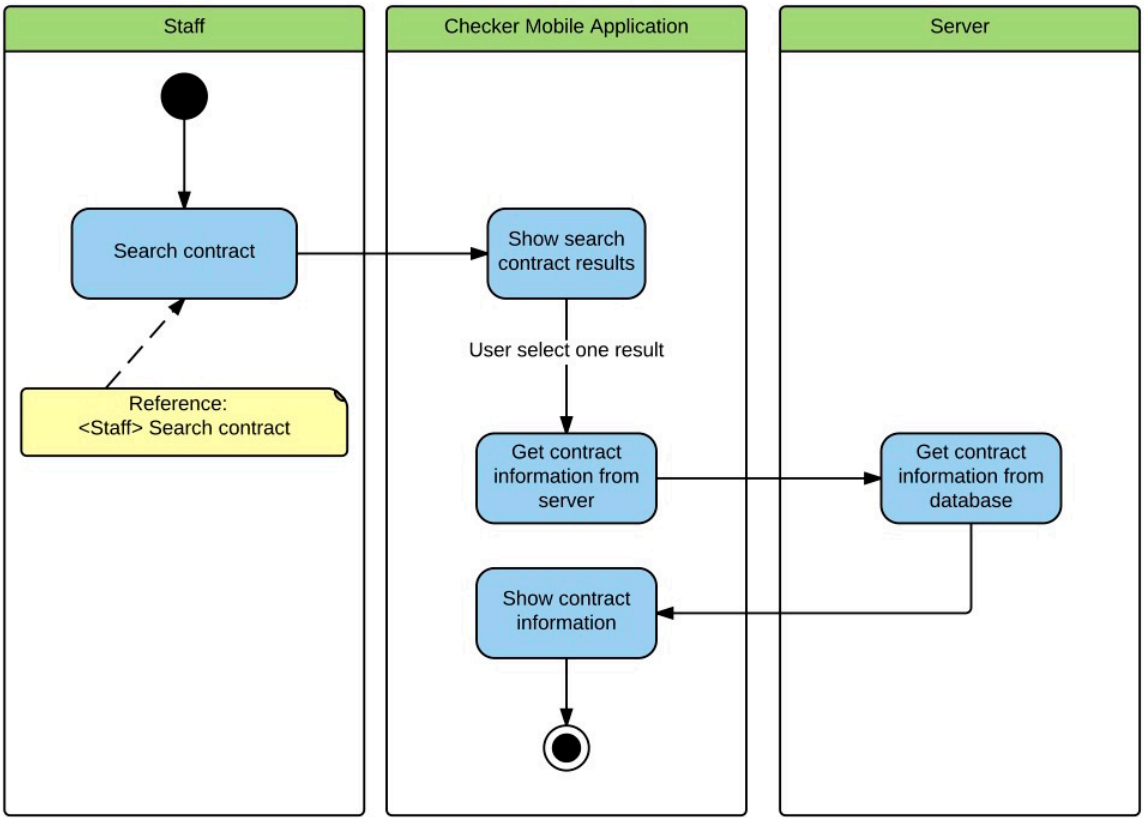


Figure 48 <Staff> View contract information

4.3.2.2.3. <Staff> Print information to NFC Card

Summary: this diagram show process of staff print information to NFC Card.

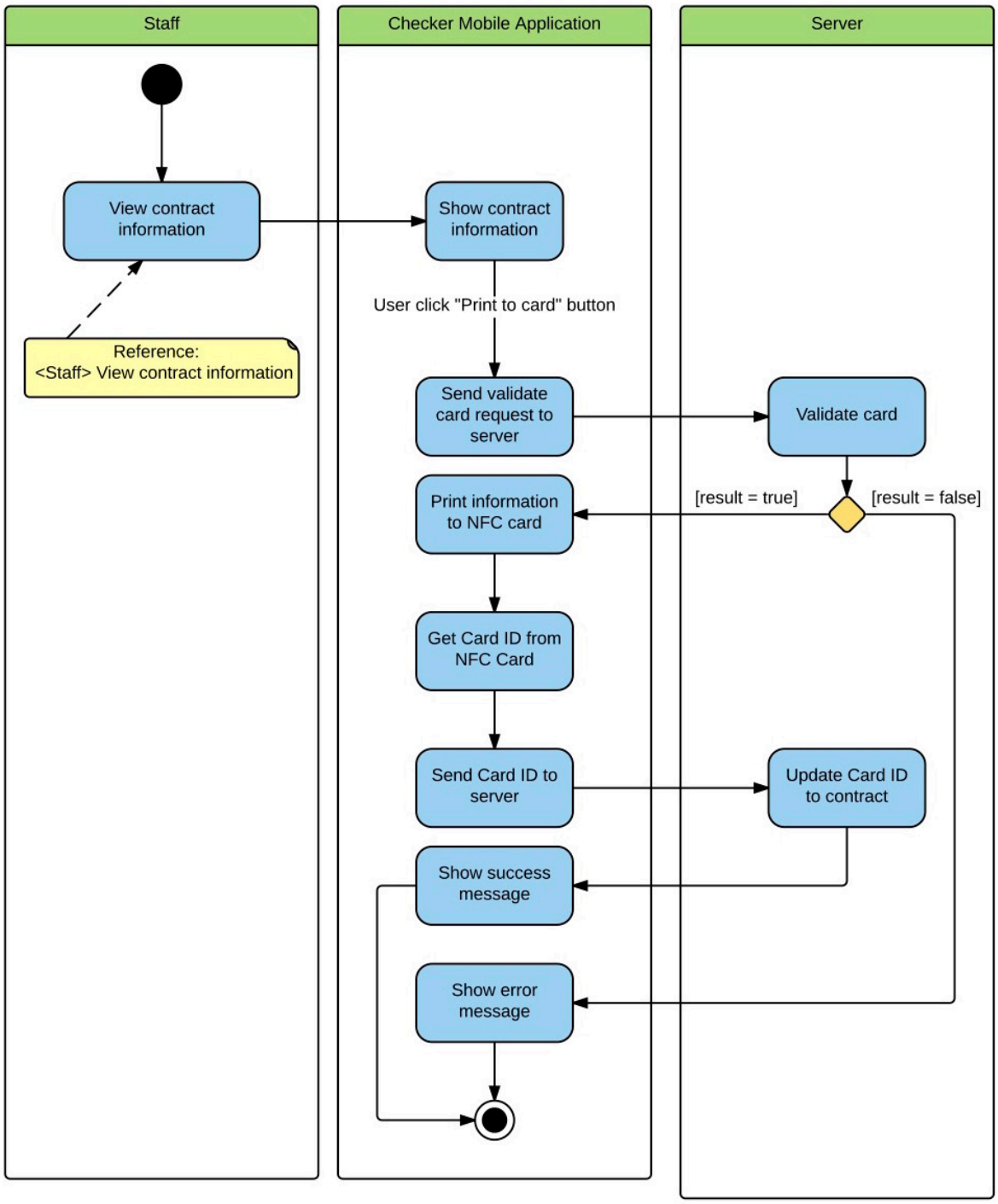


Figure 49 <Staff> Print information to NFC card

Reference to full document for complete list of Interactive Diagram

5. Interface

5.1. Component Interface

5.1.1. Web Service Interface

Signature	Description	Input	Output	Output Format	Exception
public ResponseObject getCheckConnection(R r)	Check server status	Request object r	Json Boolean the status of server	Boolean	JsonProcessingException
public ResponseObject getContracts(R r)	Search contracts by code or customer name	Request object r contains: Keyword: String	Json string List of ContractSearchResultDto	ContractEntity CustomerEntity	JsonProcessingException
public ResponseObject getUpdateCardID(R r)	Update card ID to a contract	Request object r contains: cardID: String contractCode: String	Json CardEntity object	CardEntity	JsonProcessingException NoResultException
public ResponseObject getCheckCard(R r)	Check the validation of the card	Request object r contains: cardID: String	Json CardEntity object	CardEntity	JsonProcessingException NoResultException
public ResponseObject getUpdatePunishment(R r)	Update punishment information to a contract	Request object r contains: contractCode: String punishmentInfo: PunishmentInfoObject	Json Boolean the update result	Boolean	JsonProcessingException

Table 62 Web Services Interfaces

Entity	Description
ContractEntity	contractCode startDate expiredDate status contractFee plate brand modelCode vehicleType color engine chassis capacity yearOfManufacture weight seatCapacity cancelDate cancelReason cancelNote staffCode contractTypeId customerCode needRenewPayment lastModified createdDate
CustomerEntity	customerCode name address email phone personalId password isDefaultPassword lastModified
CardEntity	cardId activatedDate deactivatedDate contractCode newCardRequestId

Table 63 Output format description

Exception	Description
JsonProcessingException	encountered when processing (parsing, generating) JSON content that are not pure I/O problems
NoResultException	Thrown by the persistence provider when getSingleResult() is executed on a query and there is no result to return

Table 64 Exception description

5.2. Web application Design

5.2.1. Staff Interface Design

5.2.1.1.Staff home page

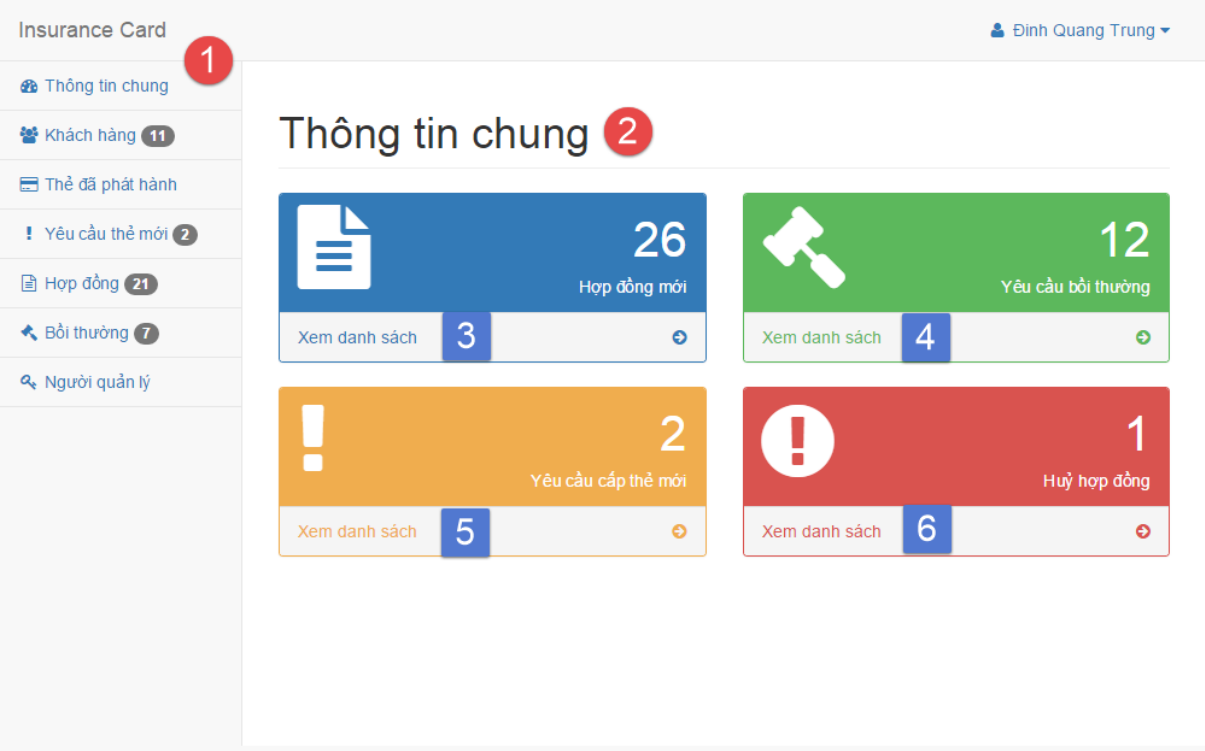


Figure 50 Interface - <Staff> Home page

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	Menu	Navigation bar	Yes	Yes	Menu bar	N/A	N/A
2	Title	Title of the page	Yes	Yes	Label	N/A	N/A

Table 65 Staff home page fields

Buttons/Hyperlinks

No	Function	Description	Validation	Outcome
3	viewContract	List all contract	N/A	Transfer to list contract page
4	viewCompensation	List all compensation	N/A	Transfer to list compensation page
5	viewNewCardReq	List all new card request	N/A	Transfer to list new card request page

6	viewCancelContract	List all cancel contract request	N/A	Transfer to list contract page

Table 66 Staff home page buttons/hyperlinks

5.2.1.2.Create contract

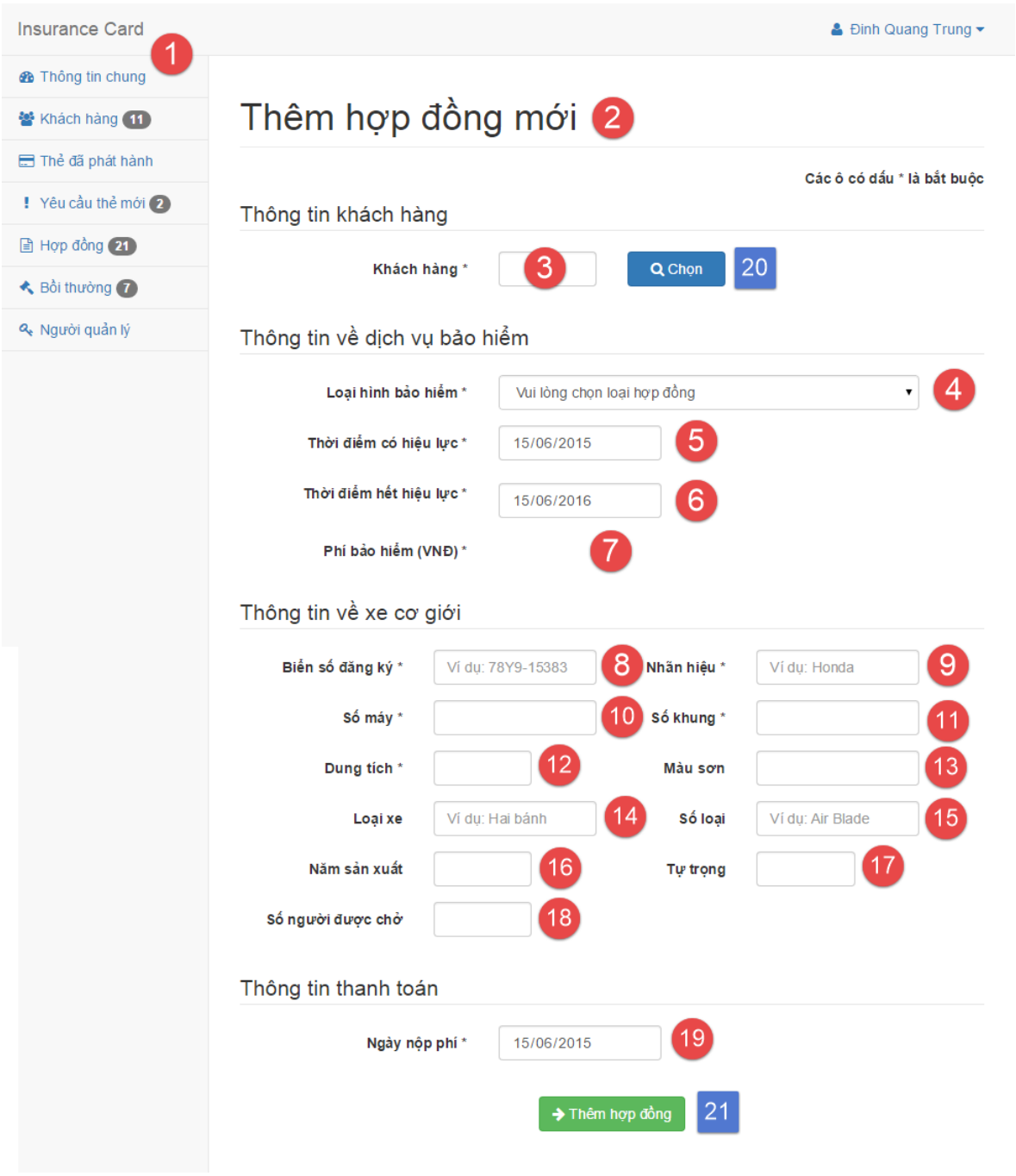


Figure 51 Interface - <Staff> Create contract

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	Menu	Navigation bar	Yes	Yes	Menu bar	N/A	N/A
2	Title	Title of the page	Yes	Yes	Label	N/A	N/A
3	txtCustomerCode	Fill customer code	No	Yes	Textbox	String	6
4	ddlContractType	Select contract type	No	Yes	Select	N/A	N/A
5	txtStartDate	Select start date	No	Yes	Date input form	N/A	N/A
6	txtExpiredDate	Select expired date	No	Yes	Date input form	N/A	N/A
7	txtContractFee	Contract fee	Yes	Yes	Label	N/A	N/A
8	txtPlate	Fill vehicle plate	No	Yes	Textbox	String	4 – 15
9	txtBrand	Fill vehicle brand	No	Yes	Textbox	String	2 – 20
10	txtEngine	Fill vehicle engine	No	Yes	Textbox	String	2 – 20
11	txtChassis	Fill vehicle chassis	No	Yes	Textbox	String	2 – 20
12	txtCapacity	Fill vehicle capacity	No	Yes	Textbox	String	2 – 20
13	txtColor	Fill vehicle color	No	No	Textbox	String	2 – 20
14	txtType	Fill vehicle type	No	No	Textbox	String	2 – 20
15	txtModelcode	Fill vehicle model code	No	No	Textbox	String	2 – 20
16	txtYearOfMan	Fill vehicle year of manufacture	No	No	Textbox	String	4
17	txtWeight	Fill vehicle empty weight	No	No	Textbox	String	1 – 4
18	txtSeatCapacity	Fill vehicle seat capacity	No	No	Textbox	String	1 – 3

19	txtPaidDate	Select paid contract fee date	No	Yes	Date input form	N/A	N/A

Table 67 <Staff> Create contract fields

Buttons/Hyperlinks

No	Function	Description	Validation	Outcome
20	btnSelect	Select customer	N/A	Transfer to select customer modal
21	btnCreate	Create new contract	N/A	Transfer to create new contract successful page

Table 68 <Staff> Create contract buttons/hyperlinks

5.2.2. Customer Interface Design

5.2.2.1.Manage contract

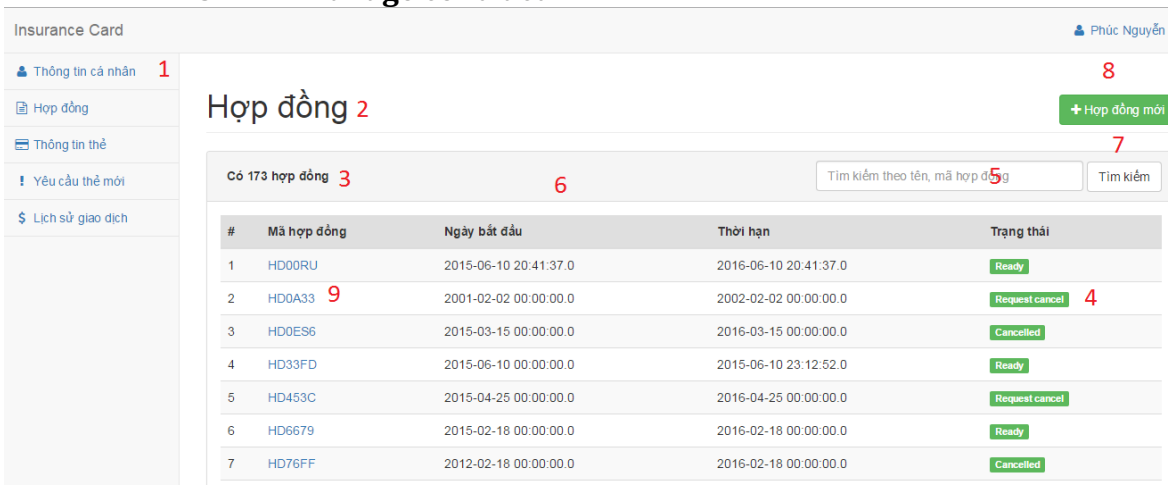


Figure 52 Interface - <Customer> Manage contract

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	lblMenu	Navigation bar	Yes	Yes	Menu bar	N/A	N/A
2	lblTitle	Title of the page	Yes	Yes	Label	N/A	N/A
3	lblAmountContract	Number of available customer	Yes	No	Label	N/A	N/A

4	lblContractStatus	Status of contract	Yes	No	Label	N/A	N/A
5	txtSearch	Fill search keyword	No	Yes	Textbox	String	N/A
6	contractTable	Table of available contract	Yes	Yes	Table	N/A	N/A

Table 69 <Customer> Manage contract fields

Buttons / Hyperlinks

No	Function	Description	Validation	Outcome
7	btnSearch	Search by input keyword	N/A	Transfer to search result page
8	btnCreate	Create new customer	N/A	Transfer to create new customer page
9	linkContractDetail	View contract detail	N/A	Transfer to contract detail page

Table 70 <Customer> Manage contract buttons/hyperlinks

5.2.2.2.Contract detail

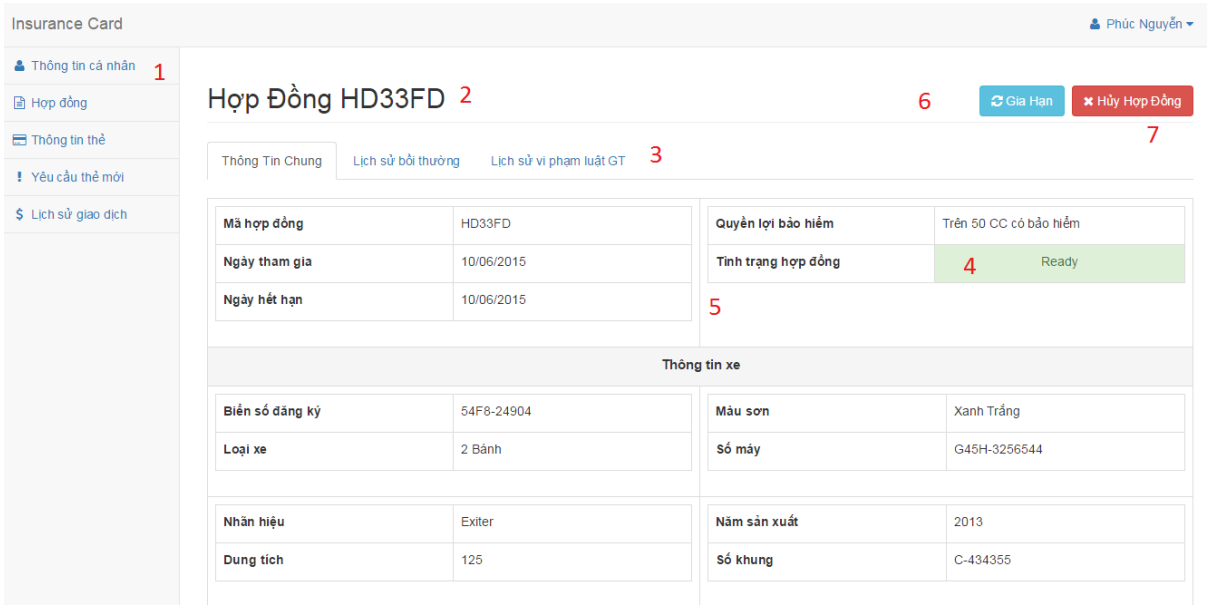


Figure 53 Interface - <Customer> Contract detail

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length

1	lblMenu	Navigation bar	Yes	Yes	Menu bar	N/A	N/A
2	lblTitle	Contract code	Yes	Yes	Label	N/A	N/A
3	txtTab	Information about contract	Yes	No	Tab	N/A	N/A
4	lblContractStatus	Status of contract	Yes	No	Label	N/A	N/A
5	txtInformation	Information about contract	No	Yes	Textbox	String	N/A

Table 71 <Customer> Contract detail fields

Buttons / Hyperlinks

No	Function	Description	Validation	Outcome
6	btnRenew	Renew contract	N/A	Show popup to choose the reason cancel
7	btnCancel	Cancel contract	N/A	Show popup to provide new time for contract and the way to pay for renew contract

Table 72 <Customer> Contract detail buttons/hyperlinks

Reference to full document for complete list of web application design.

5.3. Checker Mobile Application Design

5.3.1. Scan NFC card screen

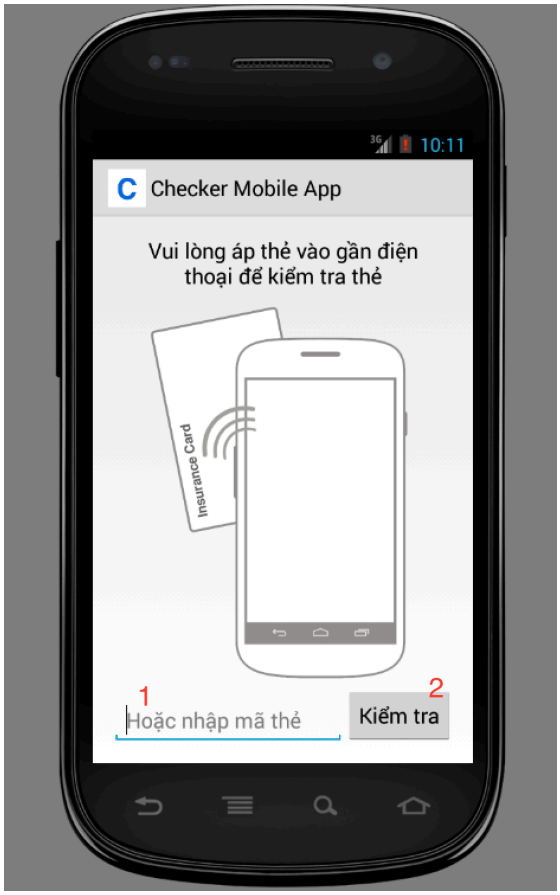


Figure 54 Scan NFC card screen

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	txtCardID	ID of the card	No	No	Textbox	String	20

Table 73 Scan NFC card screen - Fields

Buttons

No	Function	Description	Validation	Outcome
2	btnCheckCard	Verify card validation	N/A	Go to view card information screen

Table 74 Scan NFC card screen - Buttons

5.3.2. Add punishment screen

Figure 55 Add punishment screen

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	txtDescription	Description of punishment	No	No	Textbox	String	2-200
2	flePicture	Picture of punishment	No	Yes	Button	Image	Width: 100px

Table 75 Add punishment screen - Fields

Buttons

No	Function	Description	Validation	Outcome

1	btnSend	Send punishment information to server	N/A	Show information sent success, back to home screen.

Table 76 Add punishment screen - Buttons

5.4. Printer Mobile Application Design

5.4.1. Search contract screen

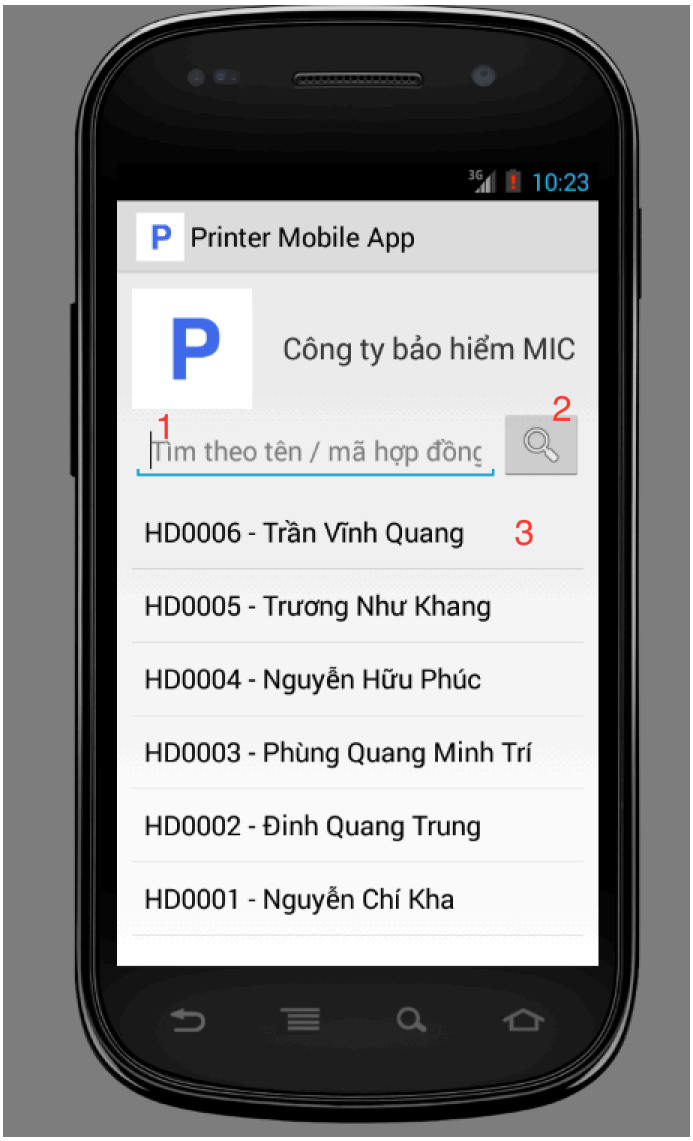


Figure 56 Search contract screen

Fields

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type	Length
1	txtSearch	Search box where user	No	No	Textbox	String	0 - 200

		input search keyword					

Table 77 Search contract screen - Fields

Buttons

No	Function	Description	Validation	Outcome
2	btnSearch	Search contract with keyword	N/A	The list will be updated with search results
3	lstContracts	Click to a row from the list to view detail information of that contract	N/A	Show view contract screen

Table 78 Search contract screen - Buttons

5.4.2. Print card screen

Figure 57 Print card screen

6. Database Design

6.1. Entity relationship diagram

This page is intentionally left blank

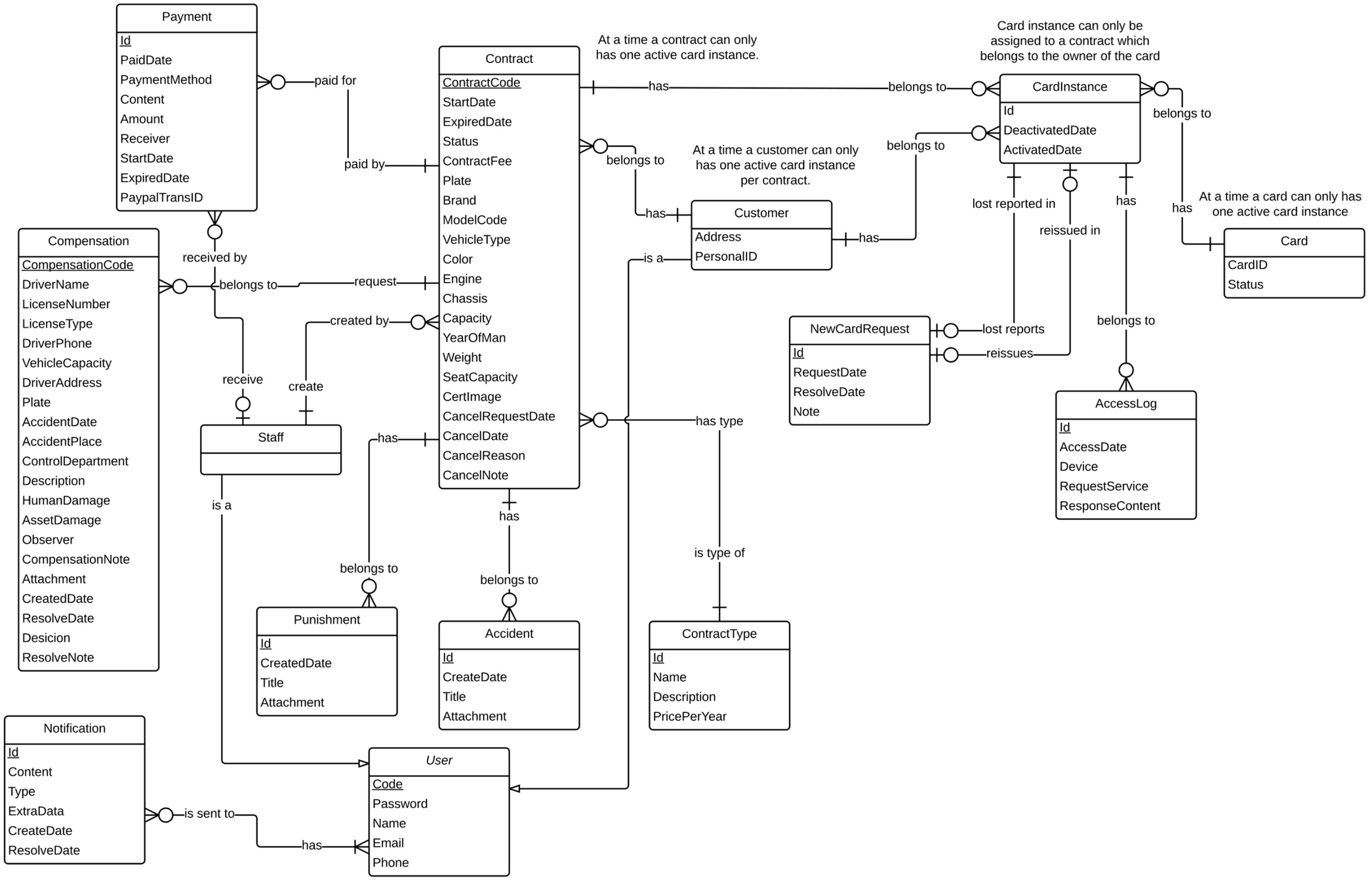


Figure 58 Entity relationship diagram

This page is intentionally left blank

6.2. Entity Dictionary

Entity Data Dictionary: describe content of all entities	
Entity name	**Description**
User	Abstract entity describes a user in system
Customer	Contain the customer information.
Contract	Contain the contract information.
Card	Contain the card information
CardInstance	Represent a card assigned to a contract
Payment	Contain the payment information.
Staff	Contain the staff information.
Compensation	Contain the compensation information.
Punishment	Contain the punishment information.
Accident	Contain the accident information.
ContractType	Contain the contract type information.
NewCardRequest	Contain the new card request information.
CardAccessLog	Contain the new card access log.
Notification	Contain the notification information

Table 79 Entity dictionary

7. Algorithms

7.1. Contract State

The contracts in MIC system is complex and can be managed differently during the operation. The state chart bellow describes all the state of a contract.

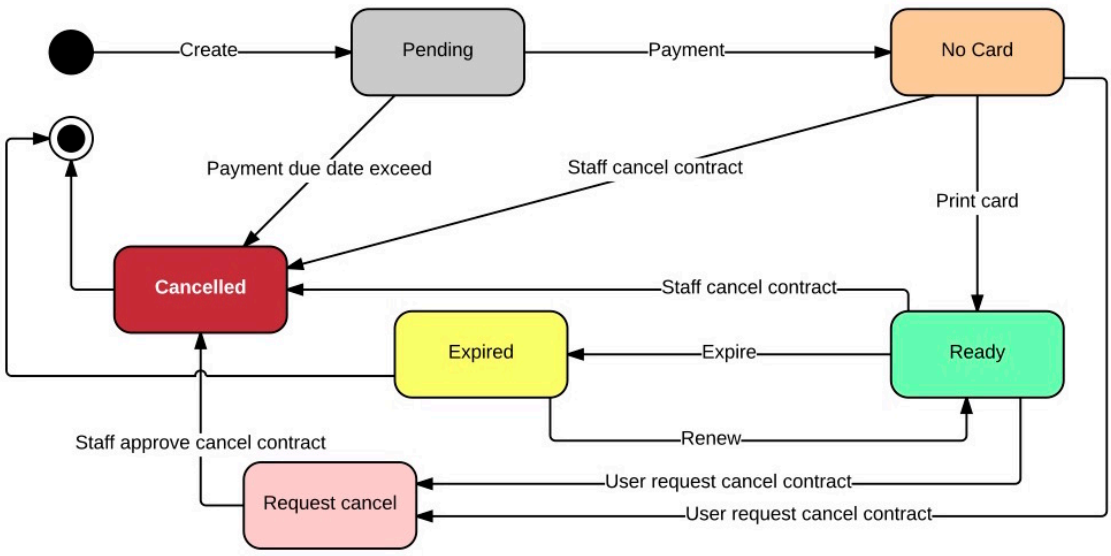


Figure 59 Contract State Chart

State	Description

Pending	The contract is created and do not have payment or the start date is not come yet
No card	The contract had have payment and contract start date is arrived but have not assigned to a card
Ready	The contract is assigned with a card and ready to use
Expired	The contract due date is exceed and no longer valuable
Request cancel	The contract is requested to cancel by customer
Cancelled	The contract is cancelled and no longer valuable

Table 80 Contract State Dictionary

Stimulus	Description	State before	State after
Create	Customer create new contract	N/A	Pending
Payment	Customer pay for the contract via PayPal or direct payment and contract start date is arrived.	Pending	No card
Payment due date exceed	When the payment due date is exceed	Pending	Cancelled
Print card	Staff use mobile app to print the card for customer, the card ID is saved to the system	No card	Ready
Expire	When the contract due date is exceed, system will change the contract status	Ready	Expired
Renew	Customer renew contract via web application or direct payment	Expired	Ready
Staff cancel contract	Staff cancel contract via web application	No card, Ready	Cancelled
Customer cancel contract	Customer cancel contract via web application	No card, Ready	Request cancel
Staff approve cancel contract	Staff approve cancel request from customer via web application	Request cancel	Cancelled

Table 81 Contract State Flow

7.2. Concurrency Control

7.2.1. Definition

Concurrency control is a method to handle data which can be accessed and modified from many sources at the same time to prevent conflict and data loss.

7.2.2. Define problem

In web application, one contract might be edited by stakeholders at the same time, this situation may cause an inconsistent state of the contract. We have to find a solution for this problem.

7.2.3. Solution

We defined a list of entity that might be modified by many sources at a same time:

- **Contract**: can be edited/renew by stakeholders at a same time. This includes the information assigned with the contract: Payment, Compensation, Accident and Punishment information.

We use timestamp concurrency control method to solve the concurrency problem. Contract information have a "last modified" value, this value represents the last time the contract is modified. When update the contract information, a validation will run to check if the last modified is changed or not. If it is changed there will be error message to users.

7.2.4. Flow chart

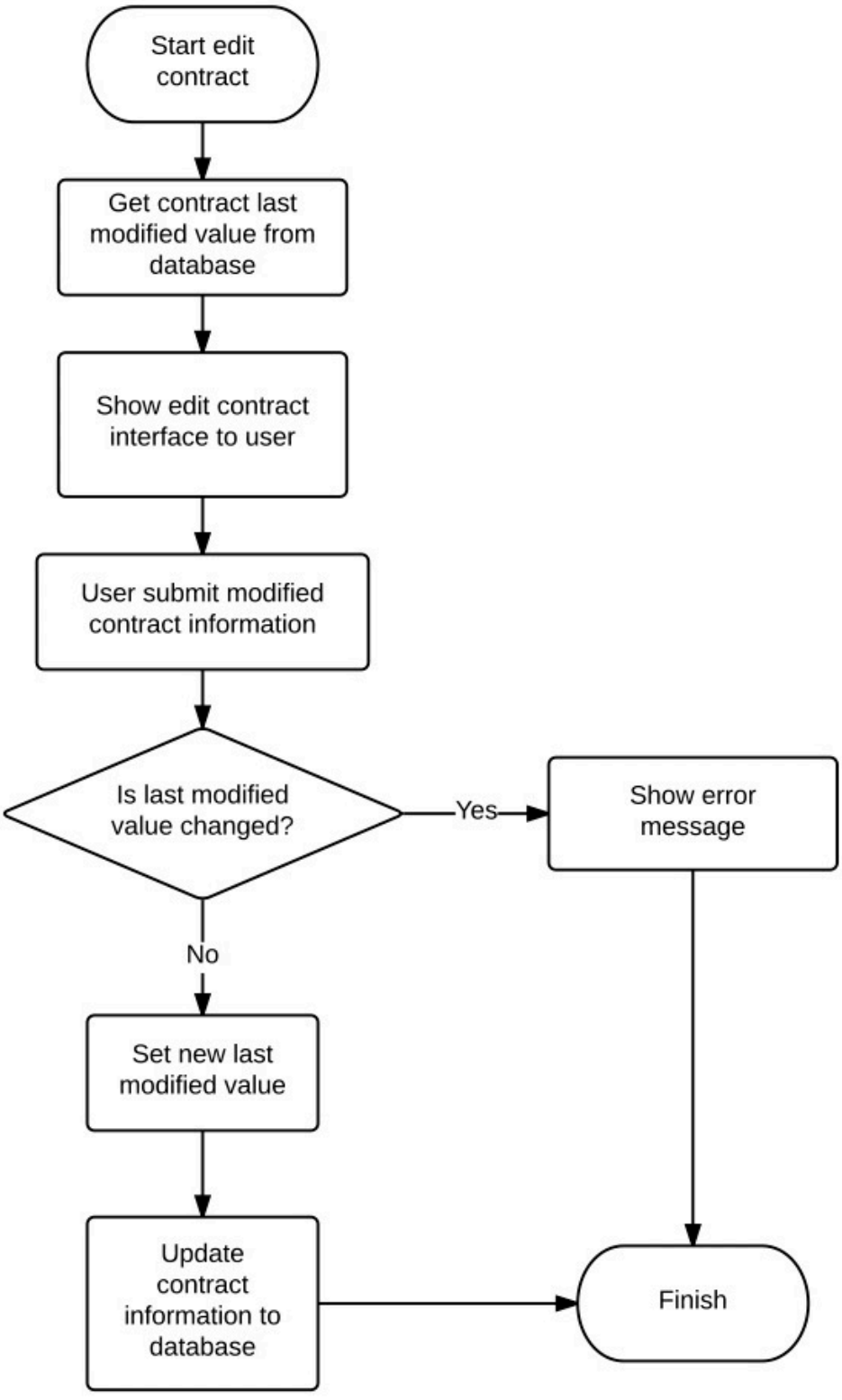


Figure 60 Concurrency control flow

7.3. Notification

7.3.1. Definition

Notification is a feature of MIC Web application to notify stakeholders when important events occurs.

7.3.2. Notification Methods

- Send email to receiver
- Show notification icon on web interface

7.3.3. Notification use cases

Type	Trigger	Receiver	Notify method
1	Customer creates new contract	All staff	Web
2	Customer sends compensation request	All staff	Web
3	Customer sends new card request	All staff	Web
4.1	Contract is nearly expired (send first)	Customer	Web, Email
4.2	Contract is nearly expired (send 2nd time)	Customer	Web, Email
4.3	Contract is nearly expired (send 3rd time)	Customer	Web, Email
5	Contract is expired	Customer	Web, Email
6	Customer sends request cancel	All staff	Web
7	Contract is cancelled because payment due date exceed	Customer	Web, Email
8	Paid Pending Contract start date come	Customer	Web
9	Compensation resolved	Customer	Web, Email

Table 82 Notification use cases

- For staffs, every notification has "status" to show that if the notification not resolved or when the notification is resolved and by whom. Except Type 1 notification has no status.
- All the notification can be handedly turned off by clicking Mark as read button
- When a notification is sent to all the staff, each staff has their own "Read" status. That's mean if staff A mark the status as read but staff B still see it as not read.

7.4. System Scheduler Process

7.4.1. Definition

System scheduler is a component of the Web application, this component is responsible for checking the changes from web application and updates information day by day.

7.4.2. Define problem

In Web application, the we need a system scheduler that runs every day at 00:00 to check the status of contracts, send notification via web, emails, changes the contract status if the due date is exceed… etc.

7.4.3. Solution

We create a scheduler in operating system that run on the same server of the Web application, scheduler will automatically run at specific time that system administrator defines when deploy the system. The scheduler will access to database to check the status and update information to database.

The checking process is described as follow:

1. Get all the contract from the system.
2. With each contract, check following information:
    a. Contract status
    b. Contract expired date
    c. Contract renew due date
    d. Contract payment due date
    e. Compensation requests
    f. New card requests
    g. Cancel request
3. Update the contract state follow the business rules.

7.4.4. Flow chart

The following flow chart diagram describes the process of the System scheduler every time it runs.

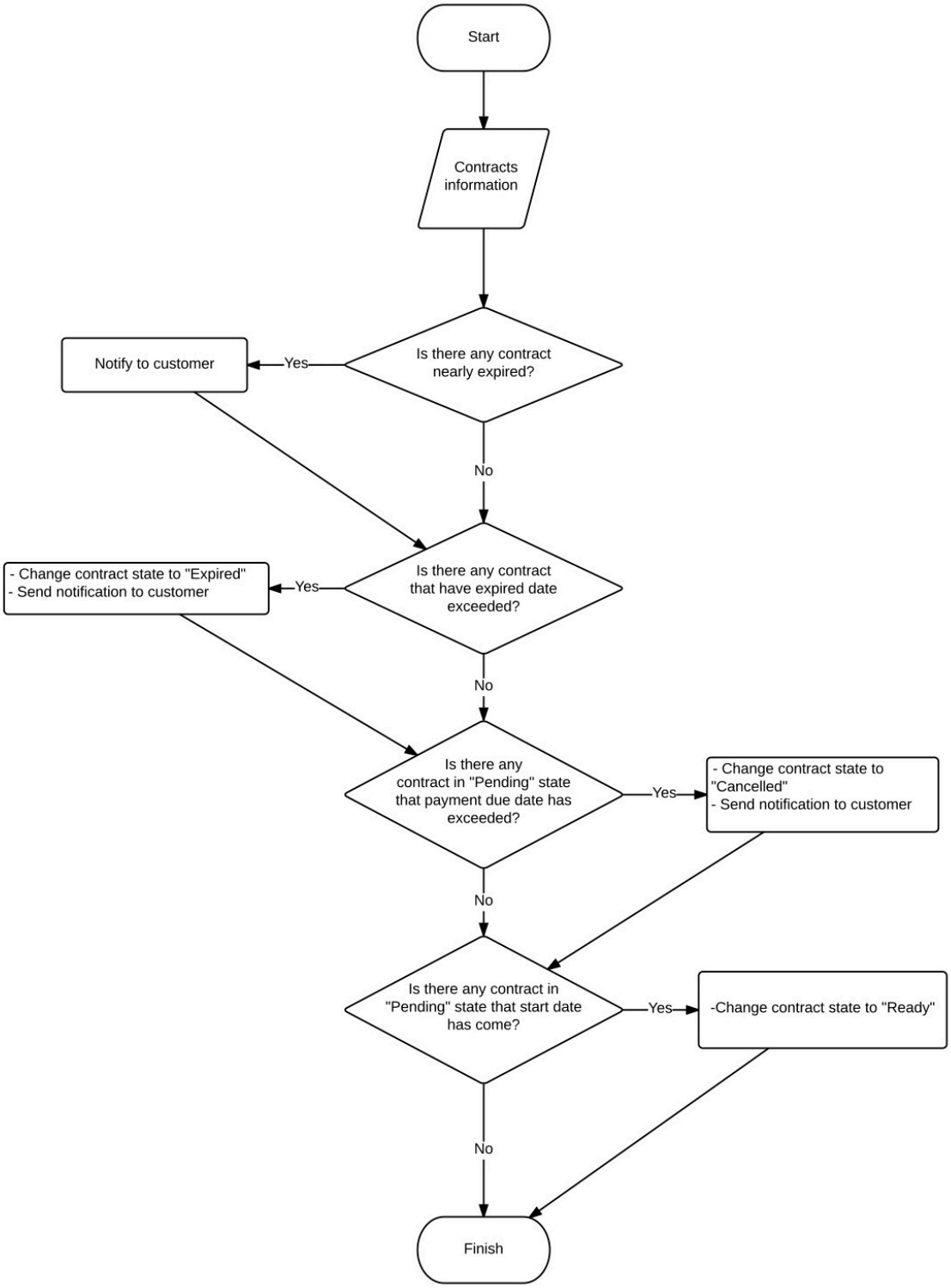


Figure 61 System Scheduler Process

7.5. NFC Card Data Format

7.5.1. Definition

According to Android Developer Forum definition, Near Field Communication (NFC) is a set of short-range wireless technologies, typically

requiring a distance of 4cm or less to initiate a connection. NFC allows you to share small payloads of data between an NFC tag and an Android-powered device, or between two Android-powered devices. NFC Card Data format is a type of format to store data in the card's storage.

7.5.2. Define problem

The data stored in the tag can be written in a variety of formats, in Printer application and Checker application we use a NFC Forum standard called NDEF (NFC Data Exchange Format) which is highly recommended from Android developer community.

According to *NFC Forum™ NDEF 1.0 Specification (2006-07-24)*, NDEF Messages is stored in tag in this structure:

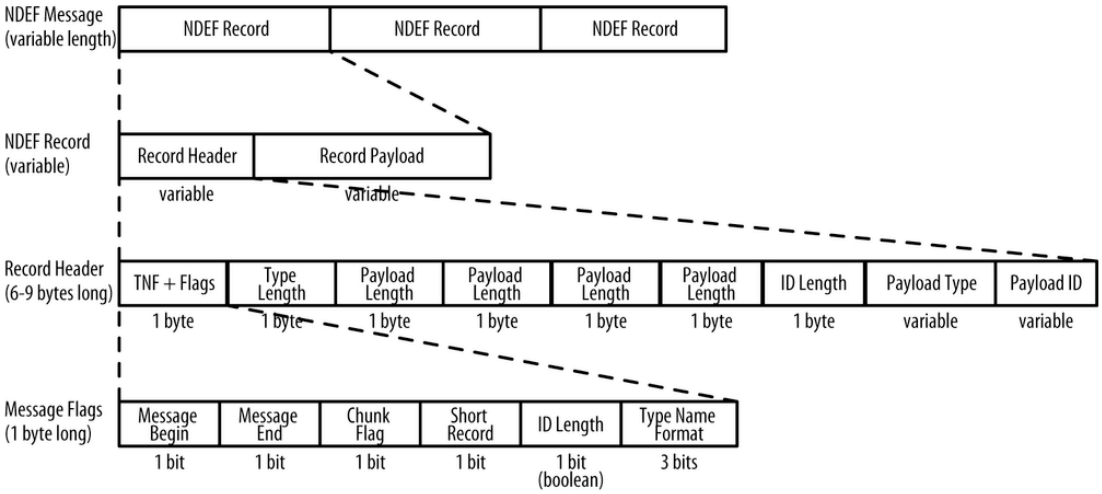


Figure 62 NDEF message structure

Reference: Brian Jepson, Don Coleman, Tom Igoe. (January 2014). *Beginning NFC, Chapter 4 - Introduction to NDEF, NDEF Structure*. O'Reilly Media, Inc.

NDEF supports followings field types of message:

Type Name Format	Description
Empty	Empty record that has no type or payload
Well-known	One of several pre-defined types laid out in the NFC Forum RTD specification.
MIME media-type	An Internet media type as defined in RFC 2046.
Absolute URI	A URI as defined in RFC 3986.
External	A user-defined value, based on rules in the NFC Forum Record Type Definition specification.
Unknown	Type is unknown. Type length must be 0.

Unchanged	Only for middle and terminating records of chunked payloads. Type length must be 0.
Reserved	Reserved by the NFC Forum for future use.

Table 83 NDEF Message Types

Reference: NFC Specification, Appendix A. NFC Specification Codes.

To prevent malicious users to override data on the NFC card or using fake card, we need to find a solution to protect data written on the card.

7.5.3. Solution

For security reason, we decided to not to write any contract information data to the tag but use only the card low-level ID to verify the card.

Card low-level ID, as known as Manufacture ID (IDm) is an 8-byte number that is used by the NFC Forum Device to address a Type 3 NFC Tag.

Reference: *NFC Forum™ Type 3 Tag Operation Specification 1.1, 2011-06-28*.

To make the card ID more friendly to users, we decided to convert the IDm to a series of 2-digit hex number which turn the 8-byte number into a string follow this regex format:

```
[0-9A-F]{16}
```

Example: 561F3ADF328D9C4B.

To make sure the card can only be read by our Checker application, we write an Android Application Record (AAR) with the package name of the Checker Mobile application.

In this project we have to use NFC card Type 4, which supports IDm and NDEF data format. Other type of NFC card might result different IDm.

Bellow is the flow to write an NFC tag for customer:

1. Printer application get contract information
2. Staff confirm the contract information is correct
3. Printer read card ID
4. If the card is not existing in system, update the card ID to the contract
5. Write AAR record to the tag and finish.

Bellow is the flow to read an NFC tag for police officer:

1. Checker application read card ID from the card
2. Checker application send card ID to system to verify the card
3. Show result to police officer

7.6. Strategy for future expand plan

MIC system is currently an application to deploy to a single company with single contract type. Our vision for the future plan is to deploy MIC system to multiple companies and with multiple types of contract (motor insurance, health insurance, asset insurance…).

7.6.1. Isolated Single Service

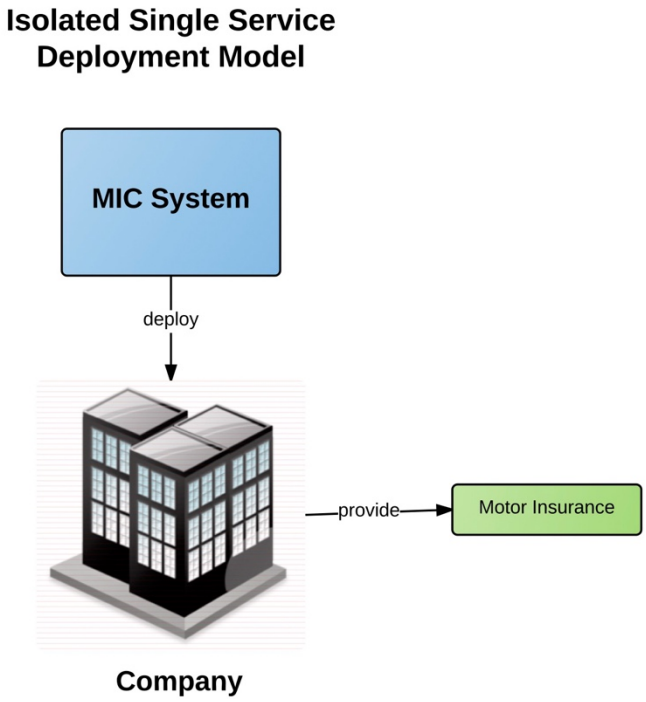


Figure 63 Isolated Single Service Deployment Model

Based on the original requirement, this is the model of MIC system which support manage motor insurance contract. The MIC System will be deployed to a company which provide a single service: motor insurance service.

As a part of the implement process, to obtains the scalability of the system, this is what we designed the components:

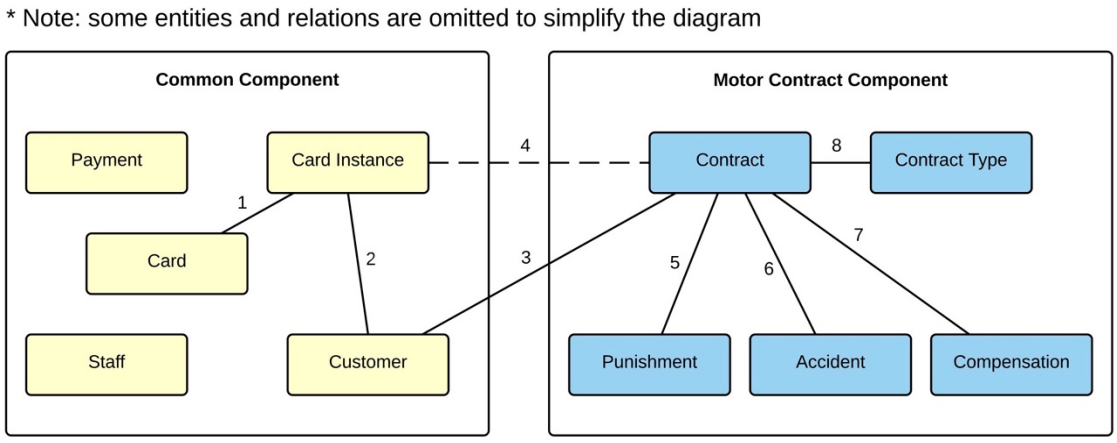


Figure 64 Entities group by components (Isolated Single Service)

From the diagram, we can see the Motor Contract Component is separated with the Common Component which contains common information such as Customer, Card, Payment… This have benefits that if we want to add a new contract type (for example heath insurance contract), the cost of modify the old system is minimize to nearly zero.

Next, we will show the process to expand this system to the new system that supports multiple service.

7.6.2. Isolated Multiple Service

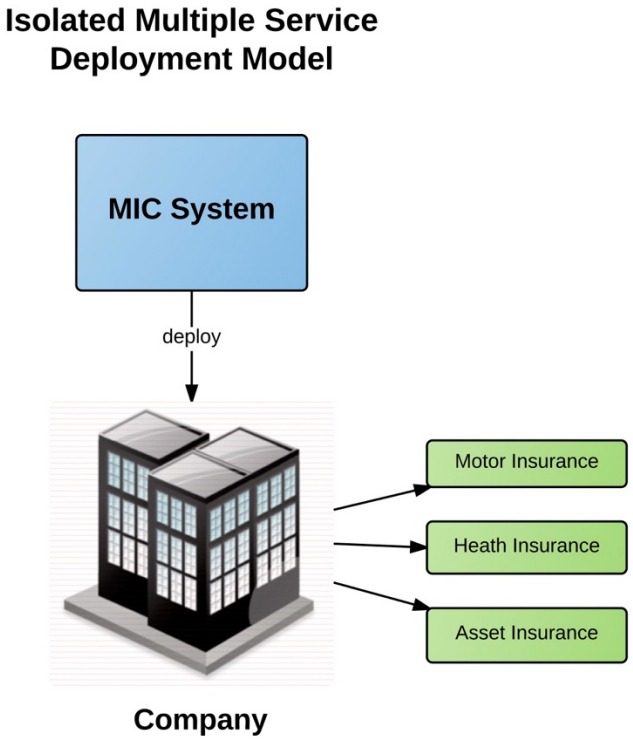


Figure 65 Isolated Multiple Service Deployment Model

This is an expanded MIC system where a company can use MIC System to manage multiple contract type for example motor insurance, health insurance and asset insurance.

Based on the original model, we can implement this model by simply add new component to the current system, this is the planned component and relationships between entities:

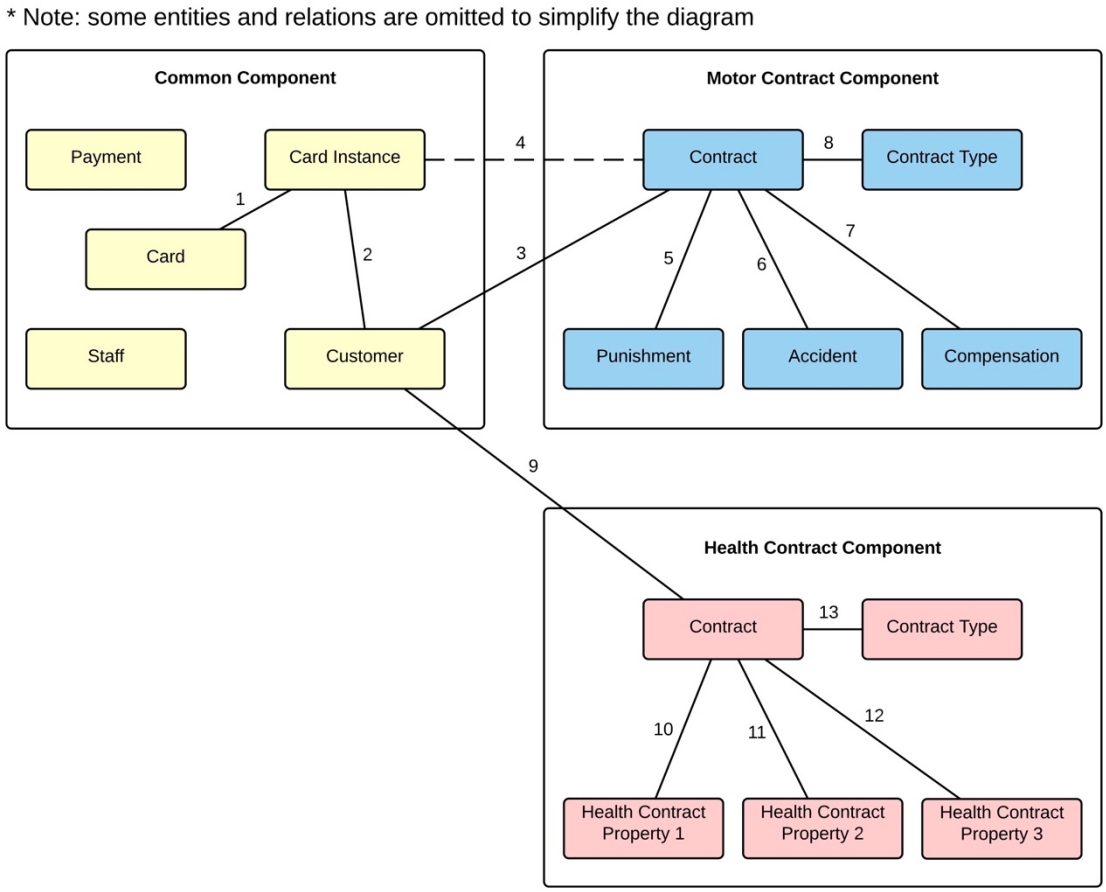


Figure 66 Entities group by components (Isolated Multiple Service)

7.6.3. Distributed Multiple Service

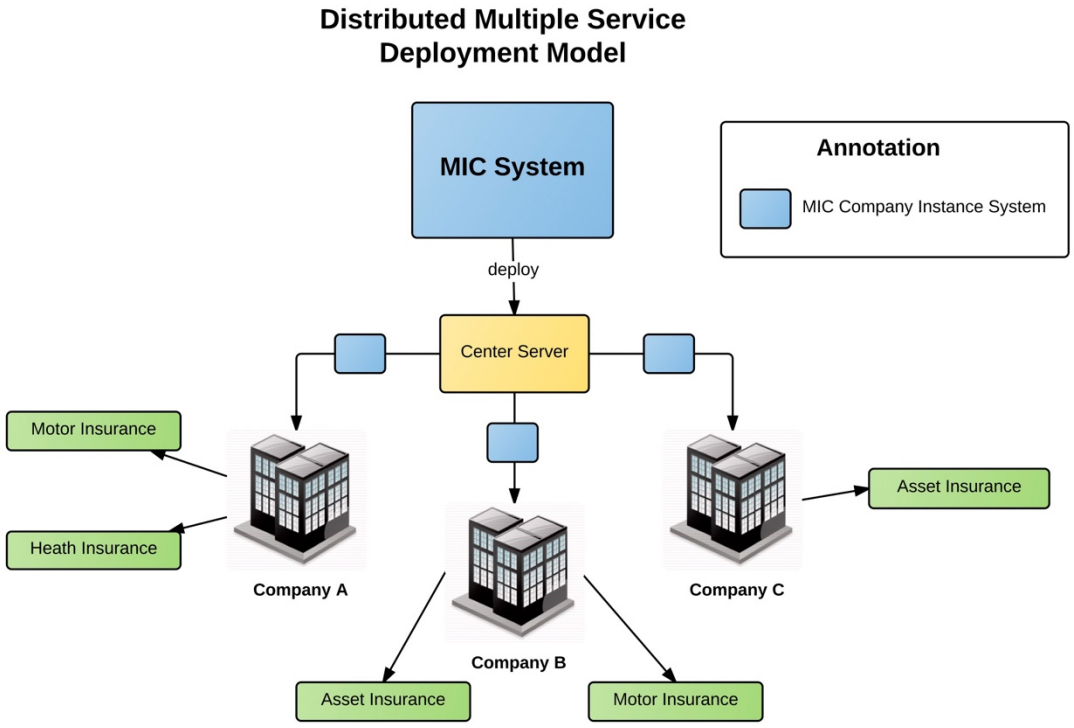


Figure 67 Distributed Multiple Service Deployment Model

This is the final model of MIC system from our vision. In this model, we deploy MIC System in a Center Server which operates multiple insurance company, each company can provide different types of contract (motor insurance, health insurance, asset insurance…). All the company use a unify card system and customer information. By this model, a customer can have multiple contracts from different companies using only one NFC card to use for all the services.

This is the expected deployment component we estimated:

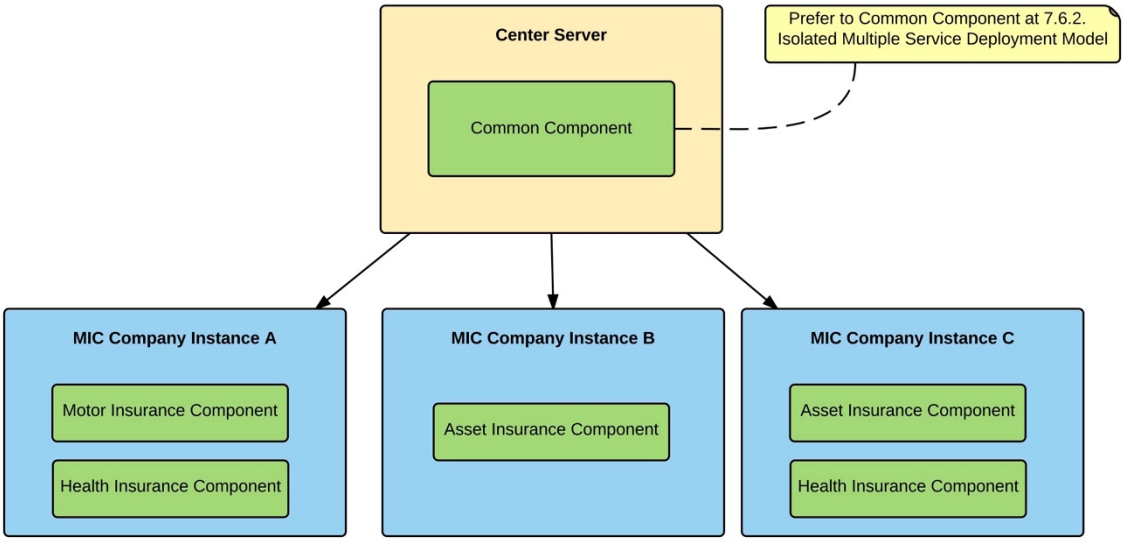


Figure 68 Components group by container (Distributed Multiple Service)

The Distributed Multiple Service plan is still need to be investigate a lot to clarify the components and relationships between entities as well as the architecture design to maintain the best performance and less impact to the current system, we also need to identify the arise problems when switching from isolated model to distributed model.

E. System Implementation & Test

1. Introduction

1.1. Overview

This section describes the approach and methodologies used by group to plan, organize and manage the testing of MIC system. It provides in the detail all necessary information about the implementation and testing procedure of the system included test plans, test cases, test result, test environments, pass/fail criteria and risks estimations as well as a checklist to cover all possible cases.

1.2. Test Approach

- Goal: Test all features in the whole MIC system based on the core flow.
- Method: black-box testing
- Technique: check list

The testing for this project will consists of Integration System test level. Testing the program which was integrated and as a complete system to ensure that the software requirements have been met.

- Integration testing would be performed by all member of team and approved by team leader.
- System testing is focused on assessing the system's reliability. This process is concerned with finding errors that result from unanticipated interactions between components and component interface problems.

2. Database Relationship Diagram

2.1. Physical Diagram

This page is intentionally left blank

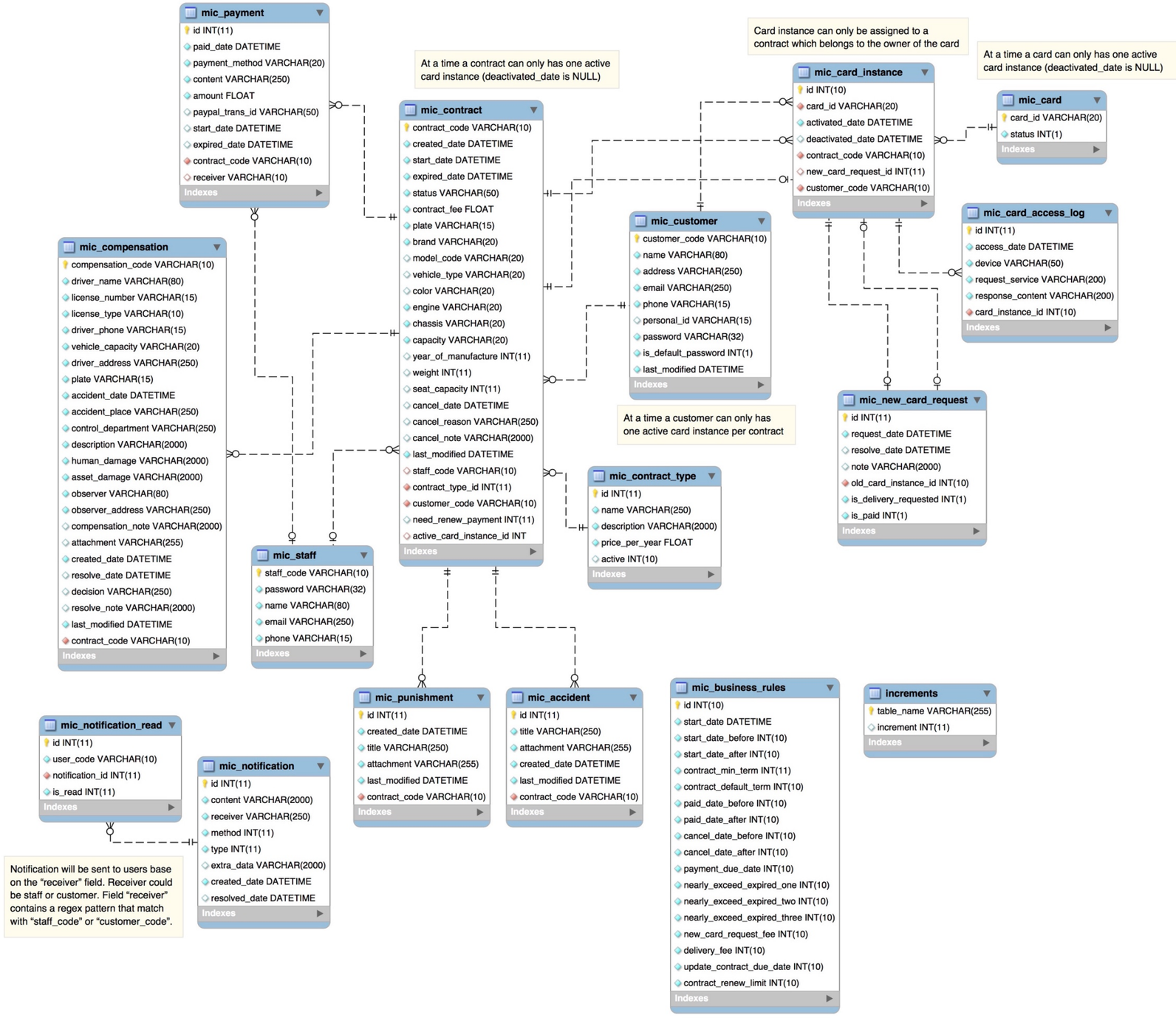


Figure 69 Physical diagram

This page is intentionally left blank

2.2. Data Dictionary

Data dictionary: describe content of all tables		
Table Name	**Mapping with Conceptual diagram**	**Description**
increments	N/A	Not exist in conceptual diagram. Needed in physical diagram to generate codes with prefixes sequentially.
mic_business_rules	N/A	Not exist in conceptual diagram. Needed in physical diagram to describe all configuration based on company's business rules.
mic_contract_type	ContractType	Describe all contract type in the system.
mic_contract	Contract	Describe all contract detail information in the system.
mic_accident	Accident	Describe all accident information belongs to contract in the system.
mic_compensation	Compensation	Describe all compensation detailed information belongs to contract in the system.
mic_punishment	Punishment	Describe all punishment information belongs to contract in the system.
mic_card	Card	Describe all card, which has been issued in the system.
mic_card_instance	CardInstance	Describe all card instance.
mic_card_access_log	N/A	Not exist in conceptual diagram. Needed in physical diagram to describe all history access to card.
mic_new_card_request	NewCardRequest	Describe all new card request in the system.
mic_customer	Customer	Describe all customer profile available in the system.
mic_notification	N/A	Not exist in conceptual diagram. Needed in physical diagram to describe all notification detail in the system.
mic_notification_read	N/A	Not exist in conceptual diagram. Needed in physical diagram to support manage all notification in the system.
mic_payment	Payment	Describe all payment information in the system.

mic_staff	Staff	Describe all staff profile available in the system.

Table 84 Data table dictionary

Table Name	Attributes	Description	Domain	Null
increments	table_name {PK}	Unique identifier of increments	varchar(255)	No
	increment			Yes
mic_business_rules	id {PK}	Unique identifier of business rules	int(10)	No
	start_date_before	Duration contract start before current date	int(10)	No
	start_date_after	Duration contract start after current date	int(10)	No
	contract_min_term	Minimum contract term	int(11)	No
	contract_default_term	Default contract term	int(10)	No
	paid_date_before	Duration pay for contract before current date	int(10)	No
	paid_date_after	Duration pay for contract after current date	int(10)	No
	cancel_date_before	Duration cancel contract before current date	int(10)	No
	cancel_date_after	Duration cancel contract after current date	int(10)	No
	payment_due_date	Duration pay for contract from create date	int(10)	No
	nearly_exceed_expired_one	Duration first time notify contract is going to expired	int(10)	No
	nearly_exceed_expired_two	Duration second time notify contract is going to expired	int(10)	No
	nearly_exceed_expired_three	Duration third time notify contract is going to expired	int(10)	No
	new_card_request_fee	The amount has to paid for new card	int(10)	No
	delivery_fee	The amount has to paid for delivery new card	int(10)	No
mic_contract_type	id {PK}	Unique identifier of contract type, auto increment.	int(11)	No
	name	Contract type name.	varchar(250)	No

	description	Contract type description.	varchar(2000)	No
	price_per_year	Contract type price per year.	float	No
mic_contract	contract_code {PK}	Unique identifier of contract.	varchar(10)	No
	created_date	Contract created date	datetime	No
	start_date	Contract start date	datetime	No
	expired_date	Contract expired date	datetime	No
	status	Contract status	varchar(50)	No
	contract_fee	Contract fee	float	No
	plate	Vehicle plate	varchar(15)	No
	brand	Vehicle brand	varchar(20)	No
	model_code	Vehicle model code	varchar(20)	Yes
	vehicle_type	Vehicle type	varchar(20)	Yes
	color	Vehicle color	varchar(20)	Yes
	engine	Vehicle engine number	varchar(20)	No
	chassis	Vehicle chassis nuber	varchar(20)	No
	capacity	Vehicle capacity	varchar(20)	No
	year_of_manufacture	Vehicle year of manufacture	int(11)	Yes
	weight	Vehicle empty weight	int(11)	Yes
	seat_capacity	Vehicle seat capacity	int(11)	Yes
	cancel_date	Contract cancel date	datetime	Yes
	cancel_reason	Contract cancel reason	varchar(250)	Yes
	cancel_note	Contract cancel note	varchar(2000)	Yes
	last_modified	Contract last modified	datetime	No
	staff_code	Code of the staff who created this contract	varchar(10)	
	contract_type_id	Contract type id	int(11)	No
	customer_code	Code of the customer who own this contract	varchar(10)	No
	active_card_instance_id	Active card instance of the contract	int(11)	Yes
mic_accident	id {PK}	Unique identifier of accident, auto increment.	int(11)	No
	title	Accident title	varchar(250)	No
	attachment	Accident attachment	varchar(255)	No
	created_date	Accident created date	datetime	No
	last_modified	Accident last modified	datetime	No
	contract_code	Code of the contract has this accident	varchar(10)	No
mic_compensation	compensation_code {PK}	Unique identifier of compensation	varchar(10)PK	No
	driver_name	Compensation driver name	varchar(80)	Yes

	license_number	Compensation driver license number	varchar(15)	Yes
	license_type	Compensation driver license type	varchar(10)	Yes
	driver_phone	Compensation driver phone	varchar(15)	Yes
	vehicle_capacity	Compensation driver vehicle capacity	varchar(20)	Yes
	driver_address	Compensation driver address	varchar(250)	Yes
	plate	Compensation driver vehicle plate	varchar(15)	Yes
	accident_date	Compensation accident date	datetime	Yes
	accident_place	Compensation accident place	varchar(250)	Yes
	control_department	Department which control this accident	varchar(250)	Yes
	description	Compensation description	varchar(2000)	Yes
	human_damage	Accident human damage	varchar(2000)	Yes
	asset_damage	Accident asset damage	varchar(2000)	Yes
	observer	Observer name	varchar(80)	Yes
	observer_address	Observer address	varchar(250)	Yes
	compensation_note	Compensation note	varchar(2000)	Yes
	attachment	Compensation attachment	varchar(255)	Yes
	created_date	Compensation created date	datetime	No
	resolve_date	Compensation resolve date	datetime	Yes
	decision	Compensation decision	varchar(250)	Yes
	resolve_note	Compensation resolve note	varchar(2000)	Yes
	last_modified	Compensation last modified time	datetime	No
	contract_code	Code of the contract has this compensation	varchar(10)	No
mic_punishment	id {PK}	Unique identifier of punishment, auto increment.	int(11)	No
	title	Punishment title	varchar(250)	No
	attachment	Punishment attachment	varchar(255)	No
	created_date	Punishment created date	datetime	No

	last_modified	Punishment last modified	datetime	No
	contract_code	Code of the contract has this punishment	varchar(10)	No
mic_card	card_id	Unique identifier of card	varchar(20)	No
	status	Status of the card	int(1)	No
mic_card_instance	Id {PK}	Unique identifier of card instance.	int(11)	No
	card_id	Card ID of the instance	varchar(20)	No
	activated_date	Card activated date	datetime	No
	deactivated_date	Card deactivated date	datetime	Yes
	contract_code	Code of contract has this card	varchar(10)	No
	new_card_request_id	New card request id	int(11)	Yes
	customer_code	Customer code, owner of the card	varchar(10)	No
mic_card_access_log	id {PK}	Unique identifier of card access log, auto increment.	int(11)	No
	access_date	Access card date	datetime	No
	device	Access card device	varchar(50)	No
	request_service	Service requested	varchar(200)	No
	response_content	Content responsed	varchar(200)	No
	card_id	Card ID	varchar(20)	No
mic_new_card_request	id {PK}	Unique identifier of new card request, auto increment.	int(11)	No
	request_date	Send request date	datetime	No
	resolve_date	Resolve request date	datetime	Yes
	note	Resolve request note	varchar(2000)	Yes
	old_card_id	ID of old card	varchar(20)	No
mic_customer	customer_code{PK}	Unique identifier of customer	varchar(10)	No
	name	Customer name	varchar(80)	No
	address	Customer address	varchar(250)	No
	email	Customer email	varchar(250)	No
	phone	Customer phone	varchar(15)	No
	personal_id	Customer personal ID	varchar(15)	Yes
	password	Customer password	varchar(32)	No
	is_default_password	Value to check customer password is default or not.	int(1)	No
	last_modified	Customer last modified time	datetime	No

mic_notification	id {PK}	Unique identifier of notification, auto increment.	int(11)	No
	content	Notification content	varchar(2000)	No
	receiver	Notification receiver	varchar(250)	No
	method	Notification method	int(11)	No
	type	Notification type	int(11)	No
	extra_data	Notification extra data	varchar(2000)	Yes
	created_date	Notification created date	datetime	No
	resolved_date	Notification resolved date	datetime	Yes
mic_notification_read	id {PK}	Unique identifier of notification read, auto increment.	int(11)	No
	user_code	Code of user read this notification	varchar(10)	No
	notification_id	Notification ID	int(11)	No
	is_read	Value to check notification is read or not yet.	int(11)	No
mic_payment	id {PK}	Unique identifier of payment, auto increment.	int(11)	No
	paid_date	The date the payment is paid	datetime	No
	payment_method	Payment method	varchar(20)	No
	content	Payment content	varchar(250)	No
	amount	Payment amount	float	No
	paypal_trans_id	PayPal transaction ID	varchar(50)	Yes
	start_date	Contract start date while payment content is renew	datetime	Yes
	expired_date	Contract expired date while payment content is renew	datetime	Yes
	contract_code	Code of contract has this payment	varchar(10)	No
	receiver	Payment receiver	varchar(10)	Yes
mic_staff	staff_code {PK}	Unique identifier of staff	varchar(10)	No
	password	Staff password	varchar(32)	No
	name	Staff name	varchar(80)	No
	email	Staff email	varchar(250)	No
	phone	Staff phone	varchar(15)	No

Table 85 Data attribute dictionary

3. Performance Measures

3.1. Web application page load speed

3.1.1. Definition

This section tests the general page load speed from all the page of the web application.

3.1.2. Test environment

Server:

- **Operating system:** Ubuntu 14.04 Server 32 bit
- **RAM:** 512 MB
- **Storage:** 40 GB
- **Processor:** 2.6 GHz Intel Core i5
- **Network:** 1 Mbps

3.1.3. Test cases

Using browser from client side and go to each page of the web application. Record the page load time using Chrome Dev Tools at client side.

Test pages:

Public	**home**
	register
	checkout
	login
Staff	dashboard
	contract: - list contract - detail contract - create contract
	compensation: - list all compensation - list compensation by contract - detail compensation - create compensation - edit compensation
	punishment - list punishment by contract - create punishment - edit punishment
	accident - list accident by contract - create accident - edit accident
	card - list card - detail card - list new card request
	customer

	- list customer - customer detail - create customer - edit customer information
Customer	dashboard
	contract - list contract - detail contract - create new contract
	compensation: - list compensation by contract - detail compensation - create compensation
	punishment - list punishment by contract - create punishment
	accident - list accident by contract - create accident
	card - list card - detail card - list new card request
	profile information

3.1.4. Test result

The test is run 10 times, each time we go through all the pages listed above and log down the average page load time.

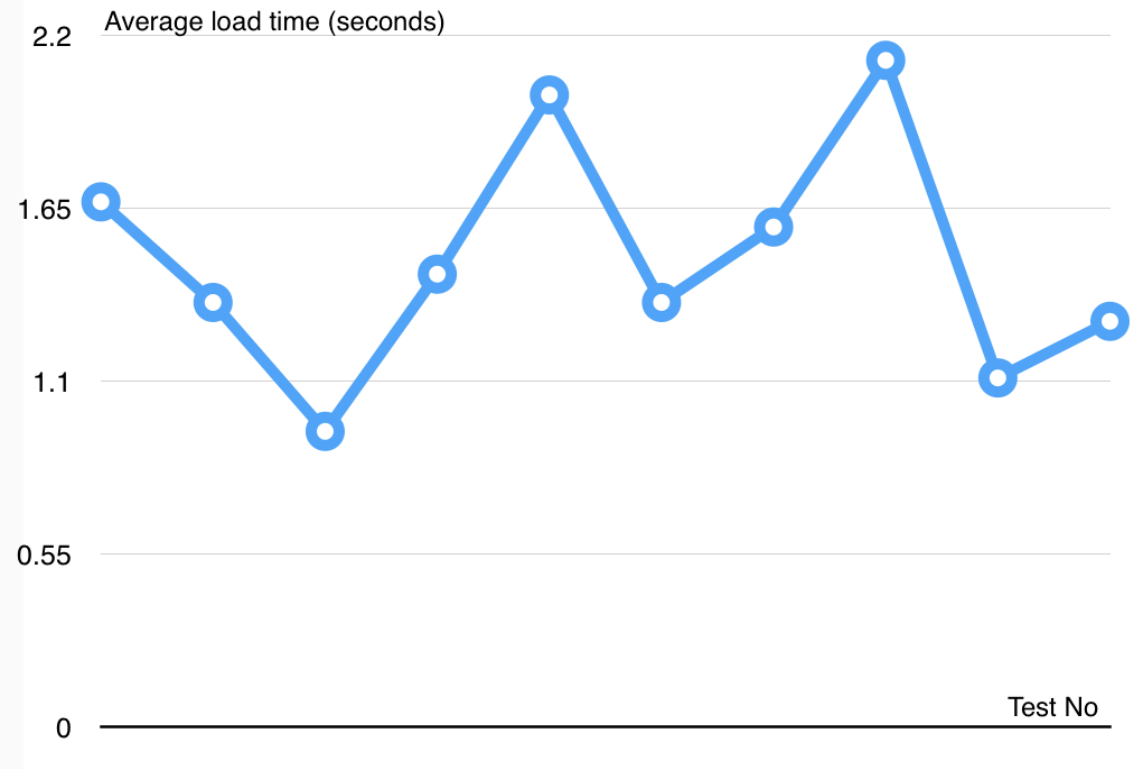


Figure 70 Web application page load speed test result

Test No.	Average page load time (second)	Execute date
1	1.67	6 July 2015
2	1.35	6 July 2015
3	0.94	6 July 2015
4	1.44	6 July 2015
5	2.01	6 July 2015
6	1.35	6 July 2015
7	1.59	6 July 2015
8	2.12	6 July 2015
9	1.11	6 July 2015
10	1.29	6 July 2015
	Average: 1.49 (seconds)	

Table 86 Web application page load speed test result

3.2. Mobile application API load speed

3.2.1. Definition

This section tests the load speed of mobile apps when connection to server through API.

3.2.2. Test environment

Server:

- **Operating system:** Ubuntu 14.04 Server 32 bit
- **RAM:** 512 MB
- **Storage:** 40 GB
- **Processor:** 2.6 GHz Intel Core i5
- **Network:** 1 Mbps

Client:

Hardware Requirements	Minimum	Recommended
Internet Connection	512Kbps	Wi-Fi Connection 12MB
Operating System	Android 4.0	Android 4.0
Hardware	NFC supported	NFC supported
Memory	128MB of RAM	1GB of RAM or more

Table 87 Mobile app API load speed client specification

3.2.3. Test cases

Using mobile application to send the entire API request to web server.

List of APIs:

No.	API	Description
1	getCheckConnection	Check server status
2	getContracts	Get contract information
3	getUpdateCardID	Update Card ID for contract
4	getCheckCard	Check card validation
5	getUpdatePunishment	Update punishment information for contract

Table 88 Mobile app API load speed test cases

3.2.4. Test result

The test is run 10 times, each time we run all the test cases listed above and log down the average API response time.

Figure 71 Mobile application API load speed test result

Test No.	Average page load time (second)	Execute date
1	2.13	6 July 2015
2	2.89	6 July 2015
3	2.21	6 July 2015
4	2.5	6 July 2015
5	2.87	6 July 2015
6	1.73	6 July 2015
7	2.17	6 July 2015
8	1.58	6 July 2015
9	1.62	6 July 2015
10	1.31	6 July 2015
	Average: 2.10 (seconds)	

Table 89 Mobile application API load speed test result

4. Test Plan

The overall purpose of testing is to ensure MIC system meets its entire technical, functional and business requirement. The purpose of this document is to describe the overall test plan and strategy for testing the MIC system. The approach described in this document provides the framework for all testing related to this application. Individual test cases will be written for each version of the application that is released. This document will also be updated as required for each release.

4.1. Features to be tested

- Staff: view contract information, view compensation information, resolve new card request, resolve compensation request, create new contract, renew contract, cancel contract.
- Customer: view contract information, view compensation information, renew contract, cancel contract, request compensation, new card request, change password.
- Guest: Create new contract request
- System: notify schedule
- Mobile application: verify card, add punishment, print card

4.2. Features not to be tested

- Admin: add staff, remove staff
- Login, logout, change profile
- View information: card, punishment, accident, customer, staff, payment.

5. System Testing Test Case

5.1. Communication Diagrams

Test cases are created from bellow communication diagram:

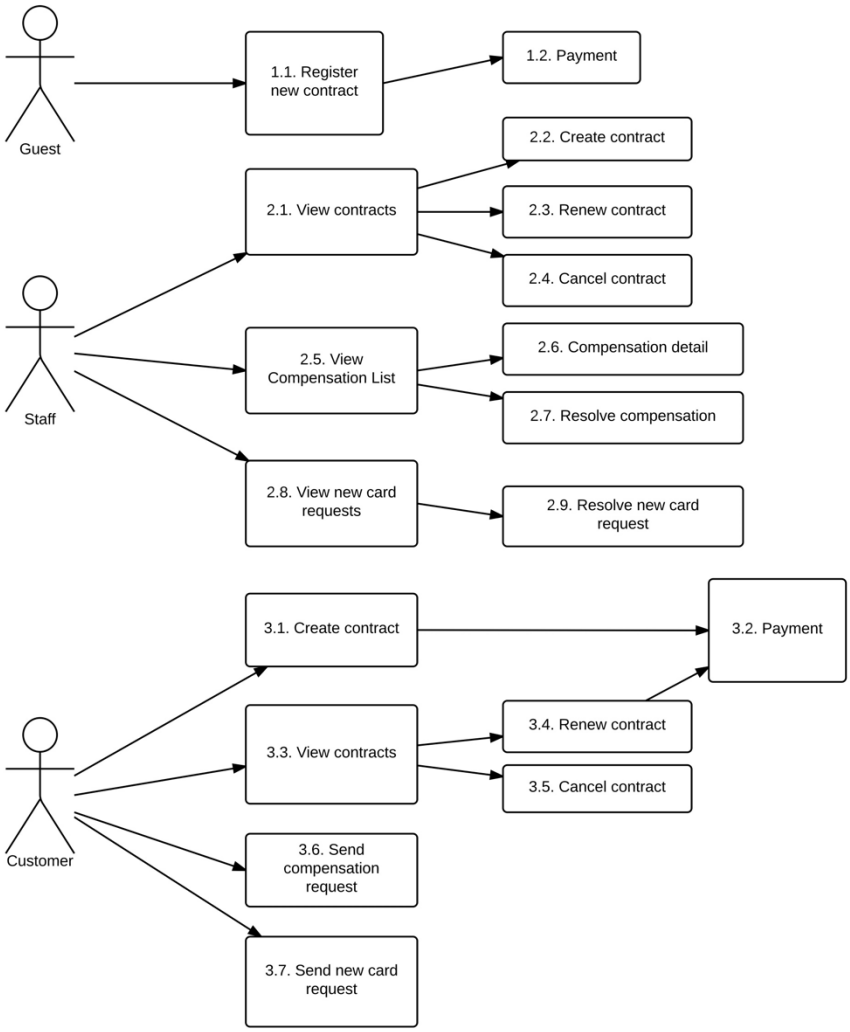


Figure 72 Web Application Communication Diagram

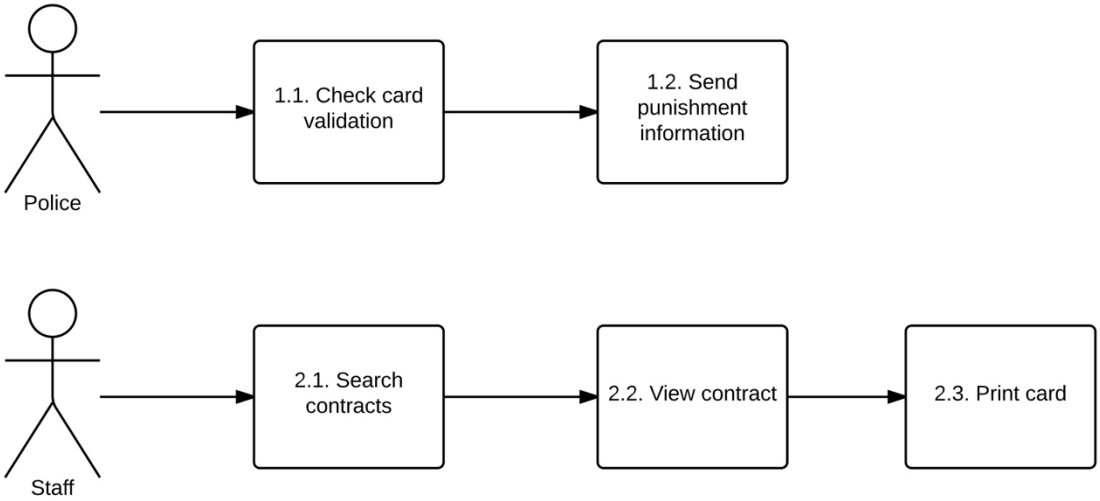


Figure 73 Mobile Applications Communication Diagram

5.2. Test cases

5.2.1. <Guest> Register new contract

ID	Test Case Description	Test case procedure	Expected output	Inter-test case dependence	Result	Test Date

RNC1	Test guest register new contract with valid information	1. Go to the home page 2. Enter personal information: - Họ tên: Nguyễn Văn A - Email: baohiem@yahoo.com - Số điện thoại:0909099099 - Số CMND/Hộ chiếu: 272185738 - Địa chỉ: 123 Chợ Lớn, Bến Thành, HCM. - Ngày bắt đầu: 13/7/2015 - Hình thức bảo hiểm: Xe trên 50cc có BH cho người trên xe 3. Click "Tiếp theo" button. 4. Click tab "2.Thông tin xe". 5. Enter vehicle information: - Biển số đăng ký:60-Y6 6666 - Nhãn hiệu: Honda - Số khung: 1001001 - Số máy: 1001001 - Dung tích: 110cc - Số loại: Air blade - Loại xe: Hai bánh - Màu sơn: Đỏ - Năm sản xuất: 2000 - Tự trọng: 100 6. Click "TIẾP THEO" button. 7. Click "TẠO HỢP ĐỒNG VÀ THANH TOÁN" button.	Create contract successful, send an email contains customer code and password to customer. Request payment		Pass	7/13/15

RNC2	Test guest register contract with an exist plate number	1. Register new contract with information like test case "RNC1". And excute payment process like test case "RNCP2". 2. Register another contract with information: - Họ tên: Nguyễn Văn B - Email: baohiem1@yahoo.com - Số điện thoại:0909888888 - Số CMND/Hộ chiếu: 123485738 - Địa chỉ: 123 Chợ Nhỏ, Bến Thành, HCM. - Ngày bắt đầu: 13/7/2015 - Hình thức bảo hiểm: Xe trên 50cc có BH cho người trên xe - Biển số đăng ký:60-Y6 6666 - Nhãn hiệu: Honda - Số khung: 1231231 - Số máy: 1231231 - Dung tích: 110cc - Số loại: Air blade - Loại xe: Hai bánh - Màu sơn: Đỏ - Năm sản xuất: 2000 - Tự trọng: 100 - Số người: 2	Show error message:"Đang có hợp đồng hiệu lực với xe có biển số này"	RNC1,RNCP2.	Pass	7/13/15

RNC3	Test guest register contract with an exist email	1. Register new contract with information like test case "RNC1". 2. Go to the home page 3. Enter personal information: - Họ tên: Nguyễn Văn B - Email: baohiem@yahoo.com - Số điện thoại:0909123123 - Số CMND/Hộ chiếu: 123456789 - Địa chỉ: 123 Chợ Bà Chiểu,Bình Thạnh, HCM. - Ngày bắt đầu: 13/7/2015 - Hình thức bảo hiểm: Xe trên 50cc có BH cho người trên xe 4. Click "Tiếp theo" button.	Show error message: "Email đã được đăng ký bởi người dùng khác"	RNC1	Pass	7/13/15

Table 90 Test case - <Guest> Register new contract

5.2.2. <Guest> Register new contract payment

ID	Test Case Description	Test case procedure	Expected output	Inter-test case dependence	Result	Test Date
RNCP1	Test guest choose direct payment for register new contract	1. Guest registered contract succeed like test case "RNC1". 2. Guest choose direct payment.	Show company address map	RNC1	Pass	7/13/15
RNCP2	Test Paypal payment successful for	1. Guest registered contract succeed like test case "RNC1". 2. Guest choose PayPal payment.	Show success message:"Thanh toán thành công"	RNC1	Pass	7/13/15

	register new contract.	3. Guest log in and pay with PayPal web page.				
RNCP3	Test guest cancel transaction while excute Paypal payment for register new contract.	1. Guest registered contract succeed like test case "RNC1". 2. Guest choose PayPal payment. 3. Guest cancel transaction	Show message to notify that guest has canceled the transaction:"Bạn đã hủy thanh toán. Xin vui lòng Đăng nhập để thanh toán lại hoặc đến thanh toán trực tiếp"	RNC1	Pass	7/13/15

Table 91 Test case - <Guest> Register new contract payment

5.2.3. <Staff> Create Contract

ID	Test case description	Test case procedure	Expected output	Inter-test case dependence	Result	Test date
CC1	Create contract success	1. Create customer "Nguyen Van A" has customer code is KH0001 2. Click create new contract, input form is displayed 3. Input all default value according to "Default_value_for_forms.xlsx" file 4. Set customer code is KH0001, contract start date to 12 July 2015 5. Click "Tạo hợp đồng" button	A contract with status is "No card" created successfully	Create Customer	Pass	7/13/15

CC2	Create contract success but not start yet	1. Create customer "Nguyen Van A" has customer code is KH0001 2. Click create new contract, input form is displayed 3. Input all default value according to "Default_value_for_forms.xlsx" file 4. Set customer code is KH0001, contract start date to 21 July 2015 5. Click "Tạo hợp đồng" button	A contract with status is "Pending" created successfully	Create Customer	Pass	7/13/15
CC3	Create contract fail because of existed vehicle plate number	1. Create customer "Nguyen Van A" has customer code is KH0001, customer "Duong Thien Hoa" has customer code is KH0002 2. Create new contract with start date is 12 July 2015, contract status is Ready, vehicle plate number is 59Y9-38482 3. Click create new contract, input form is displayed 4. Input all default value according to "Default_value_for_forms.xlsx" file 5. Set vehicle plate number is 59Y9-38482 6. Click "Tạo hợp đồng" button	A message notify this vehicle plate number is existed in the system	Create Customer CC1 - Create Contract	Pass	7/13/15
CC4	Create contract fail because of there are no customer	1. Create customer "Nguyen Van A" has customer code is KH0001 2. Click create new contract, input form is displayed 3. Input all default value according to "Default_value_for_forms.xlsx" file, customer code is KH0002 4. Set contract start date to 12 July 2015 5. Click "Tạo hợp đồng" button	A message notify this customer is not exist in the system	Create Customer	Pass	7/13/15

Table 92 Test case - <Staff> Create contract

5.2.4. <Staff> Renew Contract

ID	Test case description	Test case procedure	Expected output	Inter-test case dependence	Result	Test date
RC1	Renew contract success	1. Create new contract has start date is 10 July 2015, expired date is 1 September 2015. Contract status is "No card" 2. Click view this contract detail 3. Click "Gia hạn" button, renew contract form displayed 4. Fill out the form: - Gia hạn đến: 01/09/2016 - Ngày nộp phí: 13/07/2015 5. Click "Gia hạn hợp đồng" button	Renew contract successfully, contract status is "No card" A new payment is created with start date is 01/09/2015, expired date is 01/09/2016	CC1 - Create Contract WC1 - View Contracts	Pass	7/13/15
RC2	Renew contract success with request for new card and delivery it	1. Create new contract has start date is 9 July 2015, expired date is 13 August 2015. Contract status is "No card" 2. Print a card with code "12-34-56-78-91" for this contract. Contract new status is "Ready" 3. Click view this contract detail 4. Click "Gia hạn" button, renew contract form displayed button 5. Fill out the form: - Gia hạn đến: 13/08/2016 - Ngày nộp phí: 13/07/2015 6. Check "Cấp thẻ mới", "Vận chuyển thẻ" boxes 7. Click "Gia hạn hợp đồng" button	Renew contract successfully, its status is "No card". A new payment is created with start date is 13/08/2015, expired date is 13/08/2016 The card with code "12-34-56-78-91" is deactivated	CC1 - Create Contract Print Card WC1 - View Contracts	Pass	7/13/15

RC3	Renew contract success	1. Create new contract has start date is 6 July 2015, expired date is 12 July 2015. Contract status is "Expired" 2. Click view this contract detail 3. Click "Gia hạn" button, renew contract form displayed 4. Fill out the form: - Gia hạn đến: 13/08/2016 - Ngày nộp phí: 13/07/2015 5. Click submit button	Renew contract successfully, contract status is "No card" A new payment is created with start date is 13/07/2015, expired date is 13/07/2016	CC1 - Create Contract WC1 - View Contracts	Pass	7/13/15
RC4	Renew contract fail because of contract term is out of bound	1. Create new contract has start date is 6 July 2015, expired date is 12 August 2015. Contract status is "No card" 2. Click view this contract detail 3. Click "Gia hạn" button, renew contract form displayed 4. Fill out the form: - Gia hạn đến: 13/12/2016 - Ngày nộp phí: 13/07/2015 5. Click "Gia hạn hợp đồng" button	A message notify contract term can not exceed the limit	CC1 - Create Contract WC1 - View Contracts	Pass	7/13/15
RC5	Renew contract success with request for new card fail because this contract have no card	1. Create new contract has start date is 6 July 2015, expired date is 12 July 2015. Contract status is "Expired" 2. Click view this contract detail 3. Click "Gia hạn" button, renew contract form displayed 4. Fill out the form: - Gia hạn đến: 13/08/2016 - Ngày nộp phí: 13/07/2015 5. Check "Cấp thẻ mới" box 6. Click "Gia hạn hợp đồng" button	A message notify this contract have no card	CC1 - Create Contract	Pass	7/13/15

Table 93 Test case - <Staff> Renew contract

5.2.5. <Customer> Create New Contract

ID	Test Case Description	Test case procedure	Expected output	Inter-test case dependence	Result	Test Date
CNC1	Create new contract success	1. Click "Hợp đồng mới" button. 2. Input all default value according to Default form value.xlsx file" (Create new contract form) 3. Click "Thêm hợp đồng" 4. Click "Xác nhận" 5. Click "Thanh toán ngay bây giờ" 6. Login to paypla an pay fee	Customer can see their new contract with status "No card" and some information about compensation, punishment, accident and card	Default form value.xlsx file	Pass	7/13/15
CNC2	Create new contract fail	1. Click "Hợp đồng mới" button. 2. Click "Thêm hợp đồng"	Sytem show message error and required input value	Default form value.xlsx file	Pass	7/13/15
CNC3	Create new contract success	1. Click "Hợp đồng mới" button. 2. Input all default value according to Default form value.xlsx file" (Create new contract form) 3. Click "Thêm hợp đồng" 4. Click "Xác nhận" 5. Click "Thanh toán ngay bây giờ" 6. Click "Trực tiếp"	System shows map with detail address of company where customer can pay direct for their contract. Customer can see their new contract with status "Pending" and some information about compensation, punishment, accident and card	Default form value.xlsx file	Pass	7/13/15

CNC4	Create new contract fail	1. Click "Hợp đồng mới" button. 2. Input all default value according to Default form value.xlsx file" (Create new contract form) 3. Thời điểm có hiệu lực input "07/04/2015" 3. Click "Thêm hợp đồng"	Sytem show message error "Thời điểm có hiệu lực phải sau ngày hiện tại"	Default form value.xlsx file	Pass	7/13/15

Table 94 Test case - <Customer> Create contract

5.2.6. <Customer> Renew Contract

ID	Test Case Description	Test case procedure	Expected output	Inter-test case dependence	Result	Test Date
RC1	Renew contract success	1. Create new contract with contract code "HD0001" 2. Prepare a contract with startdate: 7/25/2014 and expireddate: 7/25/2015 and status is "Ready" 3. Click "Gia hạn" button 4. Click "Gia hạn hợp đồng" 5. Choose "Paypal" and login paypal to pay fee 6. Click "Trở về trang chi tiết hợp đồng"	System show page with message "Gia hạn thành công" and expired is updated "7/26/2016"	Create new contract	Pass	7/13/15

RC2	Renew contract success	1. Create new contract with contract code "HD0001" 2. Prepare a contract with startdate: 7/11/2014 and expireddate: 7/11/2015 and status is "Expired" 3. Click "Gia hạn" button 4. Click "Gia hạn hợp đồng" 5. Choose "Paypal" and login paypal to pay fee 6. Click "Trở về trang chi tiết hợp đồng"	System show page with message "Gia hạn thành công" and expired is updated "7/13/2016" and status change to "Ready"	Create new contract	Pass	7/13/15
RC3	Renew contract fail	1. Login with role customer 1. Create new contract with contract code "HD0001" with 2. Prepare a contract with status Expired 3. Open new browser and login with role Staff and goes into contract-detail at contract code "HD0001" 4. Click "Gia hạn hợp đồng" at browser staff 5. Do action like RC2 6 Click "Gia hạn hợp đồng" at browser customer	System will show error message "Thông tin hợp đồng đã bị sửa đổi trước đó bởi một người khác, vui lòng thực hiện lại thao tác"	Create new contract	Pass	7/13/15

Table 95 Test case - <Customer> Renew contract

5.2.7. <Customer> Send Compensation Request

ID	Test Case Description	Test case procedure	Expected output	Inter-test case dependence	Result	Test Date
SCR1	Send compensation success	1. Create contract with contract code "HD0001" 2. Click on "Hợp đồng" menu 3. Search Contract with code "HD0001" 4 Click link "HD0001" 5. Click tab "Lịch sữ bồi thường" 6. Click "Yêu cầu bồi thương" 7. Input all default value according to Default form value.xlsx file" (Create new compensation) 8. Click "Gởi yêu cầu" 9. Click "Trở về trang chi tiết hợp đồng"	System show compensation detaiil about request compensation they just send	Create new contract Default form value.xlsx file	Pass	7/13/15
SCR2	Send compensation fail	1. Create contract with contract code "HD0001" 2. Click on "Hợp đồng" menu 3. Search Contract with code "HD0001" 4 Click link "HD0001" 5. Click tab "Lịch sữ bồi thường" 6. Click "Yêu cầu bồi thương" 7. Input all default value according to Default form value.xlsx file" (Create new compensation)	System show error message "Ngày xảy ra tai nạn phải trước ngày hiện tại"	Create new contract Default form value.xlsx file	Pass	7/13/15

		8. Ngày xảy ra tai nạn input "09/13/2015" 8. Click "Gởi yêu cầu"				
SCR3	Send compensation fail	1. Create contract with contract code "HD0001" 2. Click on "Hợp đồng" menu 3. Search Contract with code "HD0001" 4 Click link "HD0001" 5. Click tab "Lịch sữ bồi thường" 6. Click "Yêu cầu bồi thương" 8. Click "Gởi yêu cầu"	System show error message "Vui lòng nhập giá trị" and required input into some field have "*" symbol	Create new contract Default form value.xlsx file	Pass	7/13/15

Table 96 Test case - <Customer> Send compensation request

Reference to full document for more test cases.

5.3. Test case results statistics

Test case	Runs	Pass rate (%)
RNC1	50	98
RNC2	50	100
RNC3	50	98
RNCP1	50	100
RNCP2	50	100
RNCP3	50	100
WC1	50	98
WC2	50	100
WC3	50	100
CC1	50	100
CC2	50	100
CC3	50	100
CC4	50	100
WCL1	50	98
WCL2	50	100
WCL3	50	100
CD1	50	100
CD2	50	98
RC1	50	100
VNCR1	50	100
VNCR2	50	100
RNCR1	50	100
CNC1	50	100
CNC2	50	98

CNC3	50	100
CNC4	50	100
RC1	50	100
RC2	50	100
RC3	50	100
CCC1	50	100
CCC2	50	100
CCC3	50	100
CCC4	50	98
CCC5	50	100
SCR1	50	100
SCR2	50	100
SCR3	50	98
SNCR1	50	100
SNCR2	50	100
SNCR3	50	100
SNCR4	50	100
SNCR5	50	100
SNCR6	50	100
SNCR7	50	98
SNCR8	50	100
SNCR9	50	100

Table 97 Test case results statistics

F. Software User's Manual

1. Installation Guide

1.1. Setting up environment at server side

Bellows are requirements for hardware and software environment to run MIC system in 10 years. The specifications are based on the dependencies requirements and performance test result from previous section of this document.

1.1.1. Hardware requirements

Hardware	Specification
Internet Connection	8 Mbps
Computer Processor	Intel® Core(TM) i5 CPU , M 460 @ 2.53GHz
Computer Memory	3GB of RAM or more
Hard Disk Drive	40GB or more

Table 98 Hardware requirements

1.1.2. Software requirements

Software	Application name / version
Operating System	Ubuntu Server 14.04.2 LTS
Java	1.7.0_79
Web Server	Apache Tomcat 7.0.52
Database	MySQL 5.6

Table 99 Software requirements

1.2. Web Application Deployment Process

1.2.1. Check environment

Check Ubuntu version: 14.04.2 LTS

```
root@MIC:~# lsb_release -a
No LSB modules are available.
Distributor ID:  Ubuntu
Description: Ubuntu 14.04.2 LTS
Release: 14.04
Codename:    trusty
```

Check Java version: 1.7.0_79

```
root@MIC:~# java -version
java version "1.7.0_79"
OpenJDK Runtime Environment (IcedTea 2.5.6) (7u79-2.5.6-
0ubuntu1.14.04.1)
OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)
```

Check MySQL version: 5.6

```
root@MIC:~# mysql --version
mysql  Ver 14.14 Distrib 5.6.19, for debian-linux-gnu
(x86_64) using  EditLine wrapper
```

Check Apache Tomcat version: 7.0.52

```
root@MIC:~# /usr/share/tomcat7/bin/version.sh
Using CATALINA_BASE:   /usr/share/tomcat7
Using CATALINA_HOME:   /usr/share/tomcat7
Using CATALINA_TMPDIR: /usr/share/tomcat7/temp
Using JRE_HOME:        /usr/lib/jvm/java-7-openjdk-amd64
Using CLASSPATH:
/usr/share/tomcat7/bin/bootstrap.jar:/usr/share/tomcat7/bin/
tomcat-juli.jar
Server version: Apache Tomcat/7.0.52 (Ubuntu)
Server built:   Jun 19 2015 08:54:46
Server number:  7.0.52.0
OS Name:        Linux
OS Version:     3.13.0-57-generic
Architecture:   amd64
JVM Version:    1.7.0_79-b14
JVM Vendor:     Oracle Corporation
```

1.2.2. Import database

Using file **database.sql** located under **Deployment** directory from this document.

```
root@MIC:~# mysql -u root -p mic_data < database.sql
```

1.2.3. Build war artifact

Under **Source code/MICWeb** directory, edit file **pom.xml** and set correct MySQL username and password at **<profiles>** tag:

```
<profiles>
    <profile>
        <id>production</id>
        <properties>
            <db.username>mysql username here</db.username>
            <db.password>mysql password here</db.password>
        </properties>
    </profile>
</profiles>
```

Under **Source code/MICWeb** directory, run bellow command to build war artifact:

```
mvn clean install –pProduction
```

A **war** artifact file will be created under **Source code/MICWeb/target** directory, we will use this file for deployment.

1.2.4. Deploy war artifact

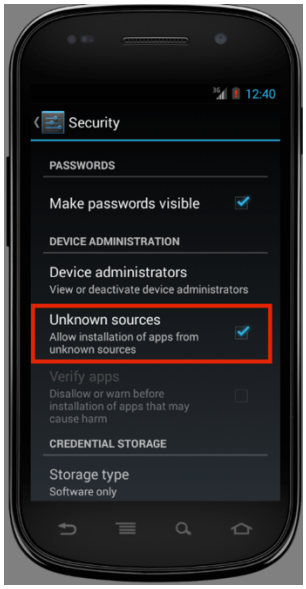
Using file **war** built from previous step.

```
root@MIC:~# mv MIC.war /var/lib/tomcat7/webapps/ROOT.war
```

Web application is now available at **http://server-ip-address**.

1.3. Mobile Application Deployment Process

At mobile device, go to Setting/Security, uncheck check box **Unknown sources**.



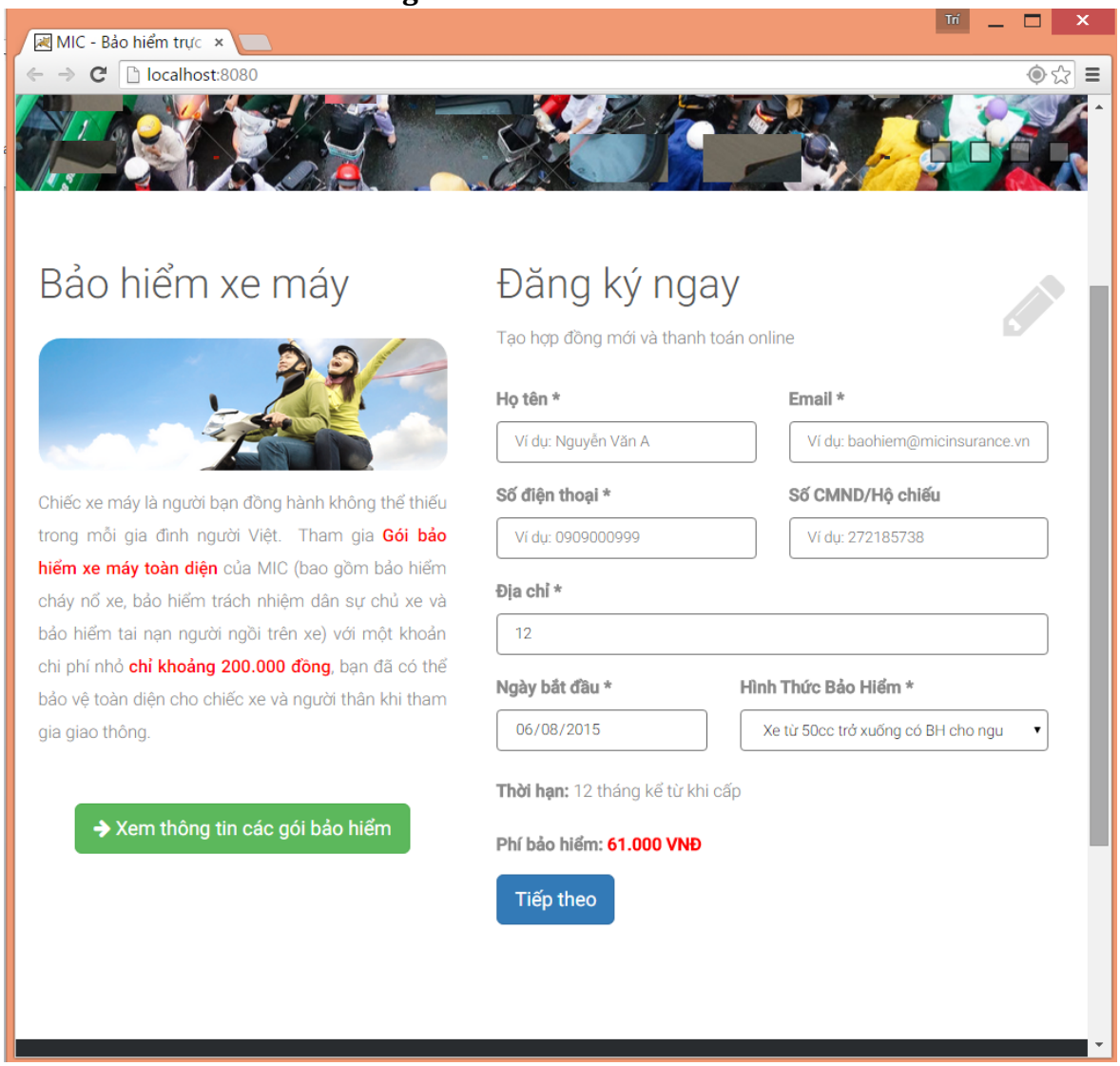
Using apk files from **Deployment** directory under this document. Click to install application.

2. User Guide

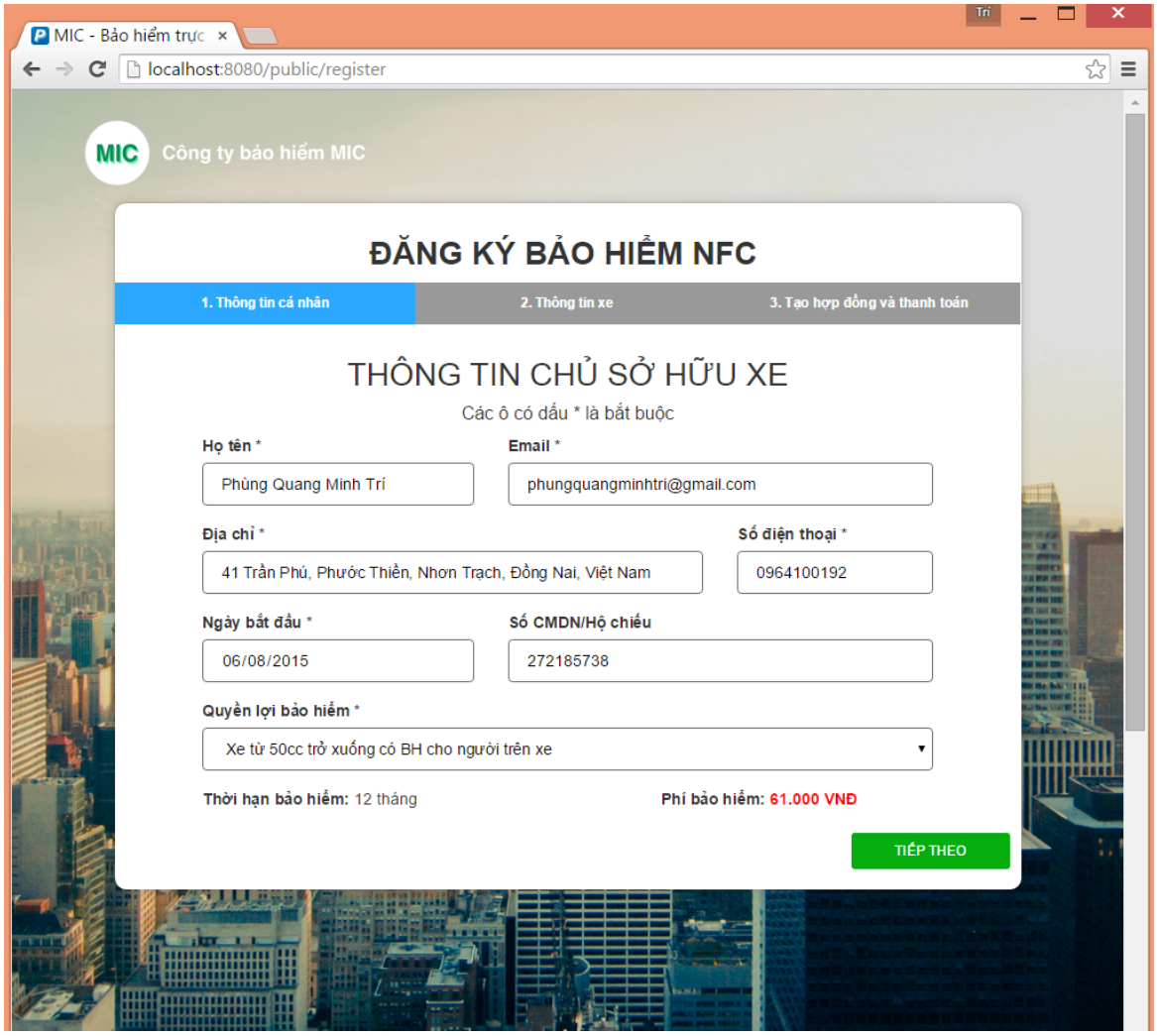
2.1. Web Application

2.1.1. Guest

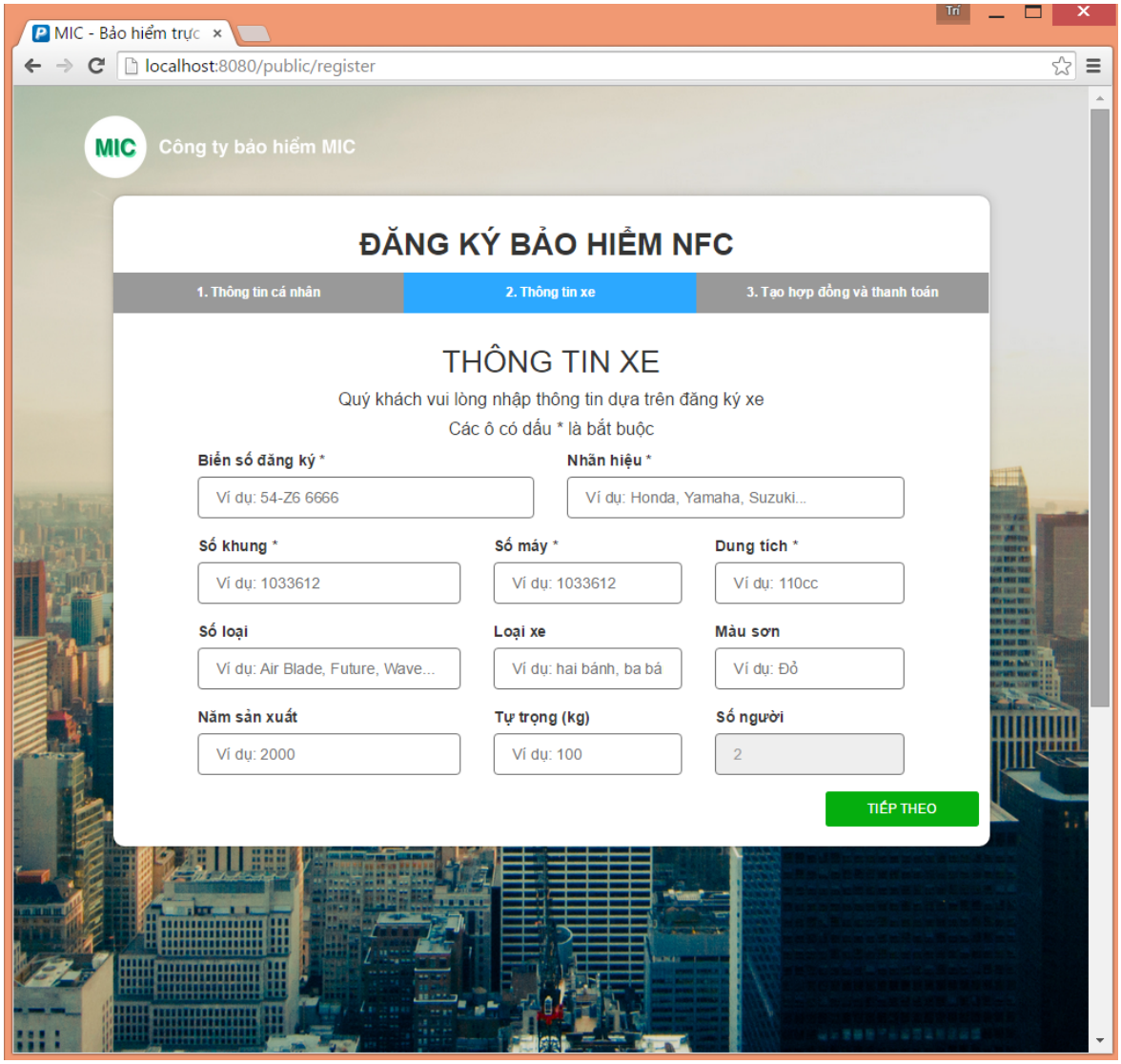
2.1.1.1.Guest register contract



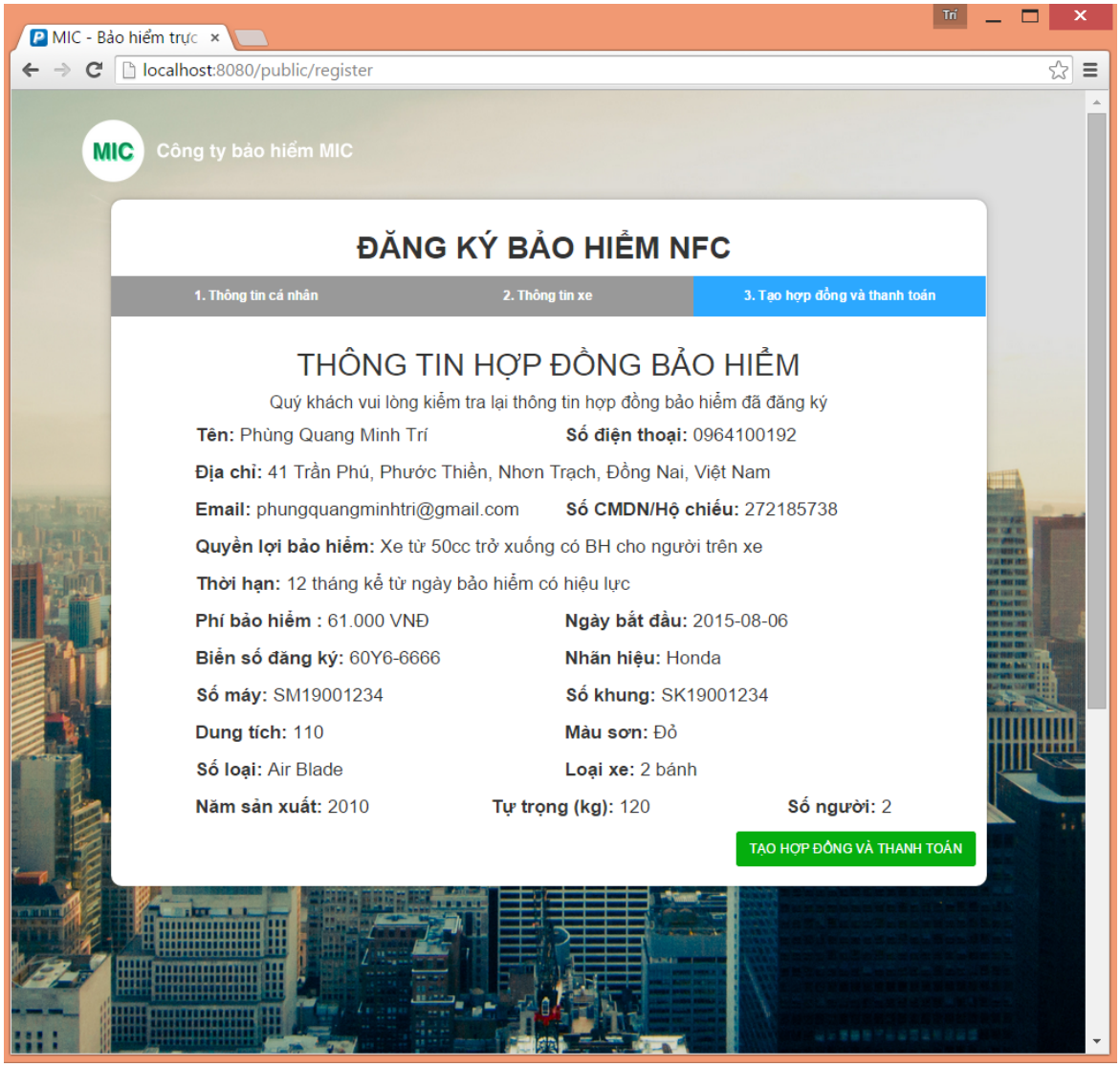
Step	Description
1	Fill in fields: “Họ tên”: Customer’s name “Email”: Customer’s email “Số điện thoại”: Phone number “Số CMND/Hộ chiếu”: Personal ID/Passport “Địa chỉ”: Address “Ngày bắt đầu”: Contract’s start date “Hình thức bảo hiểm”: Type of the contract
2	Click “Tiếp theo” button



Step	Description
3	Check and edit the previous fields if needed.
4	Click “Tiếp theo” button

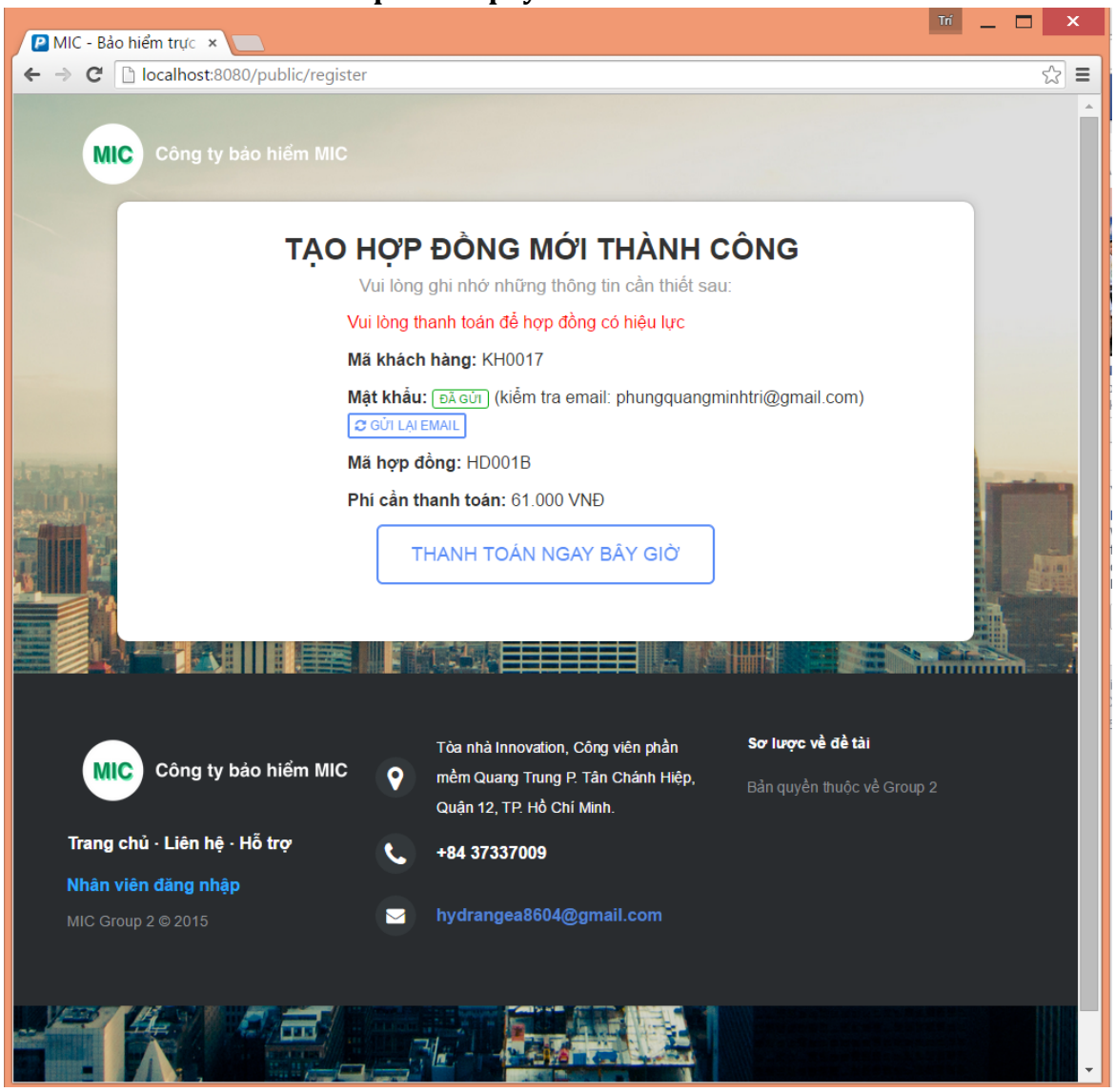


Step	Description
5	Fill in motor information: “Biến số đăng ký”: Plate number “Nhãn hiệu”: Brand “Số khung”: Chassis “Số máy”: Engine “Dung tích”: Capacity “Số loại”: Model “Loại xe”: Type “Màu sơn”: Color “Năm sản xuất”: Year of manufacturer “Tự trọng”: Weight “Số người”: Seat capacity
6	Click “Tiếp theo” button

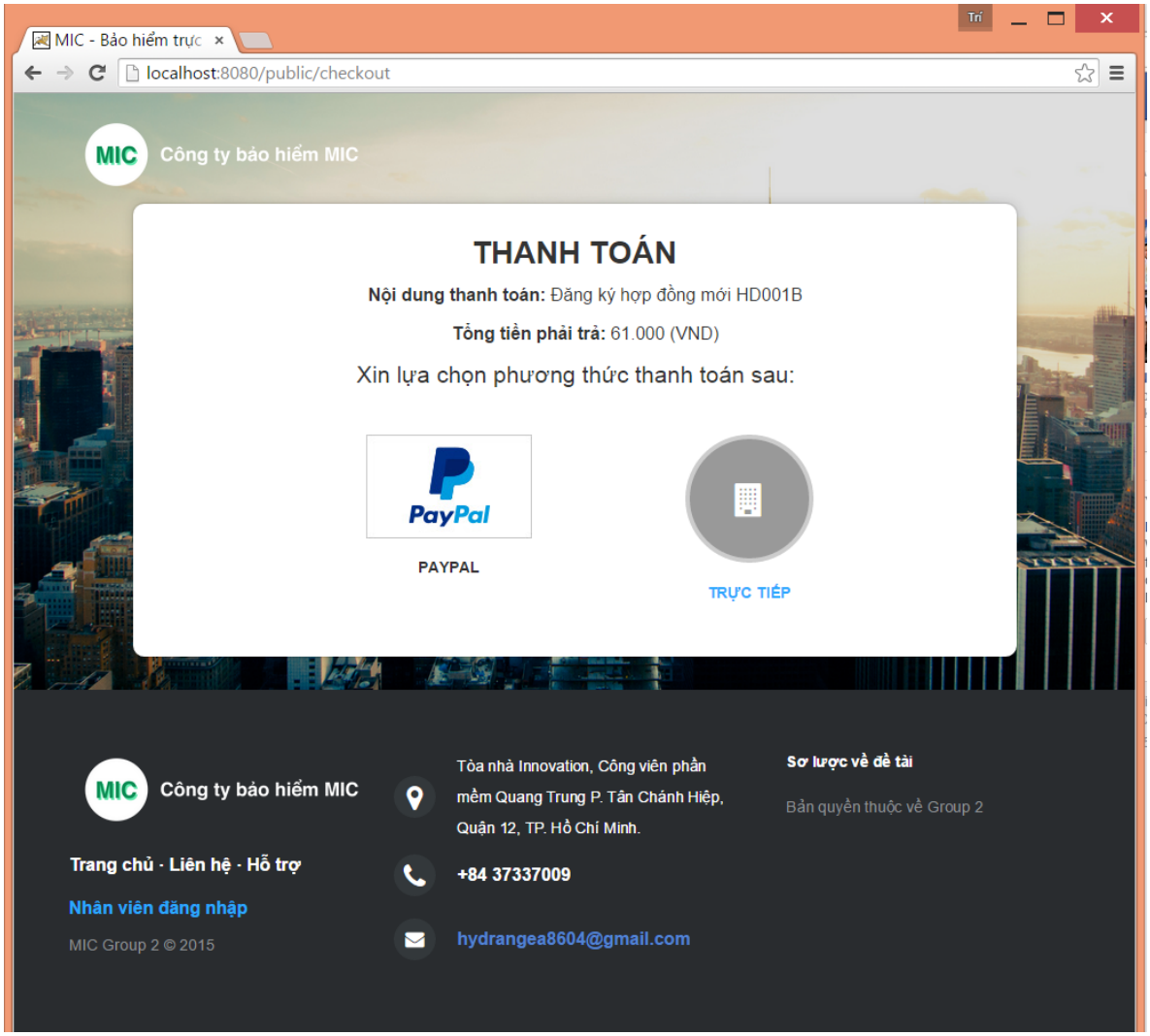


Step	Description
7	Confirm the inputted information and click “TẠO HỢP ĐỒNG VÀ THANH TOÁN” button.

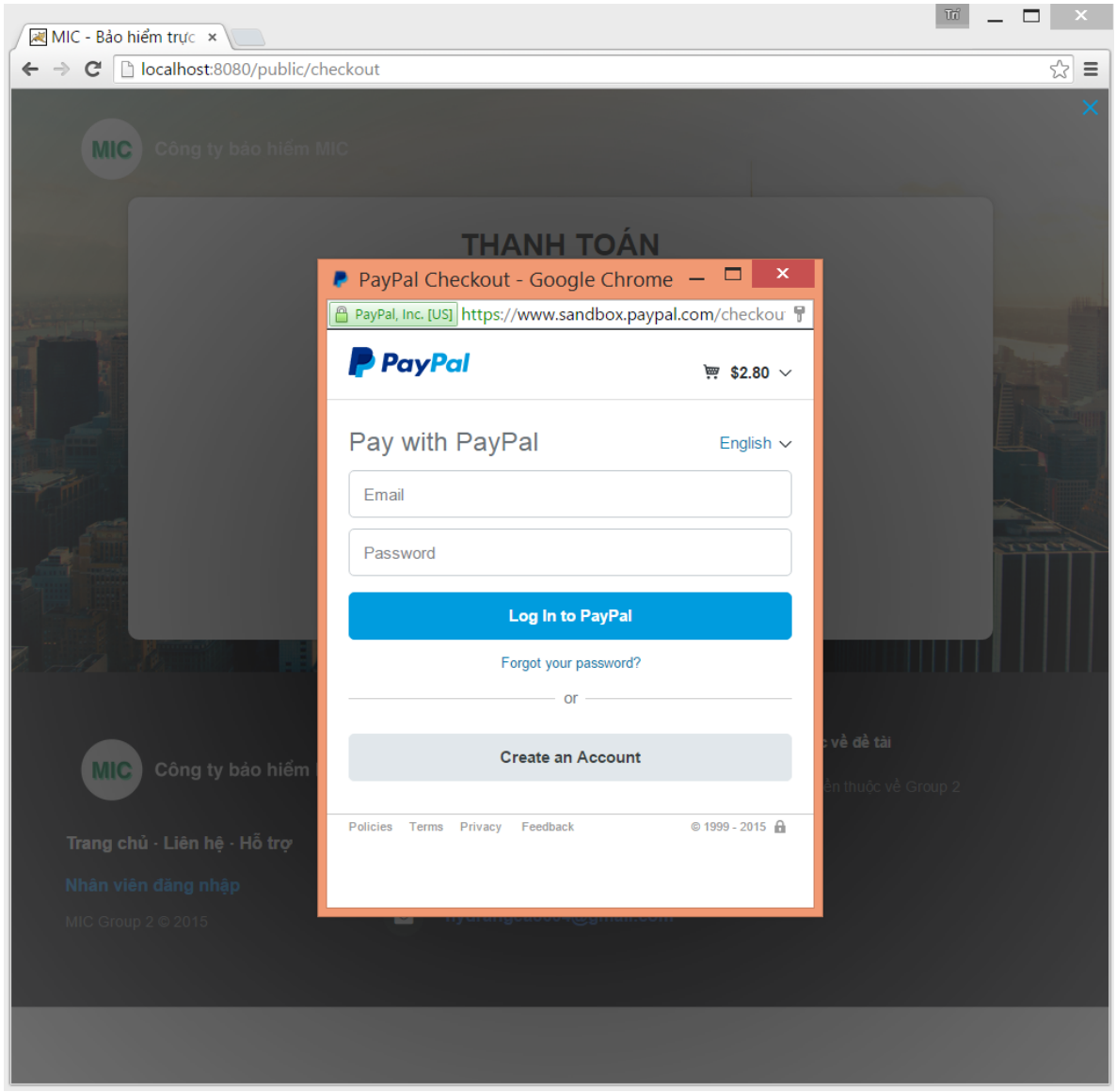
2.1.1.2.Guest process payment



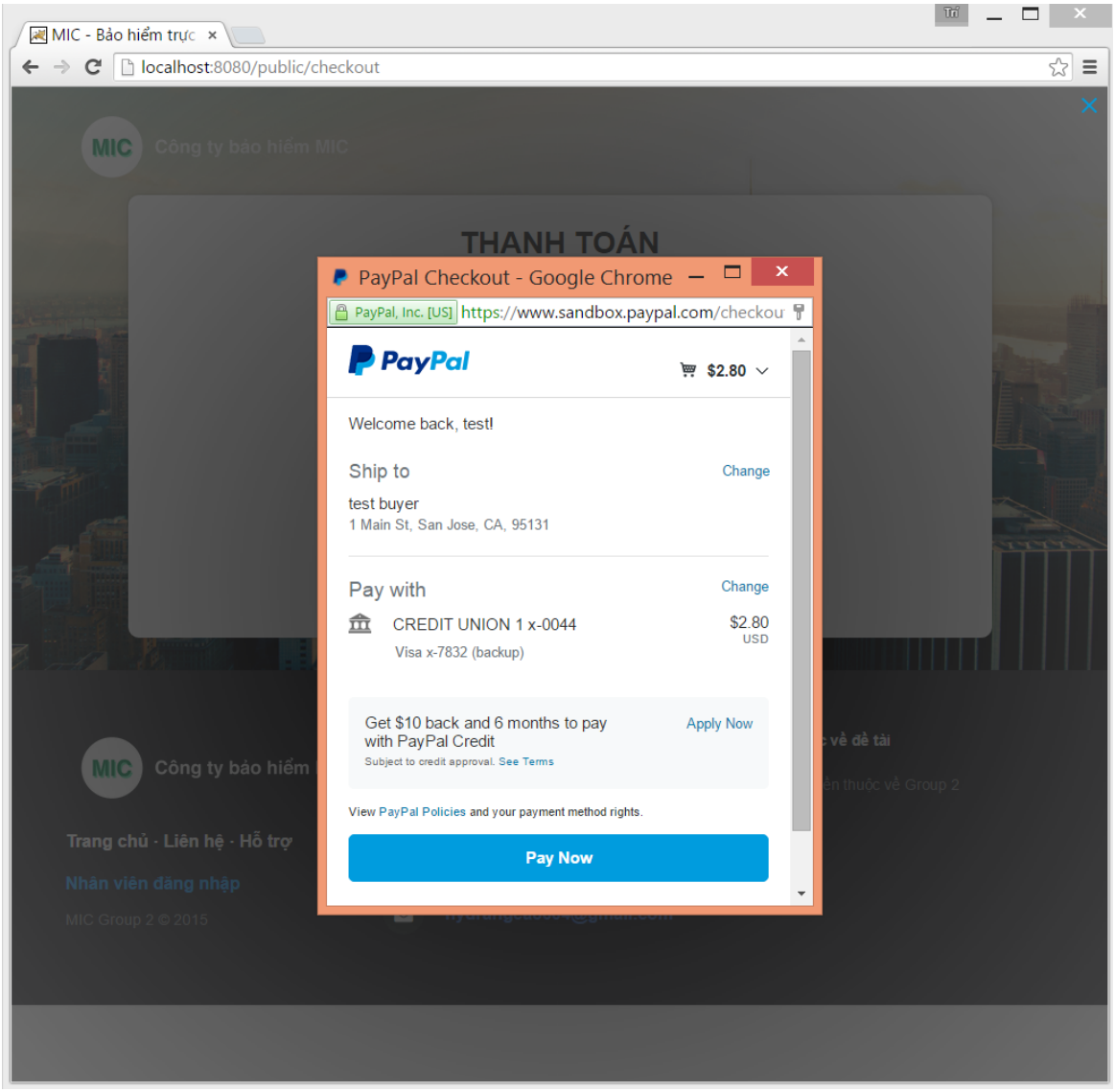
Step	Description
1	Click “THANH TOÁN NGAY BÂY GIỜ” button.



Step	Description
2	Click “PayPal” button



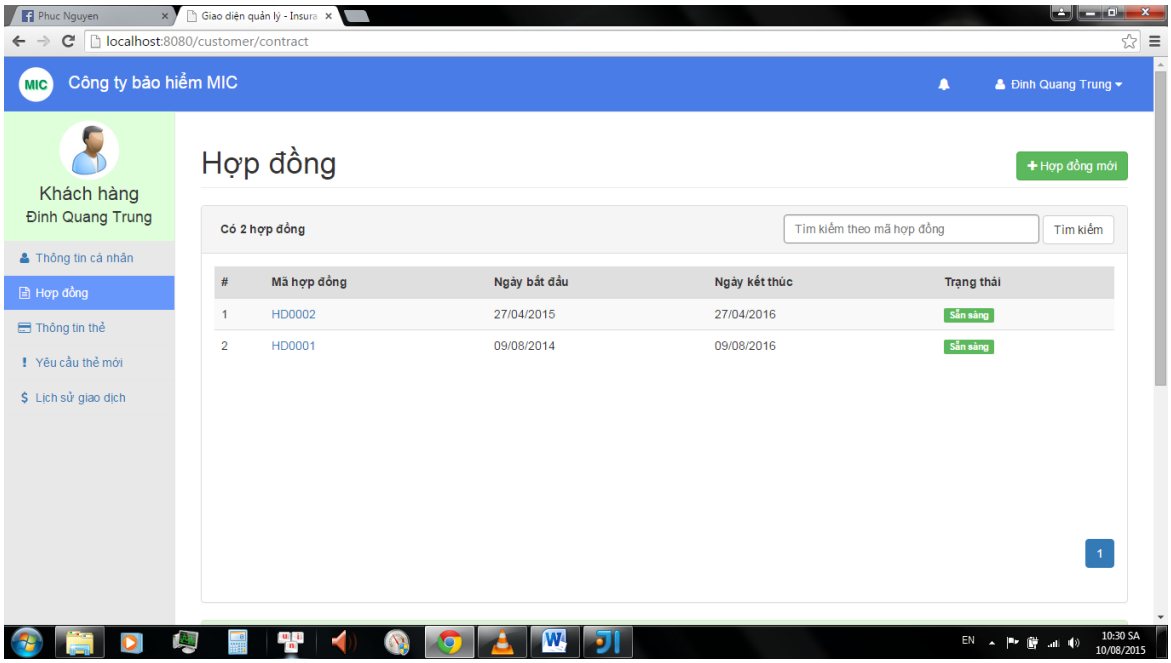
Step	Description
3	Enter your PayPal email and password
4	Click “Log In to PayPal” button



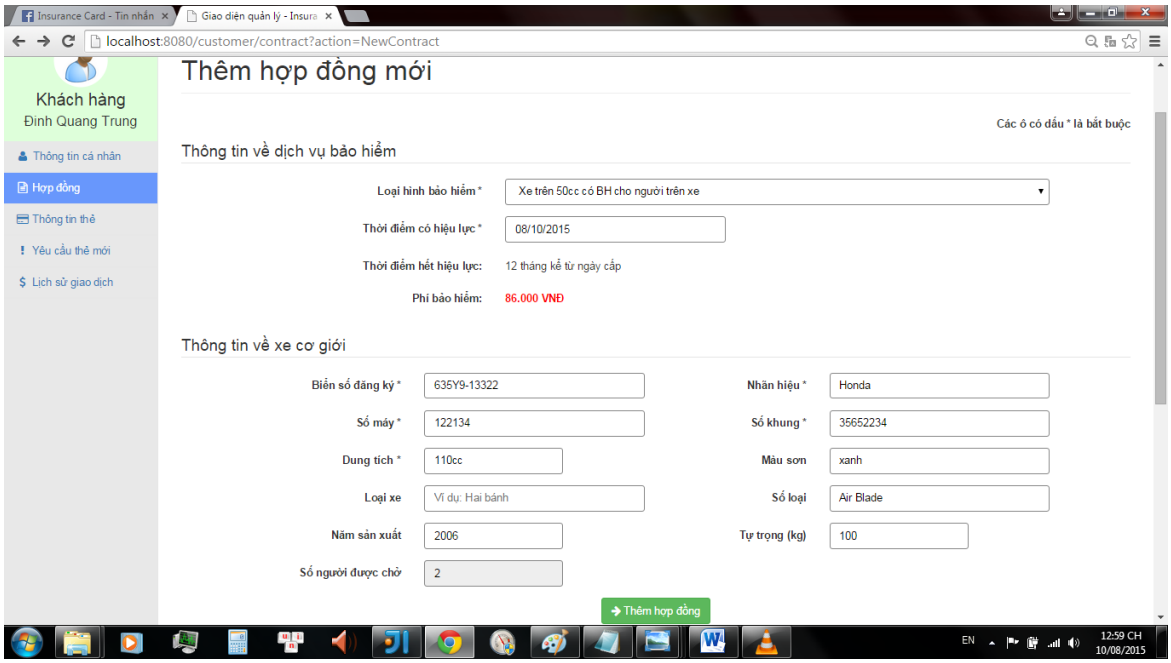
Step	Description
5	Click “Pay Now” button.

2.1.2. Customer

2.1.2.1.Customer create new contract

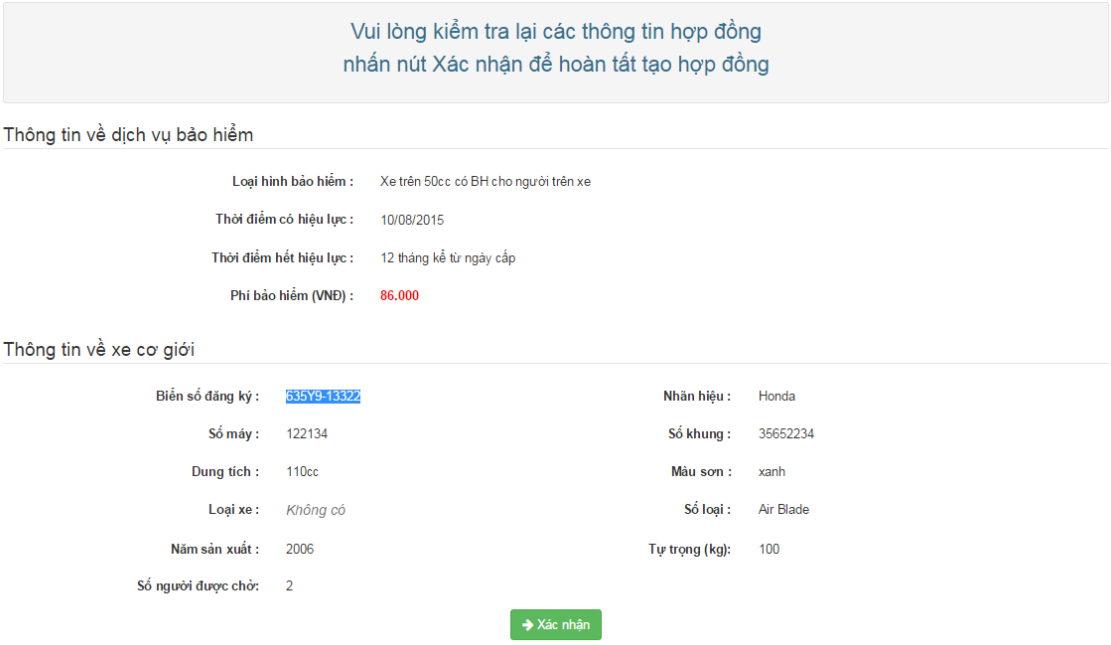


Step	Description
1	Click on “Hợp đồng mới” button

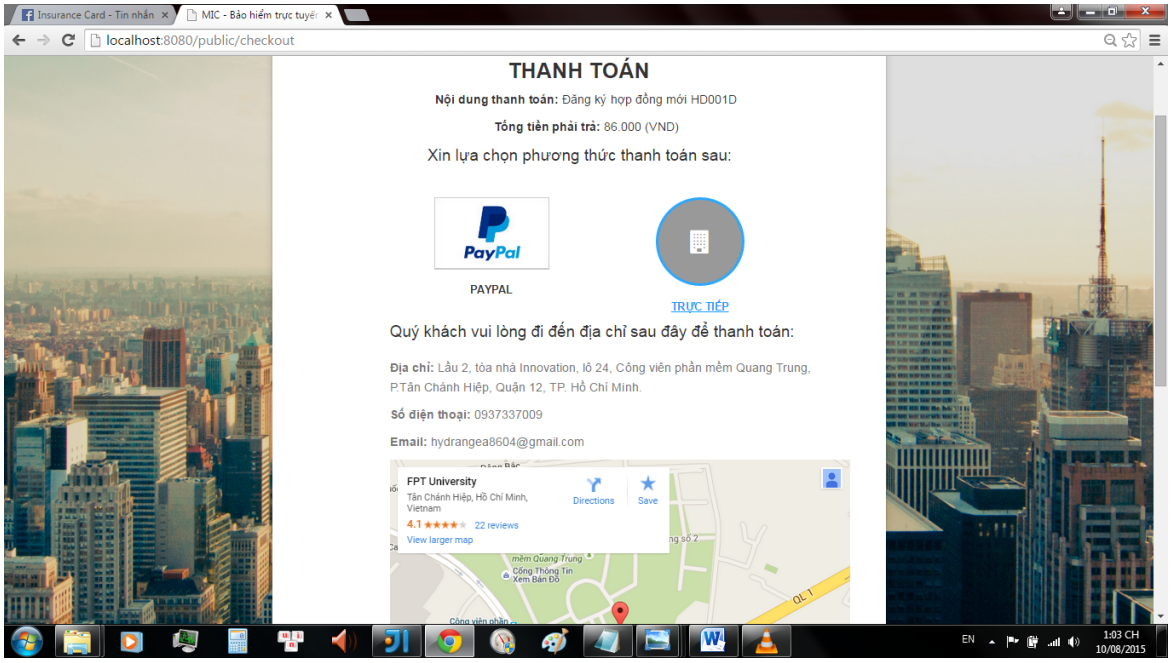


Step	Description
2	Fill information into blank fields for example:

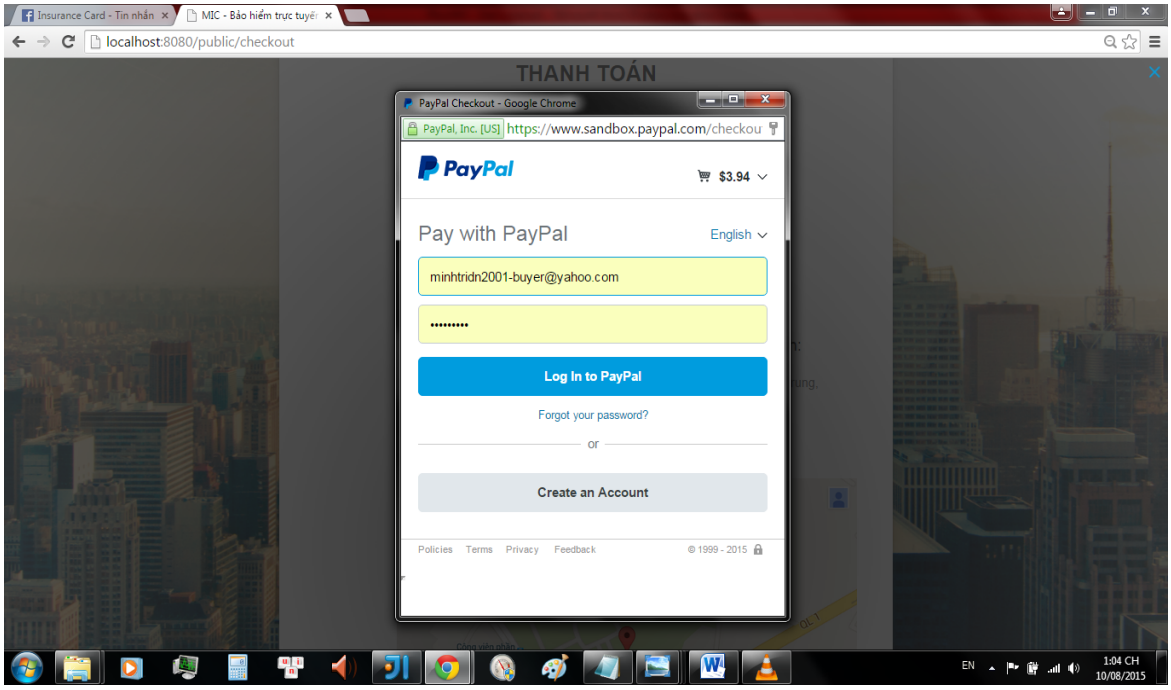
	Choose “Loại hình bảo hiểm” : Xe trên 50cc có BH cho người trên xe “Thời điểm có hiệu lực” : 10/08/2015 “Biển số đăng ký” : 635Y9-13322 “Nhãn hiệu” : Honda “Số máy” : 122134 “Số khung” : 35652234 “Dung tích” : 110cc “Màu sơn” : xanh “Loại xe” : “Số loại” : Air Blade “Năm sản xuất” : 2006 “Tự trọng (kg)” : 100 “Số người được chở” :
3	Click on “Thêm hợp đồng” button



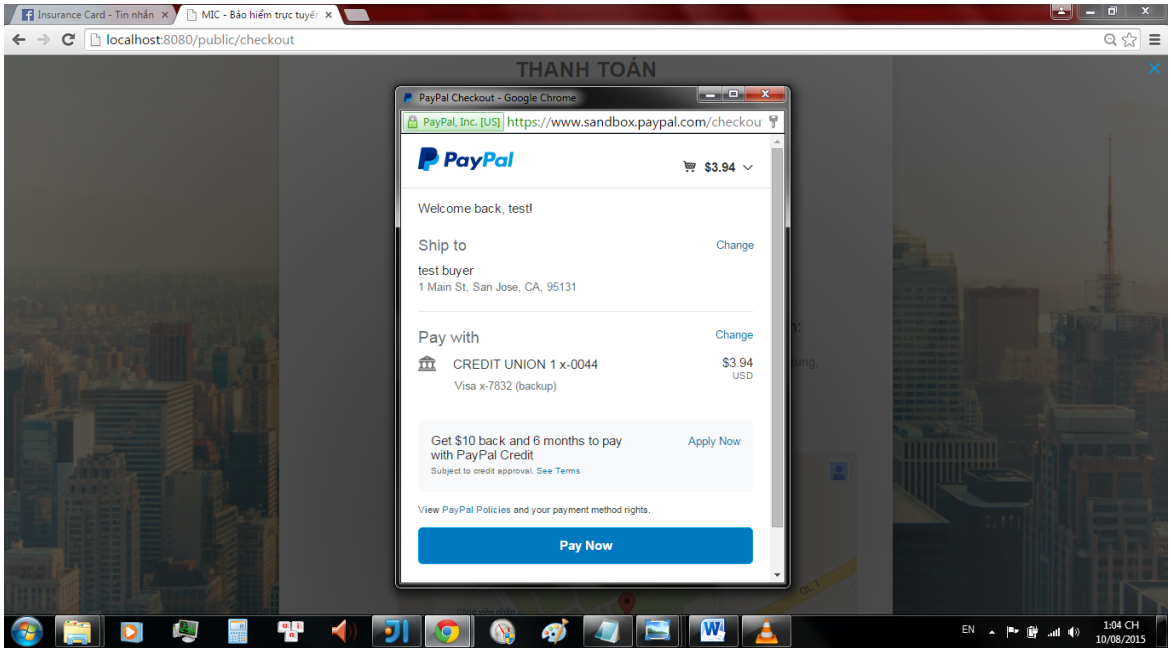
Step	Description
4	Click on “Xác nhận” button



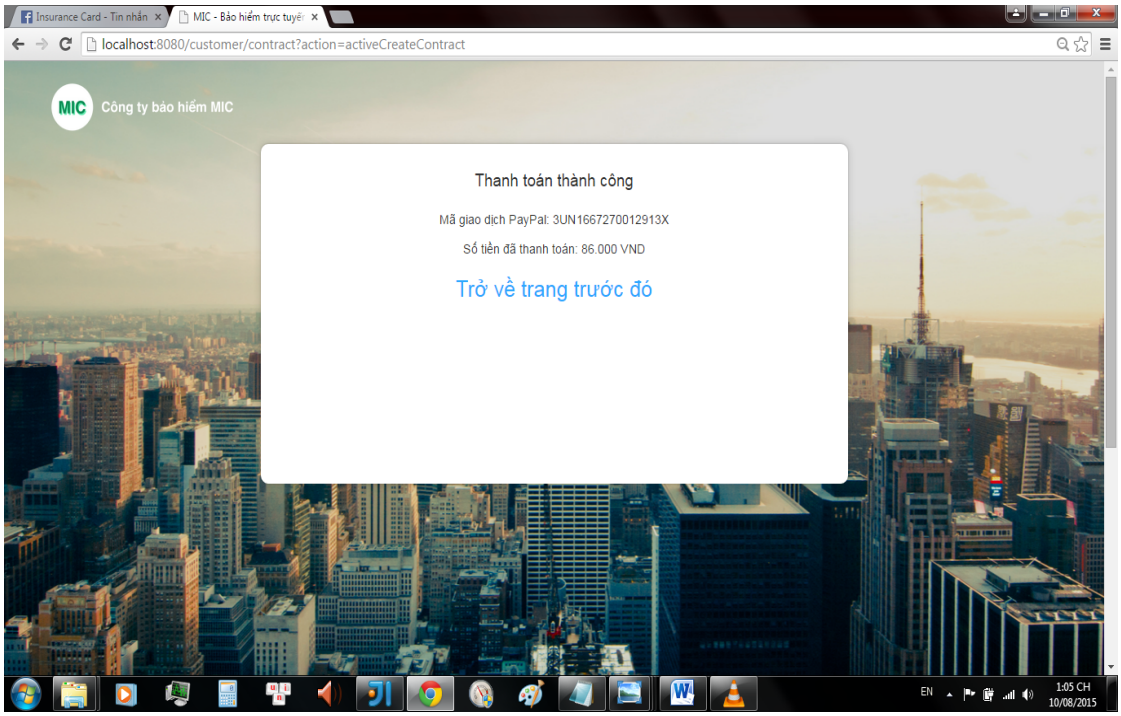
Step	Description
5	○ Click “PayPal” link image button If customer want to pay direct at company they can view address of company on map by clicking “Trực Tiếp” link image button



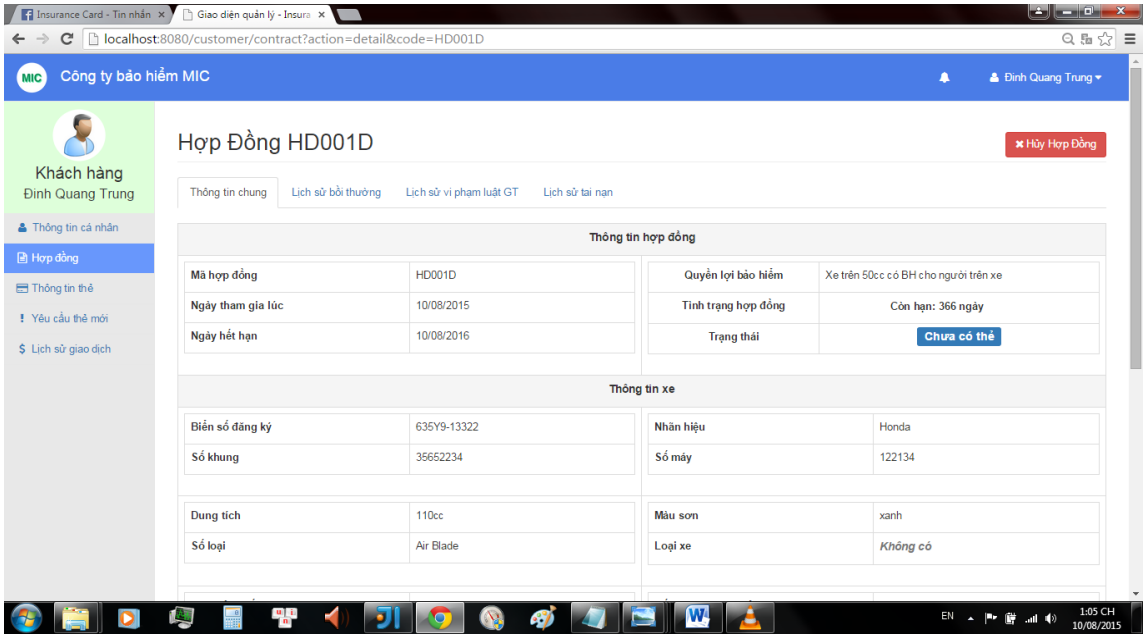
Step	Description
6	Click “Login In to PayPal” button



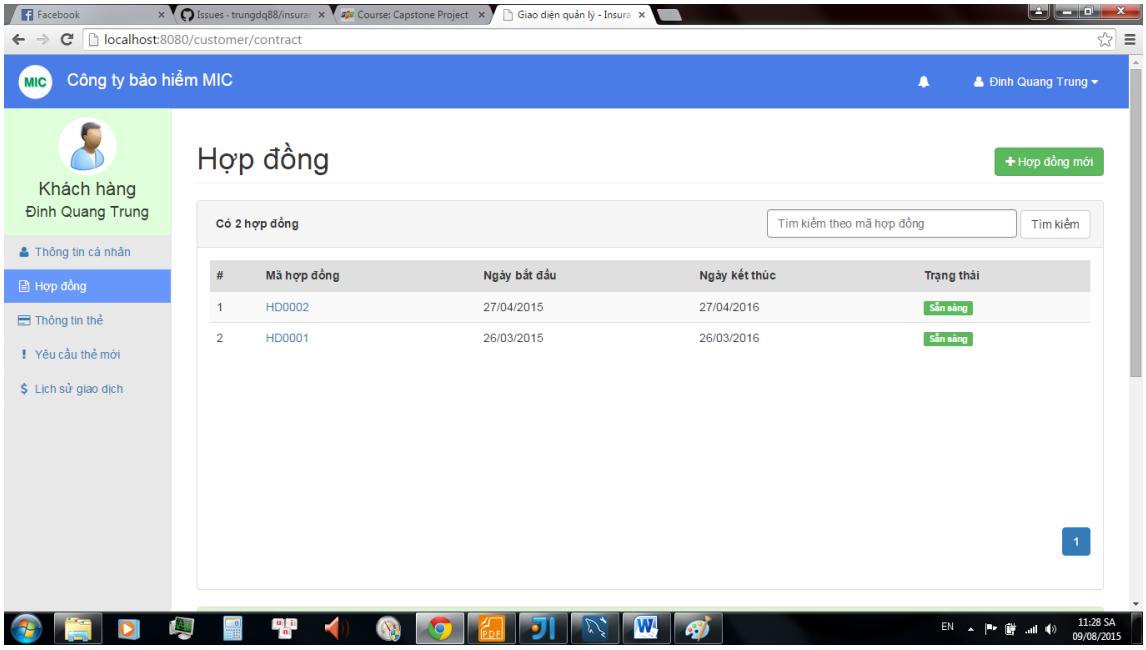
Step	Description
7	Click “Pay Now” button



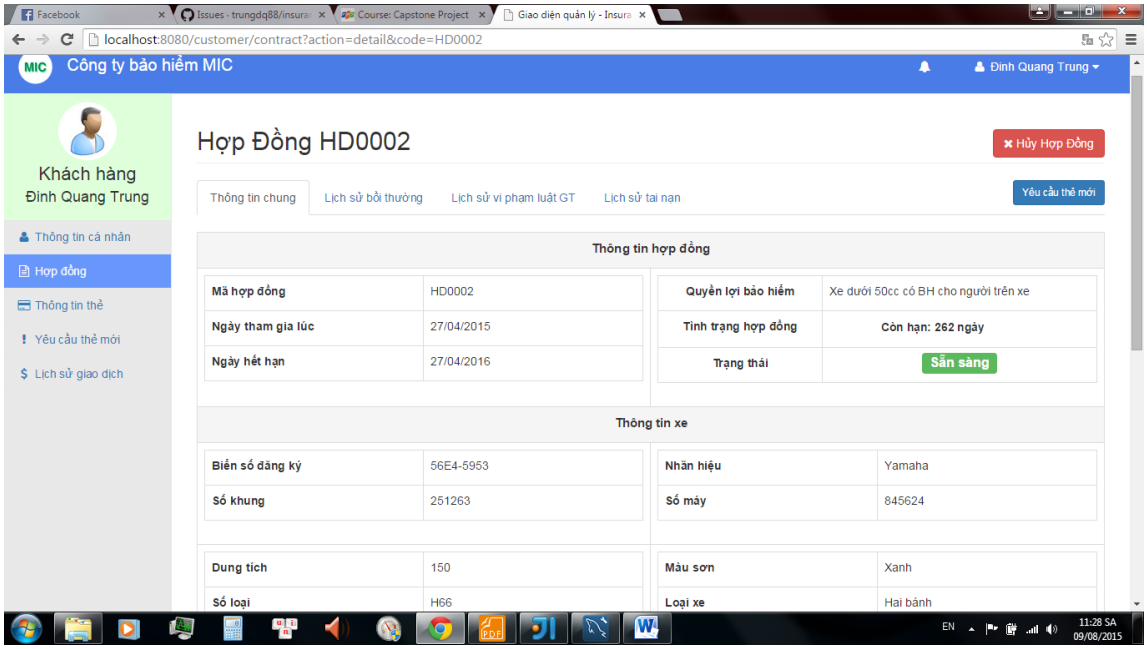
Step	Description
8	Click on “Trở về trang trước đó”



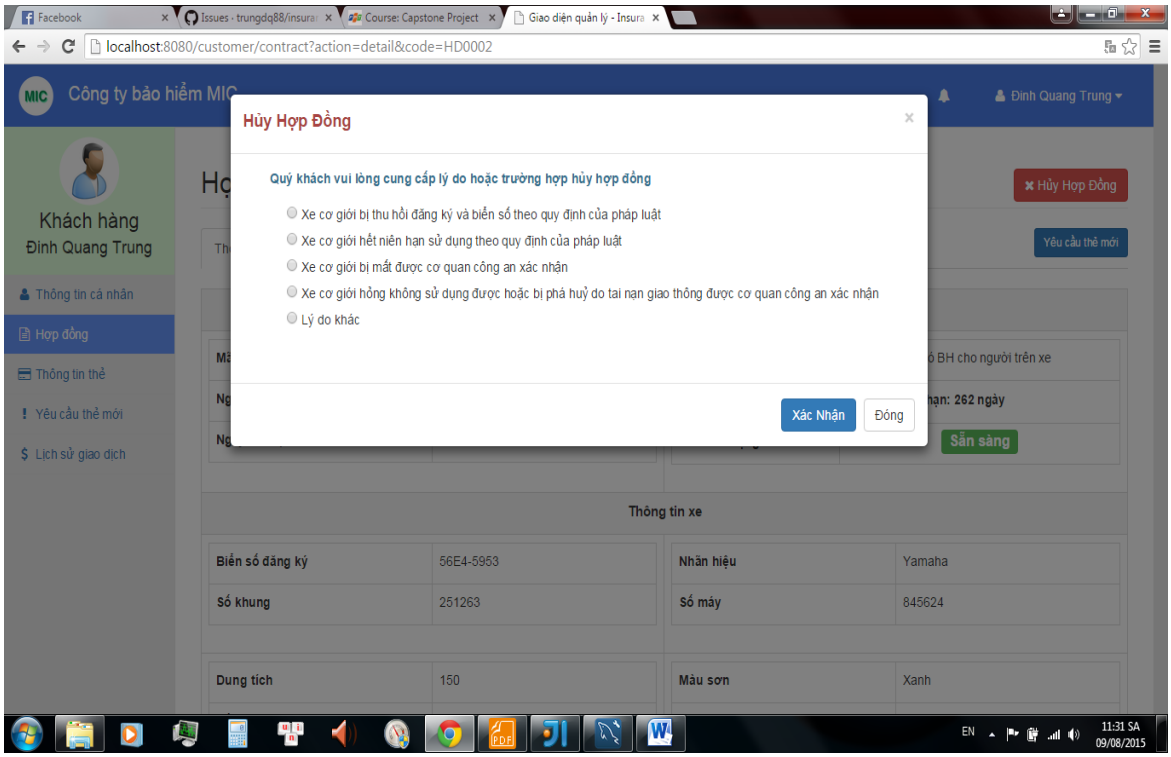
2.1.2.2.Customer cancel contract



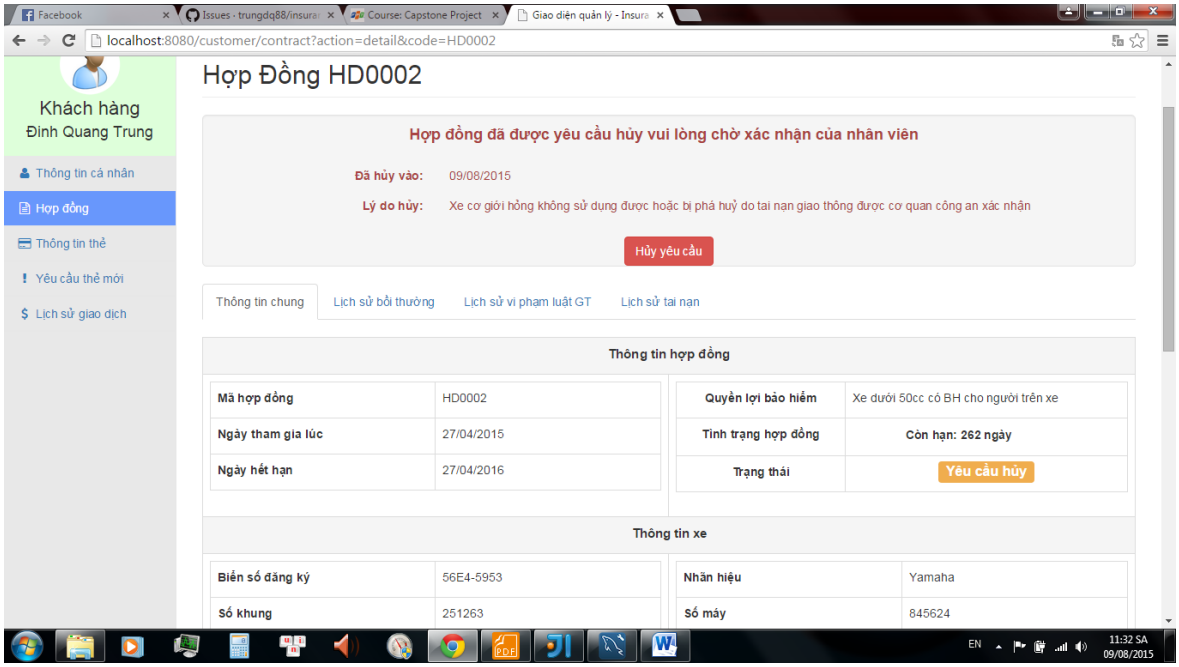
Step	Description
1	Click on contract code ‘HD002’ link



Step	Description
2	Click “Hủy Hợp Đồng” button

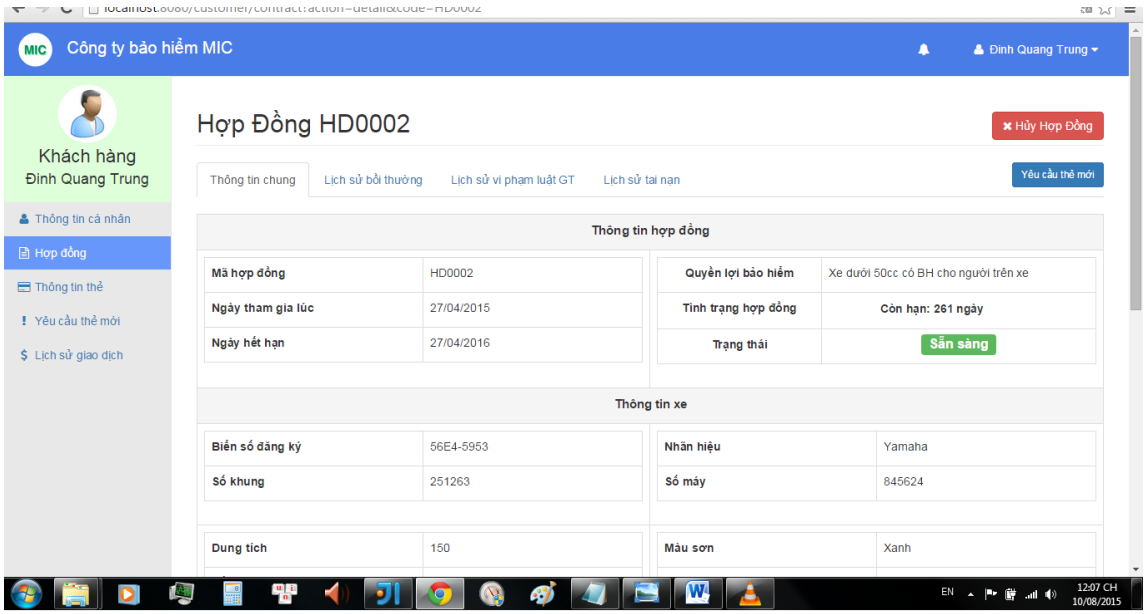


Step	Description
3	○ Choose “Xe cơ giới hỏng không sử dụng được hoặc bị phá hủy do tai nạn giao thông được cơ quan công an xác nhận” : cancel reason
4	○ Click on “Xác nhận” button

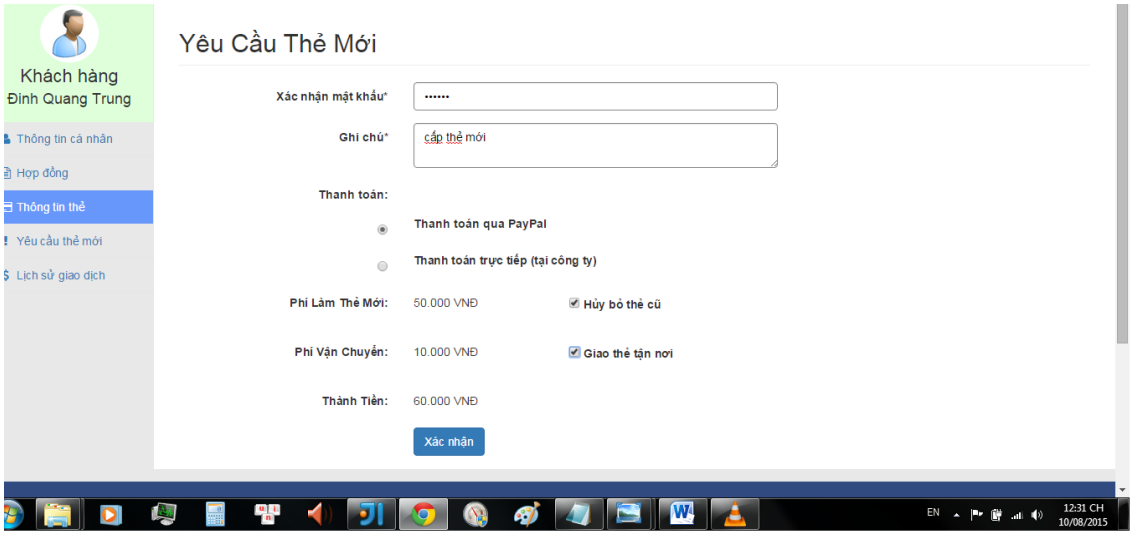


If customer wants to cancel request cancel contract they can click on “Hủy yêu cầu” button.

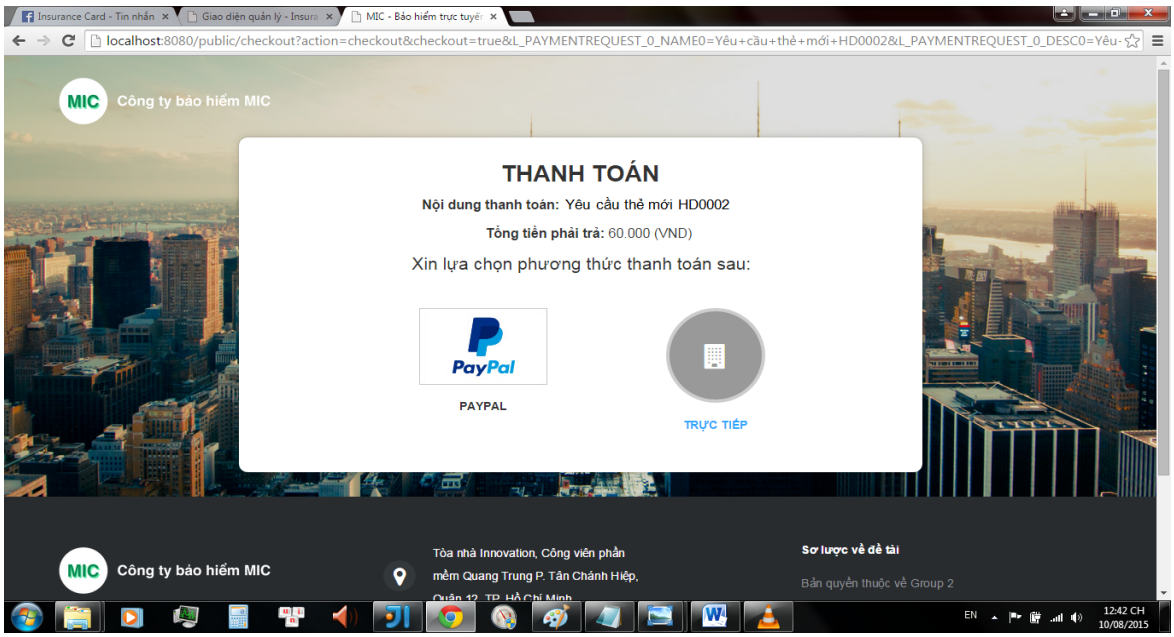
2.1.2.3.Customer request new card



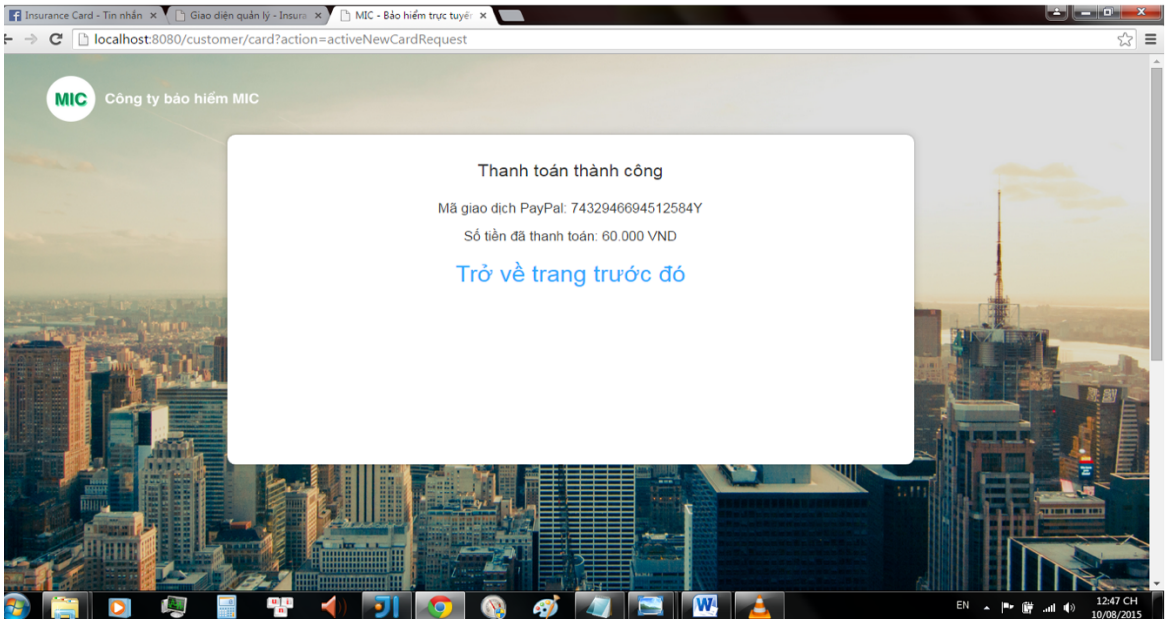
Step	Description
1	Click on “Yêu cầu thẻ mới” button



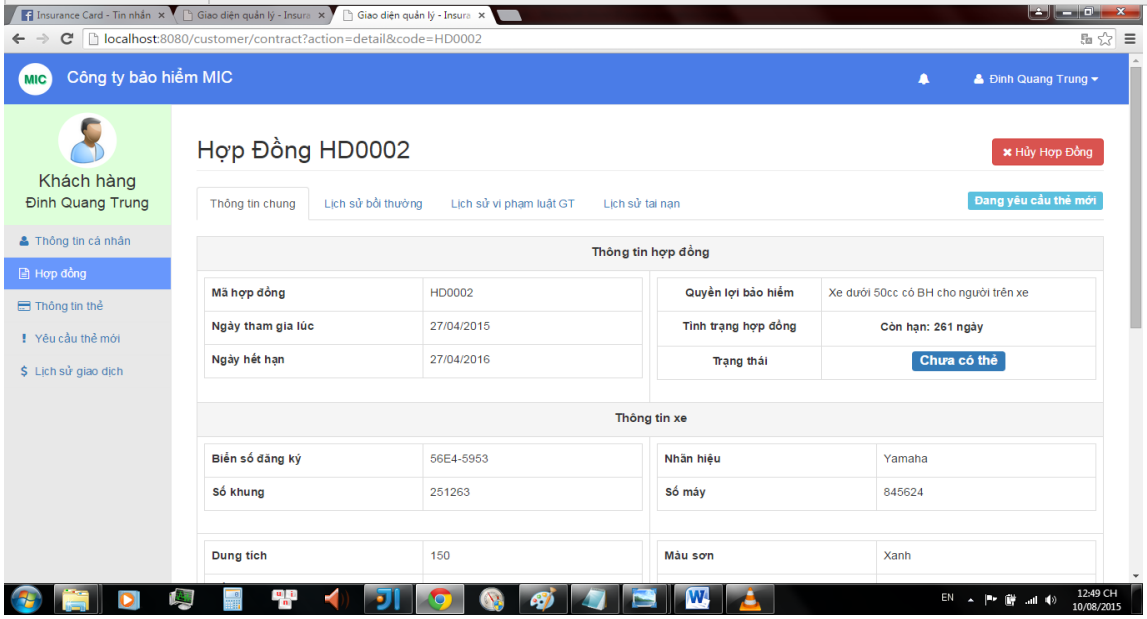
Step	Description
2	- Fill “Xác nhận mật khẩu” : 123123 (for example) - “Ghi chú” : “cấp thẻ mới” (for example) - Check “Thanh toán qua PayPal” - Check “Hủy bỏ thẻ cũ” - Check “” Giao thẻ tận nơi
3	Click on “Xác nhận” button



Step	Description
4	- Click on “Trực tiếp” image link button if customer wants to pay direct at company - Click on “PayPal” image link button



Step	Description
5	- Click on “Trở về trang trước đó” link

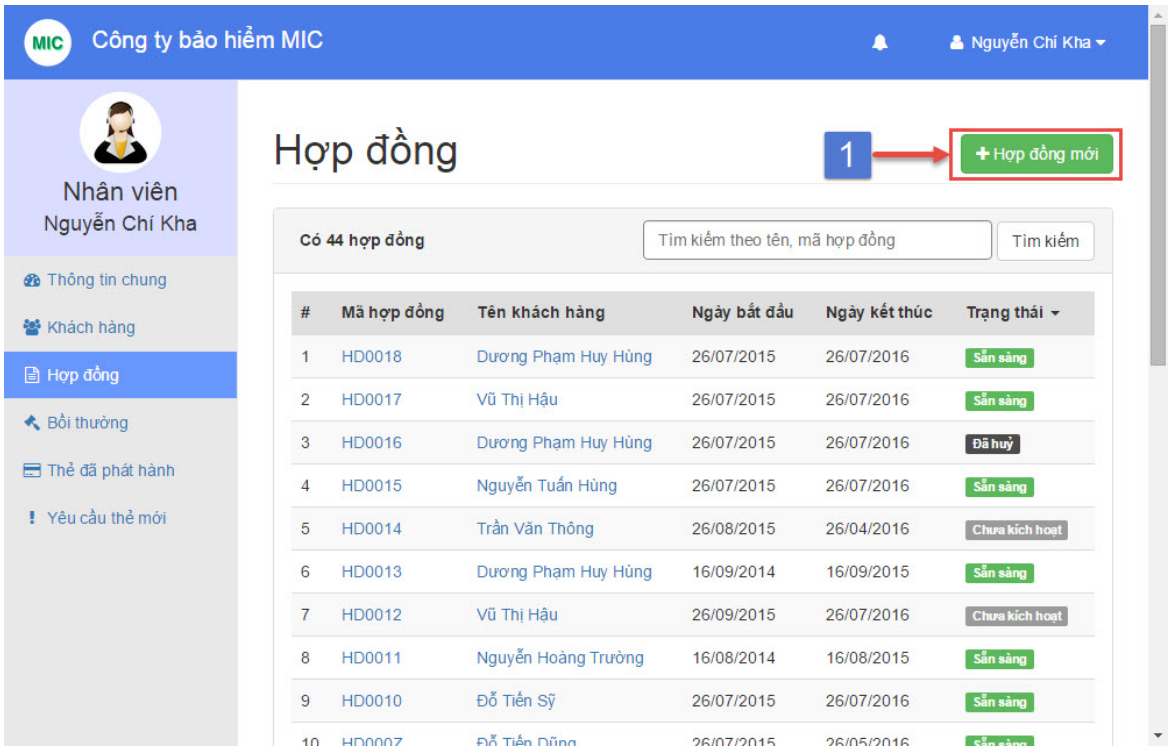


Step	Description
6	- Click on “Yêu cầu thẻ mới” link menu

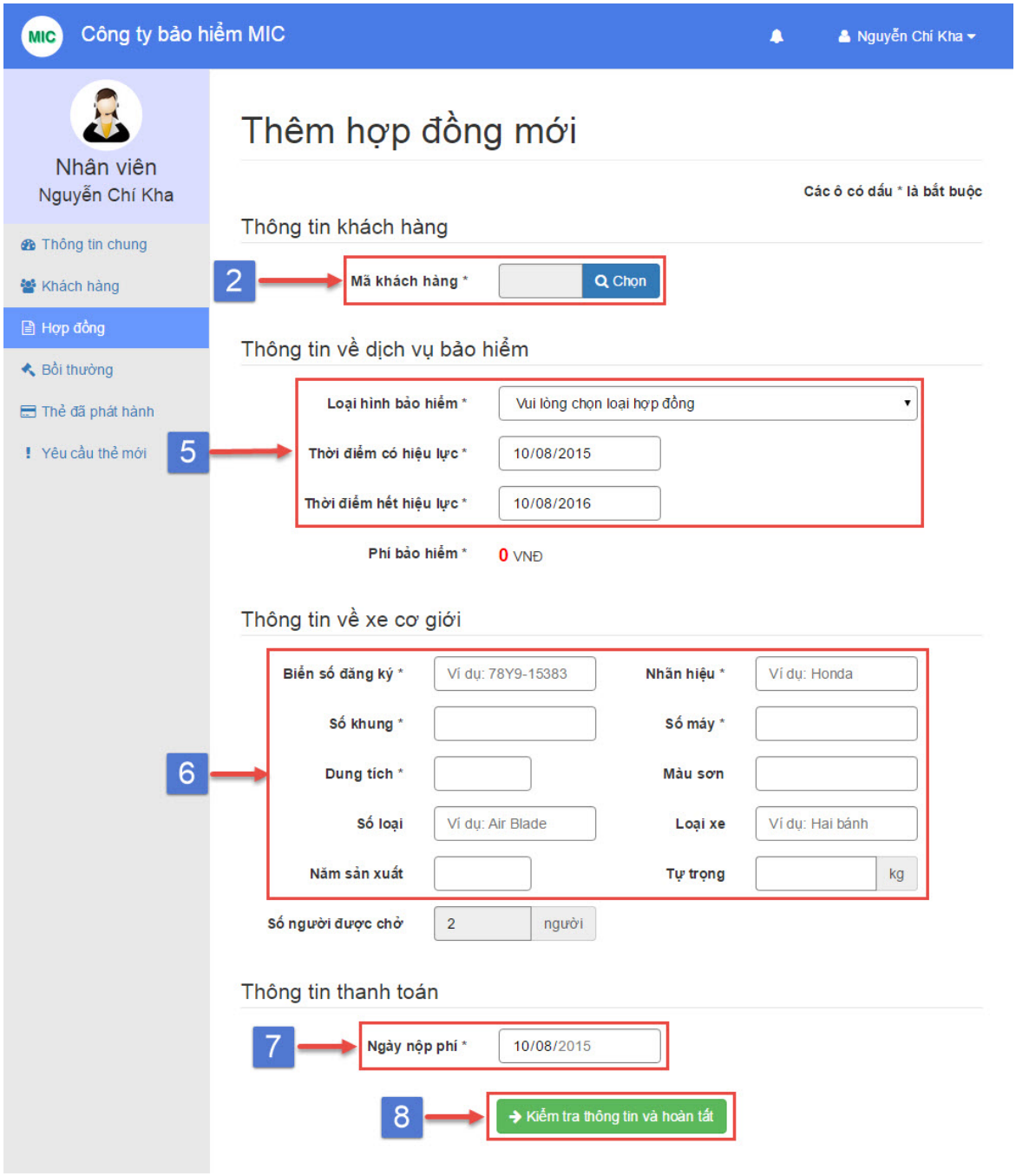

localhost:8080/customer/card?action=viewNewCardRequests
Công ty bảo hiểm MIC
Đinh Quang Trung
Khách hàng
Đinh Quang Trung
Thông tin cá nhân
Hợp đồng
Thông tin thẻ
Yêu cầu thẻ mới
Lịch sử giao dịch
Yêu cầu cấp thẻ mới
Có 1 yêu cầu cấp thẻ mới chưa xử lý
Tìm kiếm theo mã thẻ cũ
Tìm kiếm
Thời gian Ghi chú Mã thẻ cũ Hợp đồng Ngày cấp mới Thẻ mới cấp Thực hiện
1 10/08/2015 9021748528325 HD0002 Chưa cấp Đã thanh toán
1
12:50 CH
10/08/2015

2.1.3. Staff

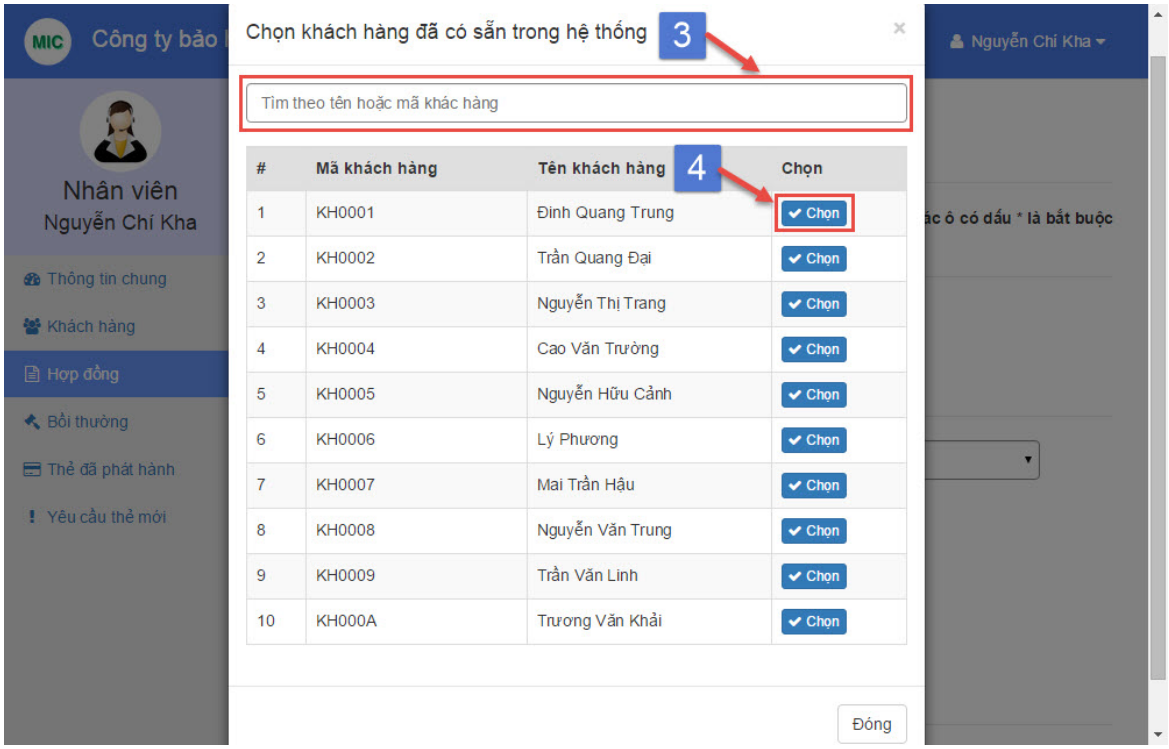
2.1.3.1.Staff create new contract



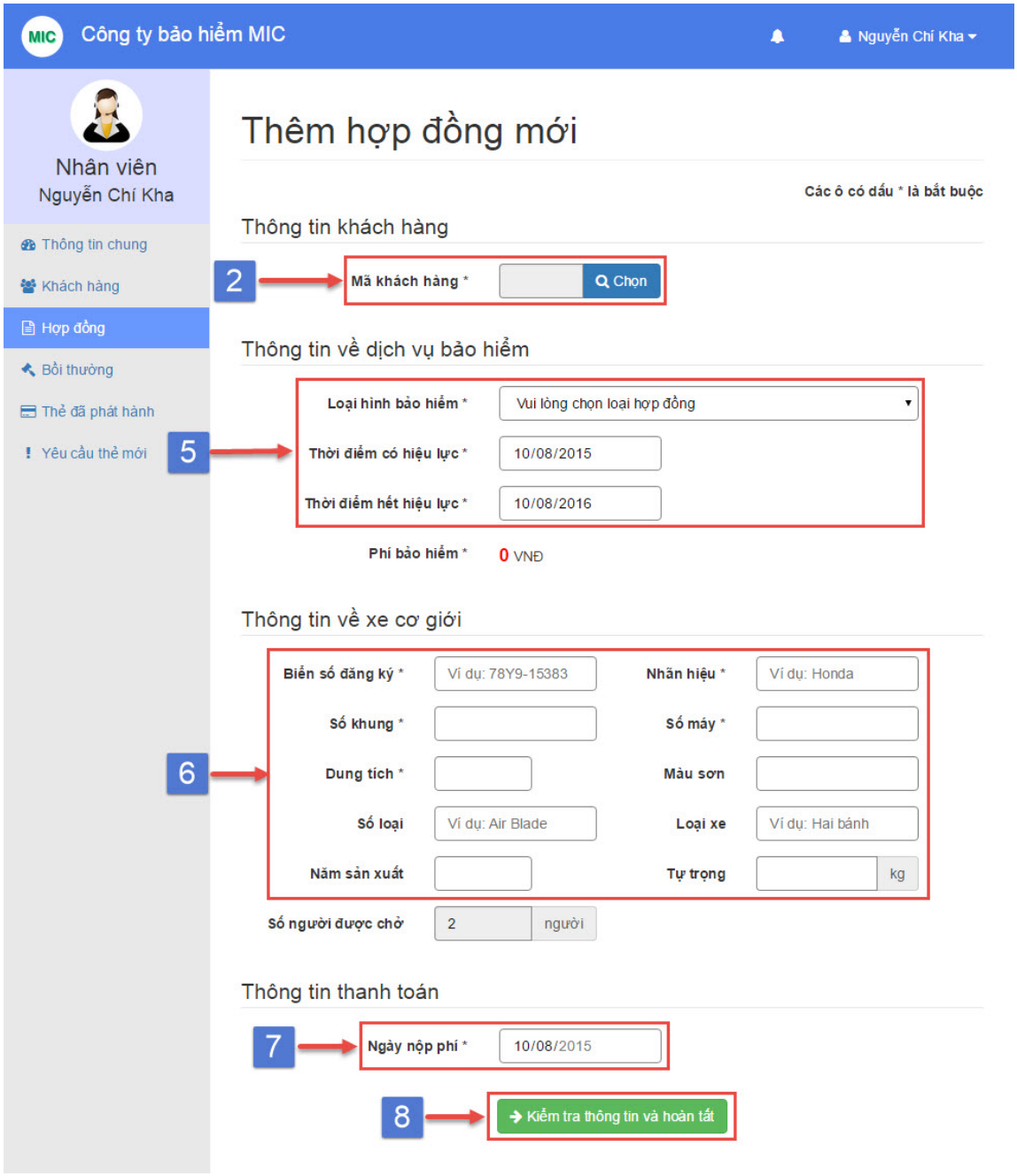
Step	Description
1	Click on button “Hợp đồng mới”



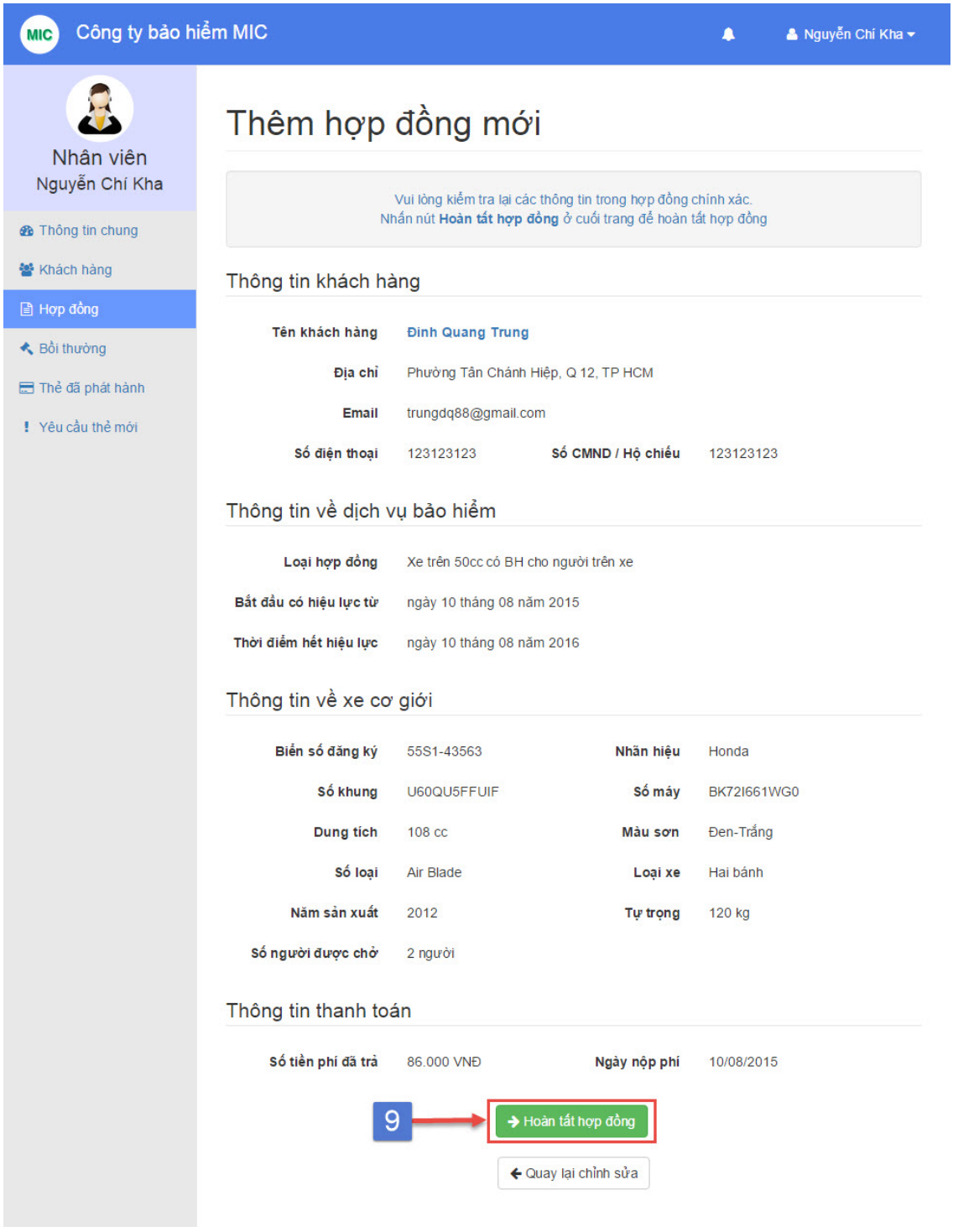
Step	Description
2	Click “Chọn” button



Step	Description
3	Search customer by name or code
4	Click “Chọn” button

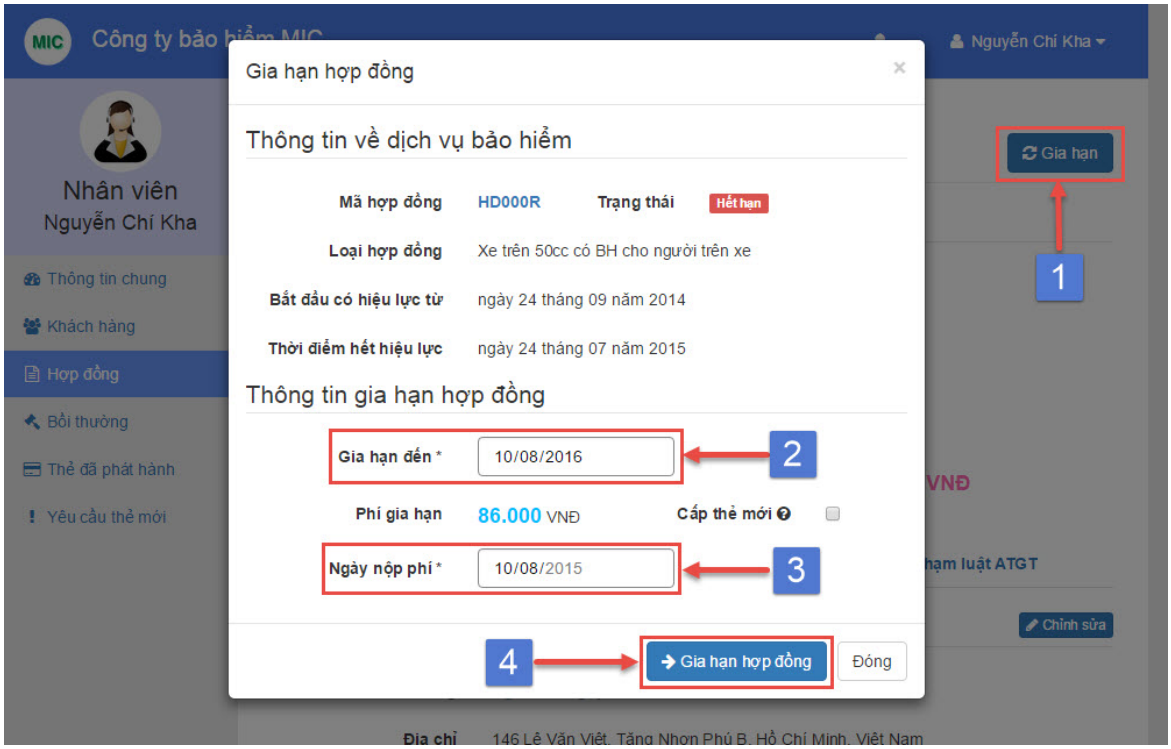


Step	Description
5	Select contract type in “Loại hình bảo hiểm” dropdown list (required) Choose “Thời điểm có hiệu lực” the start date of this contract (required) Choose “Thời điểm hết hiệu lực” the end date of this contract (required)
6	Fill vechile plate, brand, engine, chassis, capacity (required) Fill color, model code, vehicle type, year of manufacture, weight (optional)
7	Choose “Ngày nộp phí” the paid date of this contract (required)
8	Click “Kiểm tra thông tin và hoàn tất” button



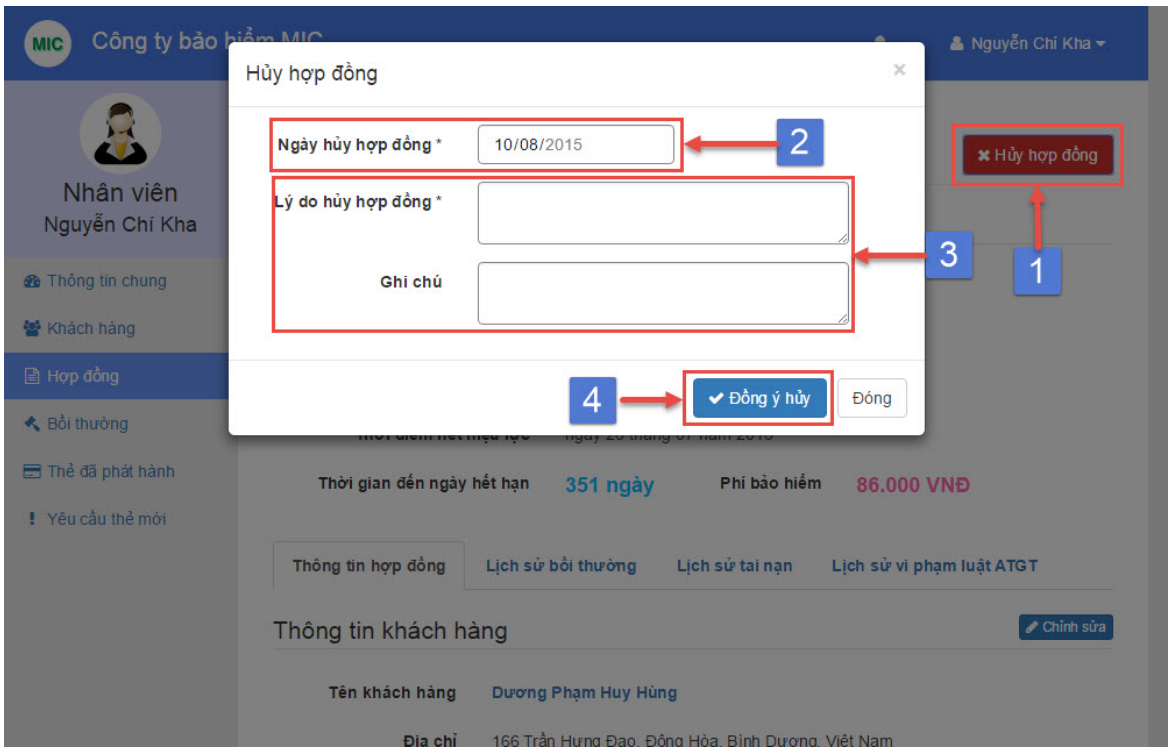
Step	Description
9	Click “Hoàn tất hợp đồng” button

2.1.3.2.Staff renew contract



Step	Description
1	Click “Gia hạn” button
2	Choose “Gia hạn đến” the contract new expired date (required)
3	Choose “Ngày nộp phí” the contract paid date (required)
4	Click “Gia hạn hợp đồng” button

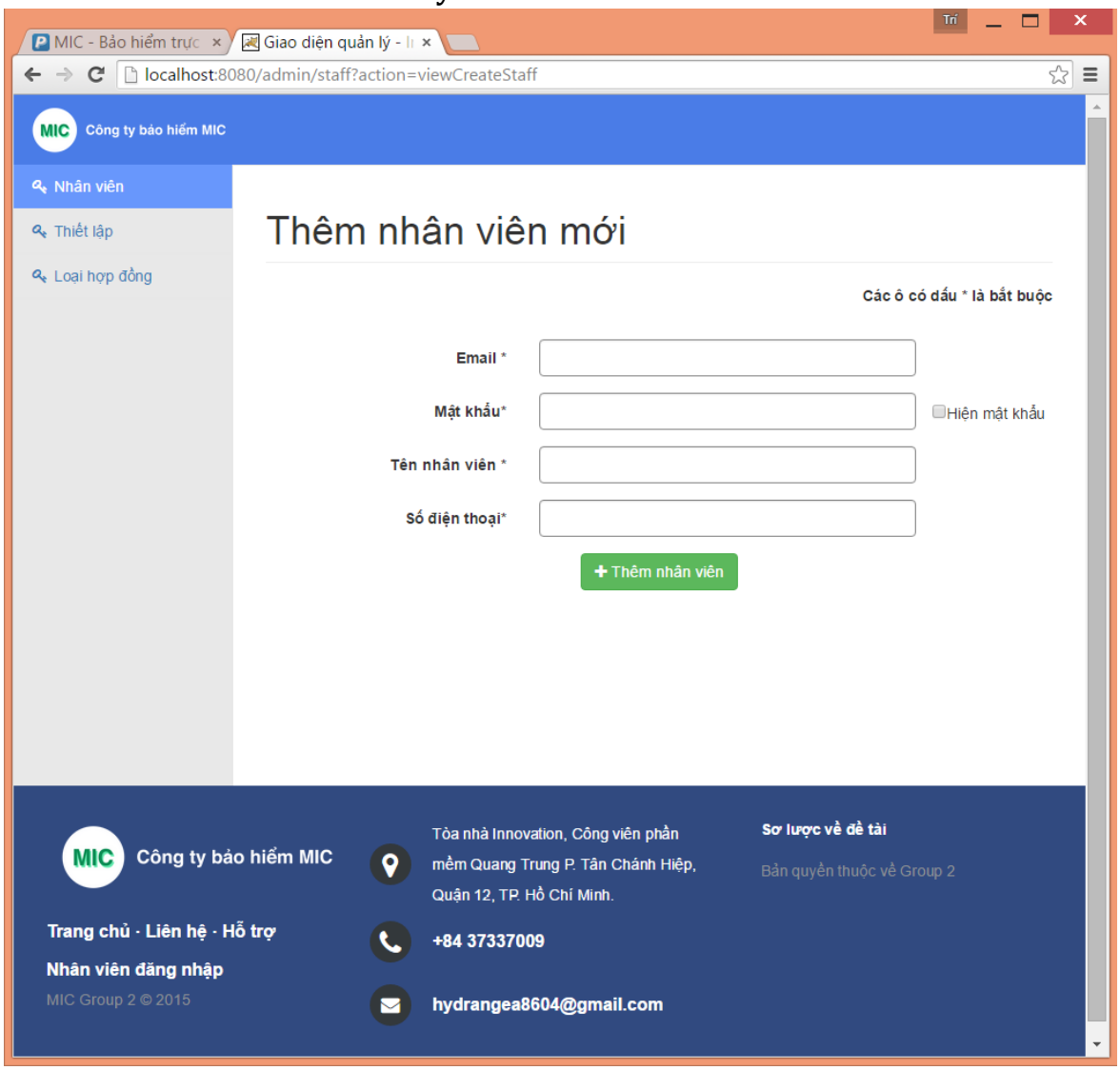
2.1.3.3.Staff cancel contract



Step	Description
1	Click “Hủy hợp đồng” button
2	Choose “Ngày hủy hợp đồng” the contract cancel date (required)
3	Fill “Lý do hủy hợp đồng” the contract cancel reason (required) Fill “Ghi chú” the contract cancel note (optional)
4	Click “Đồng ý hủy” button

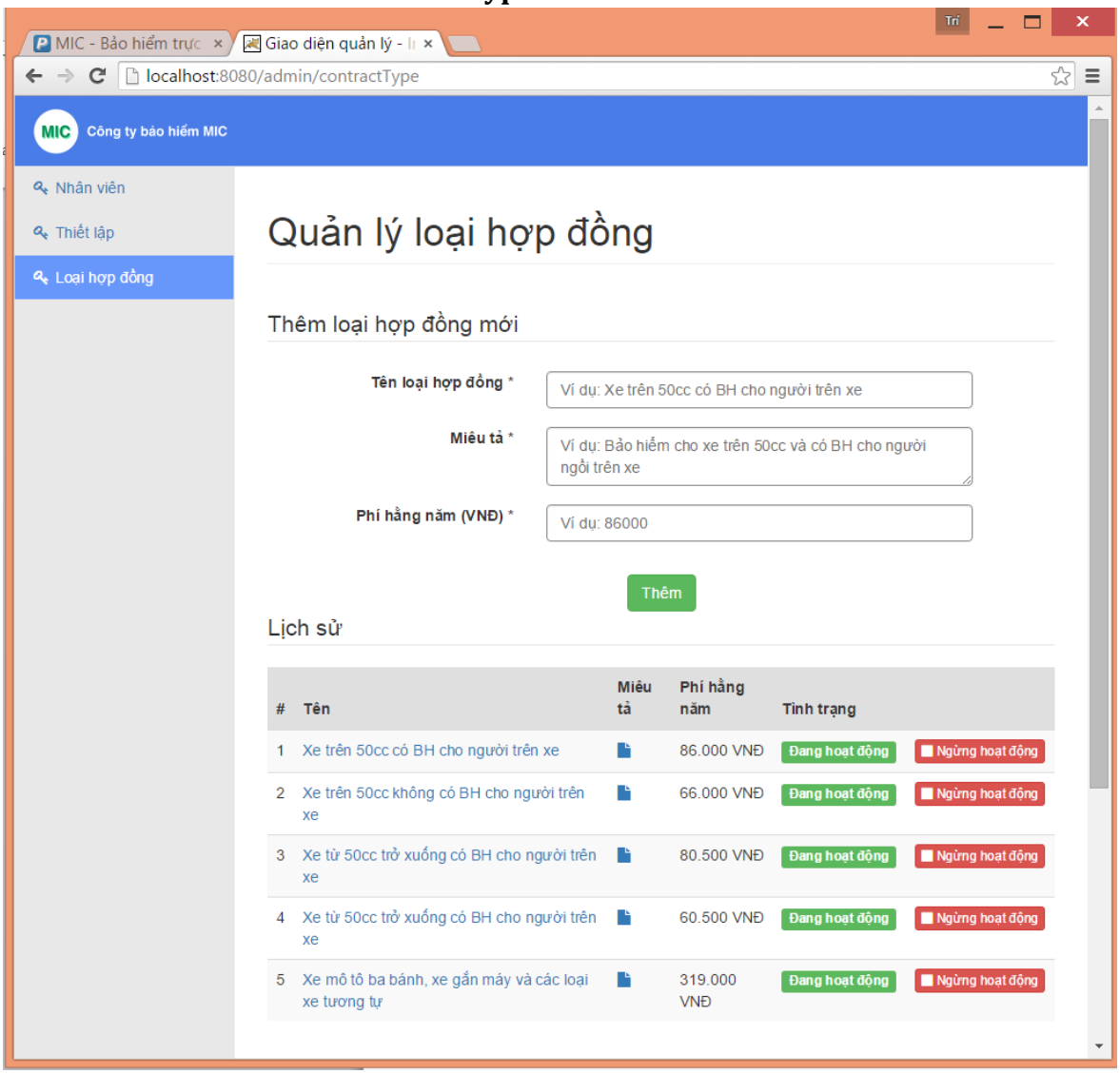
2.1.4. Admin

2.1.4.1.Add staff by admin



Step	Description
1	Fill in fields: “Email”: Staff’s email “Mật khẩu”: Password “Tên nhân viên”: Staff’s name “Số điện thoại”: Phone number
2	Click “Thêm nhân viên” button.

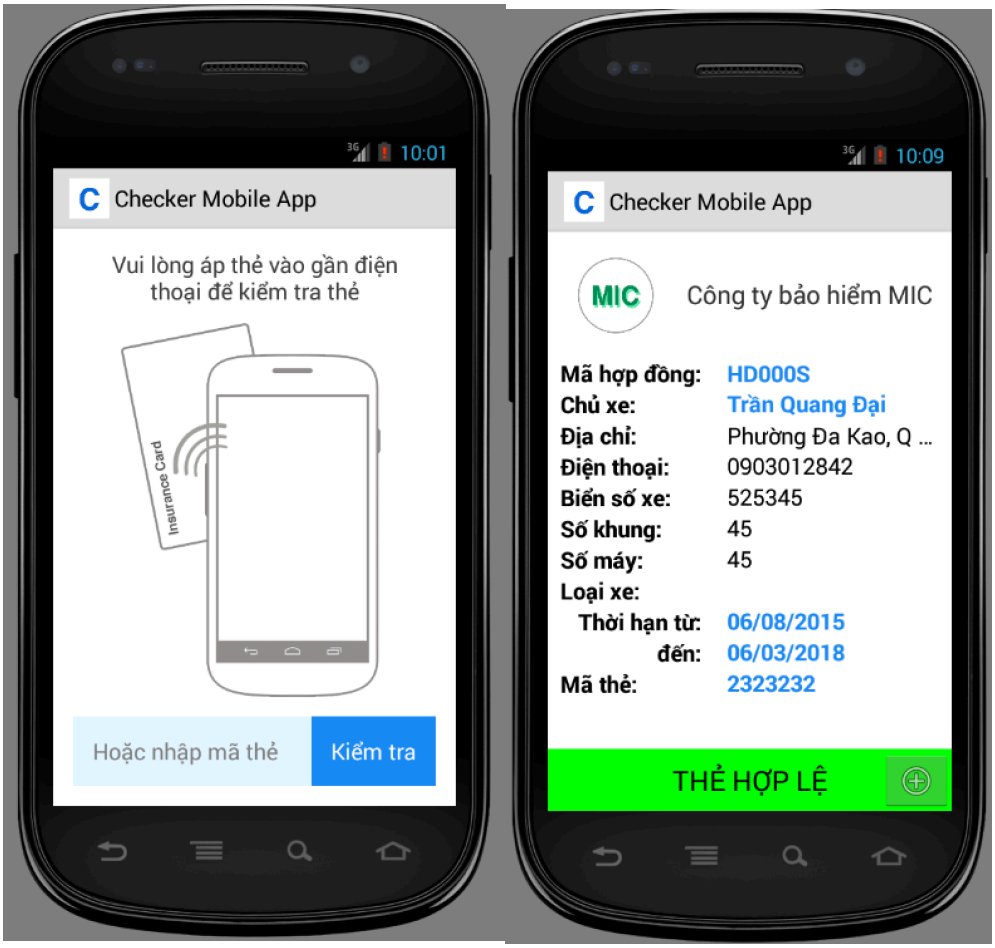
2.1.4.2.Add contract type



Step	Description
1	Fill in fields: “Tên loại hợp đồng”: Contract type’s name “Miêu tả”: Description “Phí hằng năm”: Price per year
2	Click “Thêm” button.

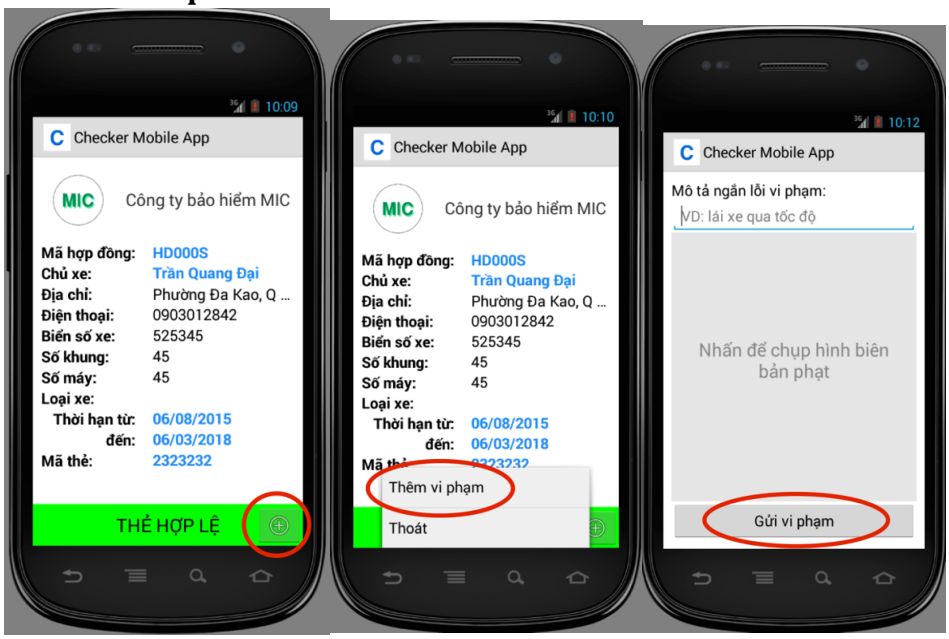
2.2. Checker Application

2.2.1. Validate card



Step	Description
1	Put Card close to the phone
2	Check information from the output screen

2.2.2. Add punishment information



Step	Description
1	At output screen, tap the plus button at bottom right corner.
2	Tap “Thêm vi phạm” button, application open new screen to enter punishment information
3	Press “Gửi vi phạm” to send punishment information

2.3. Printer Application

2.3.1. Print card

Step	Description
1	At home screen, select a contract from list.
2	Tap “Lưu thông tin vào thẻ” button
3	Put NFC card close to phone, application show print result

G. Appendix

1. SOFTWARE ENGINEERING 9th Edition, by Ian Sommerville.
2. Code Conventions for the Java TM Programming Language, by Sun Microsystems, rev April 20, 1999.
3. Android Developer Guide - Application Fundamentals http://developer.android.com/guide/components/fundamentals.html
4. NFC ForumTM Type 3 Tag Operation Specification 1.1, 2011-06-28.
5. Circular No. 126/2008/TT-BTC: On rules, terms, table of premium rates and levels of liability of compulsory insurance for civil liability of motor vehicle owners - The Ministry of Finance - Hanoi, December 22, 2008. http://thuvienphapluat.vn/van-ban/Bao-hiem/Thong-tu-126-2008-TT-BTC-quy-tac-dieu-khoan-bieu-phi-muc-trach-nhiem-bao-hiem-bat-buoc-dan-su-chu-xe-co-gioi-83284.aspx