**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

[**FPT UNIVERSITY** 1](#_Toc421171233)

[Capstone Project Document 1](#_Toc421171234)

[*This page is intentionally left blank* 2](#_Toc421171235)

[Table of Contents 3](#_Toc421171236)

[List of Tables 5](#_Toc421171237)

[List of Figures 7](#_Toc421171238)

[Definitions, Acronyms, and Abbreviations 9](#_Toc421171239)

[A. Introduction 10](#_Toc421171240)

[1. Project Information 10](#_Toc421171241)

[2. Introduction 10](#_Toc421171242)

[3. Current Situation 10](#_Toc421171243)

[4. Problem Definition 10](#_Toc421171244)

[5. Proposed Solution 11](#_Toc421171245)

[5.1. Feature functions 11](#_Toc421171246)

[5.2. Advantages and disadvantages 11](#_Toc421171247)

[6. Functional Requirements 12](#_Toc421171248)

[7. Roles and Responsibility 13](#_Toc421171249)

[B. Software Project Management Plan 14](#_Toc421171250)

[1. Problem Definition 14](#_Toc421171251)

[1.1. Name of this Capstone Project 14](#_Toc421171252)

[1.2. Problem Abstract 14](#_Toc421171253)

[1.3. Project Overview 14](#_Toc421171254)

[2. Project organization 17](#_Toc421171255)

[2.1. Software Process Model 17](#_Toc421171256)

[Figure 1: Waterfall model 18](#_Toc421171257)

[2.2. Roles and responsibilities 18](#_Toc421171258)

[2.3. Tools and Techniques 19](#_Toc421171259)

[3. Project Management Plan 19](#_Toc421171260)

[3.1. Software development life cycle 19](#_Toc421171261)

[3.2. Phase Detail 21](#_Toc421171262)

[3.3. Task sheet 22](#_Toc421171263)

[3.4. All Meeting Minutes 22](#_Toc421171264)

[4. Coding Convention 22](#_Toc421171265)

[C. Software Requirement Specification 24](#_Toc421171266)

[1. User Requirement Specification 24](#_Toc421171267)

[1.1. Customer requirement 24](#_Toc421171268)

[1.2. Staff requirement 24](#_Toc421171269)

[1.3. Police requirement 25](#_Toc421171270)

[1.4. Admin requirement 25](#_Toc421171271)

[2. System Requirement Specification 25](#_Toc421171272)

[2.1. External Interface Requirement 25](#_Toc421171273)

[2.2. System Overview Use Case 26](#_Toc421171274)

[2.3. List of Use Case 28](#_Toc421171275)

[3. Software System Attribute 93](#_Toc421171276)

[3.1. Usability 93](#_Toc421171277)

[3.2. Reliability 93](#_Toc421171278)

[3.3. Availability 93](#_Toc421171279)

[- N/A 93](#_Toc421171280)

[3.4. Security 94](#_Toc421171281)

[3.5. Maintainability 94](#_Toc421171282)

[3.6. Portability 94](#_Toc421171283)

[3.7. Performance 94](#_Toc421171284)

[4. Conceptual Diagram 94](#_Toc421171285)

List of Tables

[Table 1: Definitions, Acronyms, and Abbreviations 7](#_Toc421138201)

[Table 2 Roles and Responsibility 11](#_Toc421138202)

[Table 3 Hardware requirement for continuous integrating server 14](#_Toc421138203)

[Table 4 Hardware requirement for web development 15](#_Toc421138204)

[Table 5 Hardware requirement for mobile development 15](#_Toc421138205)

[Table 6 Software requirement 15](#_Toc421138206)

[Table 7 Roles and responsibilities 17](#_Toc421138207)

[Table 8 Tools and Techniques 17](#_Toc421138208)

[Table 9 Use case WG01 - Login 25](#_Toc421138209)

[Table 10 Use case WG02 - Create new contract request 28](#_Toc421138210)

[Table 11 Use case WC01 - Logout 29](#_Toc421138211)

[Table 12 Use case WC02 - Use Case Name Renew contract by user 31](#_Toc421138212)

[Table 13 Use case WC03 - Cancel contract 32](#_Toc421138213)

[Table 14 Use case WC04 - View payment history 33](#_Toc421138214)

[Table 15 Use case WC05 - View accident history 34](#_Toc421138215)

[Table 16 Use case WC06 - View punishment history 36](#_Toc421138216)

[Table 17 Use case WC07 - Request compensation 38](#_Toc421138217)

[Table 18 Use case WC08 - View compensation history 39](#_Toc421138218)

[Table 19 Use case WC09 - New card request 41](#_Toc421138219)

[Table 20 Use case WC10 - View card information 42](#_Toc421138220)

[Table 21 Use case WC11 - View personal information 44](#_Toc421138221)

[Table 22 Use case WC12 - Change password 45](#_Toc421138222)

[Table 23 Use case WS01 - <Staff> View profile 48](#_Toc421138223)

[Table 24 Use case WS02 - <Staff> Change password 49](#_Toc421138224)

[Table 25 Use case WS03 - <Staff> Resolve new card request 51](#_Toc421138225)

[Table 26 Use case WS04 - <Staff> Resolve compensation request 53](#_Toc421138226)

[Table 27 Use case WS05 - <Staff> View customer information 55](#_Toc421138227)

[Table 28 Use case WS06 - <Staff> Edit customer information 56](#_Toc421138228)

[Table 29 Use case WS07 - <Staff> Create new customer 58](#_Toc421138229)

[Table 30 Use case WS08 - <Staff> Create new contract 61](#_Toc421138230)

[Table 31 Use case WS09 - <Staff> Renew contract 62](#_Toc421138231)

[Table 32 Use case WS10 - <Staff> Cancel contract 64](#_Toc421138232)

[Table 33 Use case WS11 - <Staff> Update contract information 65](#_Toc421138233)

[Table 34 Use case WS12 - <Staff> Update compensation history 67](#_Toc421138234)

[Table 35 Use case WS13 - <Staff> Update punishment history 68](#_Toc421138235)

[Table 36 Use case WS14 - <Staff> Update accident history 70](#_Toc421138236)

[Table 37 Use case WS15 - <Staff> View cards information 71](#_Toc421138237)

[Table 38 Use case WS16 - <Staff> Update contract type information 72](#_Toc421138238)

[Table 39 Use case WA01 - <Admin> Remove staff 74](#_Toc421138239)

[Table 40 Use case WA02 - <Admin> Add staff 75](#_Toc421138240)

[Table 41 Use case WY01 - <System> Notify schedule 77](#_Toc421138241)

[Table 42 Use case WP01 - <Payment> Payment 79](#_Toc421138242)

[Table 43 Use case CP01 - <Police> Verify card information 82](#_Toc421138243)

[Table 44 Use case CP02 - <Police> Add punishment information 83](#_Toc421138244)

[Table 45 Use case PS01 - <Staff> Search / filter contract 85](#_Toc421138245)

[Table 46 Use case PS02 - <Staff> View contract information 86](#_Toc421138246)

[Table 47 Use case PS03 - <Staff> Print information to NFC card 88](#_Toc421138247)

List of Figures

[Figure 1: Waterfall model 17](#_Toc421139386)

[Figure 2: Web Application Overview Use Case 25](#_Toc421139387)

[Figure 3: Checker Mobile Application 26](#_Toc421139388)

[Figure 4: Printer Mobile Application 26](#_Toc421139389)

[Figure 5 <Guest> Overview Use Case 27](#_Toc421139390)

[Figure 6 <Guest> Login 27](#_Toc421139391)

[Figure 7 <Guest> Create new contract request 28](#_Toc421139392)

[Figure 8 <Customer> Overview Use Case 32](#_Toc421139393)

[Figure 9 <Customer> Renew contract 33](#_Toc421139394)

[Figure 10 <Customer> Cancel contract 35](#_Toc421139395)

[Figure 11 <Customer> View payment history 36](#_Toc421139396)

[Figure 12 <Customer> View accident history 37](#_Toc421139397)

[Figure 13 <Customer> View punishment history 39](#_Toc421139398)

[Figure 14 <Customer> Request compensation 40](#_Toc421139399)

[Figure 15 <Customer> View compensation history 42](#_Toc421139400)

[Figure 16 <Customer> New card request 44](#_Toc421139401)

[Figure 17 <Customer> View card information 45](#_Toc421139402)

[Figure 18 <Customer> View personal information 47](#_Toc421139403)

[Figure 19 <Customer> Change password 48](#_Toc421139404)

[Figure 20 <Staff> Overview Use Case 50](#_Toc421139405)

[Figure 21 <Staff> View profile 51](#_Toc421139406)

[Figure 22 <Staff> Change password 52](#_Toc421139407)

[Figure 23 <Staff> Resolve new card request 53](#_Toc421139408)

[Figure 24 <Staff> Resolve compensation request 55](#_Toc421139409)

[Figure 25 <Staff> View customer information 57](#_Toc421139410)

[Figure 26 <Staff> Edit customer information 59](#_Toc421139411)

[Figure 27 <Staff> Create new customer 61](#_Toc421139412)

[Figure 28 <Staff> Create contract 62](#_Toc421139413)

[Figure 29 <Staff> Renew contract 65](#_Toc421139414)

[Figure 30 <Staff> Cancel contract 66](#_Toc421139415)

[Figure 31 <Staff> Update contract information 68](#_Toc421139416)

[Figure 32 <Staff> Update compensation history 70](#_Toc421139417)

[Figure 33 <Staff> Update punishment history 71](#_Toc421139418)

[Figure 34 <Staff> Update accident history 73](#_Toc421139419)

[Figure 35 <Staff> View card information 74](#_Toc421139420)

[Figure 36 <Staff> Update contract type information 75](#_Toc421139421)

[Figure 37 <Admin> Overview Use Case 77](#_Toc421139422)

[Figure 38 <Admin> Remove staff 77](#_Toc421139423)

[Figure 39 <Admin> Add staff 78](#_Toc421139424)

[Figure 40 <System> Overview Use Case 80](#_Toc421139425)

[Figure 41 <System> Notify schedule 80](#_Toc421139426)

[Figure 42 <Payment> Payment 82](#_Toc421139427)

[Figure 43 <Payment> Payment 82](#_Toc421139428)

[Figure 44 <Police> Overview Use Case 84](#_Toc421139429)

[Figure 45 <Police> Verify card information 84](#_Toc421139430)

[Figure 46 <Police> Add punishment information 86](#_Toc421139431)

[Figure 47 <Staff> Overview Use Case 87](#_Toc421139432)

[Figure 48 <Staff> Search / filter contract 88](#_Toc421139433)

[Figure 49 <Staff> View contract information 89](#_Toc421139434)

[Figure 50 <Staff> Print information to NFC card 91](#_Toc421139435)

[Figure 51 Conceptual diagram 94](#_Toc421139436)

Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Name | Definition |
| MIC | Motor Insurance Card |
| NFC | Near field communication |
|  |  |

Table : Definitions, Acronyms, and Abbreviations

# Introduction

## Project Information

* Project name: **Insurance Card**
* Project Code: **MIC**
* Product Type: **Website & Android Application**
* Start Date: **May 11th, 2015**
* End Date: **September 5­th,2015**

## 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.

## Current Situation

When participating in traffic, vehicle owners is required to have compulsory insurance. 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.

## 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.
* **Insurance certificate made of paper:** It could be torn, wet, smudged and especially is counterfeited.
* **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.

## Proposed Solution

The proposed solution is to build an insurance NFC card system, include a web application and 2 mobile applications with following functions:

### 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 sends an email to notify the insured one when their insurance is expired.
  + Notify compensation state: system will sends 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.

### 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. And it is difficult to be counterfeit than insurance card made of paper.
  + Support police to check valid insurance card easier.
* Disadvantages:
  + At the present time, 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.

## Functional Requirements

Function requirements of the system are listed as below:

* **User component:**
  + New contract request
  + Check card information.
  + Renew contract.
  + Request compensation.
  + Lost card request
  + Cancel contract
* **Staff component**
  + Create new contracts
  + Manage contracts.
  + Resolve compensation requests.
  + Resolve lost 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.

## 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 Roles and Responsibility

# Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

* **Official name**: Insurance Card
* **Vietnamese name**: Thẻ bảo hiểm
* **Abbreviation**: MIC

### 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. The NFC card is supplied by insurance company when the customer buy insurance. The card contain the information of customer, if the customer join with many insurance service they just use only one card.

We provide a software to check the validation of card, the deadline of card and some insurance services that customer joined. We also provide some other advantage 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.

### Project Overview

#### Current Situation

Below is 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.
* At the present time, not consistent with the law of Vietnam about insurance card issues.

#### 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.

##### 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.

##### Checker Mobile Application

This mobile application is used by traffic officer. This application do 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.

##### 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.

#### Boundaries of the System

This section suppose that the government law 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

#### Development Environment

##### 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 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 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 Hardware requirement for mobile development

##### 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 Software requirement

## Project organization

### Software Process Model

This project is developed under waterfall model, we applies waterfall model with customizes 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 : Waterfall model

Reference: 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.

### 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 Roles and responsibilities

### 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 | Microsoft Visio 2013 |
| Document tool | Microsoft Word 2013 |

Table Tools and Techniques

## Project Management Plan

### 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. |

### Phase Detail

#### 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 |

#### 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 |

#### 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 |

#### 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 |

#### 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 |

### Task sheet

Refer to “Task sheet” folder.

### All Meeting Minutes

Refer to “Meeting minutes” folder.

## 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 4 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.

# Software Requirement Specification

## User Requirement Specification

### 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

### 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

### 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

### Admin requirement

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

* Manage staff includes:
* Remove staff
* Add staff

## System Requirement Specification

### External Interface Requirement

#### 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.

#### Hardware Interface

* N/A

#### 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).

#### 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.

### System Overview Use Case

#### Web Application

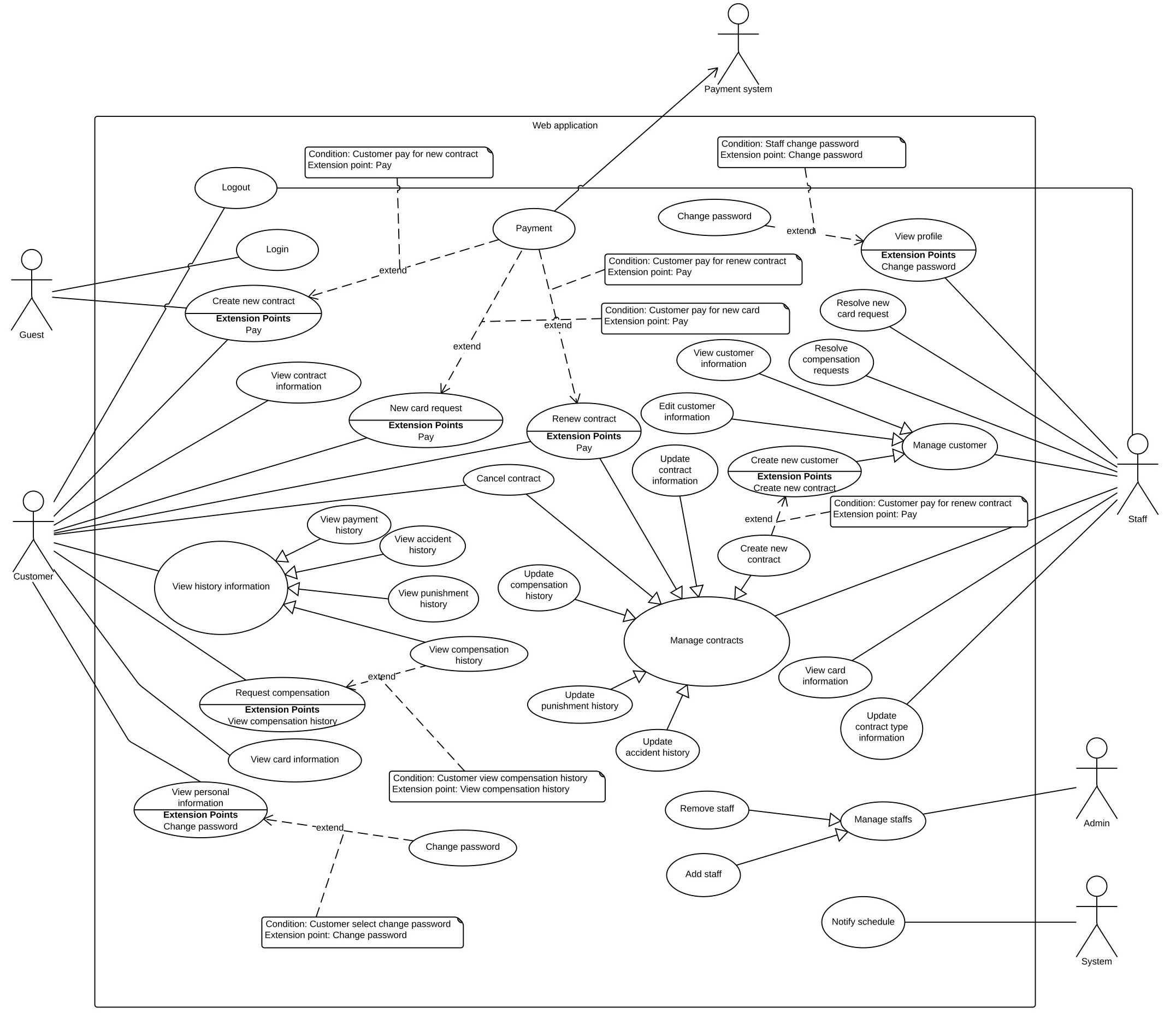


Figure : Web Application Overview Use Case

#### Checker Mobile Application

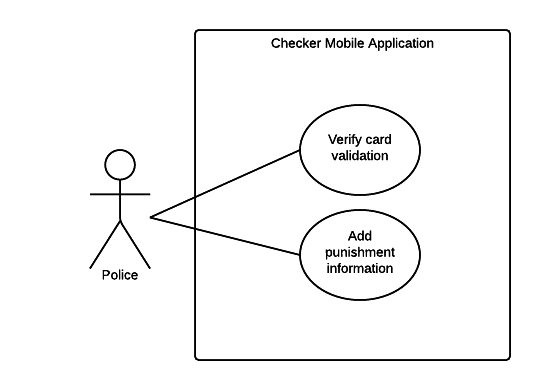


Figure : Checker Mobile Application

#### Printer Mobile Application



Figure : Printer Mobile Application

### List of Use Case

#### Web Application

##### <Guest> Overview Use Case

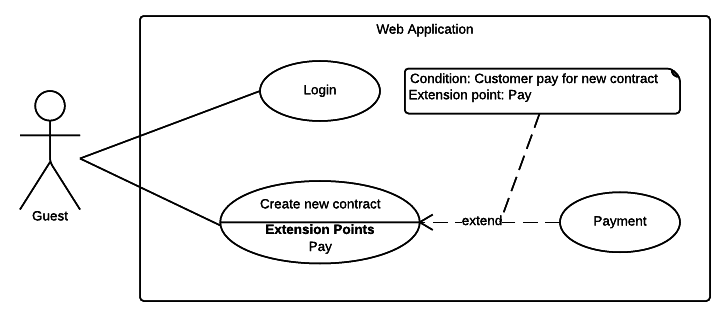


Figure <Guest> Overview Use Case

###### <Guest> Login



Figure <Guest> Login

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WG01 | | | |
| Use Case No. | WG01 | **Use Case Version** | 2.0 |
| Use Case Name | Login | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | Normal |
| Actor:   * Guest   Summary:   * This use case allows guest to log in the system.   Goal:   * Guest can log in the system.   Triggers:   * Guest sends the login command.   Preconditions:   * N/A   Post Conditions:   * Success: Guest login the system. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to login view. | System requires identity information from Guest:   * Email or customer code: free text input, required, length 3 – 250 * Password: free text input, required, length 6 – 32 | | 2 | Guest inputs information. |  | | 3 | Guest sends command to login to system | Guest will login system with their specific role  [Alternative 1]  [Exception 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest enter wrong identity information. | Wrong identity information, System shows error message. |   Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Missing of required fields | System notify guest to enter their identity information |   Relationships: N/A  Business Rules:   * After login to system, guest will be redirected to specific view based on their role on the system: staff or customer. * If role is “Customer”, the system will display to Customer view. * If role is “Staff”, the system will display to Staff Dashboard view. | | | |

Table Use case WG01 - <Guest> Login

###### <Guest> Create new contract request

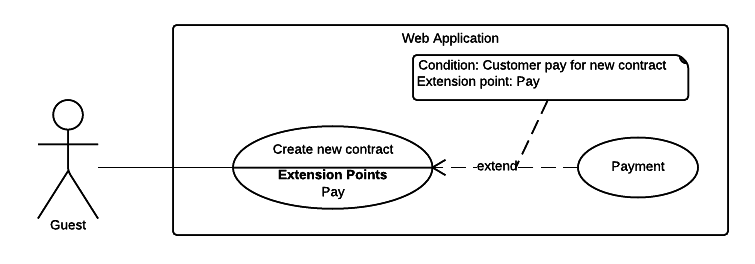


Figure <Guest> Create new contract request

|  |  |  |  |
| --- | --- | --- | --- |
| 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 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 2200. * Weight: free text input, value from 1 – 1000, unit: kilogram * Seat capacity: free text input, value from 1 – 100. | | 2 | Guest inputs information. |  | | 3 | Guest sends command to create new contract request. | System validate information, display contract details and request for confirmation.  [Exception 1, 2, 3] | | 4 | Guest sends command to create new contract request. | 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. | | 5 | If user chooses PayPal gateway and sends confirm command.  [Alternative 1] | Forward to PayPal payment view to process the payment. | | 6 | User process the PayPal payment | If payment succeed:  Add new contract information to the system.  Show message created successful.  [Exception 4] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If user chooses direct payment method and sends confirm command. | Show list of company brands address. |   Exceptions:   |  |  |  | | --- | --- | --- | | Step | 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 contract will be created in the system with inputted information. * Contract’s type must be 1 of 5 below:   + “Xe trên 50cc có BH cho người trên xe”   + “Xe trên 50cc không có BH cho người trên xe”   + “Xe dưới 50cc có BH cho người trên xe”   + “Xe dưới 50cc không có BH cho người trên xe”   + “Xe mô tô ba bánh, xe gắn máy và các loại xe tương tự” * The initial status of contract will be set to “Pending”. * When customer completed payment process, contract’s status would change from “Pending” to “No Card”. * If the contract start date is not come yet, the contract status is set to “Pending”. * 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. * Expired date and start date must not exceed 1 year. * Contract end date must not be earlier start date. * Contract price (per year) would be calculated from contract type and sets by staff. | | | |

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

##### <Customer> Overview Use Case

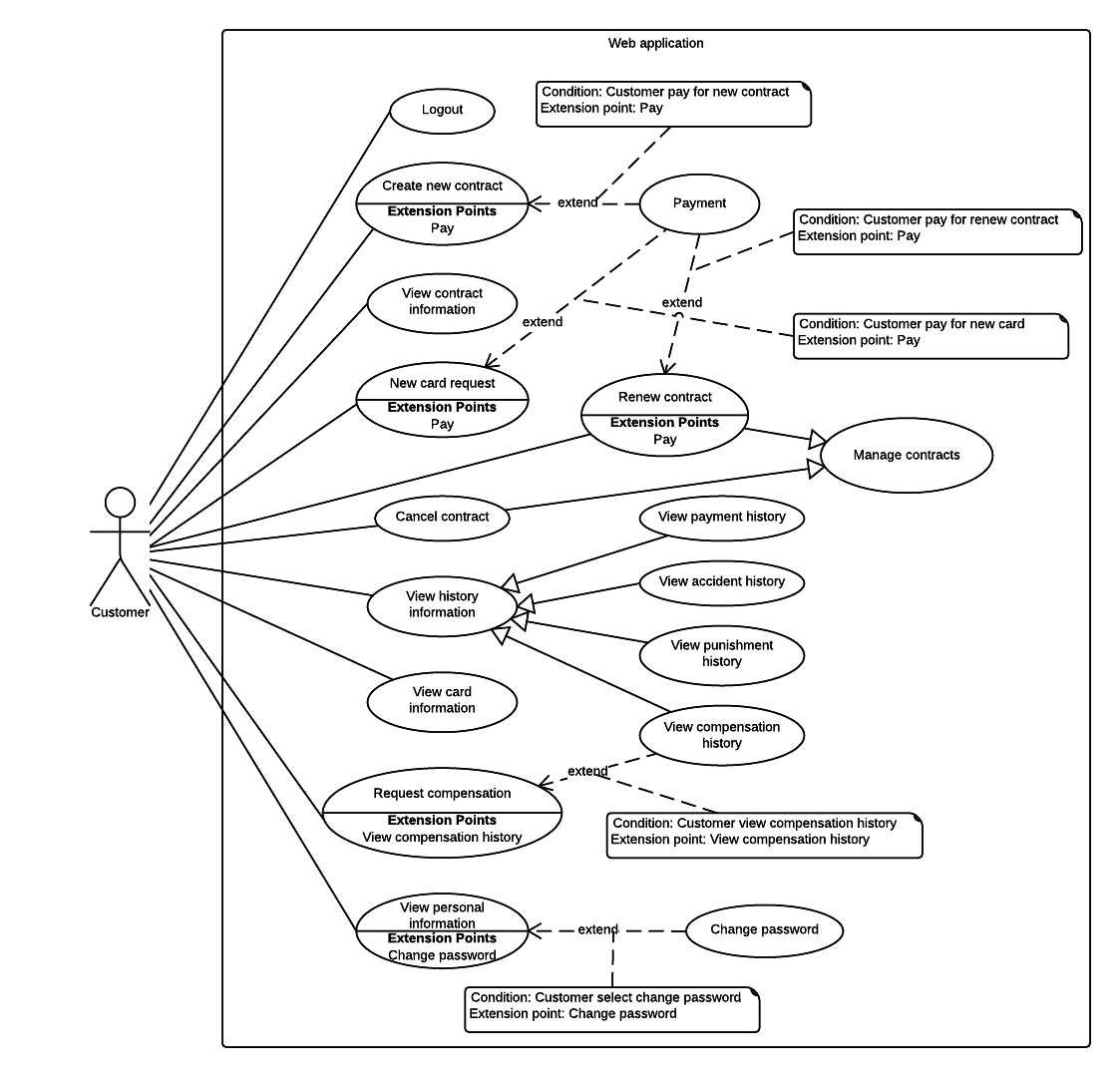


Figure <Customer> Overview Use Case

###### <Customer> Logout

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC01 | | | |
| Use Case No. | WC01 | **Use Case Version** | 2.0 |
| Use Case Name | Logout | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer, staff.   Summary:   * This use case allows user to log out the system.   Goal:   * User can log out the system.   Triggers:   * User sends the logout command.   Preconditions:   * User logged in the system.   Post Conditions:   * Success: User logs out the system. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends logout command. | User exits from the system. |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * After user logged out, return to the home. | | | |

Table Use case WC01 - <Customer> Logout

###### <Customer> Renew contract

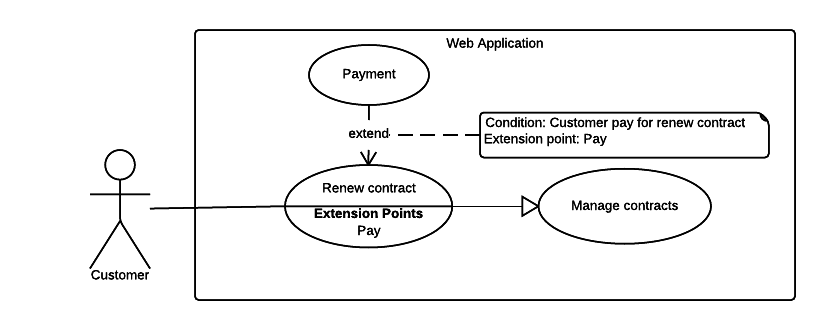


Figure <Customer> Renew contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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 expired date is less than 2 months. * Contract’s status must be “No Card”, “Ready” or “Expired”.   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:  - 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. * A notification will be sent to staff after the process is completed. | | | |

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

###### <Customer> Cancel contract

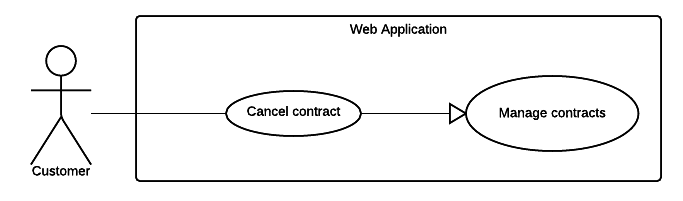


Figure <Customer> Cancel contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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:   + “Xe cơ giới bị thu hồi đăng ký”   + “Xe cơ giới hết niên hạn”   + 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 Use case WC03 - <Customer> Cancel contract

###### <Customer> View payment history

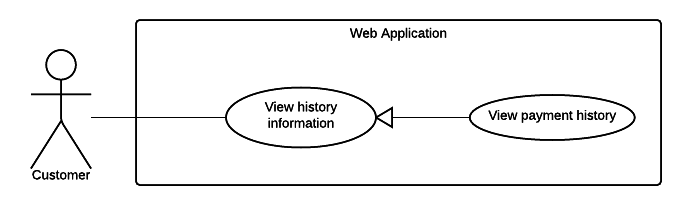


Figure <Customer> View payment history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC04 | | | |
| Use Case No. | WC04 | **Use Case Version** | 2.0 |
| Use Case Name | View payment history | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case shows user the history of customer’s transactions.   Goal:   * User can view list of transactions were made.   Triggers:   * User sends view payment history command.   Preconditions:   * User must login into the system with role Customer. * There is at least 01 transaction belongs to customer in the system.   Post Conditions:   * Success: Show the customer’s transactions history to the user. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view payment history command. | Display new view that show user list of transaction history information includes:   * Date of transaction: text * Payment method: text * Service: text * Amount: text * Receiver: text * PayPal transaction ID: text   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If there are no transaction was made. | Show message to notify that there are no transaction was made. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The information is loaded from the system. * List of transactions will be sorted by date in descending order. * Pagination must be displayed if number of transactions larger than 10. * Search bar on the top help user finding transactions faster. | | | |

Table Use case WC04 - <Customer> View payment history

###### <Customer> View accident history

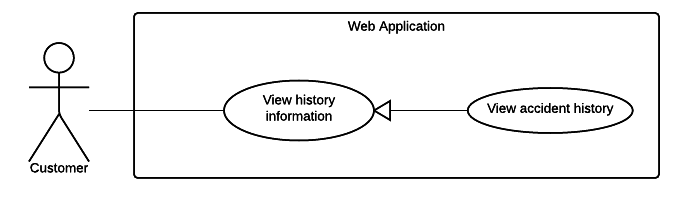


Figure <Customer> View accident history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC05 | | | |
| Use Case No. | WC05 | **Use Case Version** | 2.0 |
| Use Case Name | View accident history | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case shows user the history of customer’s accidents.   Goal:   * User can view list of their accidents history.   Triggers:   * User sends view accident history command.   Preconditions:   * User must login into the system with role Customer. * There is at least 01 accident belongs to customer in the system.   Post Conditions:   * Success: Show the customer’s accidents history to the user. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view accident history command | Display new view show list of accidents information includes:   * Date of accident: text * Title: text * Attachment file: link to file.   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | There are no accident. | Show message to notify that there are no accident. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The information is loaded from the system. * List of accidents will be sorted by date in descending order. * Pagination must be displayed if number of accidents larger than 10. * Search bar on the top help user to search accident by title. | | | |

Table Use case WC05 - <Customer> View accident history

###### <Customer> View punishment history

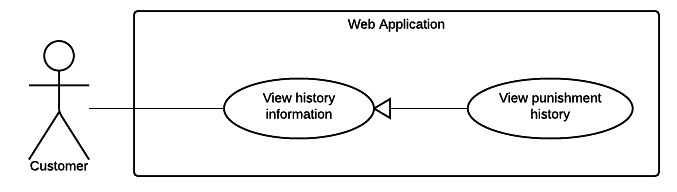


Figure <Customer> View punishment history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC06 | | | |
| Use Case No. | WC06 | **Use Case Version** | 2.0 |
| Use Case Name | View punishment history | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case shows user the history of the punishment.   Goal:   * User can view list of their punishment history.   Triggers:   * User sends view punishment history command.   Preconditions:   * User must login into the system with role Customer. * There is at least 01 punishment belongs to customer in the system.   Post Conditions:   * Success: Show the punishment history to the user. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view punishment history command | Display a table shows list of punishments information includes:   * Date of punishment: text * Punishment title: text * Attachment file: link to file.   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | There are no punishment | Show message to notify that there are no punishment. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The information is loaded from the system. * Punishments list is sorted by date order. * Search bar on the top help user finding punishment information faster. * Pagination must be displayed if number of punishments larger than 10. | | | |

Table Use case WC06 - <Customer> View punishment history

###### <Customer> Request compensation

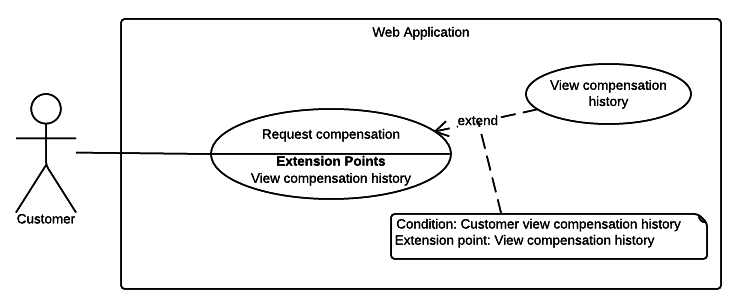


Figure <Customer> Request compensation

|  |  |  |  |
| --- | --- | --- | --- |
| 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.   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 Use case WC07 - <Customer> Request compensation

###### <Customer> View compensation history

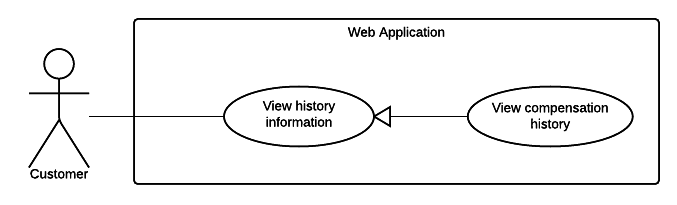


Figure <Customer> View compensation history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC08 | | | |
| Use Case No. | WC08 | **Use Case Version** | 2.0 |
| Use Case Name | View compensation history. | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case helps user to view compensation history.   Goal:   * User can view compensation history.   Triggers:   * User sends view compensation history command.   Preconditions:   * User must login into the system with role Customer. * There is at least 01 compensation belongs to customer in the system.   Post Conditions:   * Success: Show the compensation history to the user. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view compensation history command. | Display new view show a table of compensations includes:   * Compensation code: text. * Contract code: text. * Customer name: text. * Request date: text. * Resolve date: text. | | 2 | User selects a compensation to view its details. | Display new view that show the detail of the selected compensation includes:   * Driver name: text. * License number: text. * License type: text. * Driver phone: text. * Vehicle capacity: text. * Driver address: text. * Plate number of accident motor: text. * Date of accident: text. * Place of accident: text. * Control Police Department: text. * Description: text. * Human damage: text. * Asset damage: text. * Observer: text. * Compensation note: text. * Attachment file: link to file. * Created date: text. * Resolve date: text. * Decision: text. * Resolve note: text.   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | There are no compensation | Show message to notify that there are no compensation. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The information is loaded from the system. * List of compensation is sorted by resolve date in decreasing order. The unsolved compensations will be the top rows. * Pagination must be displayed if number of compensations larger than 10. * Search bar on the top help user to finding compensation faster. | | | |

Table Use case WC08 - <Customer> View compensation history

###### <Customer> New card request

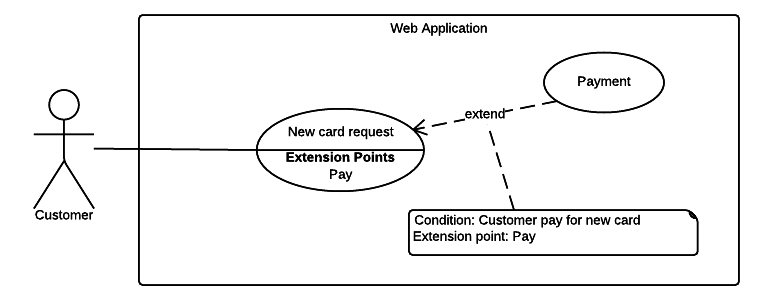


Figure <Customer> New card request

|  |  |  |  |
| --- | --- | --- | --- |
| 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’s contract has not expired. * User must has a card in the system.   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 user:   * A text box to confirm by password: free text input, required, length 6-32. * Payment gateways: can be optional selected from these selections:   + PayPal payment gateway.   + Direct payment. * The new card 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 payment is succeed and the card will be sent to customer soon.  [Exception 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User choose the direct payment method. | Show company information: address, telephone number.  [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: N/A  Business Rules:   * 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 Use case WC09 - <Customer> New card request

###### <Customer> View card information

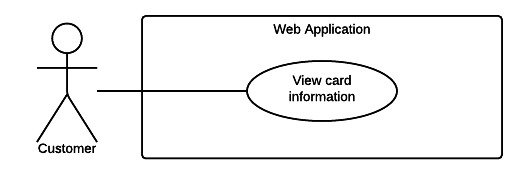


Figure <Customer> View card information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC10 | | | |
| Use Case No. | WC10 | **Use Case Version** | 2.0 |
| Use Case Name | View card information | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case shows user their card information.   Goal:   * User can view their card information.   Triggers:   * User sends view card information command.   Preconditions:   * User must login into the system with role Customer. * There is at least 01 card belongs to customer in the system.   Post Conditions:   * Success: Show the user’s card information and the access history. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view card information command | Display new view shows the card information includes:   * Card ID: text * User name: text * Card status: text * Activated date: text   And a table shows list of access history information includes:   * Access date: text * Access device: text * Request: text * Response: text |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | There are no card | Show message to notify that there are no card. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The information is loaded from the system. * List of access history is sorted by access date in descending order. * Pagination must be displayed if number of access logs larger than 10. * Filter bar on the top help user to finding access logs faster. | | | |

Table Use case WC10 - <Customer> View card information

###### <Customer> View personal information



Figure <Customer> View personal information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC11 | | | |
| Use Case No. | WC11 | **Use Case Version** | 2.0 |
| Use Case Name | View personal information | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | Medium |
| Actor:   * Customer.   Summary:   * This use case shows user their personal information.   Goal:   * User can review their card information.   Triggers:   * User sends view personal information command.   Preconditions:   * User must login into the system with role Customer.   Post Conditions:   * Success: Show the user’s personal information. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User sends view personal information command | Display new view shows the user information includes:   * User code: text * User name: text * Address: text * Email: text * Phone number: text * Personal ID: text |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules: N/A   * The information is loaded from the system. | | | |

Table Use case WC11 - <Customer> View personal information

###### <Customer> Change password



Figure <Customer> Change password

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WC12 | | | |
| Use Case No. | WC12 | **Use Case Version** | 2.0 |
| Use Case Name | Change password. | | |
| Author | TriPQM | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Customer.   Summary:   * This use case helps user to change their password.   Goal:   * User can change the password.   Triggers:   * User sends change password command.   Preconditions:   * User must login into the system.   Post Conditions:   * Success: Update new user’s password to the system. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to change password view. | Display new view contains textboxes for user to input their current and new password:   * Current password: free text input, required, length 6-32 * New password: free text input, required, length 6-32 * Confirm new password: free text input, required, length 6-32 | | 2 | User input their current password, new password and confirm new password. |  | | 3 | User sends change password command. | Show message to notify that the password has been changed successfully.  [Alternative 1, 2] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If user inputs the wrong current password | Show message to notify that the current password is wrong. | | 2 | If the new password and the confirm password not matched. | Show message to notify that the confirm password is not matched with the new password. |   Exceptions: N/A  Relationships: N/A  Business Rules:   * The new password will be saved into the system if the provided information is correct. | | | |

Table Use case WC12 - <Customer> Change password

##### <Staff> Overview Use Case

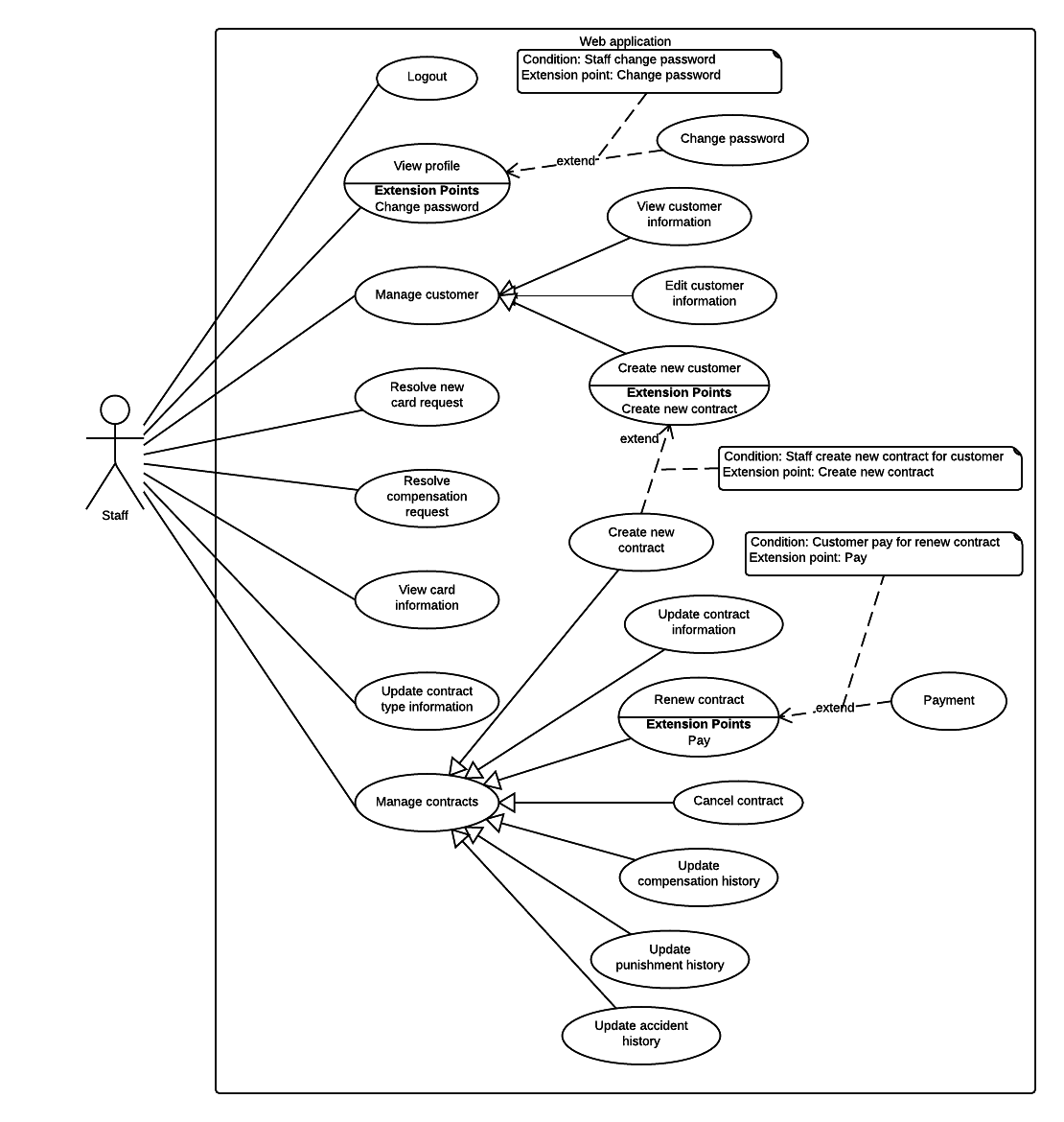


Figure <Staff> Overview Use Case

###### <Staff> View profile



Figure <Staff> View profile

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS01 | | | |
| Use Case No. | WS01 | **Use Case Version** | 2.0 |
| Use Case Name | View profile | | |
| Author | KhaNC | | |
| Date | 27/05/2015 | **Priority** | Normal |
| Actor:   * Staff   Summary:   * This use case allows staff view their profile.   Goal:   * Staff can view their detailed profile.   Triggers:   * Staff goes to view their profile.   Preconditions:   * Staff must login into the system with roles Staff.   Post Conditions:   * Success: Staff profile information is shown. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to view theirprofile. | Staff’s detail profile will be shown with following information:   * Staff’s code: text * Staff’s full name: text * Staff’s email: text * Staff’s phone: text |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * Staff’s profile information is always loaded from the system. | | | |

Table Use case WS01 - <Staff> View profile

###### <Staff> Change password



Figure <Staff> Change password

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS02 | | | |
| Use Case No. | WS02 | **Use Case Version** | 2.0 |
| Use Case Name | Change password | | |
| Author | KhaNC | | |
| Date | 27/05/2015 | **Priority** | Normal |
| Actor:   * Staff   Summary:   * This use case allows staff change their password.   Goal:   * Staff can change their account password.   Triggers:   * Staff sends change password command.   Preconditions:   * Staff must login into the system with roles Staff.   Post Conditions:   * Success: Staff’s new password is updated to the system. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to change password view. | Change password view will be shown with following information:   * Current password: free text input, required, length 6 – 32 * New password: free text input, required, length 6 – 32 * Confirm new password: free text input, required, length 6 – 32 | | 2 | User enters required information |  | | 3 | User sends change password command. | * Validate data   [Alternative 1, 2]  [Exception 1, 2]   * Update staff’s new password into the system. * Display message notify user their password has changed successfully. |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Current password does not match with staff’s password in the system | Show message notify staff entered current password does not match with staff’s password in the system | | 2 | Confirm new password does not match with new password | Show message notify staff entered confirm password is not match with their new password |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Missing of required fields | Show message to notify staff input missed fields | | 2 | Length of new password not in range | Show message notify staff’s new password is out of range |   Relationships: N/A  Business Rules:   * In case of successful scenario, staff’s new password would be updated to the system. * The user should be able to know how strong their password is when they input their new password. * Passwords could be rated in one of three ways: “Weak”, “Moderate” and “Strong”. | | | |

Table Use case WS02 - <Staff> Change password

###### <Staff> Resolve new card request

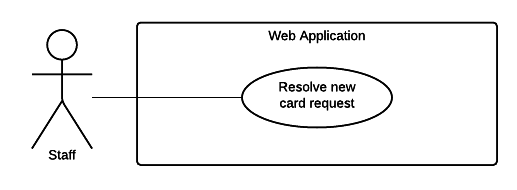


Figure <Staff> Resolve new card request

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS03 | | | |
| Use Case No. | WS03 | **Use Case Version** | 2.0 |
| 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 01 unsolved request for new card from customer. * The payment process for new card was completed.   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 lost 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 date of request in descending order. * The lost card will changes status from “Ready” to “Deactivated”. * The new card will has status is “Ready”. * Search bar on the top help user finding card or customer faster. * Pagination must be display if number of requests larger than 10 and auto-change based on staff’s selection. * Staff can select how many requests should be displayed in user view, default is 10 requests. * A solved request must have code of new card and the date this card issued. * A notification will be sent to request customer after the process is completed. | | | |

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

###### <Staff> Resolve compensation request

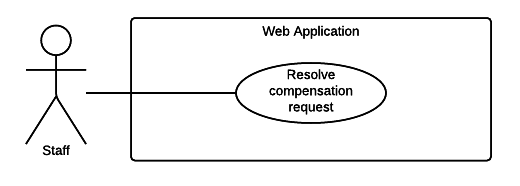


Figure <Staff> Resolve compensation request

|  |  |  |  |
| --- | --- | --- | --- |
| 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 01 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:   * Request’s code: link to request detail * Contract’s code: link to contract detail * Contract’s owner full name: link to customer detail * Get request date: text * Request solved date: text | | 2 | Staff select an unsolved request to view and resolve. | 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. * Driver’s full name: text * Driver’s address: text * Driver’s phone number: text * Driver’s license number: text * Driver’s license type: text * Vehicle’s capacity or seat capacity: text * Accident vehicle plate number: 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 * Detail of compensation request from customer: text * Note for compensation from staff: free text input. * Decision: select one of the options * Status of compensation: select one of the options. | | 3 | Staff input resolve compensation request information, included decision, description and request’s status. |  | | 4 | Staff sends save changes command | Update compensation status and notify to customer |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | There are no unsolved compensation request | Show message to notify staff there are no unsolved 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 get request date in descending order. * With unsolved request, its solved date does not exist. * A compensation request could have one of three decision is: * “Chưa quyết định” * “Chấp nhận bồi thường” * “Từ chối bồi thường” * A compensation request could have one of three status is: * “Đang xử lý” * “Hoàn tất” * “Hủy bỏ” * Search bar on the top help user finding contract or customer faster. * Pagination must be display if number of requests larger than 10 and auto-change based on staff’s selection. * Staff can select how many requests should be displayed in user view, default is 10 requests. * A notification will be sent to request customer after the process is completed. | | | |

Table Use case WS04 - <Staff> Resolve compensation request

###### <Staff> View customer information

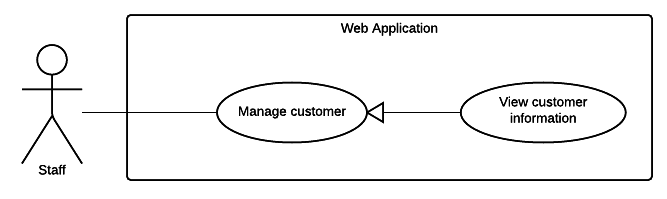


Figure <Staff> View customer information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS05 | | | |
| Use Case No. | WS05 | **Use Case Version** | 2.0 |
| Use Case Name | View customer information | | |
| Author | KhaNC | | |
| Date | 20/05/2015 | **Priority** | Medium |
| Actor:   * Staff.   Summary:   * This use case allows staff view customer details.   Goal:   * Staff can view customer’s detail information.   Triggers:   * Staff sends view customer information command.   Preconditions:   * User must login into the system with role Staff.   Post Conditions:   * Success: Customer’s detail information is shown. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to list of customer. | List of customer will be shown with following information:   * Customer’s code: link to customer detail * Customer’s full name: text * Customer’s phone: text * Customer’s NFC card ID: link to card detail. * Customer’s contract code: link to contract detail. | | 2 | Staff selects customer to view their detail information. | Customer detail will be shown with following information:   * Customer’s code: text * Customer’s full name: text * Customer’s address: text * Customer’s email address: text * Customer’s phone: text * Customer’s personal ID: text   List of customer’s card:   * Customer’s NFC card ID: link to card detail. * Card’s activated date: text * Card’s most recent access date: text * Card’s access history: link to card access history.   List of customer’s contract:   * Customer’s contract code: link to contract detail. * Customer’s contract type: text * Contract’s expired date: text * Contract’s status: text |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * List of customer and their detail information is always loaded from the system. * Staff can view all customer’s card and each card’s access history. * Search bar on the top help user find customer faster. * Pagination must be display if number of requests larger than 10 and auto-change based on staff’s selection. * Staff can select how many requests should be displayed in user view, default is 10 requests. * Contract’s status is one of 6 below: * Pending * No card * Ready * Expired * Request cancel * Cancelled | | | |

Table Use case WS05 - <Staff> View customer information

###### <Staff> Edit customer information

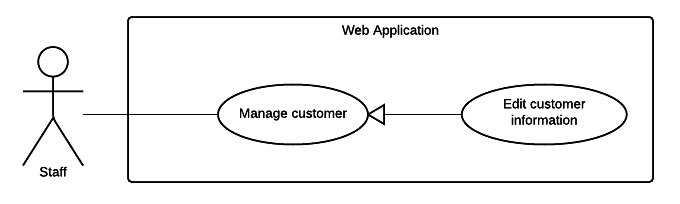


Figure <Staff> Edit customer information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS06 | | | |
| Use Case No. | WS06 | **Use Case Version** | 2.0 |
| Use Case Name | Edit customer information | | |
| Author | KhaNC | | |
| Date | 20/05/2015 | **Priority** | Medium |
| Actor:   * Staff.   Summary:   * This use case allows staff edit customer’s information.   Goal:   * Customer’s information is updated to the system.   Triggers:   * Staff sends update customer’s information command.   Preconditions:   * User must login into the system with role Staff.   Post Conditions:   * Success: Customer’s information is updated. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to edit customer information. | Edit 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 update customer’s information command. | * Validate data   [Exception 1, 2, 3]   * Update customer’s new information to the system. * Reload customer 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 | | 3 | Entered email address is not a valid email | Show message notify entered email is not valid |   Relationships: N/A.  Business Rules:   * Customer’s detail information is always loaded from the system. * In case of success scenarios, customer new information would be updated to the system. * Reloaded customer detail will display customer updated information. * An email address must be validated by this regular expression:   /^([a-z0-9\_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/   * Customer’s current information must be shown in its respective fields. | | | |

Table Use case WS06 - <Staff> Edit customer information

###### <Staff> Create new customer

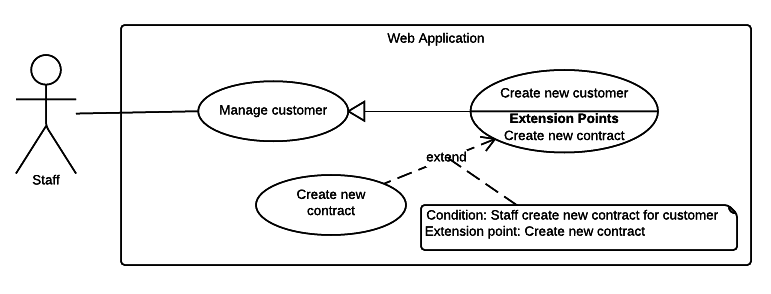


Figure <Staff> Create new customer

|  |  |  |  |
| --- | --- | --- | --- |
| 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]   * 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 |   Relationships: N/A  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 Use case WS07 - <Staff> Create new customer

###### <Staff> Create contract

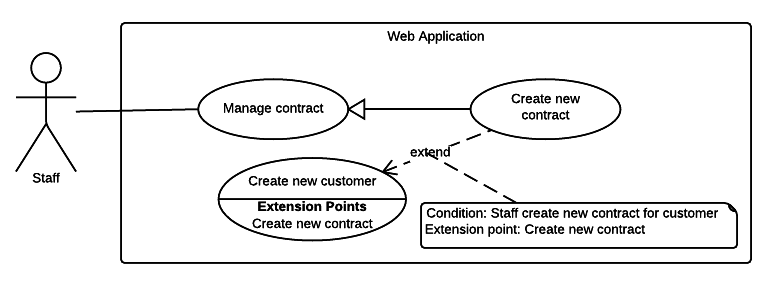


Figure <Staff> Create contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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 01 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   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: N/A  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. * A new contract created successfully will has the initial status is “No Card”. * Contract’s start date must not be earlier than create contract date. * Contract’s term must not exceed 1 year. * Default contract’s term is 1 year. * Contract’s expired date must not be earlier than contract’s start date. * Contract’s type must be one of the following values: * “Xe trên 50cc có BH cho người trên xe” * “Xe trên 50cc không có BH cho người trên xe” * “Xe dưới 50cc có BH cho người trên xe” * “Xe dưới 50cc không có BH cho người trên xe” * “Xe mô tô ba bánh, xe gắn máy và các loại xe tương tự” | | | |

Table Use case WS08 - <Staff> Create new contract

###### <Staff> Renew contract

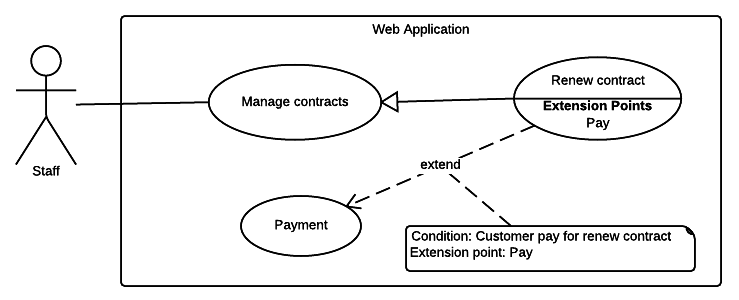


Figure <Staff> Renew contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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. * Contract’s status must be “No Card”, “Ready” or “Expired”. * The payment process for renew contract was completed.   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 start date: text * Contract’s new expired date: date time input, required * Renew contract fee: text   Payment information:   * 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: N/A  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 start date must not be earlier than create contract date. * Contract’s term must not exceed 1 year. * Default contract’s term is 1 year. * Contract’s expired date must not be earlier than contract’s start date. | | | |

Table Use case WS09 - <Staff> Renew contract

###### <Staff> Cancel contract

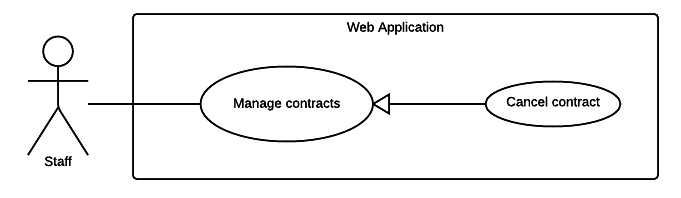


Figure <Staff> Cancel contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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 not be “Cancelled” or “Expired”. * The expired date is less than 2 months. * The compensation for breach of contract was completed.   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:   * No notification will be sends to user when a staff cancel a contract. * A contract cancelled successfully will change status from “No card”, “Ready” or “Request cancel” to “Cancelled”. | | | |

Table Use case WS10 - <Staff> Cancel contract

###### <Staff> Update contract information

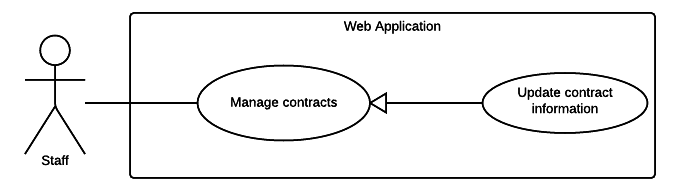


Figure <Staff> Update contract information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS11 | | | |
| Use Case No. | WS11 | **Use Case Version** | 2.0 |
| Use Case Name | Update contract information | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Staff   Summary:   * This use case allows staff to update contract information.   Goal:   * Staff can update contract information.   Triggers:   * Staff sends command to update contract information.   Preconditions:   * User has to logged in to the system as Staff role * Contract is existed in the system * Contract status must not be “Cancelled” or “Expired”   Post Conditions:   * Success: Contract information will be updated * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to update contract information. | System list out information of contract:  **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 15.   **Contract information** (all information below are required)   * Contract’s type: select one from the options. * Start date: date time input * Expired date: date time input   **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 2200. * Weight: free text input, value from 1 – 1000, unit: kilogram * Seat capacity: free text input, value from 1 – 100. | | 2 | Staff inputs required information. |  | | 3 | Staff sends command to save new information. | System shows message notify contract information is updated successfully.  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends command to save new information. | System shows error message to ask staff input missing required fields. |   Relationships: N/A  Business Rules:   * Contract end date must not be earlier start date. * Expired date and start date must not exceed 1 year. * Contract information will be updated to system. * Contract’s type must be one of the following values: * “Xe trên 50cc có BH cho người trên xe” * “Xe trên 50cc không có BH cho người trên xe” * “Xe dưới 50cc có BH cho người trên xe” * “Xe dưới 50cc không có BH cho người trên xe” * “Xe mô tô ba bánh, xe gắn máy và các loại xe tương tự” | | | |

Table Use case WS11 - <Staff> Update contract information

###### <Staff> Update compensation history

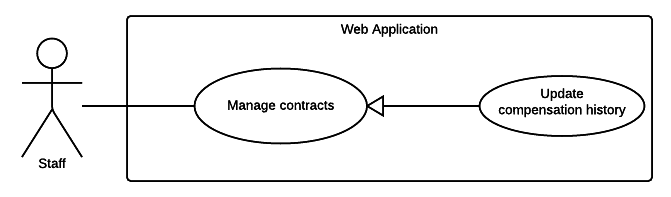


Figure <Staff> Update compensation history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS12 | | | |
| Use Case No. | WS12 | **Use Case Version** | 2.0 |
| Use Case Name | Update compensation history | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Staff   Summary:   * This use case allows staff to update compensation information.   Goal:   * Staff can update compensation information.   Triggers:   * Staff sends command to update compensation information.   Preconditions:   * User has to logged in to the system as Staff role * Compensation is existed in the system   Post Conditions:   * Success: Compensation information will be updated * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to update compensation information. | System list out information of compensation:   * 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: free text input, required, length 4 – 15. * Date: date time input, required. * Place: free text input, required, length 4 – 15. * Control 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 upload input * Created date: date time input, required. * Resolve date: date time input, required. * Decision: free text input, required, length 1 – 250. * Resolve note: free text input, required, length 1 – 2000. | | 2 | Staff inputs information |  | | 3 | Staff sends command to save new information. | System shows message notify compensation information is updated successfully.  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends command to save new information. | System shows error message to ask staff input missing required fields. |   Relationships: N/A  Business Rules:   * Compensation information will be updated to system. | | | |

Table Use case WS12 - <Staff> Update compensation history

###### <Staff> Update punishment history

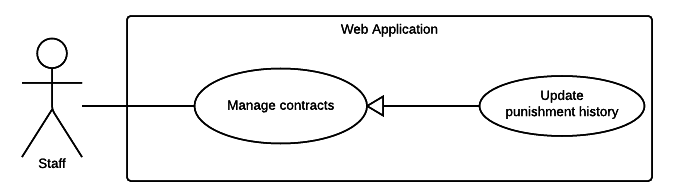


Figure <Staff> Update punishment history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS13 | | | |
| Use Case No. | WS13 | **Use Case Version** | 2.0 |
| Use Case Name | Update punishment history | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Staff   Summary:   * This use case allows staff to update punishment information.   Goal:   * Staff can update punishment information.   Triggers:   * Staff sends command to update punishment information.   Preconditions:   * User has to logged in to the system as Staff role * Punishment is existed in the system   Post Conditions:   * Success: Punishment information will be updated * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to update punishment information. | System list out information of punishment:   * Date: date time input, required. * Title: free text input, required, length 1 – 250. * Attachment: file upload input, required. | | 2 | Staff inputs information |  | | 3 | Staff sends command to save new information. | System shows message notify punishment information is updated successfully.  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends command to save new information. | System shows error message to ask staff input missing required fields. |   Relationships: N/A  Business Rules:   * Punishment information will be updated to system. * Punishment date must not exceed current date. | | | |

Table Use case WS13 - <Staff> Update punishment history

###### <Staff> Update accident history

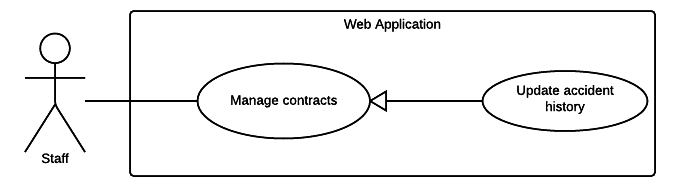


Figure <Staff> Update accident history

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS14 | | | |
| Use Case No. | WS14 | **Use Case Version** | 2.0 |
| Use Case Name | Update accident history | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Staff   Summary:   * This use case allows staff to update accident information.   Goal:   * Staff can update accident information.   Triggers:   * Staff sends command to update accident information.   Preconditions:   * User has to logged in to the system as Staff role * Contract of the accident is existed in the system   Post Conditions:   * Success: Punishment information will be updated * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to accident information. | System list out information of accident:   * Date: date time input, required. * Title: free text input, required, length 1 – 250. * Attachment: file upload input, required. | | 2 | Staff inputs information. |  | | 3 | Staff sends command to save new information. | System shows message notify compensation information is updated successfully.  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends command to save new information. | System shows error message to ask staff input missing required fields. |   Relationships: N/A  Business Rules:   * Accident information will be updated to the system. * Accident date must not exceed current date. | | | |

Table Use case WS14 - <Staff> Update accident history

###### <Staff> View cards information

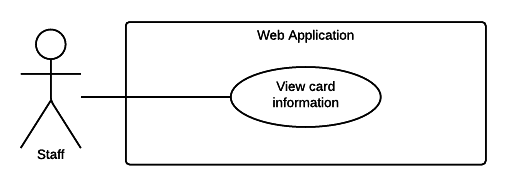


Figure <Staff> View card information

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WS15 | | | |
| Use Case No. | WS15 | **Use Case Version** | 2.0 |
| Use Case Name | View cards information | | |
| Author | TrungDQ | | |
| Date | 27/05/2015 | **Priority** | High |
| Actor:   * Staff   Summary:   * This use case allows staff to view published cards information.   Goal:   * Staff can view cards information.   Triggers:   * Staff goes to view cards information.   Preconditions:   * User has to logged in to the system as Staff role * Cards information is existed in the system   Post Conditions:   * Success: Cards information is shown * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to view cards information | System list out information of published cards:   * Card ID: text link to card detail * Card owner: text * Last access time: text * Status: text   [Exception 1] | | 2 | Staff sends command to view card detail information | System shows card detail information:   * Card ID: text * Card owner: text * Status: text * Activated date: text * Last access time: text * History of access   [Exception 2] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff goes to view cards information | System shows no connection error message. | | 2 | Staff sends command to view card detail information | System shows no connection error message. |   Relationships:  Business Rules:   * Cards information will be loaded from the system. * History of access will show 10 results at a time, with pagination links at the bottom. | | | |

Table Use case WS15 - <Staff> View cards information

###### <Staff> Update contract type information



Figure <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:   * Staff   Summary:   * This use case allows staff to update contract type.   Goal:   * Staff can update contract type.   Triggers:   * Staff sends command to update contract type.   Preconditions:   * User has to logged in to the system as Staff 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 | Staff goes to view contract type information. | System list out information of accident:   * 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: VND. | | 2 | Staff inputs information |  | | 3 | Staff sends command to save new information. | System shows message notify contract type information is updated successfully.  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends command to save new information. | System shows error message to ask staff input 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. | | | |

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

##### <Admin> Overview Use Case

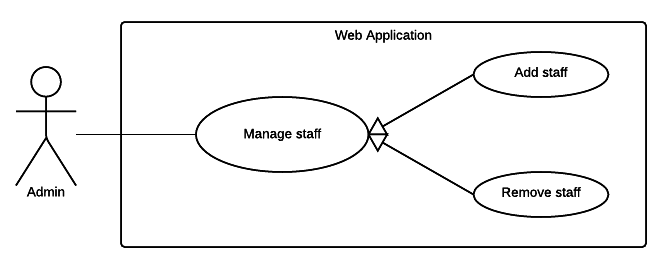


Figure <Admin> Overview Use Case

###### < Admin> Remove staff

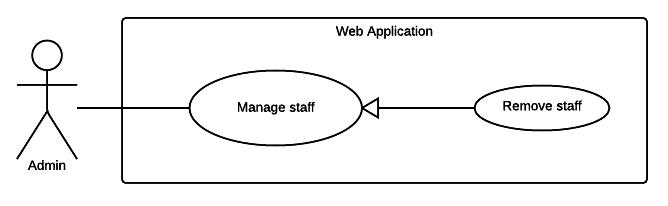


Figure <Admin> Remove staff

|  |  |  |  |
| --- | --- | --- | --- |
| 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. | | | |

Table Use case WA01 - <Admin> Remove staff

###### < Admin> Add staff

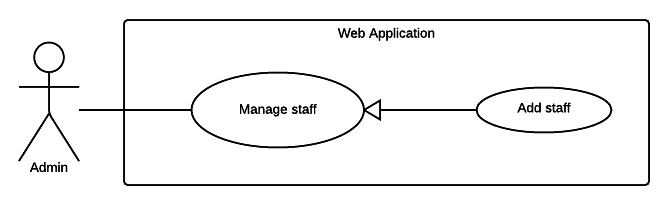


Figure <Admin> Add staff

|  |  |  |  |
| --- | --- | --- | --- |
| 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: Admin be able to add new staff to the system * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin goes to add staff. | System require information of staff:   * Staff code: text, required, length 3 - 80 * Name: text, required, length 3 – 80. * Email: text, required, length 3 – 250. * 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 the code used in physical system maintained by insurance company, it probably already exists for each staff when join in the company. If the company has no staff code for their staff, this field can be enter personal id of the staff. * Staff code must not be duplicated. | | | |

Table Use case WA02 - <Admin> Add staff

##### <System> Overview Use Case



Figure <System> Overview Use Case

###### <System> Notify schedule



Figure <System> Notify schedule

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – WY01 | | | |
| Use Case No. | WY01 | **Use Case Version** | 2.0 |
| Use Case Name | Notify schedule | | |
| Author | TrungDQ | | |
| Date | 26/05/2015 | **Priority** | Medium |
| Actor:   * System   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:   * 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 everyday. * If a contract has expired date remaining is less than 15 days, system send notification to users to ask users to renew their contract. * Notification will repeat every 4 days if the user do not renew the contract. * Contracts that are not renewed will has status “Expired” and have no value. * If there is new compensation, or the compensation status changes, a notification will be sent to users. | | | |

Table Use case WY01 - <System> Notify schedule

##### <Payment system> Overview Use Case



Figure <Payment> Payment

###### <Payment system> Payment



Figure <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: N/A.  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 Use case WP01 - <Payment> Payment

#### Checker Mobile Application

##### <Police> Overview Use Case

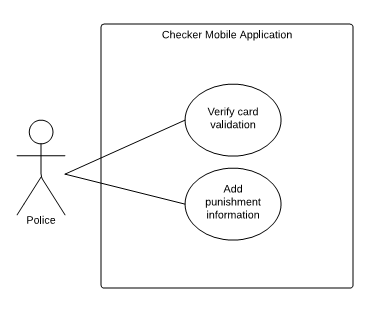


Figure <Police> Overview Use Case

###### <Police> Verify card information

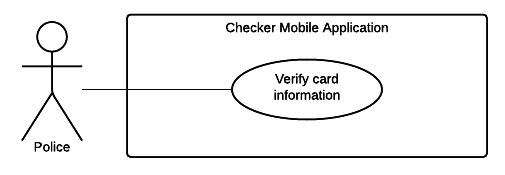


Figure <Police> Verify card information

|  |  |  |  |
| --- | --- | --- | --- |
| 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:   * The application in home screen.   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:  Business Rules:   * Mobile application sends request about information card to server and receives information about contract. * A card is “Valid” if all of the following conditions are true:   + The card ID is in activated status.   + 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 15 days. * Highlight the status of the card include:   + Valid card   + Expired card   + Card is nearly expired, show remaining days | | | |

Table Use case CP01 - <Police> Verify card information

###### <Police> Add punishment information

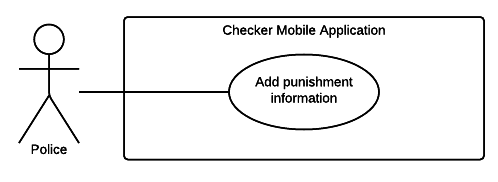


Figure <Police> Add punishment information

|  |  |  |  |
| --- | --- | --- | --- |
| 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 reads NFC card successful.   Preconditions:   * The application in Verify card screen. * The NFC card is valid.   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 allows for police add new punishment. | | 2 | Police chooses add more punishment. | Mobile required police input description about punishment and picture about it.   * Description: free text input, required, length 1 - 200 * Picture: file upload input, required | | 3 | Police fill the punishment information and sends punishment. | 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 that police forgets take a photo for punishment. | | 2 | Cannot connect to server. | Warning that cannot connect to server. |   Relationships: N/A  Business Rules:   * Information about punishment will be sent in server and update in contract of user. | | | |

Table Use case CP02 - <Police> Add punishment information

#### Printer Mobile Application

##### <Staff> Overview Use Case



Figure <Staff> Overview Use Case

###### <Staff> Search / filter contract

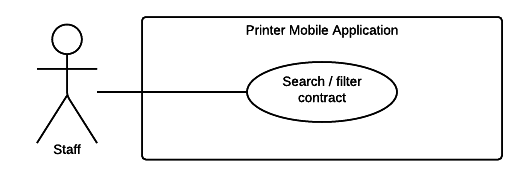


Figure <Staff> Search / filter contract

|  |  |  |  |
| --- | --- | --- | --- |
| 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:   * Staff must login successful.   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 sends command to search/ filter contract  Staff chooses contract | System shows:  Information about request search.  System shows information about contract:   * Contract code: text * Customer name: text * Date start: text * Time of contract: text * Status: text   [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. It is include:   * Invalid the value of textbox * Fill some especial character | Website show message to notify. |   Relationships:  Business Rules:  Search/ filter with some criterion:   * Search/ filter by contract code * Search/ filter by customer name * Search/ filter by start date * Search/ filter by time of contract | | | |

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

###### <Staff> View contract information



Figure <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 user view their contract’s information.   Goal:   * User can review their contract’s information.   Triggers:   * User chooses information insurance tab in the navigation bar.   Preconditions:   * User must login into the system.   Post Conditions:   * Success: Show contract’s information to user. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User chooses information insurance in the navigation bar. | Show information about the insurance and the motor’s 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 |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Cannot connect to server. | System shows error that cannot connect to server. |   Relationships:  Business Rules:   * System application shows information about contract. | | | |

Table Use case PS02 - <Staff> View contract information

###### <Staff> Print information to NFC card



Figure <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 finds the information of contract then command to request print information into card.   Preconditions:   * Staff must login successful in mobile application. * Contract must NOT be in status “Pending” or “Cancelled”   Post Conditions:   * Success: Notify that print successfully. * Fail: Notify that print fail.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff filters by name or contract code to find contract | Show information of customer:   * Motor owner: text * Address: text * Phone Number: text * Plate: text * Chassis: text * Engine: text * Type: text * Time: text * Insurance premium: text * Fee payer: text * People issue: text * Date issue: text * Hot line: text | | 2 | Staff choose print information into card | Notify print success  [Alternative 1]  [Alternative 2]  [Exception 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Staff commands to search/ filter but cannot see the result of contract | Staff commands a request to server to notify | | 2 | The contract is already assigned with a card. | System show message to notify staff that the contract is already assigned to a card. Ask staff if he/she want to override the old card. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If the information of contract cannot print into the NFC card | Show message print fail |   Relationships:  Business Rules:   * The information of staff who prints that card will be saved and information of card will be printed and transfer for user. * In case staff decided to override old card, the old card will be deactivated and no longer work. | | | |

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

## Software System Attribute

### Usability

#### Graphic User Interface

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

#### Usability

* The system usability are 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.

#### Installation

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

### Reliability

* N/A

### Availability

* N/A

### 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.

### Maintainability

* The system is separated into modules.

### 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 smartphones that have version greater than 4.0.

### 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.

## Conceptual Diagram

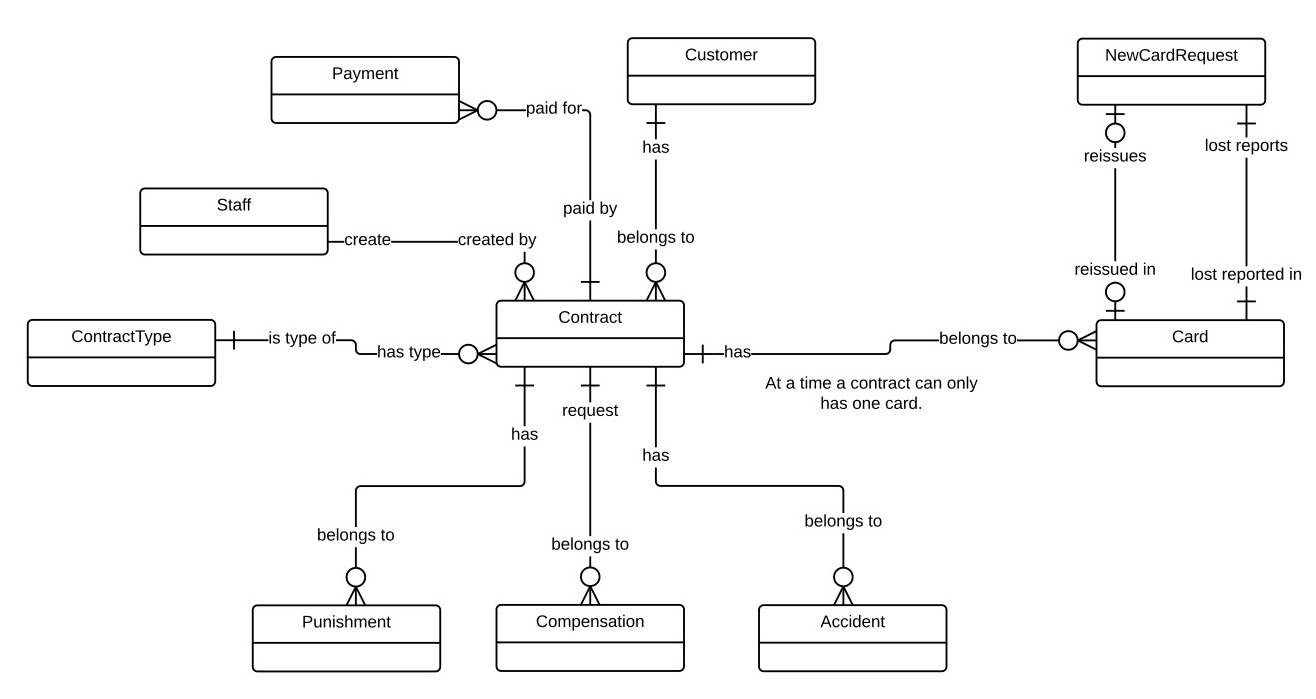


Figure Conceptual diagram

**Data Dictionary**

|  |  |
| --- | --- |
| Entity Data dictionary: describe all content of all entities | |
| Entity Name | **Description** |
| Customer | Describe all customer in the system |
| Contract | Describe all customer in the system |
| ContractType | Describe all contract’s type in the system |
| Card | Describe all card in the system |
| NewCardRequest | Describe all request for new card in the system |
| Compensation | Describe all compensation information in the system |
| Punishment | Describe all punishment information in the system |
| Accident | Describe all accident in the system in the system |
| Staff | Describe all staff in the system |
| Payment | Describe all payment method in the system |

# Software Design Description

## 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.

## System Architecture Design

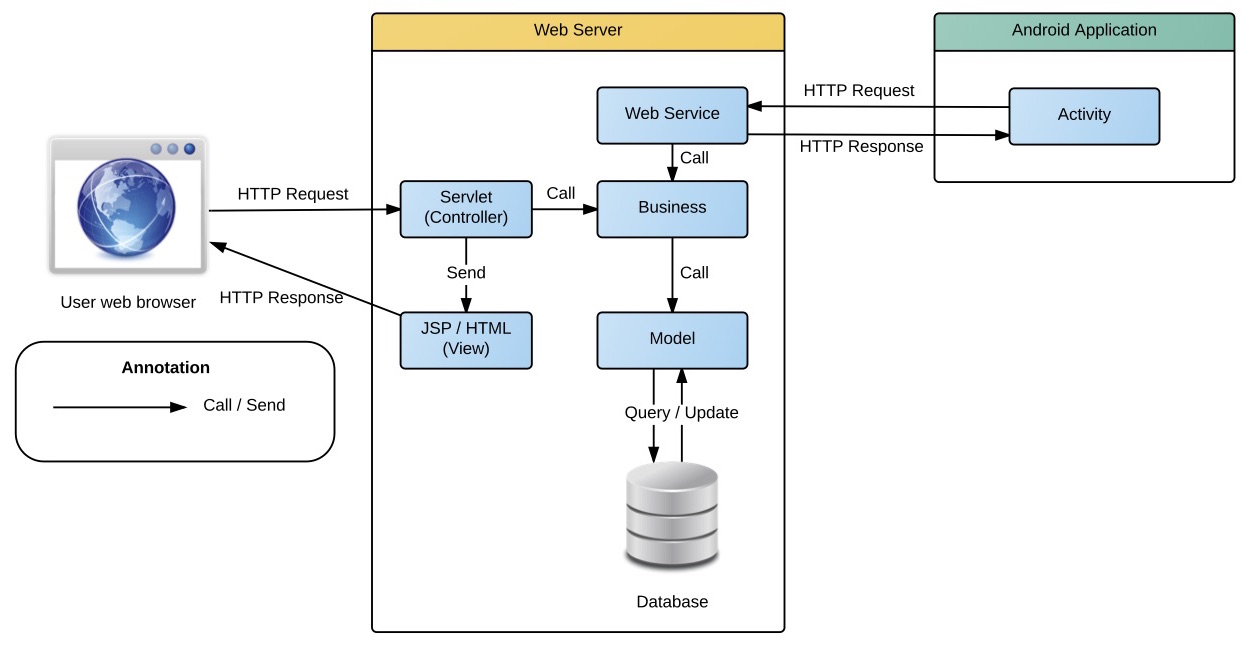


Figure System architecture design

### 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 this, 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 follow 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.

### 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 follow Android application architecture with following components:

* **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 ...

## Component Diagram

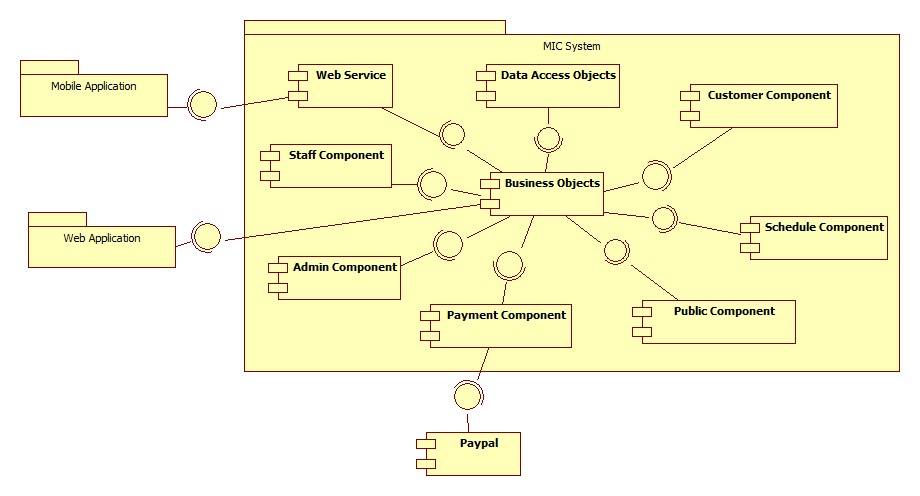


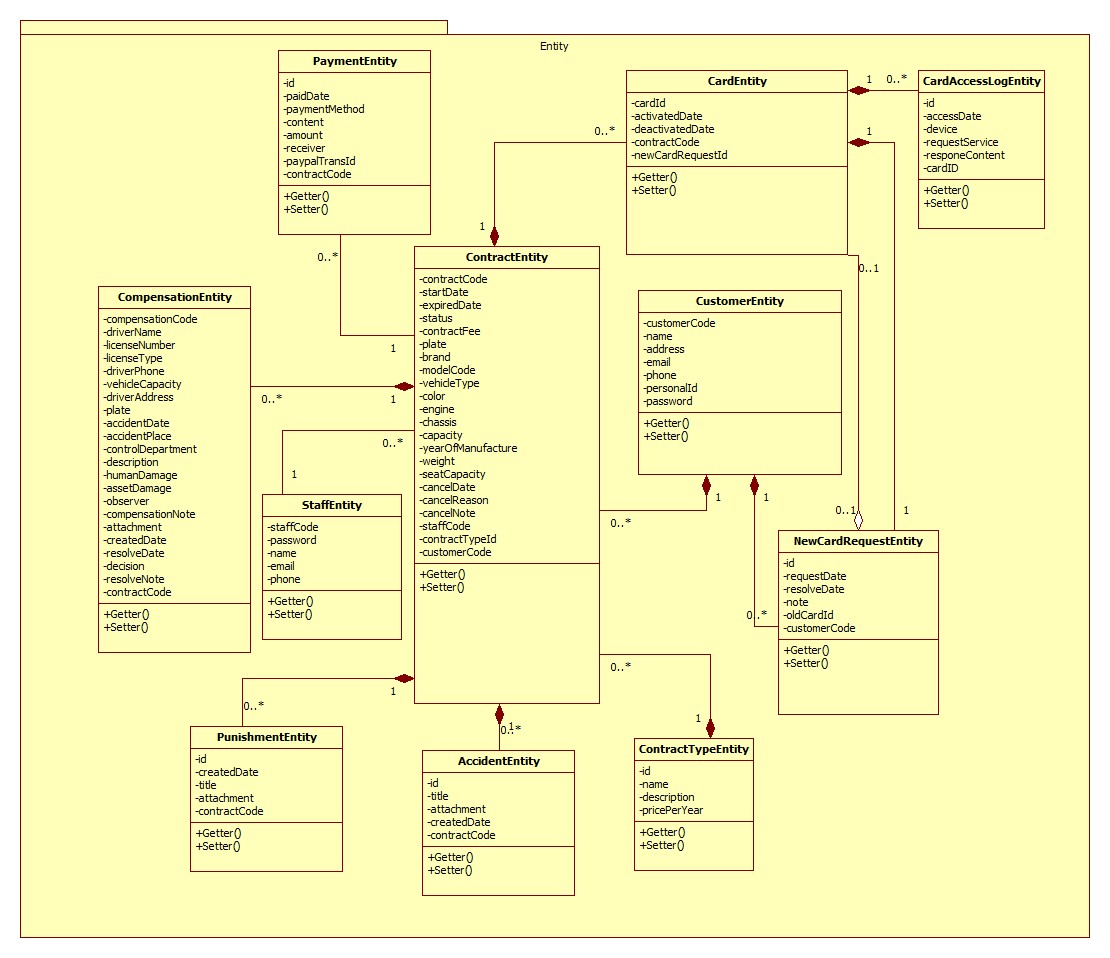
Figure Component Diagram

|  |  |
| --- | --- |
| Component Dictionary: Describes components | |
| Web Application | Web application package: View, Controller |
| Mobile Application | Mobile application package |
| PayPal | Handle payment process with PayPal API |
| Web Service | operations o handle interaction between the system and database.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 Component Dictionary

## Detailed Description

### Class Diagram



|  |  |  |
| --- | --- | --- |
| Class dictionary: describe Class | | |
| Class Name | **Mapping column with Conceptual diagram** | **Description** |
| PaymentEntity | Payment | Contain the payment information. |
| CardEntity | Card | Contain the card 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. |

### Class Diagram Explanation

#### 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. |

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

#### CardEntity

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| cardId | String | Private | Unique identifier of an card |
| 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

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

#### 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. |

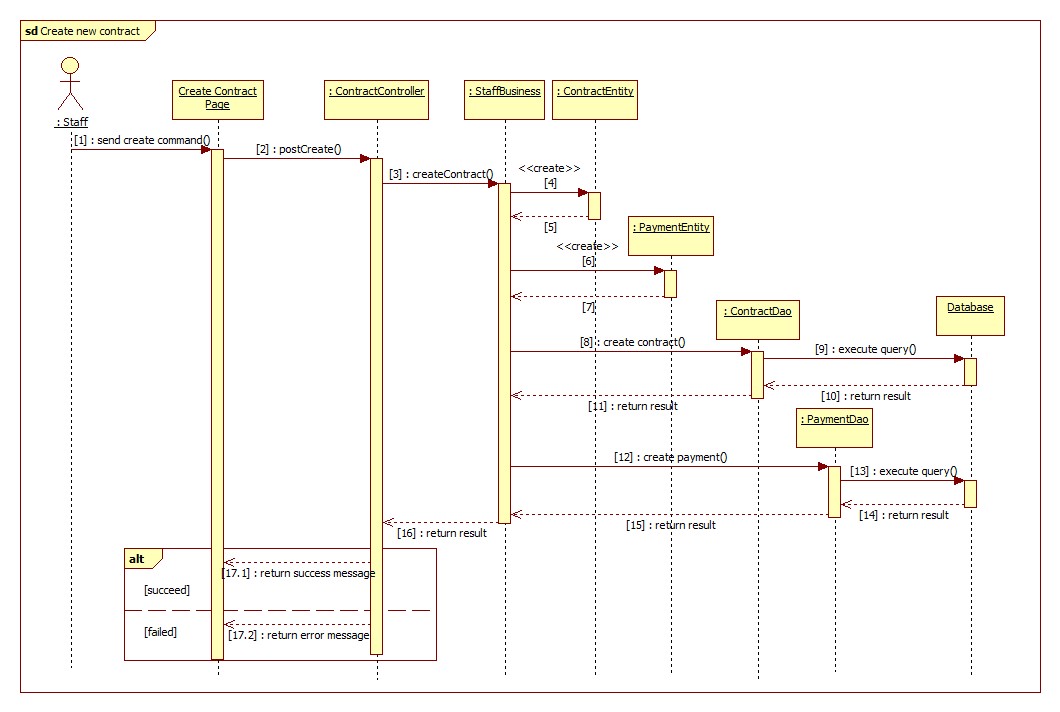
|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return Type | Visibility | Description |
| Getter | Attribute type | Public | Get attribute value. |
| Setter | Void | Public | Set value of attribute. |

### Interactive Diagram

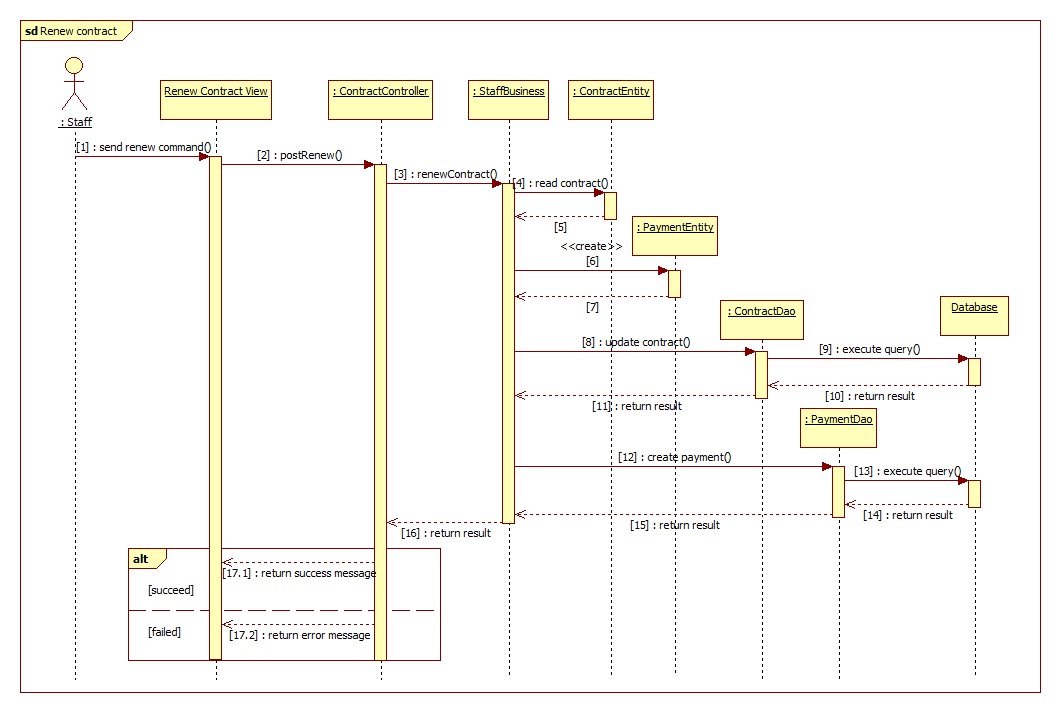
#### Web Application

##### Staff

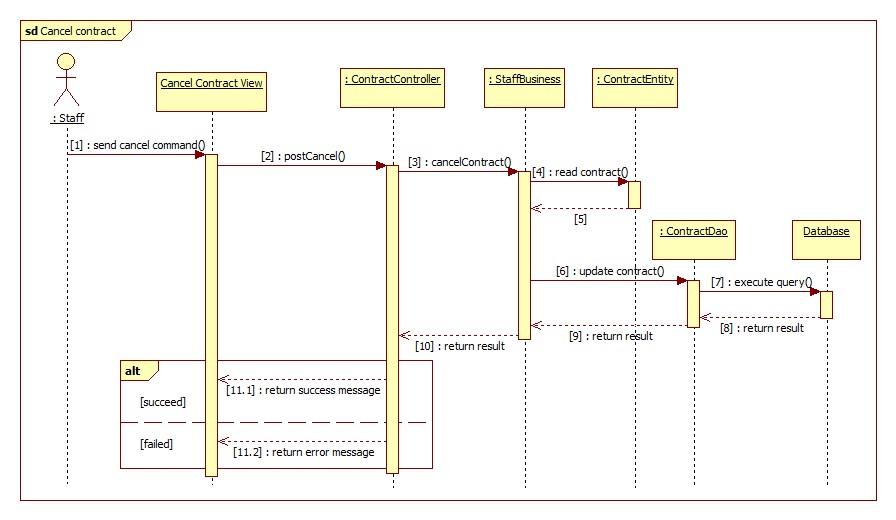
###### Create new contract



###### Renew contract

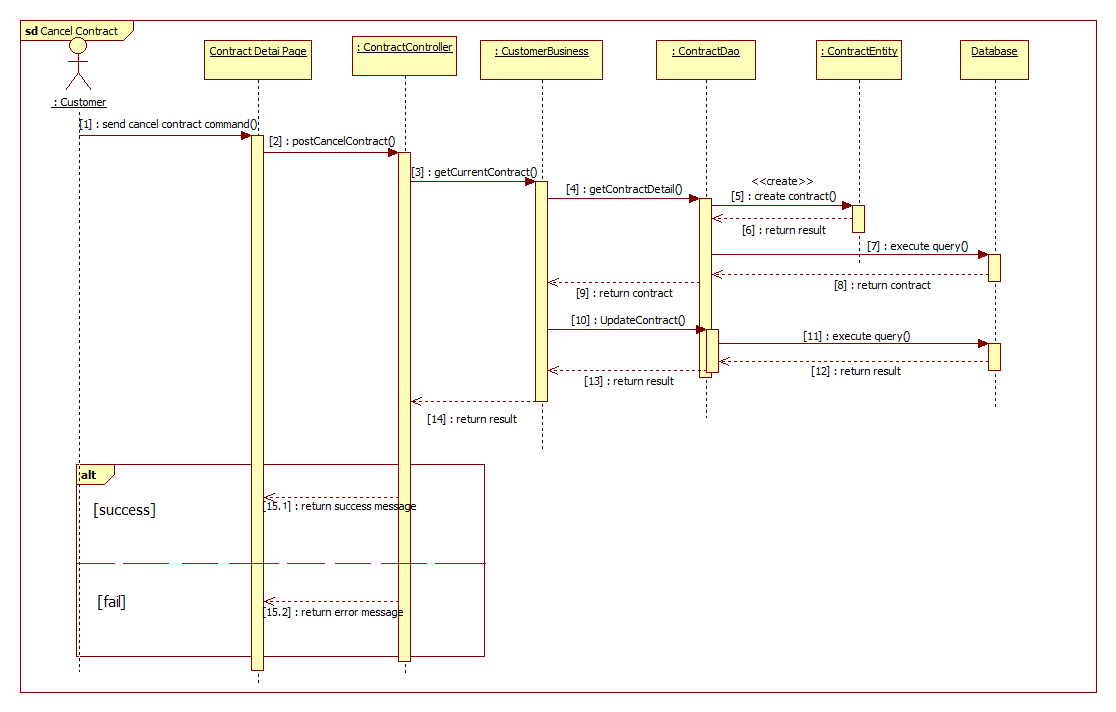


###### Cancel contract

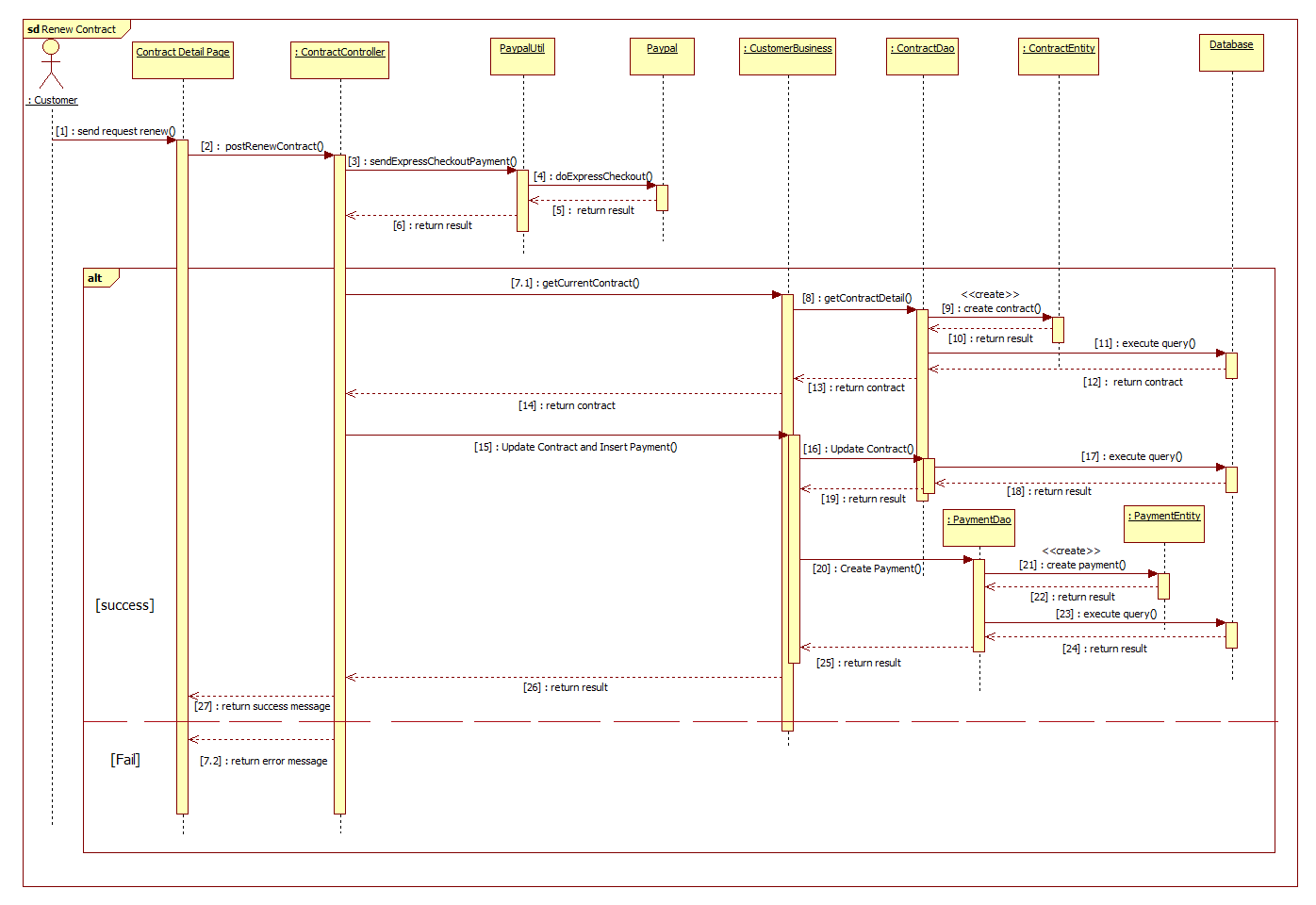


##### Customer

###### Cancel contract

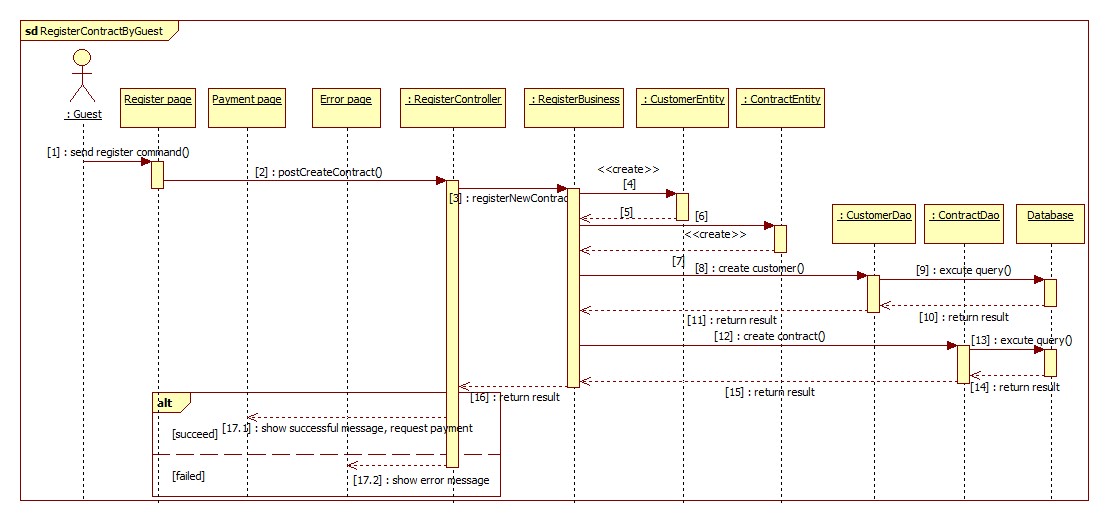


###### Renew contract

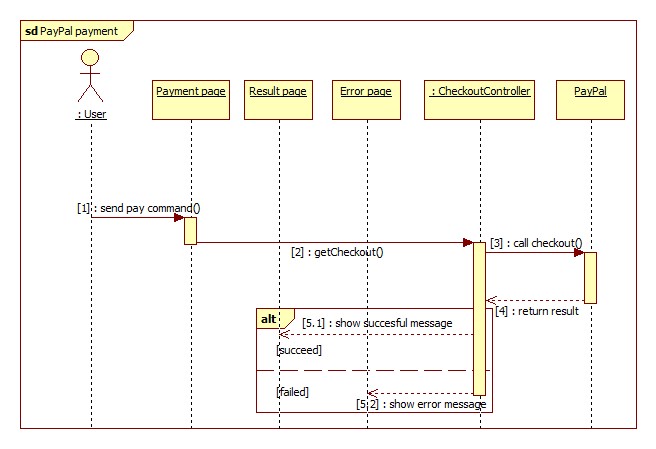


##### Guest

###### Register new contract



###### PayPal payment



#### Mobile Application

##### Checker Mobile Application

###### <Police> Verify card validation

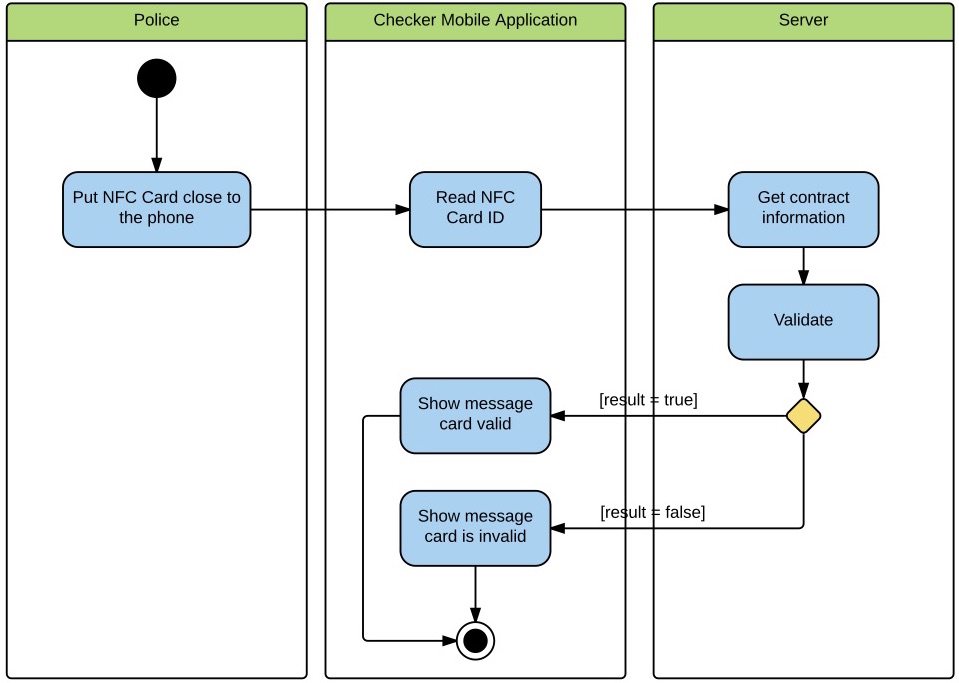


Figure <Police> Verify card validation

###### <Police> Add punishment information

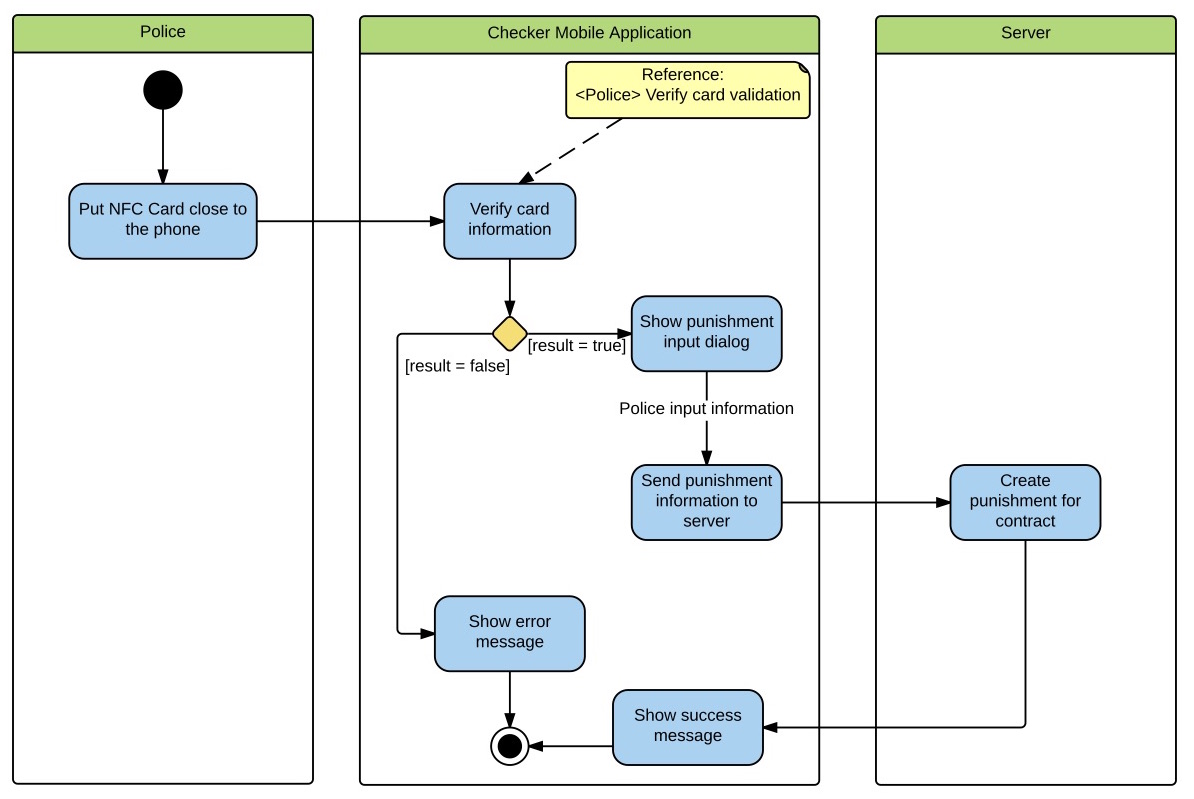


Figure <Police> Add punishment information

##### Printer Mobile Application

###### <Staff> Search contract

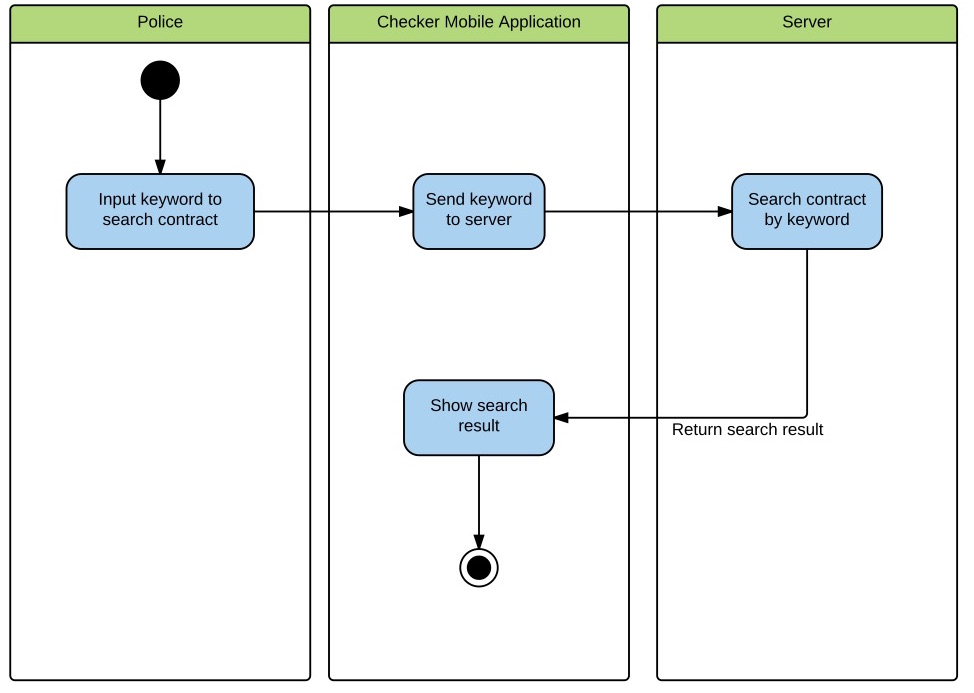


Figure <Staff> Search contract

###### <Staff> View contract information

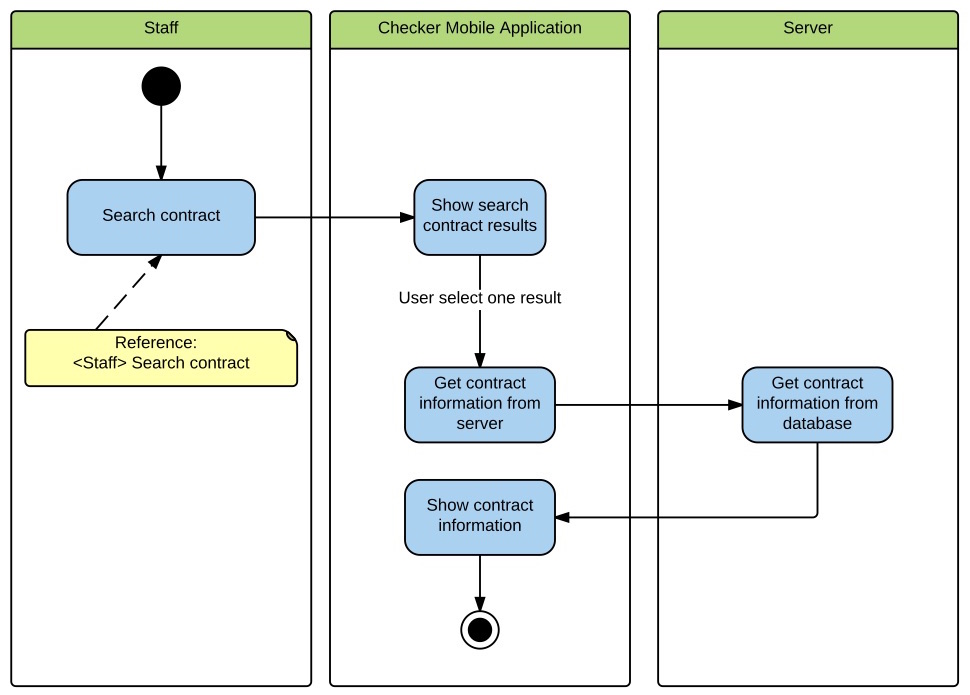


Figure <Staff> View contract information

###### <Staff> Print information to NFC Card

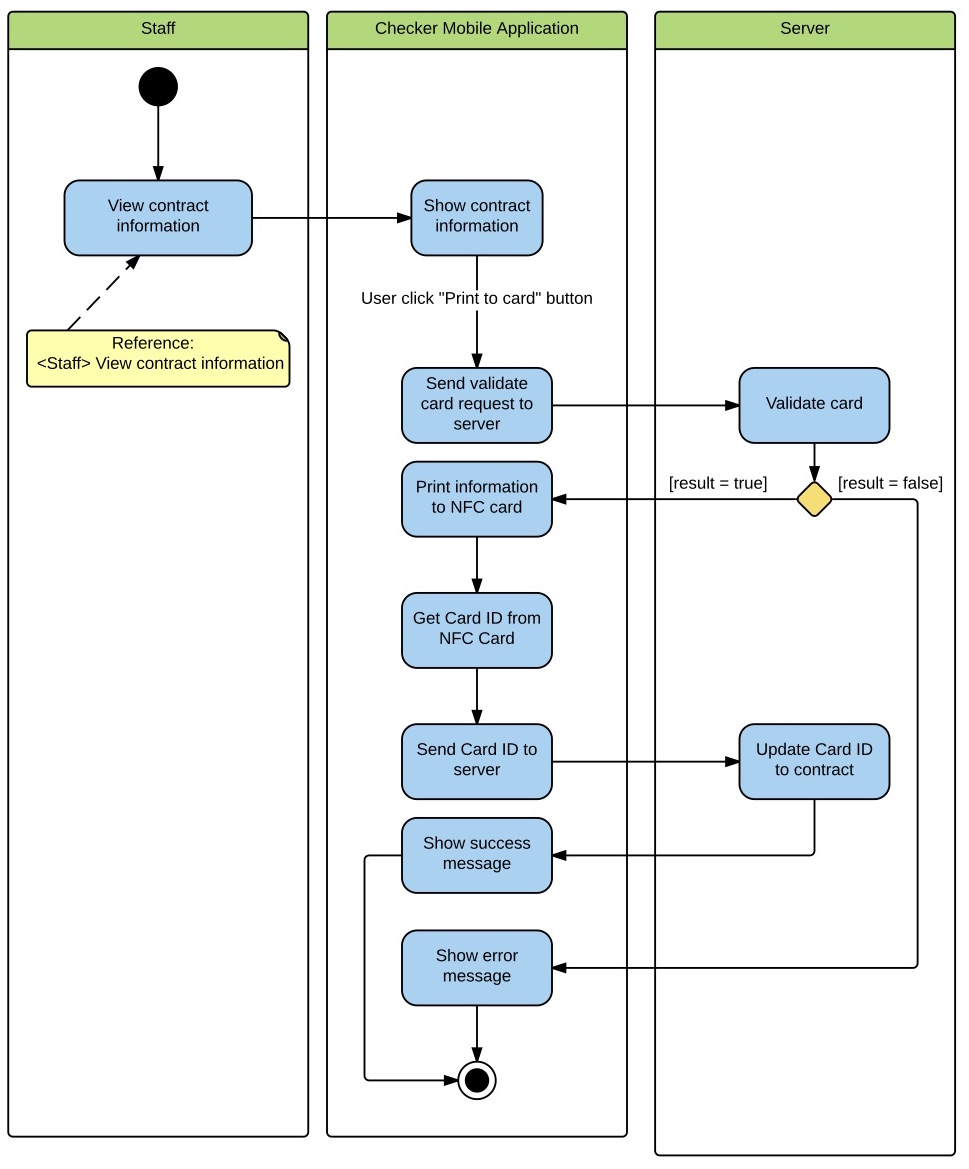


Figure <Staff> Print information to NFC card

## Interface

### Component Interface

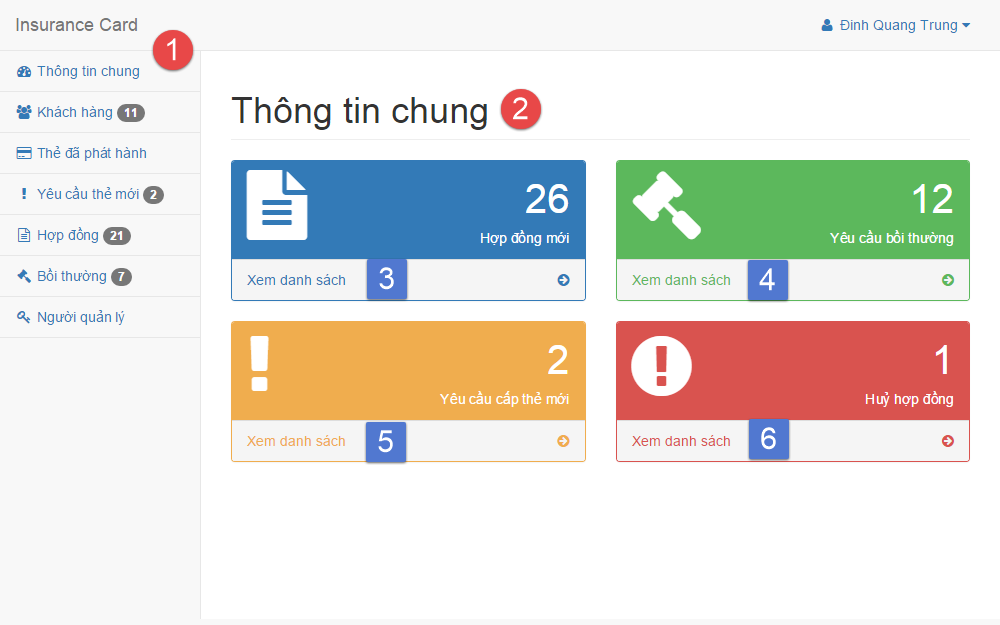
#### Web Service Interface

|  |  |  |  |
| --- | --- | --- | --- |
| Signature | Description | Input | Output |
| Contractlean if the update is successntractpublic ResponseObject getContracts(R r) | Search contracts by code or customer name | Request object r contains:  Keyword: String | Json string List of contracts |
| public ResponseObject getUpdateCardID(R r) | Update card ID to a contract | Request object r contains:  cardID: String  contractCode: String | Json Boolean the update result |
| public ResponseObject getCheckCard | Check the validation of the card | Request object r contains:  cardID: String | Json CardResult object |
| public ResponseObject getUpdatePunishment | Update punishment information to a contract | Request object r contains:  contractCode: String  punishmentInfo: PunishmentInfoObject | Json Boolean the update result |

### Web application Design

#### Staff Interface Design

##### Staff home page



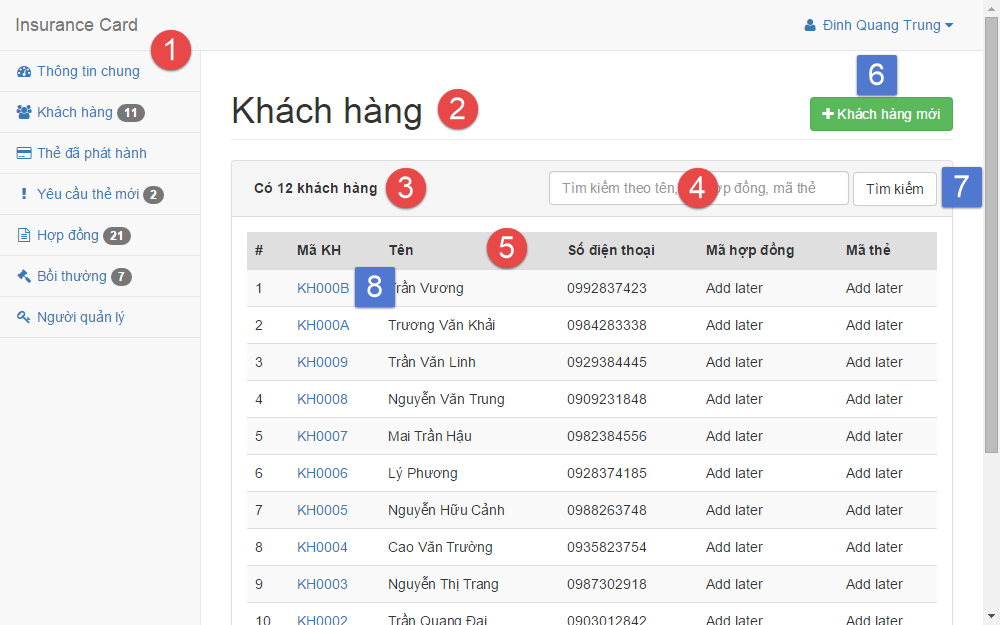
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

**Button/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 |

##### Manage customer



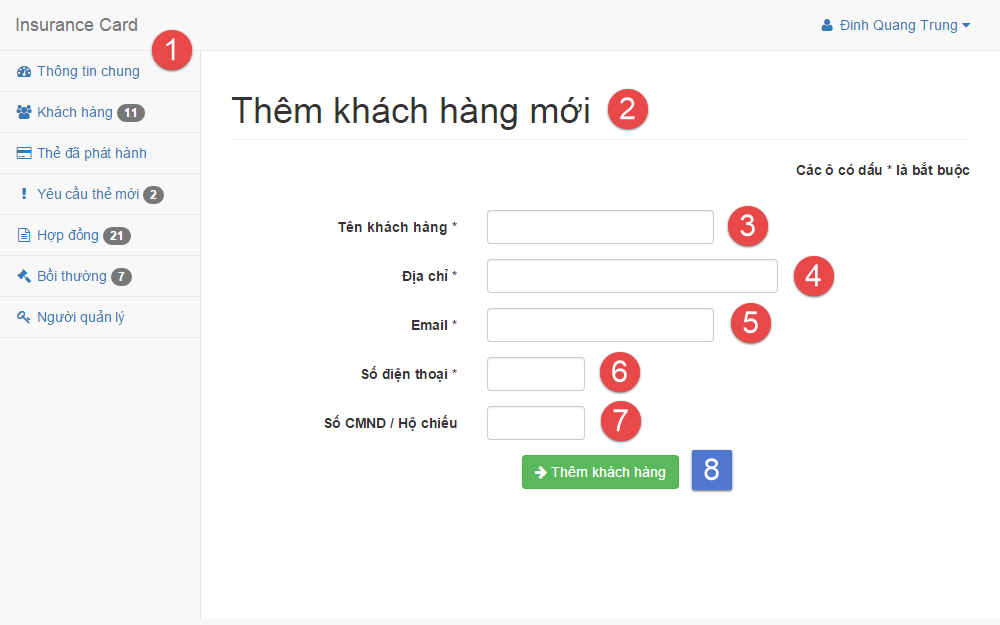
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | Number | Number of available customer | Yes | No | Label | N/A | N/A |
| 4 | txtSearchValue | Fill search keyword | No | Yes | Textbox | String | N/A |
| 5 | Table | Table of available customer | Yes | Yes | Table | N/A | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 6 | btnCreate | Create new customer | N/A | Transfer to create new customer page |
| 7 | btnSearch | Search by input keyword | N/A | Transfer to search result page |
| 8 | customerDetail | View customer detail | N/A | Transfer to customer detail page |

##### Create customer



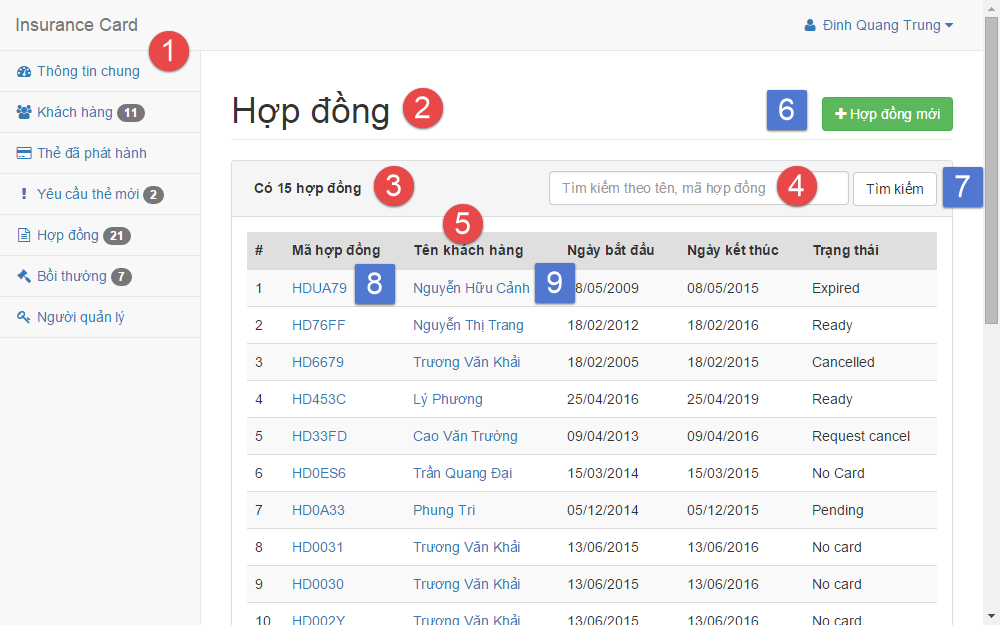
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | txtName | Fill customer name | No | Yes | Textbox | String | 3 – 80 |
| 4 | txtAddress | Fill address | No | Yes | Textbox | String | 3 – 250 |
| 5 | txtEmail | Fill email | No | Yes | Textbox | String | 3 – 250 |
| 6 | txtPhone | Fill phone | No | Yes | Textbox | Integer | 8 – 15 |
| 7 | txtPersonal ID | Fill personal ID | No | No | Textbox | Integer | 8 – 15 |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 8 | btnCreate | Create new customer | N/A | Transfer to create new customer successful page |

##### Manage contract



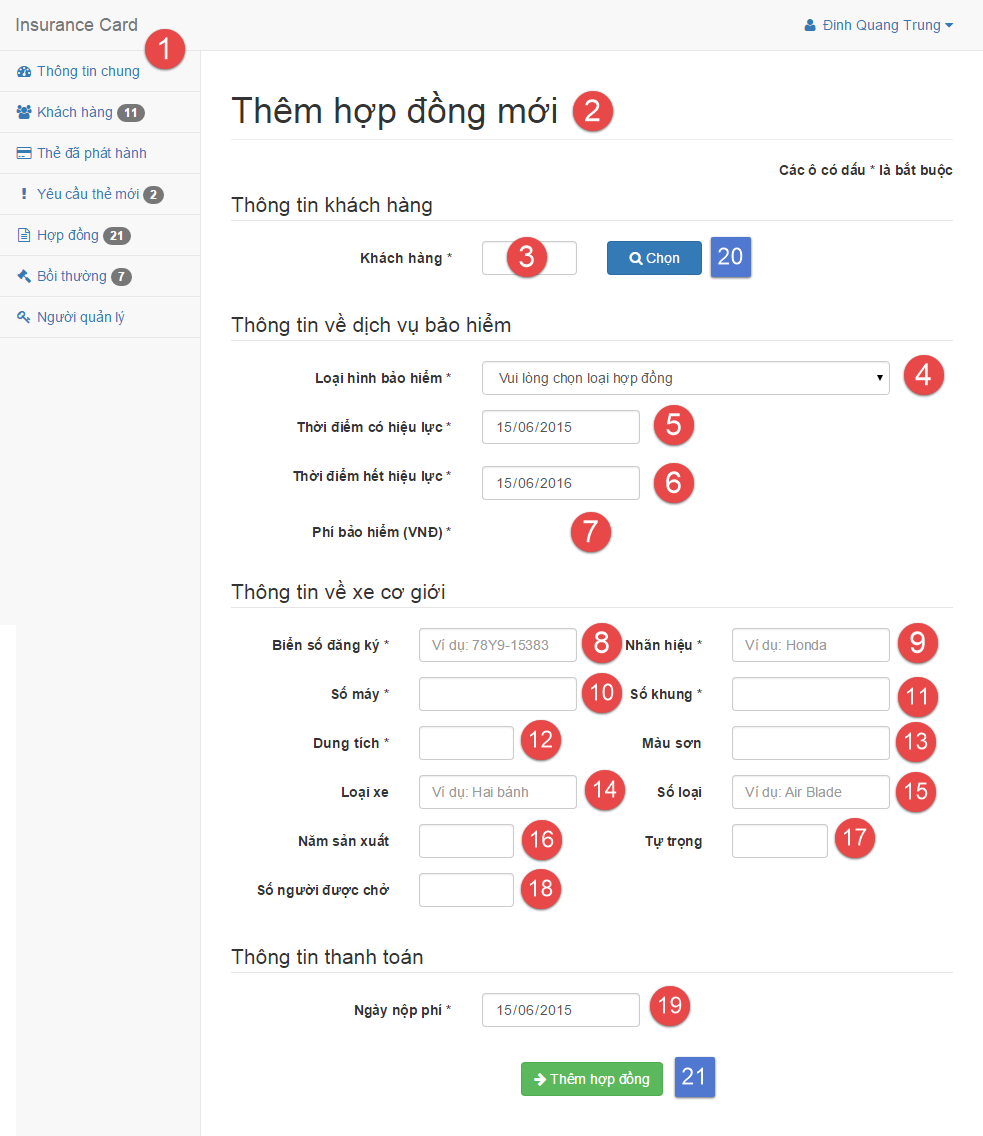
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | Number | Number of available contract | Yes | No | Label | N/A | N/A |
| 4 | txtSearchValue | Fill search keyword | No | Yes | Textbox | String | N/A |
| 5 | Table | Table of available customer | Yes | Yes | Table | N/A | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 6 | btnCreate | Create new contract | N/A | Transfer to create new contract page |
| 7 | btnSearch | Search by input keyword | N/A | Transfer to search result page |
| 8 | contractDetail | View contract detail | N/A | Transfer to contract detail page |
| 9 | customerDetail | View customer detail | N/A | Transfer to customer detail page |

##### 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 | txtModel code | 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 |

**Button/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 |

##### Renew contract



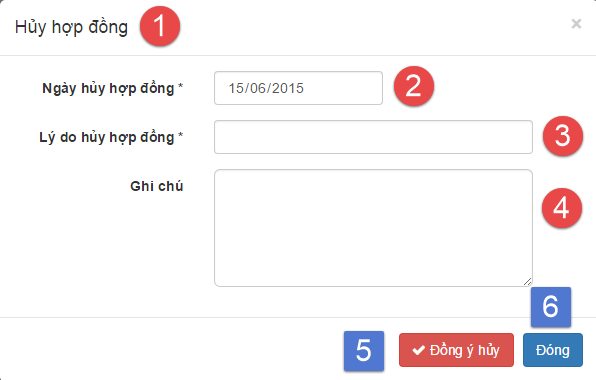
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | Title | Title of the modal | Yes | Yes | Label | N/A | N/A |
| 2 | Start date | Contract’s start date | Yes | Yes | Label | N/A | N/A |
| 3 | txtExpiredDate | Select expired date | No | Yes | Date input form | N/A | N/A |
| 4 | txtAmount | Renew contract fee | Yes | Yes | Label | N/A | N/A |
| 5 | txtPaidDate | Select paid contract fee date | No | Yes | Date input form | N/A | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 6 | btnRenew | Renew contract | N/A | Transfer to renew contract successful page |
| 7 | btnClose | Close modal | N/A | Transfer to contract detail page |

##### Cancel contract



**Fields**

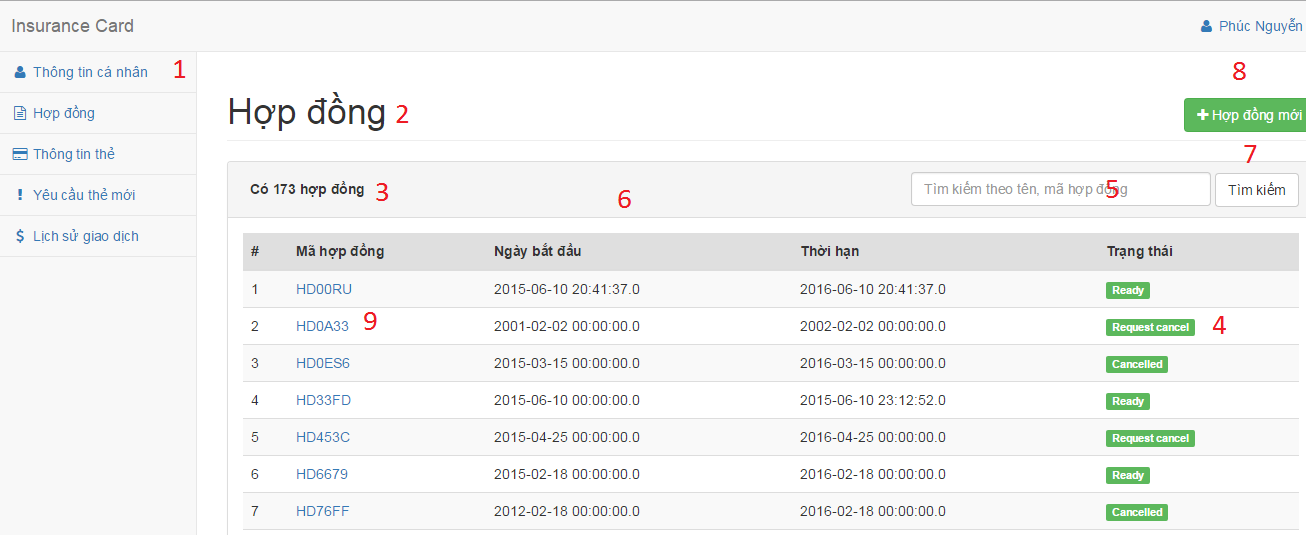
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | Title | Title of the modal | Yes | Yes | Label | N/A | N/A |
| 2 | txtCancelDate | Select cancel contract date | No | Yes | Date input form | N/A | N/A |
| 3 | txtCancelReason | Fill cancel contract reason | No | Yes | Textbox | String | 1 – 250 |
| 4 | txtCancelNote | Fill note about cancel contract | No | No | Text area | String | 1 - 2000 |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 5 | btnCancel | Cancel contract | N/A | Transfer to cancel contract successful page |
| 6 | btnClose | Close modal | N/A | Transfer to contract detail page |

#### Customer Interface Design

##### Manage contract



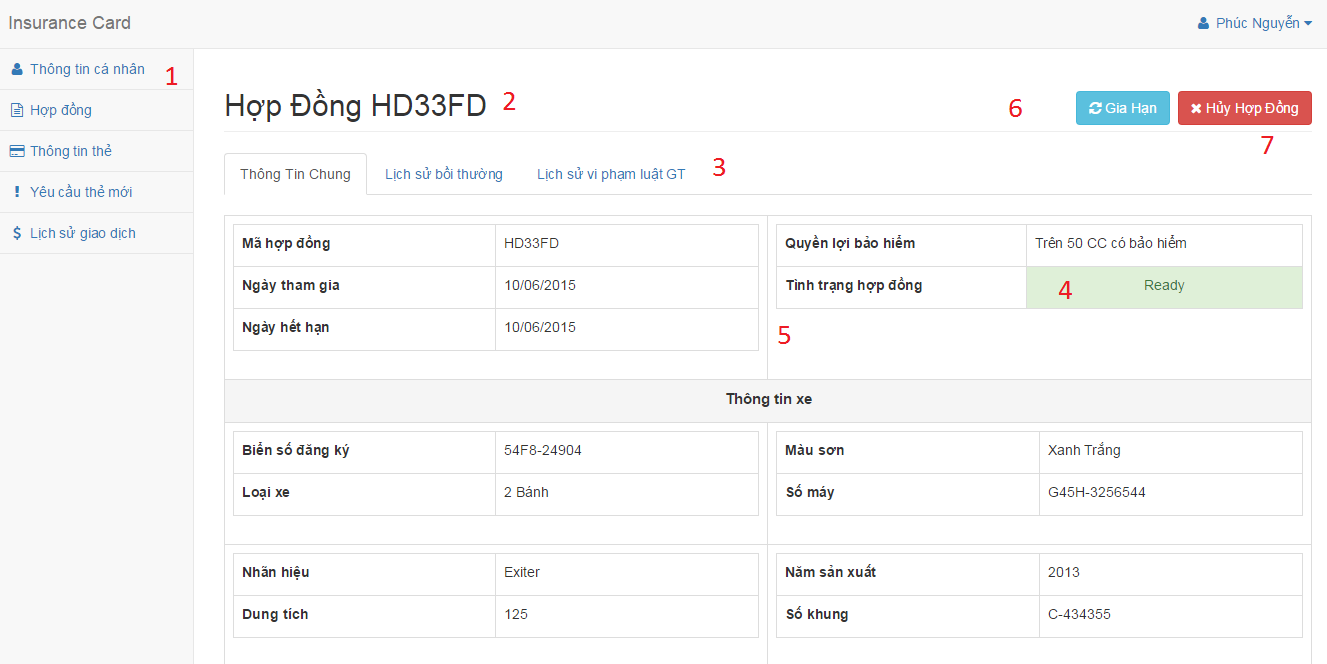
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

**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 |

##### Contract detail



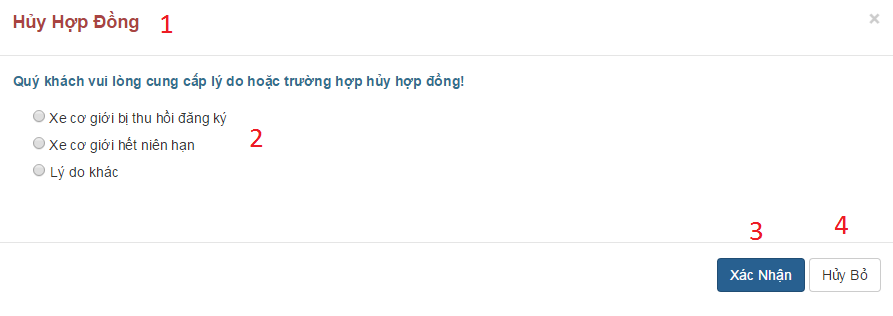
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

**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 |

##### Cancel contract



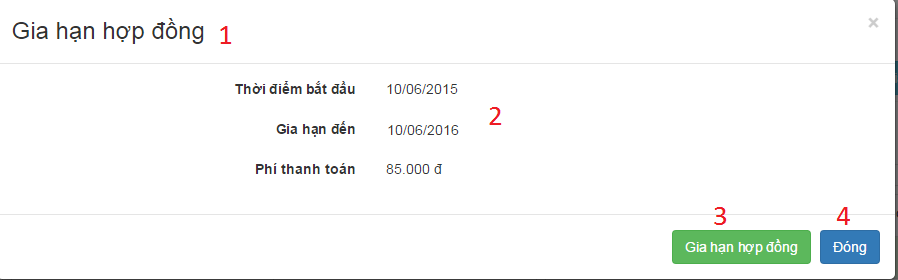
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | lblTitle | Title | Yes | Yes | Label | N/A | N/A |

**Buttons / Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 2 | chkOptionCancel | Choose reason why customer cancel contract | N/A | N/A |
| 3 | btnConfirm | Confirm to cancel contract | If customer doesn’t input the reason in case they choose another reason, the system will notify. | Reload contract detail with information about contract is canceled |
| 4 | btnLeave | Leave request cancel | N/A | Return to contract detail |

##### Renew contract



**Select paypment method**

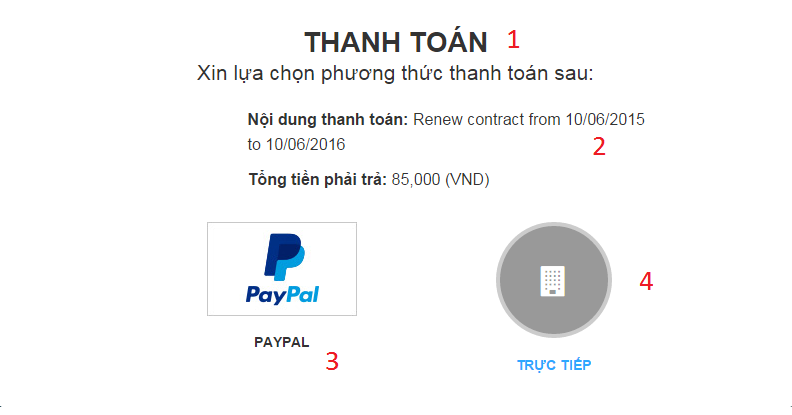
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | lblTitle | Title | Yes | Yes | Label | N/A | N/A |
| 2 | txtInformation | Information about renew contract | Yes | Yes | Label | N/A | N/A |

**Buttons / Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 3 | btnConfirm | Confirm to renew contract | N/A | Transfer to checkout page |
| 4 | btnLeave | Leave request renew | N/A | Return to contract detail |

##### Select payment method



**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | lblTitle | Title | Yes | Yes | Label | N/A | N/A |
| 2 | txtInformation | Information about renew contract | Yes | Yes | Label | N/A | N/A |

**Buttons / Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 3 | btnPaypal | Payment by paypal | N/A | Transfer to paypal page |
| 4 | btnDirect | Payment direct at company | N/A | Show address of list company |

**Select paypment method**

### Checker Mobile Application Design

#### Scan NFC card screen

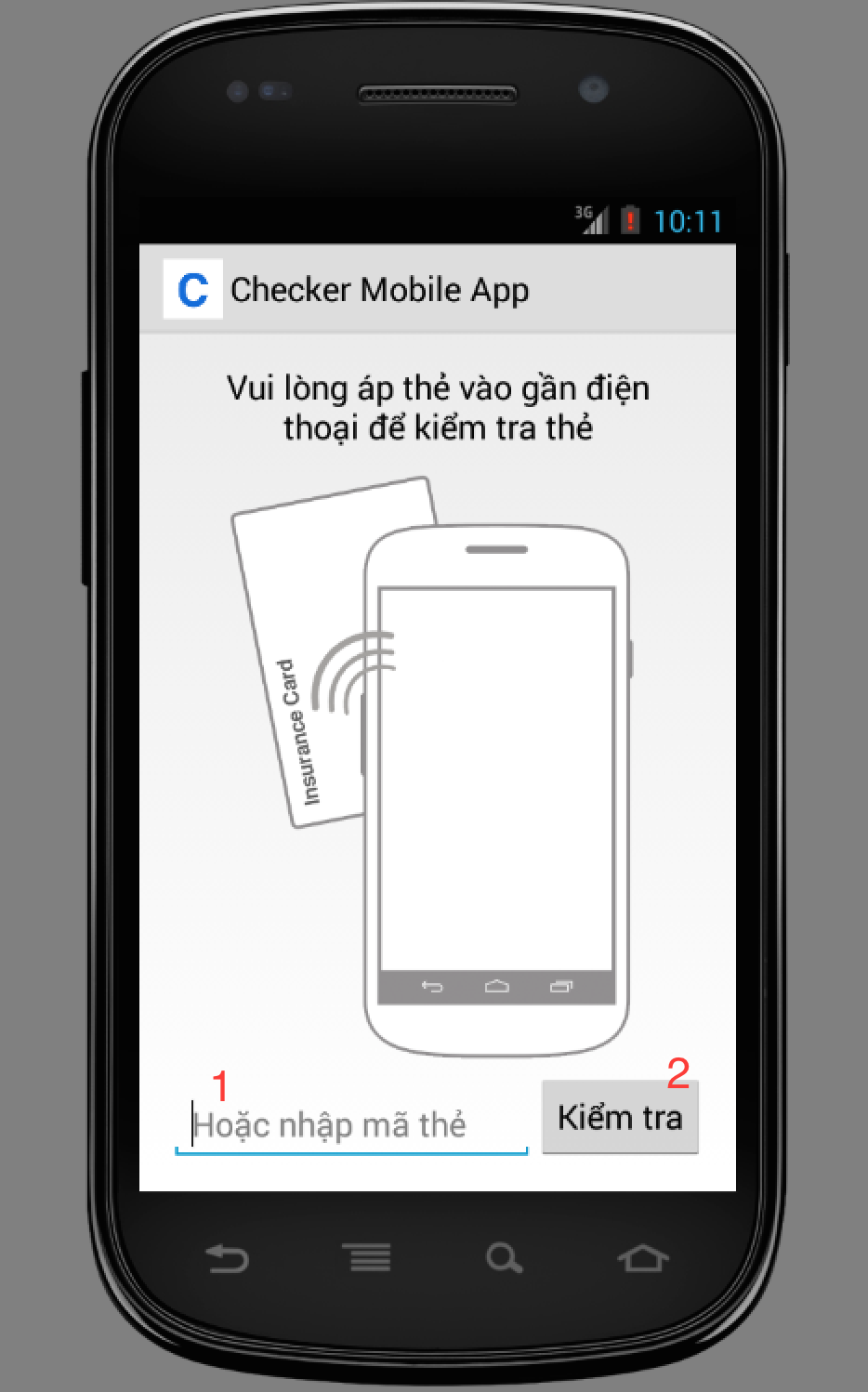


Figure Scan NFC card screen

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control Type | Data Type | Length |
| 1 | txtCardIDnpen print card screen supporto view detail information of that contractCard | ID of the card | No | No | Textbox | String | 20 |

Table 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 Scan NFC card screen - Buttons

#### View card information screen

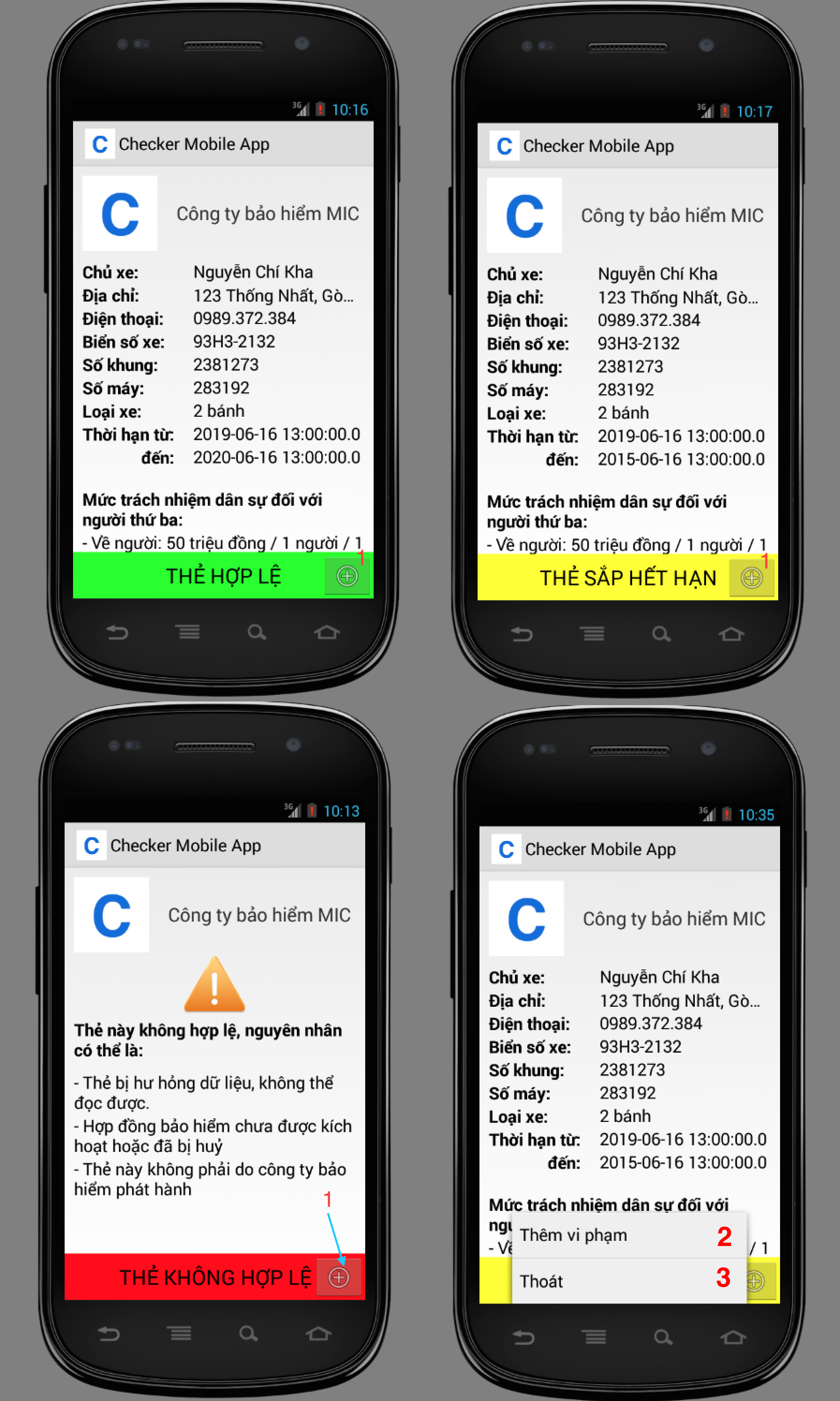


Figure View card information screen

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 1 | btnMenu | Show menu which contains “Add punishment” button and “Exit” button | N/A | Open menu |
| 2 | mnuAddPunishment | Add punishment information | N/A | Open add punishment dialog |
| 3 | mnuExit | Exit the current page | N/A | Return to Scan screen |

Table View card information screen - Buttons

#### Add punishment screen

### Printer Mobile Application Design

#### Search contract screen

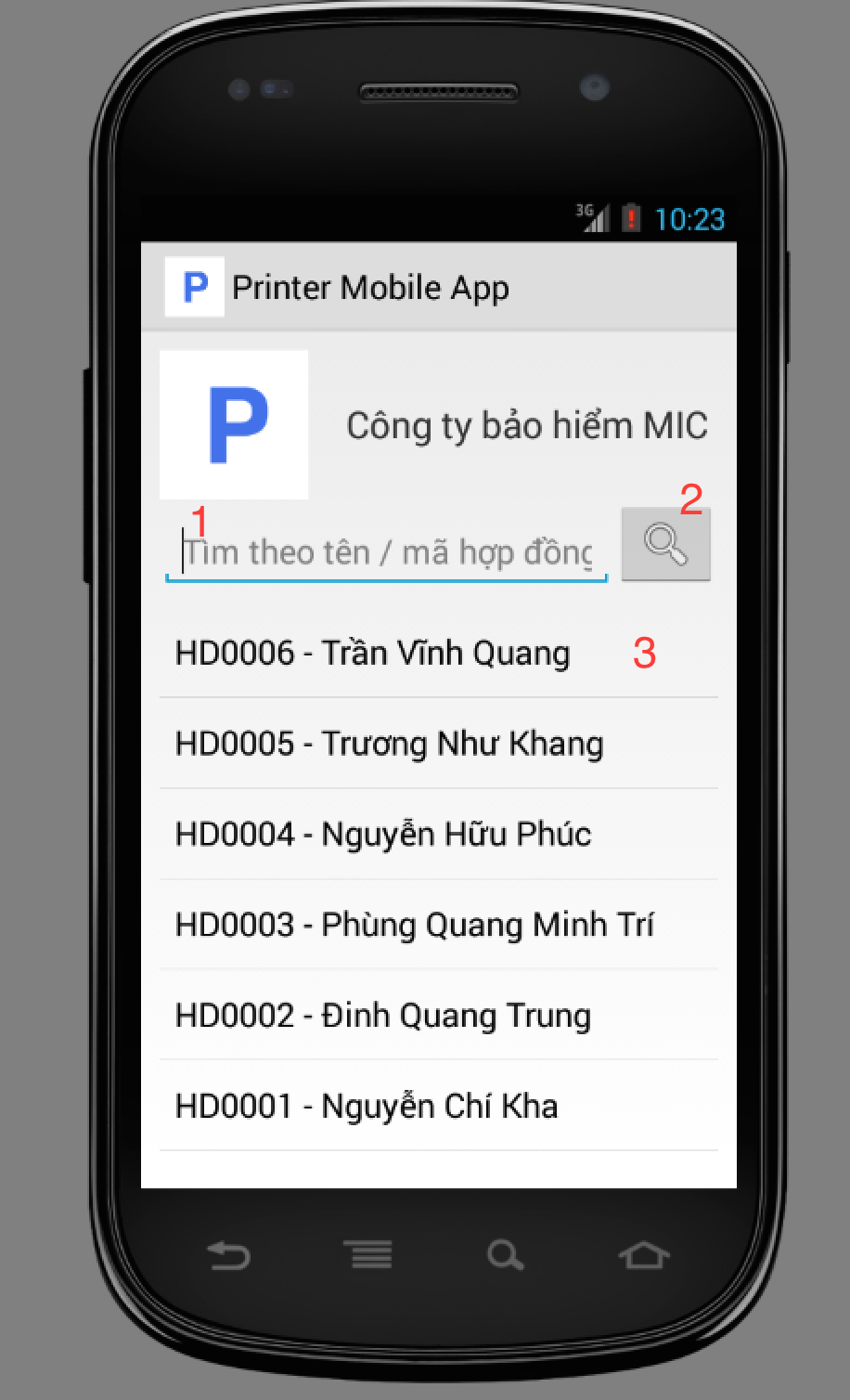


Figure Search contract screen

**Fields**

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

Table 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 Search contract screen - Buttons

#### View contract screen



Figure View contract screen

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 1 | btnPrint | Click to print information to NFC card | Phone have to have NFC support | Open print card screen |

Table View npen print card screen supporto view detail information of that contractcontract screen - Buttons

#### Print card screen



Figure Print card screen

## Database Design

### Entity relationship diagram



Figure Entity relationship diagram

### Entity Dictionary

|  |  |
| --- | --- |
| Entity Data Dictionary: describe content of all entities | |
| Entity name | **Description** |
| Customer | Contain the customer information. |
| Contract | Contain the contract information. |
| Card | Contain the card information. |
| 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. |

Table Entity dictionary

## Algorithms

### 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.

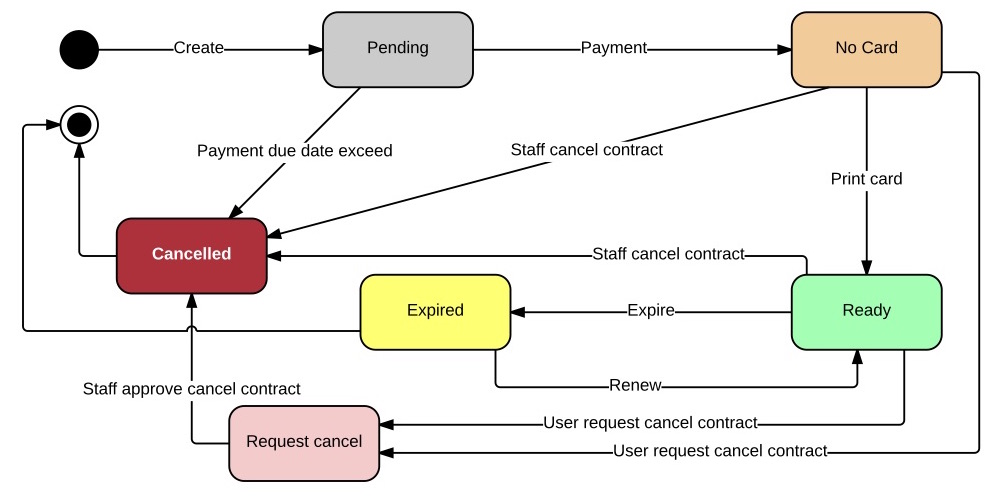


Figure Contract State Chart

|  |  |
| --- | --- |
| SStimulusion﷽﷽﷽﷽﷽> ERDquence numberecution bar em and database.tate | 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 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 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 | 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 | Rancelledancelcancel request from customer via web application .eady | 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 cancelledancelcancel request from customer via web application .ancel contract | Staff approve cancel request from customer via web application | Request cancel | Cancelled |

Table Contract State Flow

### Concurrency Control

#### 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.

#### 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.

#### 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 represent 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.

#### Flow chart

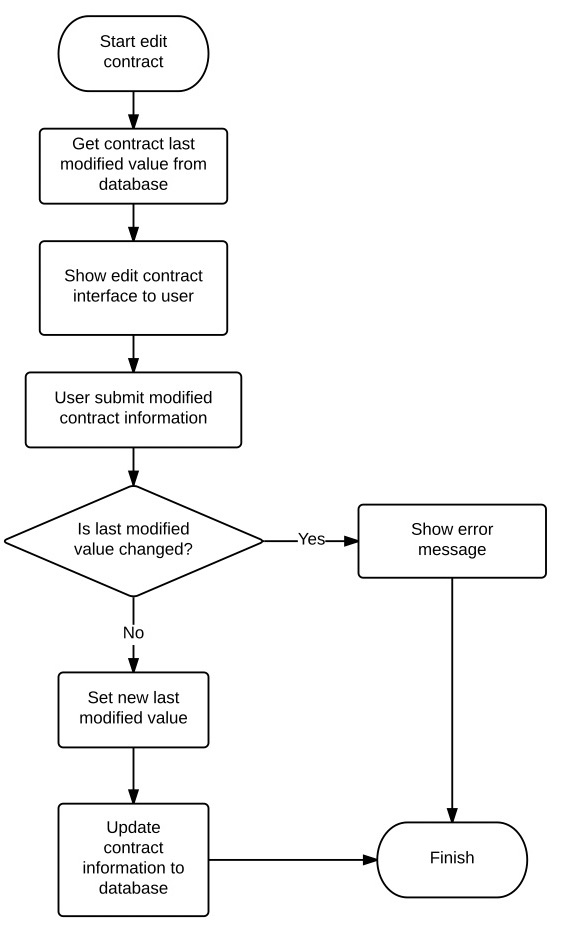


Figure Concurrency control flow

### Notification

#### Definition

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

#### Notification Methods

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

#### Notification use cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Trigger | Receiver | Notify method | Close notify trigger |
| 1 | Customer creates new contract | All staff | Web | Mark as read |
| 2 | Customer sends compensation request | All staff | Web | Staff resolves compensation |
| 3 | Customer sends new card request | All staff | Web | Staff prints new card |
| 4.1 | Contract is nearly expired (send first) | Customer | (by scheduler) Web, Email | Customer or Staff renew the contract |
| 4.2 | Contract is nearly expired (send 2nd time) | Customer | (by scheduler) Web, Email | Customer or Staff renew the contract |
| 4.3 | Contract is nearly expired (send 3rd time) | Customer | (by scheduler) Web, Email | Customer or Staff renew the contract |
| 5 | Contract is expired | Customer | (by scheduler) Web, Email | Customer or Staff renew the contract |
| 6 | Customer sends request cancel | All staff | Web | Staff resolves request cancel or Customer rejects the request cancel |
| 7 | Contract is cancelled because payment due date exceed | Customer | (by scheduler) Web (Staff and Customer), Email (Customer) | Mark as read |
| 8 | Paid Pending Contract start date come | Customer | (by scheduler) Web | Mark as read |

Table 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.

### System Scheduler Process

#### 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.

#### 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 dates is exceed… etc.

#### Solution

We create a Cronjob in Unix operating system that run on the same server of the Web application, Cronjob will automatically run at specific time that system administrator define when deploy the system. The Cronjob 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:
   1. Contract status
   2. Contract expired date
   3. Contract renew due date
   4. Contract payment due date
   5. Compensation requests
   6. New card requests
   7. Cancel request
3. Update the contract state follow the business rules.

#### Flow chart

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

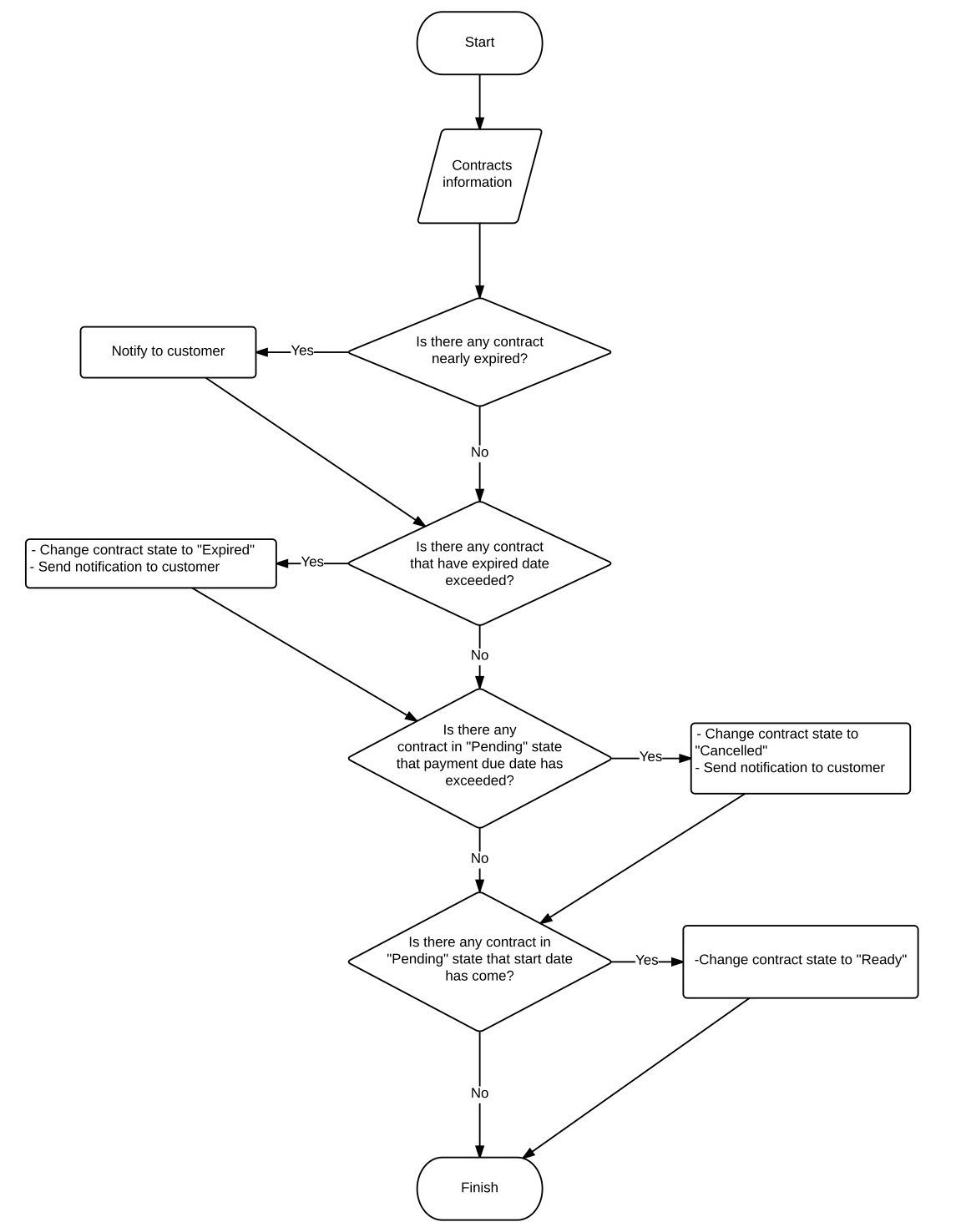


Figure System Scheduler Process

### NFC Card Data Format

#### Definition

According to [Android Developer Forum](https://developer.android.com/guide/topics/connectivity/nfc/index.html) 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.

#### 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 ForumTM NDEF 1.0 Specification (2006-07-24)*, NDEF Messages is stored in tag in this structure:

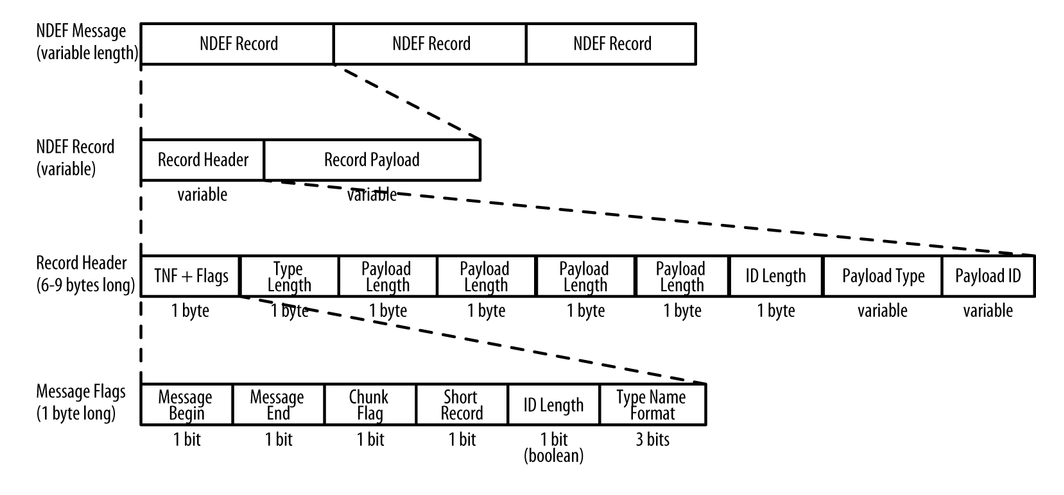


Figure NDEF message structure

**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 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.

#### 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 ForumTM 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.

**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 exists 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

# System Implementation & Test

## Introduction

### 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.

### Test Approach

* Goal: Test all features in the whole MIC system based on the core flow.
* Method: white-box testing, 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 will 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.

## Database Relationship Diagram

### Physical Diagram

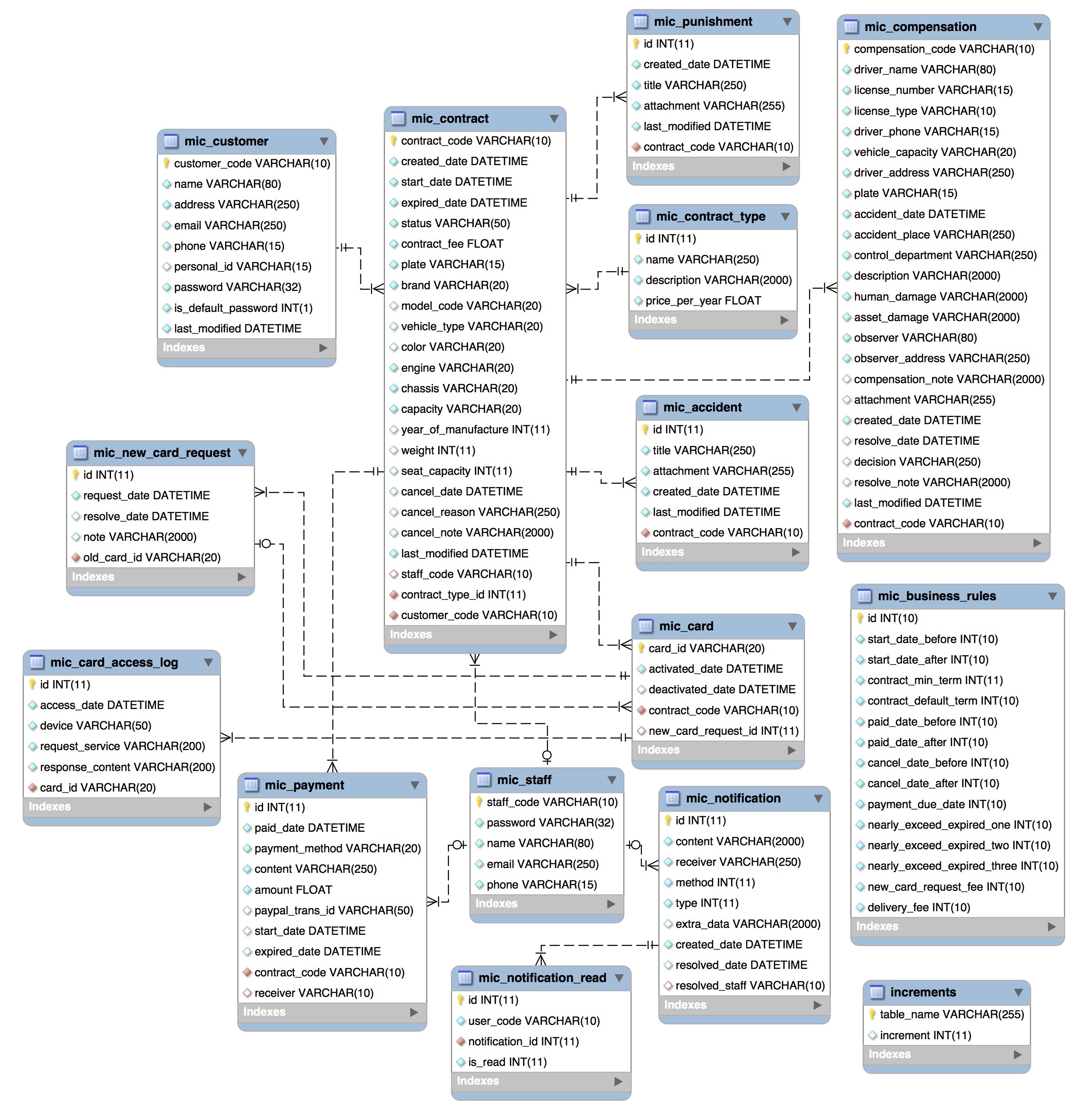


Figure 69 Physical diagram

### Data Dictionary

|  |  |
| --- | --- |
| Data dictionary: describe content of all tables | |
| Table Name | **Description** |
| increments | Support generates codes with prefixes sequentially. |
| mic\_business\_rules | Describe all configuration based on company’s business rules. |
| mic\_contract\_type | Describe all contract type in the system. |
| mic\_contract | Describe all contract detail information in the system. |
| mic\_accident | Describe all accident information belongs to contract in the system. |
| mic\_compensation | Describe all compensation detailed information belongs to contract in the system. |
| mic\_punishment | Describe all punishment information belongs to contract in the system. |
| mic\_card | Describe all card which have been issued in the system. |
| mic\_card\_access\_log | Describe all history access to card. |
| mic\_new\_card\_request | Describe all new card request in the system. |
| mic\_customer | Describe all customer profile available in the system. |
| mic\_notification | Describe all notification detail in the system. |
| mic\_notification\_read | Support to manage all notification in the system. |
| mic\_payment | Describe all payment information in the system. |
| mic\_staff | Describe all staff profile available in the system. |

Table 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 |
| 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) | No |
| license\_number | Compensation driver license number | varchar(15) | No |
| license\_type | Compensation driver license type | varchar(10) | No |
| driver\_phone | Compensation driver phone | varchar(15) | No |
| vehicle\_capacity | Compensation driver vehicle capacity | varchar(20) | No |
| driver\_address | Compensation driver address | varchar(250) | No |
| plate | Compensation driver vehicle plate | varchar(15) | No |
| accident\_date | Compensation accident date | datetime | No |
| accident\_place | Compensation accident place | varchar(250) | No |
| control\_department | Department which control this accident | varchar(250) | No |
| description | Compensation description | varchar(2000) | No |
| human\_damage | Accident human damage | varchar(2000) | No |
| asset\_damage | Accident asset damage | varchar(2000) | No |
| observer | Observer name | varchar(80) | No |
| observer\_address | Observer address | varchar(250) | No |
| 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 {PK} | Unique identifier of card. | 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 |
| 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 |
| resolved\_staff | Notification resolved staff | varchar(10) | 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 Data attribute dictionary

## Performance Measures

### Web application page load speed

#### Definition

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

#### Test environment

**Server:**

* **Operating system:** Ubuntu 14.04 Server 32 bit
* **RAM:** 512 MB
* **Storage:** 20 GB
* **Processor:** 2.6 GHz Intel Core i5
* **Network:** 1 Mbps

#### 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:**

|  |  |
| --- | --- |
| * Home * Register * Login * Staff / dashboard * Staff / list contract * Staff / detail contract * Staff / compensation * Staff / punishment * Staff / accident | * Staff / card * Staff / customer * Customer / dashboard * Customer / list contract * Customer / detail contract * Customer / compensation * Customer / punishment * Customer /accident * Customer / card |

#### 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.

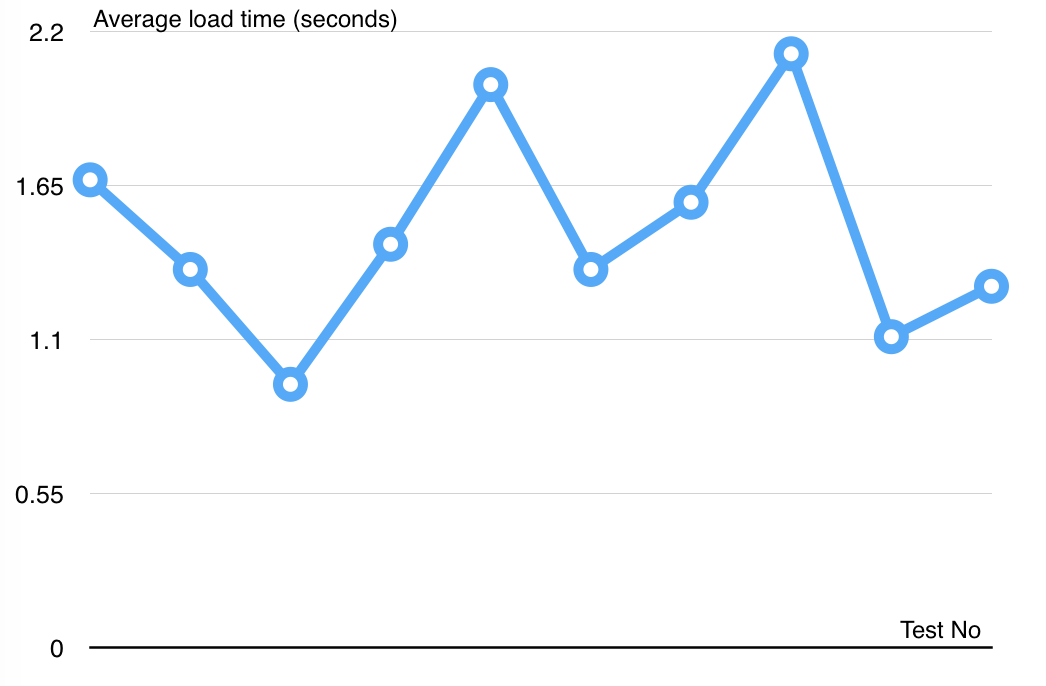


Figure 70 Web application page load speed test result

|  |  |  |
| --- | --- | --- |
| 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 Web application page load speed test result

### Mobile application API load speed

#### Definition

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

#### Test environment

**Server:**

* **Operating system:** Ubuntu 14.04 Server 32 bit
* **RAM:** 512 MB
* **Storage:** 20 GB
* **Processor:** 2.6 GHz Intel Core i5
* **Network:** 1 Mbps

**Client:**

* **Operating system:** Android 4.0
* **RAM:** 1GB

#### Test cases

Using mobile application to send all the 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 |

#### 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.

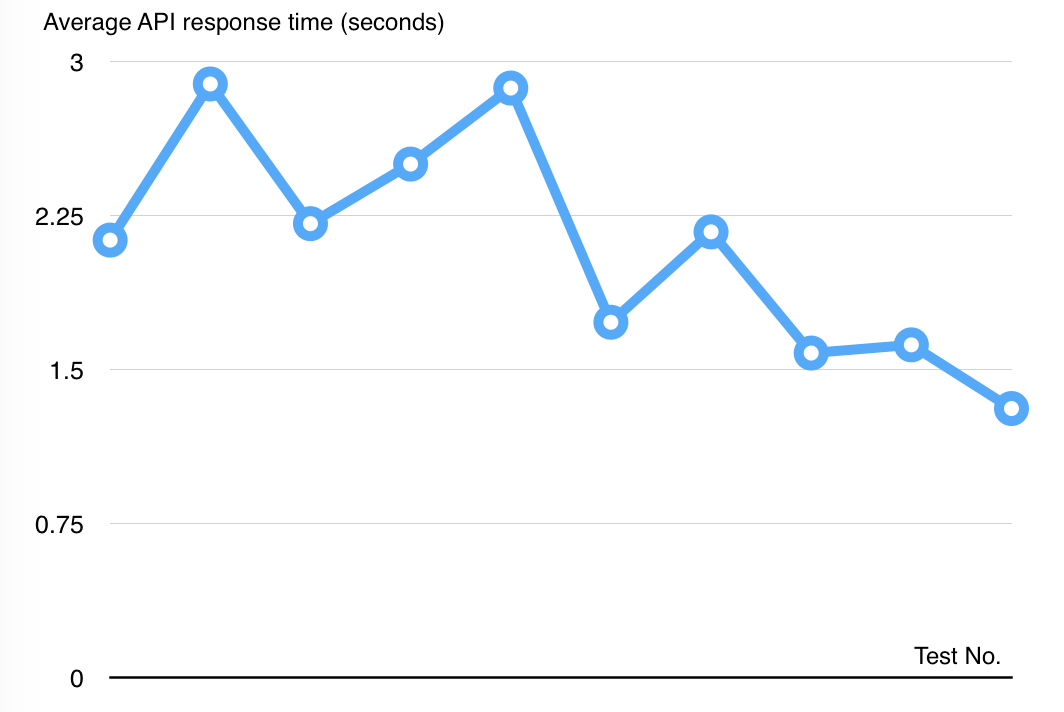


Figure 71 Mobile application API load speed test result

|  |  |  |
| --- | --- | --- |
| 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 Mobile application API load speed test result

## Test Plan

The overall purpose of testing is to ensure MIC system meets all of its 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.

### 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

### Features not to be tested

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

## System Testing Test Case