**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Green Bus Ticket System**

|  |  |
| --- | --- |
| **Group 1** | |
| **Group members** | Đỗ Ngọc Hoàng – SE61246  Trần Quang Trường – SE61129  Đoàn Minh Đức – SE61486 |
| **Supervisor** | Kiều Trọng Khánh |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | GBTS |

- Ho Chi Minh city, September 5th 2016 -

Table of Contents

[List of Tables 5](#_Toc464338621)

[List of Figures 6](#_Toc464338622)

[Definitions, Acronyms, and Abbreviations 7](#_Toc464338623)

[A. Report No. 1 Introduction 8](#_Toc464338624)

[1. Project Information 8](#_Toc464338625)

[2. Introduction 8](#_Toc464338626)

[3. Current Situation 8](#_Toc464338627)

[4. Problem Definition 8](#_Toc464338628)

[5. Proposed Solution 9](#_Toc464338629)

[5.1 Feature functions 9](#_Toc464338630)

[5.2 Advantages and disadvantages 9](#_Toc464338631)

[6. Functional Requirements 10](#_Toc464338632)

[7. Role and Responsibility 10](#_Toc464338633)

[B. Report No.2 Software Project Management Plan 12](#_Toc464338634)

[1. Problem Definition 12](#_Toc464338635)

[1.1. Name of this Capstone Project 12](#_Toc464338636)

[1.2. Problem Abstract 12](#_Toc464338637)

[1.3. Project Overview 12](#_Toc464338638)

[2. Project organization 15](#_Toc464338639)

[2.1. Software Process Model 15](#_Toc464338642)

[2.2. Roles and responsibilities 16](#_Toc464338643)

[2.3. Tools and Techniques 16](#_Toc464338644)

[3. Project Management Plan 18](#_Toc464338645)

[3.1. Software development life cycle 18](#_Toc464338647)

[3.2. Phase Detail 20](#_Toc464338648)

[3.3. All Meeting Minutes 22](#_Toc464338649)

[4. Coding Convention 22](#_Toc464338650)

[C. Report No. 3 Software Requirement Specification 23](#_Toc464338653)

[1. User Requirement Specification 23](#_Toc464338654)

[1. Unauthorized User Requirement 23](#_Toc464338655)

[2. Authorized User Requirement 23](#_Toc464338656)

[3. Passenger Requirement 23](#_Toc464338657)

[4. Staff Requirement 23](#_Toc464338658)

[5. Manager Requirement 23](#_Toc464338659)

[6. Admin Requirement 23](#_Toc464338660)

[7. Emulator Requirement 23](#_Toc464338661)

[8. Auto Handler Requirement 24](#_Toc464338662)

[2. System Requirement Specification 24](#_Toc464338663)

[1. External Interface Requirement 24](#_Toc464338665)

[2. System Overview Use Case 25](#_Toc464338666)

[3. List of Use Case 26](#_Toc464338667)

[3. Software System Attribute 84](#_Toc464338668)

[1. Usability 84](#_Toc464338670)

[2. Reliability 85](#_Toc464338671)

[3. Availability 85](#_Toc464338672)

[4. Security 85](#_Toc464338673)

[5. Maintenanability 85](#_Toc464338674)

[6. Portability 85](#_Toc464338675)

[7. Performance 85](#_Toc464338676)

[4. Conceptual Diagram 86](#_Toc464338677)

[D. Report No. 3 Software Requirement Specification 87](#_Toc464338679)

[1. Design Overview 87](#_Toc464338680)

[2. System Architecture Design 87](#_Toc464338681)

[4.2 Web Application Architecture Description 88](#_Toc464338682)

[4.3 Android Application Architecture Description 88](#_Toc464338683)

[3. Component Diagram 88](#_Toc464338684)

[4. Detailed Description 90](#_Toc464338685)

[4.1 Class Diagram 90](#_Toc464338688)

[4.2 Class diagram explanation 90](#_Toc464338689)

[4.3 Interactive Diagram 93](#_Toc464338690)

[4.4 User Interface 99](#_Toc464338691)

## List of Tables

[Table 1 : Roles and Responsibilities 11](#_Toc468572487)

[Table 2 : Hardware Requirement for Server 14](#_Toc468572488)

[Table 3 : Hardware Requirement for Mobile 14](#_Toc468572489)

[Table 4 : Software requirements 15](#_Toc468572490)

[Table 5 : Roles and responsibilities 17](#_Toc468572491)

[Table 6: Tools List 17](#_Toc468572492)

[Table 7: Technique List 17](#_Toc468572493)

[Table 8: Software Development Life Cycle Detail 19](#_Toc468572494)

[Table 9: Phase 1: Infrastructure 20](#_Toc468572495)

[Table 10: Phase 2: System & Web app 21](#_Toc468572496)

[Table 11: Phase 3: Web service 21](#_Toc468572497)

[Table 12: Phase 4: Mobile app 22](#_Toc468572498)

[Table 13: <Unauthorized User> Login 27](#_Toc468572499)

[Table 14: <Unauthorized User> Activate account 29](#_Toc468572500)

[Table 15: < Authorized User> Overview use case 31](#_Toc468572501)

[Table 16: < Authorized User> Log out 32](#_Toc468572502)

[Table 17: < Passenger> Get NFC card 35](#_Toc468572503)

[Table 18: < Passenger> Add credit 37](#_Toc468572504)

[Table 19: < Passenger> Edit card name 38](#_Toc468572505)

[Table 20: < Passenger> Get outcome report 41](#_Toc468572506)

[Table 21: < Passenger> Find bus 42](#_Toc468572507)

[Table 22: < Passenger> Get route 44](#_Toc468572508)

[Table 22: < Passenger> Get route 45](#_Toc468572509)

[Table 22: < Passenger> Get route 47](#_Toc468572510)

[Table 22: < Passenger> Get route 48](#_Toc468572511)

[Table 23: < Staff > Add card 51](#_Toc468572512)

[Table 24: < Staff > Activate/Deactivate card 53](#_Toc468572513)

[Table 25: < Staff > Search passenger 54](#_Toc468572514)

[Table 26: < Staff > Edit passenger 57](#_Toc468572515)

[Table 26: < Staff > Edit passenger 58](#_Toc468572516)

[Table 27: < Manager > Add ticket type 61](#_Toc468572517)

[Table 28: < Manager > Edit ticket type 63](#_Toc468572518)

[Table 29: < Manager > Delete ticket type 65](#_Toc468572519)

[Table 30: < Manager > Add credit plan 67](#_Toc468572520)

[Table 31: < Manager > Edit credit plan 69](#_Toc468572521)

[Table 32: < Manager > Delete credit plan 71](#_Toc468572522)

[Table 33: < Manager > Get income report 73](#_Toc468572523)

[Table 34: < Manager > Create promotion 75](#_Toc468572524)

[Table 35: < Admin> Search account 76](#_Toc468572525)

[Table 36: < Admin> Add account 79](#_Toc468572526)

[Table 37: < Admin> Edit account 80](#_Toc468572527)

[Table 39: < Auto Handler > Parse bus route 82](#_Toc468572528)

[Table 40: < Auto Handler > Suggest promotion 84](#_Toc468572529)

[Table 41: < Auto Handler > Send notification 85](#_Toc468572530)

[Table 42: < Emulator > Write card 87](#_Toc468572531)

[Table 43: < Emulator > Verify ticket 89](#_Toc468572532)

[Table 44: Conceptual Diagram Data Dictionary 91](#_Toc468572533)

[Table 45: Component Dictionary 94](#_Toc468572534)

## List of Figures

[Figure 1 : Scrum model 17](#_Toc464338739)

[Figure 2 : <Use case> Web application overview 26](#_Toc464338740)

[Figure 3 : <Use case> Mobile application overview 27](#_Toc464338741)

[Figure 4: <Unauthorized User> Overview use case 27](#_Toc464338742)

[Figure 5: <Unauthorized User> Login 27](#_Toc464338743)

[Figure 6: <Unauthorized User> Activate account 29](#_Toc464338744)

[Figure 7: < Authorized User> Overview use case 30](#_Toc464338745)

[Figure 8: < Authorized User> Overview use case 31](#_Toc464338746)

[Figure 9: < Authorized User> Log out 32](#_Toc464338747)

[Figure 10: < Passenger> Overview use case 34](#_Toc464338748)

[Figure 11: < Passenger> Get NFC card 34](#_Toc464338749)

[Figure 12: < Passenger> Add credit 36](#_Toc464338750)

[Figure 13: < Passenger> Edit card name 38](#_Toc464338751)

[Figure 14: < Passenger> Get outcome report 40](#_Toc464338752)

[Figure 15: < Passenger> Find bus 42](#_Toc464338753)

[Figure 16: < Passenger> Get route 44](#_Toc464338754)

[Figure 17: < Staff > Overview use case 45](#_Toc464338755)

[Figure 18: < Staff > Add card 46](#_Toc464338756)

[Figure 19: < Staff > Activate/Deactivate card 48](#_Toc464338757)

[Figure 20: < Staff > Search passenger 49](#_Toc464338758)

[Figure 21: < Staff > Edit passenger 51](#_Toc464338759)

[Figure 22: < Manager > Overview use case 53](#_Toc464338760)

[Figure 23: < Manager > Add ticket type 53](#_Toc464338761)

[Figure 24: < Manager > Edit ticket type 56](#_Toc464338762)

[Figure 25: < Manager > Delete ticket type 58](#_Toc464338763)

[Figure 26: < Manager > Add credit plan 59](#_Toc464338764)

[Figure 27: < Manager > Edit credit plan 61](#_Toc464338765)

[Figure 28: < Manager > Delete credit plan 63](#_Toc464338766)

[Figure 29: < Manager > Get income report 65](#_Toc464338767)

[Figure 30: < Manager > Create promotion 67](#_Toc464338768)

[Figure 31: < Admin> Overview use case 69](#_Toc464338769)

[Figure 32: < Admin> Search account 69](#_Toc464338770)

[Figure 33: < Admin> Add account 71](#_Toc464338771)

[Figure 34: < Admin> Edit account 73](#_Toc464338772)

[Figure 35: < Admin> Delete account 75](#_Toc464338773)

[Figure 36: < Auto Handler > Overview use case 77](#_Toc464338774)

[Figure 37: < Auto Handler > Parse bus route 77](#_Toc464338775)

[Figure 38: < Auto Handler > Suggest promotion 79](#_Toc464338776)

[Figure 39: < Auto Handler > Send notification 81](#_Toc464338777)­

[Figure 40: < Emulator > Overview use case 82](#_Toc464338778)

[Figure 41: < Emulator > Write card 83](#_Toc464338779)

[Figure 42: < Emulator > Verify ticket 84](#_Toc464338780)

[Figure 43: Conceptual Diagram 87](#_Toc464338781)

[Figure 44: System Architecture 88](#_Toc464338782)

[Figure 45: Component Diagram 89](#_Toc464338783)

[Figure 46: Class Diagram 91](#_Toc464338784)

[Figure 47: <Sequence Diagram> Add Credit 95](#_Toc464338785)

[Figure 48: <Sequence Diagram> Get Cards 95](#_Toc464338786)

[Figure 49: <Sequence Diagram> Get Income Report 96](#_Toc464338787)

[Figure 50: <Sequence Diagram> Edit Passenger 96](#_Toc464338788)

[Figure 51: <Sequence Diagram> Activate/Deactivate Card 97](#_Toc464338789)

[Figure 52: <Sequence Diagram>Add Card 97](#_Toc464338790)

[Figure 53: <Sequence Diagram>Get Outcome Report 98](#_Toc464338791)

[Figure 54: <Sequence Diagram>Create Promotion 98](#_Toc464338792)

[Figure 55 : <Activity Diagram> Add Credit 99](#_Toc464338793)

[Figure 56: <Activity Diagram>Find Bus 100](#_Toc464338794)

[Figure 57: <Activity Diagram> Get Outcome Report 100](#_Toc464338795)

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| GBTS | Green Bus Ticket System |
| NFC | Near Field Communication |

# Report No. 1 Introduction

## Project Information

* Project name: **Green Bus Ticket System**
* Abbreviation: **GBTS**
* Product Type: **Web app & Mobile app**
* Start Date: **September 5th 2016**
* End Date: **December 2016**

## Introduction

Nowadays, bus is the most popular public transportation, buses are very safe, time efficient and cheap. In Vietnam, many people daily travel in these local buses from their house to their offices, schools and other places. For traveling by bus, people have to buy a paper bus ticket on a bus. The bus sometimes is crowded, buying ticket is not comfortable and inefficient. Moreover, the ticket is thrown anywhere after had used. The cost is spent very large and the ticket is created to garbage that cause wasting of wood resource. Based on researches and analysis, we proposed a solution for bus managers, passengers in Vietnam.

We build a system, which help bus managers, passengers to solve their current problems. In the process of analysis, we believe that NFC technology is suitable to resolve the problem by using NFC card to save passenger’s account information, so these NFC cards will be electronic bus tickets. NFC cards are cheap, small, convenient to bring with and easy to perform checking or validating. NFC cards are very durable, no-battery required and easy to rewrite information which is suitable to resuse. Beside of that, we also provide a system to help passengers manage their account, NFC cards and find buses based on their starting point and destination. Moreover, this system can help bus company manage their buses and tickets for income report.

## Current Situation

* For using bus, the passengers have to buy a paper bus ticket on a bus and pay cash for the driver assistant.
* Some buses route has already provided the buy ticket machine but the driver must take a cash, then must pay cash in return.

## Problem Definition

Research the current buying bus ticket process, we found that the traditional process has many advantages and disavantages below:

* Advantages:
  + Easy for anyone to use the bus, they need to bring cash along and pay for the paper ticket.
  + No technical skills needed.
  + No need to bring NFC card along.
* Disadvantages:
  + Buying a paper bus ticket on a bus which sometimes is crowded is not comfortable and inefficient.
  + The ticket is thrown anywhere after had used is wasting of money and wood resource.
  + The ticket sets (monthy or annually) are easily being lost or fray. Besides, the driver assistance has not good behavior with those passengers.
  + Some buses route has already provided the buy ticket machine but the driver must take cash, then pay cash in return. It is not comfortable and inefficient both passenger and driver.

## Proposed Solution

Our solution is a new system, which will cover the whole bus ticket buy and sell process combine with NFC technology for buying bus ticket easy and efficient. In addition, our solution is also help passengers find buses route.

GBTS includes a web app and a mobile app, with the following features.

### Feature functions

* + Using NFC technology for storing passenger’s account information. The passengers can buy NFC card from bus companies or via online obly one time for using many times. The passengers use this NFC card as an electronic ticket each time they use bus.
  + There is an emulator on each bus which directly communicates with NFC card to access passenger account information. The emulator then sends information to the system to process. Next, the system checks, minus credit on related account and notify tiket details to passenger’s mobile phone (if they have already installed mobile application).
  + With web application, passengers can manage their account and outcome report. In the other hand, bus managers can manage their buses, tickets, passengers and income report. They can also notify or suggest the promotion campaign to passengers.
  + With mobile application, passengers can receive ticket details or promotion notification, manage their account and outcome report. The mobile application can also help them to find buses route.
  + In the other hand, bus companies also issue anonymous NFC cards with fixed balance. Passengers who forget their card can buy these anonymous cards for temporary using.

### Advantages and disadvantages

* Advantages:
  + - NFC card is cheaper and very durable, no-battery requied, small and easy to bring with.
    - The system can replace the traditional way which always need paper bus tickets. We try to reduce garbages and save wood resource.
    - The system provides a new way to accost passengers with promotion or promotion.
    - Standardize the process and make them available to more and more transportation services.
  + Disadvantages:
    - Each passenger must have at least one primary NFC card to use the system and up to three additional NFC cards.
    - Cost of buying each bus an emulator in order to communicate with NFC card.

## Functional Requirements

Passenger component:

* + Activate for a new account.
  + Get NFC cards.
  + Add credit to card.
  + Edit card name
  + Get outcome report.
  + Find bus.

Staff component:

* + Search for passenger.
  + Edit passengers.
  + Manage NFC cards.

Manager component:

* + Manage ticket type.
  + Manage credit plan.
  + Get income report.
  + Create promotions.

Admin component:

* + Manage all accounts.

Emulator component:

* + Read & write NFC card.
  + Verify card.

Auto Handler component:

* + Suggest promotions
  + Send notification
  + Parse bus route

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| 1 | Kiều Trọng Khánh | Project Manager | Supervisor | [khanhkt@fpt.edu.vn](mailto:khanhkt@fpt.edu.vn) |
| 2 | Đỗ Ngọc Hoàng | Developer | Leader | [hoangdnse61246@fpt.edu.vn](mailto:hoangdnse61246@fpt.edu.vn) |
| 3 | Trần Quang Trường | Developer | Member | [truongtqse61129@fpt.edu.vn](mailto:truongtqse61129@fpt.edu.vn) |
| 4 | Đoàn Minh Đức | Developer | Member | [ducdmse61486@fpt.edu.vn](mailto:ducdmse61486@fpt.edu.vn) |

Table 1 : Roles and Responsibilities

# Report No.2 Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

* **Official name**: Green Bus Ticket System
* **Vietnamese name**: Hệ thống bán vé xe buýt tiện lợi
* **Abbreviation**: GBTS

### Problem Abstract

For the goal of improving current bus system, especially the buying ticket process. We provide the solution for both passengers in buying, using bus ticket and the bus managers in managing efficiently. But there are many kinds of passengers may use our system, some of them don’t have any information technology skills like the old people. So we have to find the best convenient way to make our system simplest and easy to use for anyone.

Our system use NFC technology, this technology is quite new, so we may need times to research and integrate NFC to our system. Currently, only Android and Window Phone are supporting NFC technology, so we need knowledges on these operating systems for implementation.

### Project Overview

#### Current Situation

Below are the problems encountered in this project:

* **NFC security**: working with NFC, there are some problems may happen, any device support NFC like smartphone can read and write to this, so it can be counterfeited, attacked during data transmission caused data loss, data, corruption.
* **Emulator’s problem:** using emulator on bus in order to validate and process NFC cards is sometime not work or damaged.
* **Passengers’s habitat**: passengers are used to buying paper ticket with cash, so deploy the system in real life may take long time.
* **Account information secutiry**: the system allow passenger to buy ticket credits, this function may becom tatget for hacking and cheating.

#### The Proposed System

* After doing many researches on technology for saving information, we choose NFC technology as this technology is very capable of resolve the current situations in selling bus tikets. The basic idea is to use a NFC tag that each NFC card contain a unique card ID as a ticket that can be reused instead of using paper.
* In task assignment, we assign to member using vertical model 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.
* Our system includes three subsystems:
  + An online web application for passengers, staffs, managers and administrator.
  + A mobile application for passengers.
  + A mobile application for emulator.

##### ***Web Application***

Web application consists of three main parts:

* For passengers:
  + Activate for a new account.
  + Get NFC cards.
  + Add credit to card.
  + Edit card name
  + Get outcome report.
  + Find bus.
* For staff:
  + Search for passenger.
  + Edit passengers.
  + Manage NFC cards.
  + Publish scratch cards.
* For manager:
  + Manage ticket type.
  + Manage credit plan.
  + Get income report.
  + Manage offer subscription
  + Create promotions.
* For administrator:
  + Manage accounts.

Besides, website application also provides an API interface for two mobile applications to retrieve, update data from mobile applications.

##### ***Mobile Application***

There will be 2 applications which will be used by passengers and emulator. The mobile applications included functions as below:

* For passengers:
  + Activate for a new account.
  + Get NFC cards.
  + Add credit to card.
  + Edit card name
  + Get outcome report.
  + Find bus.
  + Buy ticket by phone.
* For emulator:
  + Read & write NFC card.
  + Verify card.

#### Boundaries of the System

* The system is mostly built based on real processes of bus ticket in Ho Chi Minh City. Our main target is improving the current process and makes it more convenient and efficient in Ho Chi Minh City.
* Any bus system which deployed this system must set up devices to operate, includes:
  + Emulator can read a NFC Card, with internet connection.
  + NFC cards with account information.
* The completed product includes:
  + Website application
  + Android mobile application for passengers and for emulator.

#### Future Plans

With further research and development, the system can apply the following features:

* Bus companies can place many emulators at bus stop with an assistant for helping the passengers recharge their NFC card credit without using mobile app or website. It is suitable for any passenger who lack of knowledge in information technology.
* Allow passengers add credit to their account. The system will minus credit on their account automatically while buying ticket in case of their cards is out of credit.

#### Development Environment

##### ***Hardware requirements***

* **For web application server**

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Window Server 2008 R2 | Window Server 2012 R2 |
| Computer Processor | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core |
| Computer Memory | 2GB of RAM | 4GB of RAM or more |

Table 2 : Hardware Requirement for Server

* **For Mobile**

|  |  |  |
| --- | --- | --- |
| Android | Minimum | Recommended |
| Internet Connection | Wi-Fi or 3G (1 Mbps) | Wi-Fi or 3G (8 Mbps) |
| Operating System | Android 4.4.2 | Android 6.0.0 |
| Mobile Processor | Cortex-A7 Dual-Core 1.3GHz | Cortex-A7 Dual-Core 1.3GHz |
| Mobile Memory | 1GB of RAM | 2GB of RAM or more |

Table 3 : Hardware Requirement for Mobile

##### ***Software requirements***

|  |  |  |
| --- | --- | --- |
| Software | Name / Version | Description |
| Operating system | Window Server 2012 R2 | Operating system and platform for development |
| Environment | .NET Framework 4.5 | Specification for developing web application |
| IDE | Visual Studio 2015, Android Studio v2.1 | Used for implement website and Android Mobile App. |
| Design Model tool | StartUML v2.5.1 | Used for creating modal and diagrams. |
| DBMS | Microsoft SQL Server 2014 | Used to create & manage the database for system |
| Document storage | Github | Used for storing document |
| Store and manage source code | Github & SourceTree | Used to store all source code |

Table 4 : Software requirements

## Project organization



### Software Process Model

The project is developed under scrum model. Scrum model is capable with current situation in our team. We choose this model because the following reasons:

* The bus ticket problem not fully defined and the bus business in company cannot be fully understood. The users of our system are vary, so we may have many changes during development process to adapt the requirements.
* This project use NFC technology, which is a new technology that may need many times to research and implement.
* Scrum adopts an empirical approach, accepting that the problem is not fully understood or defined, focusing instead on maximizing the team's ability to deliver quickly, to respond to emerging requirements and to adapt to evolving technologies and changes in market conditions.

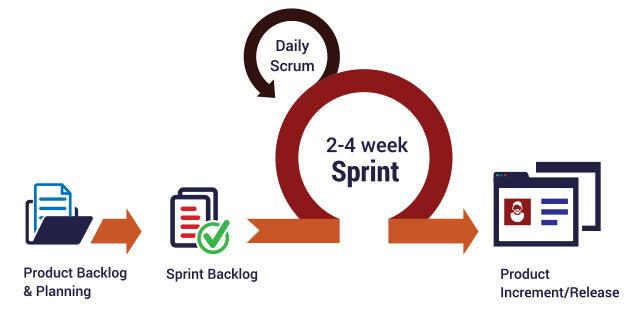


Figure 1 : Scrum model

Reference: <http://skytechnovation.com/scrum-development-model/>

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| No | Full name | Role in Group | Responsibilities |
| 1 | Kiều Trọng Khánh | Supervisor, Project Manager | * Specify user requirements * Control the development process * Give out technique and business analysis support |
| 2 | Đỗ Ngọc Hoàng | Team leader, B.A, Developer, Tester | * Managing process * Designing database * Clarifying requirements * Prepare documents * GUI design * Create test plan * Coding * Testing |
| 3 | Trần Quang Trường | Team member,  B.A, Developer,  Tester | * Designing database * Clarifying requirements * Prepare documents * GUI design * Create test plan * Coding * Test |
| 4 | Đoàn Minh Đức | Team member,  B.A, Developer,  Tester | * Designing database * Clarifying requirements * Prepare documents * Create test plan * Coding * Test |

Table 5 : Roles and responsibilities

### Tools and Techniques

|  |  |
| --- | --- |
| Tool | Name / version |
| Web server | IIS |
| Development tool | Visual Studio, Android Studio |
| DBMS | SQL Server 2014 |
| Source control | Github & SourceTree |
| Modeling tool | StarUML v5.0.1 |
| Document tool | Microsoft Word 2010 |

Table 6: Tools List

|  |  |
| --- | --- |
| Technique | Name / version |
| Frontend | HTML5, CSS, JavaScript, jQuery |
| Backend | ASP.Net, Android, NFC |

Table 7: Technique List

## Project Management Plan



### Software development life cycle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Phase | Description | Deliverables | Resource needed | Dependencies and Constrains | Risks |
| Infrastructure | - Identify and clarify overall requirements.  - Determine the system architecture.  - Build infrastructure for the project. | - Database design.  - System main structure. | 20 man-days |  | - Unclear project scope.  - Lack of member share of understand. |
| System  &  Web app | - Identify software and hardware requirements.  - Implements all web app modules.  - Design the web UI  - Build the web app | - Complete web app for all roles of the system. | 60 man-days | - Depends on “Infrastructure” | - Unclear project scope.  - Lack of experience. |
| Web services | - Identify requirements for mobile app.  - Build required API for mobile app. | - API for mobile app. | 20 man-days | - Depends on “Web app & System” | - Lack of experience. |
| Mobile apps | - Design the mobile UI  - Build mobile apps for end users and emulator. | - Complete Android Apps | 20 man-days | - Depends on “Web services” | - Lack of experience.  - Lack of NFC knowledge |

Table 8: Software Development Life Cycle Detail

### Phase Detail

#### Phase 1: Infrastructure

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| 1. Assessment | - Determine requirements.  - Create product backlog. | * HoangDN * DucDM * TruongTQ |
| 2. Selection | - Determine system architecture: ASP .NET MVC.  - Determine software design pattern: Repository & Service.  - Determind all core functions. | * HoangDN * DucDM * TruongTQ |
| 3. Development | - Create the main structure of project. | * HoangDN * DucDM * TruongTQ |
| 4. Review | - Review all completed works and presentation.  - Create sprint backlog. | * HoangDN * DucDM * TruongTQ |

Table 9: Phase 1: Infrastructure

#### Phase 2: System & Web app

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| 1. Assessment | - Determine requirements for System and Web app.  - Update product backlog. | * HoangDN * DucDM * TruongTQ |
| 2. Selection | - Determind all functions according to requirements of System and Web app. | * HoangDN * DucDM * TruongTQ |
| 3. Development | - Design and build prototype for web UI  - Create conceptual diagram  - Design class diagram  - Design database  - Implement the entire web UI: layouts, detail pages, etc.  - Implement all the functions in controllers.  - Build needed utility classes | * HoangDN * DucDM * TruongTQ |
| 4. Review | - Review all completed works and presentation.  - Create sprint backlog. | * HoangDN * DucDM * TruongTQ |

Table 10: Phase 2: System & Web app

#### Phase 3: Web service

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| 1. Assessment | - Determine requirements for Web service.  - Update product backlog. | * HoangDN * DucDM * TruongTQ |
| 2. Selection | - Determind all functions according to requirements of Web service. | * HoangDN * DucDM * TruongTQ |
| 3. Development | - Create API for mobile app based on functions on the web app. | * HoangDN * DucDM * TruongTQ |
| 4. Review | - Review all completed works and presentation.  - Create sprint backlog. | * HoangDN * DucDM * TruongTQ |

Table 11: Phase 3: Web service

#### Phase 4: Mobile app

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| 1. Assessment | - Determine requirements for System and Mobile app.  - Update product backlog. | * HoangDN * DucDM * TruongTQ |
| 2. Selection | - Determind all functions according to requirements of Mobile app. | * HoangDN * DucDM * TruongTQ |
| 3. Development | - Implement all the functions based on the designed UI and the provided API. | * HoangDN * DucDM * TruongTQ |
| 4. Review | - Review all completed works and presentation.  - Create sprint backlog. | * HoangDN * DucDM * TruongTQ |

Table 12: Phase 4: Mobile app

### All Meeting Minutes

Meeting minutes are contained in folder “Meeting minutes” in the attached CD.

## Coding Convention

**C#:** Using to develop website and web service.

Summary:

* Naming Convention:
  + For variable’s name, use camel case. Eg: minValue, maxValue…
  + For function name, class name, use Pascal case. Eg: AddIncome, AddExpense…
* Layout Convention:
  + Indent continuation one tab stop (four spaces).
  + Add at least one blank line between method definitions and property definitions.
  + Use parentheses to make clauses in an expression apparent.

**Android:** Using to develop mobile application

Summary:

* Naming Convention:
  + For variable’s name, use camel case. Eg: minValue, maxValue…
  + For function name, class name, use Pascal case. Eg: AddIncome, AddExpense…
  + For resource file names are written in lowercase\_underscore. Eg: my\_name
  + Declarations Convention:
    - One declaration per line is recommended.
  + Using C# Code Convention from:
    - <https://msdn.microsoft.com/en-us/library/ff926074.aspx>
  + Using Android Code Convention from
    - <https://source.android.com/source/code-style.html>



# Report No. 3 Software Requirement Specification

## User Requirement Specification

### Unauthorized User Requirement

Unauthorized is user does not login to this system. Unauthorized only has two functions.

* Activate new account.
* Login.

### Authorized User Requirement

Authorized User is user who has logged into system. Authorized User only has two functions.

* Edit profile.
* Logout.

### Passenger Requirement

Passenger can use some following functions:

* + Get NFC cards
* Add credit to card
* Edit card name
* Get outcome report
* Find bus

### Staff Requirement

Staff is the user who interacts directly to passengers, they can use some following functions:

* Search for passenger
* Edit passengers
* Manage NFC cards
* Publish scratch cards

### Manager Requirement

Manager is the Staff supervisor, they can use some following functions:

* Manage ticket type
* Manage credit plan
* Get income report
* Manage offer subscription
* Create promotions

### Admin Requirement

Admin is the user who responsible for managing accounts for the whole system, they can use some following functions:

* Manage account

### Emulator Requirement

Emulator is the device which can interact with NFC card, it can use some following functions:

* Read & write NFC card
* Verify card

### Auto Handler Requirement

Auto Handler can use some following functions:

* + Suggest promotions
  + Auto extend subscription
  + Crawl bus routes
  + Auto generate server token

## System Requirement Specification



### External Interface Requirement

#### User Interface

* The user interface uses Vietnamese language in both web application and mobile application.
* The user interface displays best on 1366x768 resolutions on desktop, and 1080x1920 on mobile.

#### Hardware Interface

* Desktop:
  + RAM: 2GB
  + CPU: 2.5Ghz
* Android Smartphone:
  + RAM: 1GB
  + CPU: 1.7 Ghz

#### Software Interface

* Web application: Works with Firefox (v30 or above), Google Chrome (v25 or later) or with any web browser that supports HTML5 & CSS3.
* Mobile application: Android operating system (v4.0 or later)

#### Communication Protocol

* Use HTTP 1.1 to communicate between the web browser and the web server.
* Use HTTP 1.1 to communicate between the mobile application and the web service.

### System Overview Use Case

#### Web Application



Figure 2 : <Use case> Web application overview

#### Mobile Application



Figure 3 : <Use case> Mobile application overview

### List of Use Case

#### Web Application

#### Unauthorized User

#### <Unauthorized User> Overview use case



Figure 4: <Unauthorized User> Overview use case

#### <Unauthorized User> Login



Figure 5: <Unauthorized User> Login

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS01** | | | |
| **Use Case No.** | GBTS01 | **Use Case Version** | 1.0 |
| **Use Case Name** | Login | | |
| **Author** | Tran Quang Truong | | |
| **Date** | September 18, 2016 | **Priority** | High |
| **Actor:**   * Unauthorized User   **Summary:**   * This use case allows unauthorized user to log into the system with a specific role   **Goal:**   * Authenticate user, redirect user to a specific view base on the role.   **Triggers:**   * Unauthorized User sends Login account command.   **Preconditions:**   * Valid phone number and password is inputted.   **Post Conditions:**   * **Success:** Unauthorized User is logged into the system as an Authorized User. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 2 | Unauthorized User inputs information then send Login command.  [Exception 1] | Unauthorized user will login system with their specified role.  The system redirects to the role’s view.  [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Unauthorized user leaves phone number and password field empty | System display an error message. |   **Exception:**  [Exception 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Unauthorized user inputs wrong phone number and password | System displays an error message. |   **Relationships: N/A**  **Business Rules:**   * System authenticate user by checking the phone number and password. * User enter password in the hidden text field and password must be encrypted before sending to server. * After logged in to the system, actor will be redirect to a specific view: * If the role is “Hành khách”, the system will display Passenger view. * If the role is “Quản lý”, the system will display Bus Manager view. * If the role is “Nhân viên”, the system will display Bus Manager view. * If the role is “Quản trị”, the system will display Administrator view. | | | |

Table 13: <Unauthorized User> Login

#### <Unauthorized User> Activate account

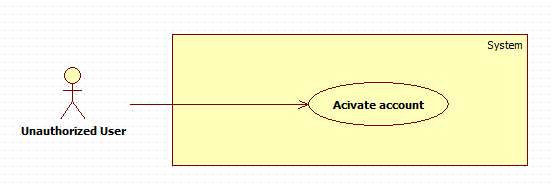


Figure 6: <Unauthorized User> Activate account

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS02** | | | |
| **Use Case No.** | GBTS02 | **Use Case Version** | 1.0 |
| **Use Case Name** | Activate account | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Unauthorized User   **Summary:**   * This use case allows Unauthorized User to activate a new account.   **Goal:**   * Unauthorized User can activate a new account with an NFC card to become a Passenger.   **Triggers:**   * Unauthorized User sends “Activate account” command.   **Preconditions:**   * Actor logged in system with role “Unauthorized User”. * Actor has a NFC card which isn’t activated yet.   **Post Conditions:**   * **Success:** New account is created. * **Fail:** Send error SMS.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Unauthorized User sends “Activate account” command by sending an SMS with their NFC card id to the system. | System gets and verifies the sender’s phone number and card id from the SMS.  System creates a new account with the phone number and maps the NFC card which has card id from SMS with this account.  System sends a SMS contains account login detail back to user’s phone number.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1,2]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Unauthorized User sends SMS with wrong format. | System sends an error back to user’s phone number: “Tin nhan sai cu phap, vui long kiem tra lai.” |   **Relationships:** N/A  **Business Rules:**   * New account will be created with input information and its status is “True”. * New account role is Passenger. * After activating account successful, user can login to the system with account login detail. | | | |

Table 14: <Unauthorized User> Activate account

#### Authorized User

#### < Authorized User> Overview use case



Figure 7: < Authorized User> Overview use case

#### < Authorized User> Edit profile



Figure 8: < Authorized User> Overview use case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS03** | | | |
| **Use Case No.** | GBTS03 | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit profile | | |
| **Author** | Tran Quang Truong | | |
| **Date** | September 18, 2016 | **Priority** | Low |
| **Actor:**   * Authorized User   **Summary:**   * This use case allows authorized users to edit their personal information.   **Goal:**   * Authorized user’s personal information is updated when they want to change their password or full name.   **Triggers:**   * Authorized User sends Edit profile command.   **Preconditions:**   * User must have logged in.   **Post Conditions:**   * **Success:** Authorized user’s personal information is updated in the database. * **Fail:** Error message is displayed.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Authorized User sends Edit profile command.  [Alternative 1] | System list out information of account in system and components:   * “Số điện thoại”: label * “Họ tên”: free text input, length 6-50 * “Password:: free password input, length 6-50 * Update command. | | 2 | Authorized User updates information. |  | | 3 | Authorized User sends Update command. | System validates the input then updates the account’ profile.  System displays a successful message: “Cập nhật thành công”  [Exeption 1] |   **Alternative Scenario: N/A**  **Exception:**   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Authorized User input password field with wrong format. | System displays error message: “Password từ 6-50 ký tự” |   **Relationships: N/A**  **Business Rules:**   * User update account profile in case of changing full name or password * Password must be encrypted before saving to the system. * User cannot update phone number. | | | |

Table 15: < Authorized User> Overview use case

#### < Authorized User> Log out

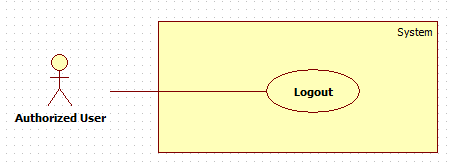


Figure 9: < Authorized User> Log out

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS04** | | | |
| **Use Case No.** | GBTS04 | **Use Case Version** | 1.0 |
| **Use Case Name** | Logout | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Authorized User   **Summary:**   * This use case allows Authorized User to logout.   **Goal:**   * Authorized User can logout of the system.   **Triggers:**   * Authorized User sends “Logout” command.   **Preconditions:**   * Actor logged in system with role “Authorized User”.   **Post Conditions:**   * **Success:** Authorized User is logged out. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Authorized User sends “Logout” command. | System clears the current session of the user then redirects to home page. |   **Alternative Scenario:** N/A  **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * After logging out of the system, user is no longer has permission to access any authorized function of the system. | | | |

Table 16: < Authorized User> Log out

#### Passenger

#### < Passenger> Overview use case



Figure 10: < Passenger> Overview use case

#### < Passenger> Get NFC card



Figure 11: < Passenger> Get NFC card

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS05** | | | |
| **Use Case No.** | GBTS05 | **Use Case Version** | 1.0 |
| **Use Case Name** | Get NFC card | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to get their NFC cards list.   **Goal:**   * The passenger can view all their NFC cards and the information such as card id, credit remaining of a specific card.   **Triggers:**   * Passenger sends “Get NFC cards” command.   **Preconditions:**   * Actor logged in system with role “Passenger”. * Passenger has at least one NFC card in the system.   **Post Conditions:**   * **Success:** The NFC card list is show. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Get NFC cards” command. | System gets NFC card list which belong to current logged account then display “Card list” view, each card has following information:  + “Mã thẻ”: label  + “Tên thẻ”: free text input, length 3-50, required  + “Ngày đăng kí”: label  + “Số dư”: label  + “Trạng thái”: label  + “Add credit” command.  [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Passenger hs no card on the system | System shows a message "Bạn chưa có thẻ nào trên hệ thống." |   **Exceptions:** N/A  **Relationships:** Add credit   * Extension point: Add credit * Condition: Passenger selects a NFC card in card list and sends “Add credit” command.   **Business Rules:**   * Card status is one of the following status: “Đã kích hoạt”, “Tạm khóa”, “Chưa kích hoạt” * Cards with status “Đã kích hoạt” or “Tạm khóa” will be showed to Passenger. | | | |

Table 17: < Passenger> Get NFC card

#### < Passenger> Add credit

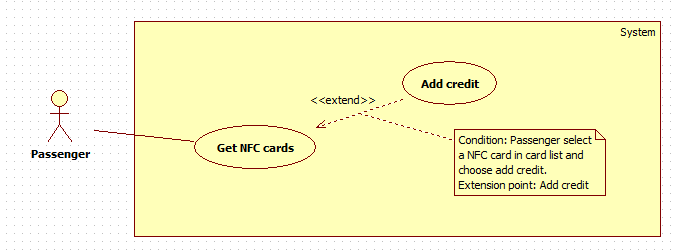


Figure 12: < Passenger> Add credit

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS06** | | | |
| **Use Case No.** | GBTS06 | **Use Case Version** | 1.0 |
| **Use Case Name** | Add credit | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to add more credit to their NFC card.   **Goal:**   * Passenger can add more credit to their specific NFC card for using this card to buy ticket when traveling by bus.   **Triggers:**   * Passenger sends “Add credit” command of one card on their NFC card list.   **Preconditions:**   * Actor logged in system with role “Passenger”. * Passenger has at least one NFC card in the system. * The NFC card which passenger wants to add credit must have status “Đã Kích Hoạt”   **Post Conditions:**   * **Success:** Credits is added to corresponding card balance. * **Fail:** Show an error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Add credit” command of one card on their NFC card list. | System shows “Add credit” view with following information:   * “Gói nạp tiền”: select one of the options such as “50.000đ”, “100.000đ”, etc. based on Credit plans created by Manager. * “Checkout” command. * “Cancel” command. | | 2 | Passenger selects one option.  [Alternative 1] |  | | 3 | Passenger sends “Checkout” command. | System redirects to Paypal payment gateway. | | 4 | Pssenger processes the PayPal payment. | If the payment is succeed: Credits will be added to corresponding card balance. System redirects to “Card list” view and display successful message: “Thanh toán thành công!”  [Exception 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends cancel command | System returns to “Card list” view. |   **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Payment failed | System returns to “Card list” view and displays error message: “Thanh toán không thành công!” |   **Relationships:** Get NFC cards   * Extension point: Add credit * Condition: Passenger selects a NFC card in card list and sends “Add credit” command.   **Business Rules:**   * Credit in the system mean the card’s balance which will be used to buy ticket. * After payment successfully, credits will be added to corresponding card balance and payment transaction will be saved to database for tracking by Bus Manager. * Credit plans and its price are loaded from system. These plans are created and managed by Bus Manager. | | | |

Table 18: < Passenger> Add credit

#### < Passenger> Edit card name



Figure 13: < Passenger> Edit card name

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS07** | | | |
| **Use Case No.** | GBTS07 | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit card name | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger edit their NFC card name.   **Goal:**   * Passenger can update new name for their card which make them easy to identify each card in their card list.   **Triggers:**   * Passenger changing name of a card in card list.   **Preconditions:**   * Actor logged in system with role “Passenger”. * Passenger has at least one NFC card in the system.   **Post Conditions:**   * **Success:** New card name is updated. * **Fail:** Show an error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger changing name of a card in card list. | System waits for user inputting. | | 2 | Passenger stops inputting. | System updates new name for corresponding card.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Passenger inputs card name with wrong format. | System displays error message: “Vui lòng nhập tên thẻ từ 3-50 ký tự ”. |   **Relationships:** Get NFC cards   * Extension point: Edit card name * Condition: Passenger edit card name of a card on card list.   **Business Rules:**   * New default card name when a new card is added to the system is “Thẻ {Card id}” * Card name will be used in outcome report for representing which card paid for ticket. | | | |

Table 19: < Passenger> Edit card name

#### < Passenger> Get outcome report

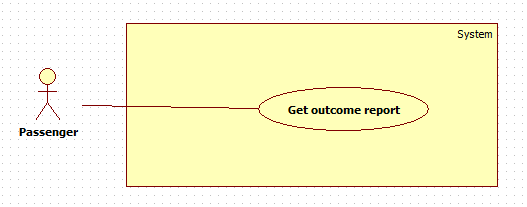


Figure 14: < Passenger> Get outcome report

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS08** | | | |
| **Use Case No.** | GBTS08 | **Use Case Version** | 1.0 |
| **Use Case Name** | Get outcome report | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger get their outcome report.   **Goal:**   * Passenger can view how much they spend for bus in a specific date range and also the details list of bought tickets.   **Triggers:**   * Passenger sends “Get outcome report” command.   **Preconditions:**   * Actor logged in system with role “Passenger”.   **Post Conditions:**   * **Success:** Outcome report is showed. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Get outcome report” command. | System shows “Get outcome report” view with following information:   * “Từ ngày”: date input, default date is 1st of current month, required. * “Đến ngày”: date input, default date is the current date, required. * “Get report” command. | | 2 | Passenger inputs date for both two field. |  | | 3 | Passenger sends “Get report” command. | System verifies input information. Get outcome report of current passenger based on input date range. Display report as a table, each row has following information:   * “Tên thẻ”: label * “Số vé đã mua”: label * “Tổng tiền”: label * “Tuyến thường đi”: label   [Exception 1,2,3,4]  [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | If there is no data to show. | System shows a message “Không có dữ liệu trong khoảng thời gian đã chọn”. |   **Exceptions:**  [Exception 1,2,3,4]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Passenger does not input required fields. | System notices that user need to input all required field.   * “Từ ngày”: System displays warning message “Vui lòng chọn ngày bắt đầu”. * “Đến ngày”: System displays warning message “Vui lòng chọn ngày kết thúc”. | | 2 | Passenger inputs start date greater than end date. | System displays error message: “Ngày bắt đầu phải nhỏ hơn ngày kết thúc.” | | 3 | Passenger inputs date range over 30 days. | System displays error message: “Khoảng thời gian không vượt quá 30 ngày.” | | 4 | Passenger inputs end date greater than current date. | System displays error message: “Ngày bắt đầu phải nhỏ hơn ngày kết thúc.” |   **Relationships:** N/A  **Business Rules:**   * Start date and end date format are “dd/mm/yyyy”. * End date must not be earlier than start date. * End date must not greater than current date. * Date range for getting report must be within 30 days. * System allows Passenger to search and sort report on all fields. | | | |

Table 20: < Passenger> Get outcome report

#### < Passenger> Find bus

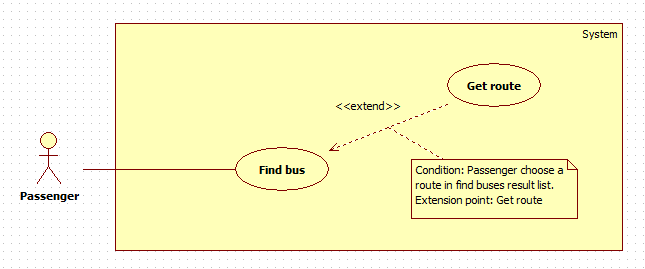


Figure 15: < Passenger> Find bus

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS09** | | | |
| **Use Case No.** | GBTS09 | **Use Case Version** | 1.0 |
| **Use Case Name** | Find bus | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to find bus routes.   **Goal:**   * Passenger can find bus routes based on starting point and destination for getting instructions to get to the destination.   **Triggers:**   * Passenger sends “Find bus” command.   **Preconditions:**   * Actor logged in system with role “Passenger”.   **Post Conditions:**   * **Success:** Bus routes result list is showed. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Find bus” command. | System shows “Find bus” view with following information:   * “Điểm khởi hành”: free text input. * “Điểm đến”: free text input. * “Find” command. | | 2 | Passenger inputs information. |  | | 3 | Passenger sends “Find” command. | System verifies input information then finds bus routes. System displays routes list, each route has following information:   * “Số xe”: label * “Thời gian ước tính”: label   [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | No route found. | If there is no route as the result, system shows a message "Không tìm thấy kết quả nào phù hợp". |   **Exceptions:** N/A  **Relationships:** Get route   * Condition: Passenger chooses a route in route result list. * Extension point: Get route   **Business Rules:**   * If Passenger doesn’t input starting point or destination, system will use the current location automatically. * System finds and display result based on starting point and destination, top 5 results will be displayed to Passenger. | | | |

Table 21: < Passenger> Find bus

#### < Passenger> Get route

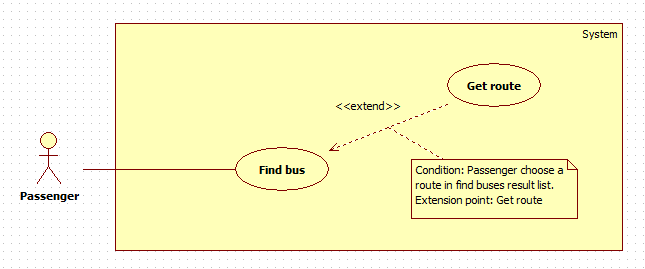


Figure 16: < Passenger> Get route

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS10** | | | |
| **Use Case No.** | GBTS10 | **Use Case Version** | 1.0 |
| **Use Case Name** | Get route | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to get a bus route.   **Goal:**   * Passenger can view a bus route details and instruction for getting bus to get to the destination.   **Triggers:**   * Passenger sends “Get route” command of one route on bus route result list.   **Preconditions:**   * Actor logged in system with role “Passenger”. * The bus route result list has at least one route item.   **Post Conditions:**   * **Success:** Bus route detail is show. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger send “Get route” command of one route on bus route result list.  [Alternative 1] | System gets correspond route details then shows “Route details” view:   * Details instructions * Route preview on map. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends back command | System returns to bus route result list. |   **Exceptions:** N/A  **Relationships:** Find bus   * Condition: Passenger chooses a route in route result list. * Extension point: Get route   **Business Rules:**   * Route instruction will be display step by step to make sure it’s easy to follow. * Route instruction format:   {Estimated time} {Route Number List}  {Action 1}: {Destination1 or Route Number1}  {Action 2}: {Destination2 or Route Number2}  {Action 3}: {Destination3 or Route Number3} | | | |

Table 22: < Passenger> Get route

#### < Passenger> Look up bus route



Figure 16: < Passenger> Get route

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS10** | | | |
| **Use Case No.** | GBTS10 | **Use Case Version** | 1.0 |
| **Use Case Name** | Look up bus route | | |
| **Author** | Doan Minh Duc | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to get a specific bus route’s information.   **Goal:**   * Passenger can view a bus route’s stops by route code.   **Triggers:**   * Passenger sends “Look up bus route” command of one route on bus route result list.   **Preconditions:**   * Actor logged in system with role “Passenger”. * The bus route result list has at least one route item.   **Post Conditions:**   * **Success:** Bus route detail is show. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger send “Look up bus route” command of one route on bus route result list.  [Alternative 1] | System gets correspond route details then shows “Route details” view:   * Stops name * Route preview on map. |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends back command | System returns to bus route result list. |   **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * Bus stops name will be display in a list. | | | |

Table 22: < Passenger> Get route

#### < Passenger> Register subscription



Figure 16: < Passenger> Get route

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS10** | | | |
| **Use Case No.** | GBTS10 | **Use Case Version** | 1.0 |
| **Use Case Name** | Register subscription | | |
| **Author** | Doan Minh Duc | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to register a subscription package.   **Goal:**   * Registered subscription passengers can buy ticket with cheaper price.   **Triggers:**   * Passenger sends “Register subscription” command.   **Preconditions:**   * Actor logged in system with role “Passenger”.   **Post Conditions:**   * **Success:** System sends an SMS with successful message to passenger. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Register subscription” command by sending an SMS with their NFC card id and subscription name to the system. | System sends an SMS with successful message to passenger.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:** N/A  [Exception 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Passenger sends SMS with wrong format. | System sends an error back to user’s phone number: “Tin nhan sai cu phap, vui long kiem tra lai.” |   **Relationships:** N/A  **Business Rules:**   * Subscription will be automatically extend until passenger sends cancel command. * With subscription, passengers can buy ticket at cheap price with limited amount of tickets. | | | |

Table 22: < Passenger> Get route

#### < Passenger> Cancel subscription



Figure 16: < Passenger> Get route

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS10** | | | |
| **Use Case No.** | GBTS10 | **Use Case Version** | 1.0 |
| **Use Case Name** | Cancel subscription | | |
| **Author** | Doan Minh Duc | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Passenger   **Summary:**   * This use case allows Passenger to cancel a registerd subscription package.   **Goal:**   * Passenger cancels a specific subscription package.   **Triggers:**   * Passenger sends “Cancel subscription” command.   **Preconditions:**   * Actor logged in system with role “Passenger”.   **Post Conditions:**   * **Success:** System sends an SMS with successful message to passenger. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Passenger sends “Cancel subscription” command by sending an SMS with their NFC card id and subscription name to the system. | System sends an SMS with successful message to passenger.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:** N/A  [Exception 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Passenger sends SMS with wrong format. | System sends an error back to user’s phone number: “Tin nhan sai cu phap, vui long kiem tra lai.” |   **Relationships:** N/A  **Business Rules:**   * After cancelling a subscription, passengers buy ticket at listed price. | | | |

Table 22: < Passenger> Get route

#### Staff

#### < Staff > Overview use case



Figure 17: < Staff > Overview use case

#### < Staff > Add card

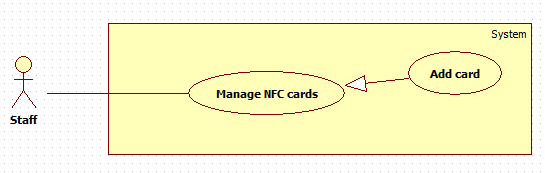


Figure 18: < Staff > Add card

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS11** | | | |
| **Use Case No.** | GBTS11 | **Use Case Version** | 1.0 |
| **Use Case Name** | Add card | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Staff.   **Summary:**   * This use case allows Staff to add a new NFC card to the system.   **Goal:**   * Staff can add a new NFC card to the system to make this card available to use or activate by Passenger.   **Triggers:**   * Staff sends “Add card” command.   **Preconditions:**   * Actor logged in the system with role “Staff”.   **Post Conditions:**   * **Success:** New NFC card added to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends “Add card” command. | System displays “Add card” view with following information:   * “Mã thẻ”: free text input, length 6 – 250, required. * “Số dư”: number input, positive integer number from 0 to 10000000. * “Save” command. * “Cancel” command. | | 2 | Staff taps NFC card on the NFC reader then inputs information.  [Alternative 1] |  | | 3 | Staff sends “Save” command. | System verifies input information, adds new card to the system then displays success message: “Thêm thẻ thành công”.  [Exception 1,2,3] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends cancel command. | System return to main view. |   **Exceptions:**  [Exception 1,2,3]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Staff does not input required fields. | System notices that user need to input all required field.   * “Mã thẻ”: System displays warning message “Vui lòng nhập mã thẻ”. | | 2 | Staff inputs some fields with wrong format. | System notices that user need to input all required field.   * “Mã thẻ”: System displays warning message “Vui lòng nhập mã thẻ từ 6-250 ký tự”. * “Số dư”: System displays warning message “Vui lòng nhập số dư từ 0 – 10.000.000”. | | 3 | Card ID is already existed in the system. | System displays warning message “Mã thẻ đã tồn tại trên hệ thống”. |   **Relationships: N/A**  **Business Rules:**   * New card will be added to the system with input information with status “Chưa kích hoạt”. * Tap card on NFC reader to get card’s ID. * If no card balance is input. New card balance is 15000. * New card is not belonged to any passenger but is still valid for using to buy ticket, but not allow adding more credit. * Bus company publishes many card for selling to temporary Passenger or in case of forgetting card. | | | |

Table 23: < Staff > Add card

#### < Staff > Activate/Deactivate card

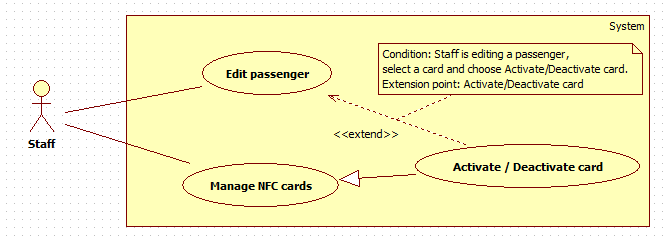


Figure 19: < Staff > Activate/Deactivate card

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS12** | | | |
| **Use Case No.** | GBTS12 | **Use Case Version** | 1.0 |
| **Use Case Name** | Activate / Deactivate card | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Staff.   **Summary:**   * This use case allows Staff to activate / deactivate NFC card in the system.   **Goal:**   * Staff can activate NFC card which will allow using this card to buy ticket. * Staff can deactivate NFC card which will lock this card from buying ticket.   **Triggers:**   * Staff sends “Activate/Deactivate” command of a card on card list.   **Preconditions:**   * Actor logged in the system with role “Staff”.   **Post Conditions:**   * **Success:** Target card status is changed. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends “Activate/Deactivate” command of a card on card list | If selected card is currently new and not activated yet. System will do nothing and display message “Thẻ này chưa kích hoạt!”.  If selected card is currently activated. System will deactivate corresponding card and display message “Đã khóa thẻ”.  If selected card is currently deactivated. System will activate corresponding card and display message “Đã mở khóa thẻ”. |   **Alternative Scenario:** N/A  **Exceptions:** N/A  **Relationships:** Activate/Deactivate card   * Condition: Staff is editing a passenger, select a card and choose Activate/Deactivate card. * Extension point: Activate/Deactivate card.   **Business Rules:**   * System activate NFC card by changing card status to “Đã kích hoạt” which will allow using this card to buy ticket. * System deactivate NFC card by changing card status to “Tạm khóa” which will lock this card from using to buy ticket. * System cannot activate/deactivate new card with status “Chưa kích hoạt”. | | | |

Table 24: < Staff > Activate/Deactivate card

#### < Staff > Search passenger

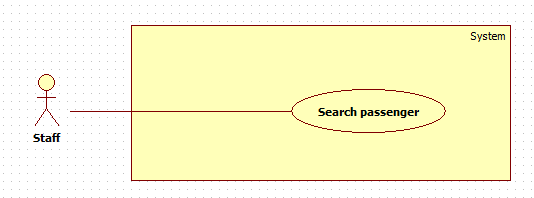


Figure 20: < Staff > Search passenger

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS13** | | | |
| **Use Case No.** | GBTS13 | **Use Case Version** | 1.0 |
| **Use Case Name** | Search passenger | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Staff.   **Summary:**   * This use case allows Staff to search a passenger.   **Goal:**   * Staff can get passenger list match input phone number and view the information of a specific passenger.   **Triggers:**   * Staff changes content of the search text input.   **Preconditions:**   * Actor logged in the system with role “Staff”. * There is at least one passenger in the system with status “True”.   **Post Conditions:**   * **Success:** Passenger list match searching criteria is showed. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff inputs phone number to search text input field. | System loads the search results each time the search query changed. Information for each result:   * “Số điện thoại”: label * “Họ và tên: label * “Edit passenger” command.   [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | There is no passenger match searching criteria. | System shows message: “Không tìm thấy khách hàng nào.” |   **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * Each passenger has a unique phone number. * Any passengers with status “False” will be ignore from searching process and will not be display in search result. | | | |

Table 25: < Staff > Search passenger

#### < Staff > Edit passenger

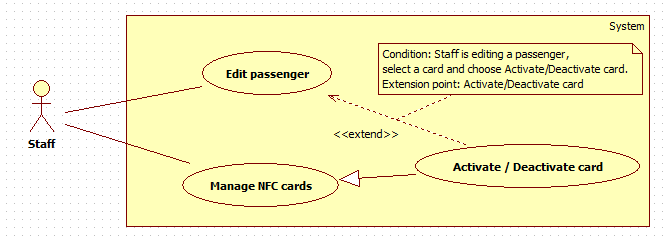


Figure 21: < Staff > Edit passenger

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS14** | | | |
| **Use Case No.** | GBTS14 | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit passenger | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Staff.   **Summary:**   * This use case allows Staff to edit a passenger.   **Goal:**   * Staff can update a passenger details and activate / deactivate their card in the system.   **Triggers:**   * Staff sends “Edit passenger” command of a passenger on passenger list.   **Preconditions:**   * Actor logged in the system with role “Staff”.   **Post Conditions:**   * **Success:** Passenger’s detail is updated to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends “Edit passenger” command of a passenger on passenger list. | System displays “Edit passenger” view with following information:   * “Số điện thoại”: label, read-only. * “Họ tên”: free text input, length 3 – 80, required. * NFC card list, each card has following information:   + “Mã thẻ”: label   + “Số dư”: label   + “Trạng thái”: select one of the options such as “Đã kích hoạt”, “Tạm khóa”.   + “Activate/Deactivate” command * “Save” command. * “Cancel” command. | | 2 | Staff edits information.  [Alternative 1,2] |  | | 3 | Staff sends “Save” command. | System verifies input information. System updates passenger details to the system then displays success message: “Cập nhật thông tin khách hàng thành công”.  [Exception 1,2] |   **Alternative Scenario:**  [Alternative 1,2]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends cancel command. | System returns to main view. | | 2 | Staff sends “Activate/Deactivate” command of a NFC card on card list. | If selected card is currently activated. System will deactivate corresponding card and display message “Đã khóa thẻ”.  If selected card is currently deactivated. System will activate corresponding card and display message “Đã mở khóa thẻ”. |   **Exceptions:**  [Exception 1,2]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Staff does not input required fields. | System notices that user need to input all required field.   * “Họ tên”: System displays warning message “Vui lòng nhập họ tên”. | | 2 | Staff inputs some fields with wrong format. | System notices that user need to input again.   * “Họ tên”: System displays warning message “Vui lòng nhập họ tên từ 3-80 ký tự”. |   **Relationships:** Activate/Deactivate card   * Condition: Staff is editing a passenger, select a card and choose Activate/Deactivate card. * Extension point: Activate/Deactivate card   **Business Rules:**   * Passenger details will be updated to the system with input information. Passenger status won’t change. * Each passenger has a unique phone number. The system does not allow Staff to edit phone number. * System activate NFC card by changing card status to “Đã kích hoạt”. * System deactivate NFC card by changing card status to “Tạm khóa”. | | | |

Table 26: < Staff > Edit passenger

#### < Staff > Publish scratch cards



Figure 21: < Staff > Edit passenger

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS14** | | | |
| **Use Case No.** | GBTS14 | **Use Case Version** | 1.0 |
| **Use Case Name** | Publish scratch cards | | |
| **Author** | Doan Minh Duc | | |
| **Date** | 1/11/2016 | **Priority** | Normal |
| **Actor:**   * Staff.   **Summary:**   * This use case allows Staff to publish scratch cards.   **Goal:**   * Staff can publish scratch cards for sale. Passenger can buy scratch card to top up card   **Triggers:**   * Staff sends “Publish scratch cards” command of a passenger on passenger list.   **Preconditions:**   * Actor logged in the system with role “Staff”.   **Post Conditions:**   * **Success:** Newscratch cards are generated in the database. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends “Publish scratch cards” command. | System displays “Publish scratch cards” view with following information:   * “Số lượng”: number input, required. * “Mệnh giá”: select at least one of price options. * “Publish” command. * “Cancel” command. | | 2 | Staff input information.  [Alternative 1,2] |  | | 3 | Staff sends “Save” command. | System generates the scratch cards  [Exception 1] |   **Alternative Scenario:**  [Alternative 1,2]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Staff sends cancel command. | System returns to main view. |   **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Staff does not input required fields. | System notices user need to input all required field. |   **Relationships:** N/A  **Business Rules:**   * Each scratch card must have the expired date. * Each scratch card can only be used once. * Scratch card’s code is case sensitive. | | | |

Table 26: < Staff > Edit passenger

#### Manager

#### < Manager > Overview use case



Figure 22: < Manager > Overview use case

#### < Manager > Add ticket type

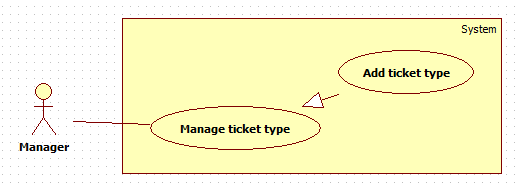


Figure 23: < Manager > Add ticket type

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS15** | | | |
| **Use Case No.** | GBTS15 | **Use Case Version** | 1.0 |
| **Use Case Name** | Add ticket type | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to add a ticket type.   **Goal:**   * Manager can create and add a ticket type to the system with specific ticket price that is suitable for correspond route distance.   **Triggers:**   * Manager sends “Add ticket type” command.   **Preconditions:**   * Actor logged in the system with role “Manager”.   **Post Conditions:**   * **Success:** New ticket type is added to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Add ticket type” command. | System displays “Add ticket type” view with following information:   * “Loại vé”: free text input, length 6-250, required. * “Mô tả”: free text area, required. * “Giá vé”: number input, positive integer number from 500 to 100000, required. * “Save” command. * “Cancel” command. | | 2 | Manager inputs information.  [Alternative 1] |  | | 3 | Manager sends “Save” command. | System verifies input information, adds new ticket type to the system then displays success message: “Thêm loại vé thành công”.  [Exception 1,2,3] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System return to main view. |   **Exceptions:**  [Exception 1,2,3]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Tên loại vé”: System displays warning message “Vui lòng nhập tên loại vé”. * “Mô tả”: System displays warning message “Vui lòng nhập mô tả”. * “Giá vé”: System displays warning message “Vui lòng nhập giá vé”. | | 2 | Manager inputs some fields with wrong format. | System notices that user need to input again.   * “Tên loại vé”: System displays warning message “Vui lòng nhập tên loại vé từ 6-250 ký tự”. * “Giá vé”: System displays warning message “Vui lòng nhập giá vé từ 500-100000”. | | 3 | Ticket type name is already existed in the system. | System displays warning message “Tên loại vé không được trùng”. |   **Relationships: N/A**  **Business Rules:**   * New ticket type will be added to the system with input information with status “True”. * Ticket type and its price depend on route distance and will be specify by company. * Each emulator on bus will be configured to a corresponding ticket type. * Passenger pays ticket price according to the ticket type of the bus they use. * Ticket type name is unique to make sure no ticket type is duplicated. | | | |

Table 27: < Manager > Add ticket type

#### < Manager > Edit ticket type

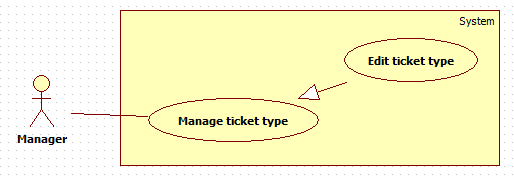


Figure 24: < Manager > Edit ticket type

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS16** | | | |
| **Use Case No.** | GBTS16 | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit ticket type | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to edit a ticket type in the system.   **Goal:**   * Manager can update a ticket type details in the system for adapting the ticket price change.   **Triggers:**   * Manager sends “Edit ticket type” command of one ticket type on ticket type list.   **Preconditions:**   * Actor logged in the system with role “Manager”. * There is at least one ticket type in the system.   **Post Conditions:**   * **Success:** Ticket type detail is updated to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Edit ticket type” command of one ticket type on ticket type list. | System displays “Edit ticket type” view and loads bus detail to a form with following information:   * “Loại vé”: free text input, length 6-250, required. * “Mô tả”: free text area, required. * “Giá vé”: number input, positive integer number from 500 to 100000, required. * “Save” command. * “Cancel” command. | | 2 | Manager edits information.  [Alternative 1] |  | | 3 | Manager sends “Save” command. | System verifies input information. Update ticket type detail to the system then displays success message: “Sửa thông tin loại vé thành công”.  [Exception 1,2,3] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System return to main view. |   **Exceptions:**  [Exception 1,2,3]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Tên loại vé”: System displays warning message “Vui lòng nhập tên loại vé”. * “Mô tả”: System displays warning message “Vui lòng nhập mô tả”. * “Giá vé”: System displays warning message “Vui lòng nhập giá vé”. | | 2 | Manager inputs some fields with wrong format. | System notices that user need to input again.   * “Tên loại vé”: System displays warning message “Vui lòng nhập tên loại vé từ 6-250 ký tự”. * “Giá vé”: System displays warning message “Vui lòng nhập giá vé từ 500-100000”. | | 3 | Ticket type name is already existed in the system. | System displays warning message “Tên loại vé không được trùng”. |   **Relationships: N/A**  **Business Rules:**   * Ticket type detail will be updated to the system with input information. * Ticket type and its price depend on route distance and will be specify by company. * Each emulator on bus will be configured to a corresponding ticket type. * Passenger pays ticket price according to the ticket type of the bus they use. * Ticket type name is unique to make sure no ticket type is duplicated. | | | |

Table 28: < Manager > Edit ticket type

#### < Manager > Delete ticket type

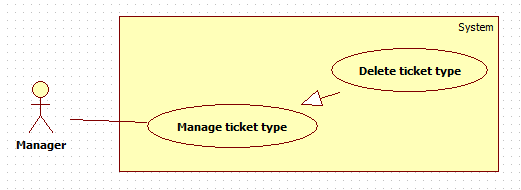


Figure 25: < Manager > Delete ticket type

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS17** | | | |
| **Use Case No.** | GBTS17 | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete ticket type | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to delete a ticket type from the system.   **Goal:**   * Manager can delete a ticket type from the system.   **Triggers:**   * Manager sends “Delete ticket type” command of one ticket type on ticket type list.   **Preconditions:**   * Actor logged in the system with role “Manager”. * There is at least one ticket type in the system. * The target ticket type has status “True”   **Post Conditions:**   * **Success:** Ticket type is deleted from the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Delete ticket type” command of one ticket type on ticket type list. | System displays confirmation message: “Bạn có chắc chắn muốn xóa?” | | 2 | Manager sends “Yes” command.  [Alternative 1] | System deletes target ticket type from the system and displays success message: “Xóa thành công”; |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System return to main view. |   **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * The target ticket type must have status “True”. * If target ticket type has not been used yet. System delete this ticket type completely, otherwise system set its status to “False”. | | | |

Table 29: < Manager > Delete ticket type

#### < Manager > Add credit plan

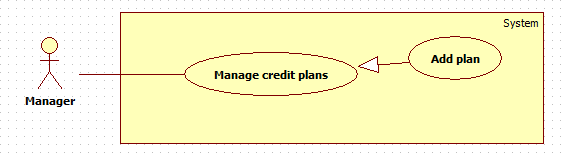


Figure 26: < Manager > Add credit plan

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS18** | | | |
| **Use Case No.** | GBTS18 | **Use Case Version** | 1.0 |
| **Use Case Name** | Add plan | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to add a new credit plan.   **Goal:**   * Manager can create and add a new credit plan to the system. Passenger can add credit to their card by buying these credit plans.   **Triggers:**   * Manager sends “Add plan” command.   **Preconditions:**   * Actor logged in the system with role “Manager”.   **Post Conditions:**   * **Success:** New credit plan is added to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Add plan” command. | System displays “Add plan” view with following information:   * “Tên gói”: free text input, length 3 – 250, required. * “Mô tả”: free text area, optional. * “Giá gói”: number input, positive integer number from 50000 to 10000000, required. * “Save” command. * “Cancel” command. | | 2 | Manager inputs information.  [Alternative 1] |  | | 3 | Manager sends “Save” command. | System verifies input information, adds new plan to the system then displays success message: “Thêm gói thành công”.  [Exception 1,2,3] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System return to main view. |   **Exceptions:**  [Exception 1,2,3]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Tên gói”: System displays warning message “Vui lòng nhập tên gói”. * “Giá gói”: System displays warning message “Vui lòng nhập giá gói”. | | 2 | Manager inputs some fields with wrong format | System notices that user need to input all required field.   * “Tên gói”: System displays warning message “Vui lòng nhập tên gói từ 3-250 ký tự”. * “Giá gói”: System displays warning message “Vui lòng nhập giá gói từ 50.000-10.000.000”. | | 3 | Credit plan name is already existed in the system. | System displays warning message “Tên gói không được trùng, vui lòng chọn tên gói khác.”. |   **Relationships: N/A**  **Business Rules:**   * New credit plan will be added to the system with input information with status “True” and will available to buy when Passenger want to add credit to their card. * Credit plan name is unique to make sure no plan is duplicated. | | | |

Table 30: < Manager > Add credit plan

#### < Manager > Edit credit plan

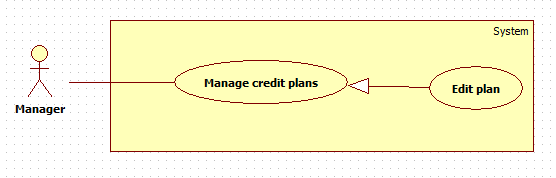


Figure 27: < Manager > Edit credit plan

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS19** | | | |
| **Use Case No.** | GBTS19 | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit plan | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to edit a credit plan.   **Goal:**   * Manager can update a credit plan details in the system for adapting price change.   **Triggers:**   * Manager sends “Edit plan” command of one plan on plan list.   **Preconditions:**   * Actor logged in the system with role “Manager”. * There is at least one credit plan in the system with status “True”   **Post Conditions:**   * **Success:** Credit plan details updated to the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Edit plan” command of one plan on plan list. | System displays “Edit plan” view and loads plan detail to a form with following information:   * “Tên gói”: free text input, length 3 – 250, required. * “Mô tả”: free text area, optional. * “Giá gói”: number input, positive integer number from 50000 to 10000000, required. * “Save” command. * “Cancel” command. | | 2 | Manager edits information.  [Alternative ] |  | | 3 | Manager sends “Save” command. | System verifies input information, updates credit plan details then displays success message: “Sửa thông tin gói thành công”.  [Exception 1,2,3] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System returns to main view. |   **Exceptions:**  [Exception 1,2,3]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Tên gói”: System displays warning message “Vui lòng nhập tên gói”. * “Giá gói”: System displays warning message “Vui lòng nhập giá gói”. | | 2 | Manager inputs some fields with wrong format | System notices that user need to input all required field.   * “Tên gói”: System displays warning message “Vui lòng nhập tên gói từ 3-250 ký tự”. * “Giá gói”: System displays warning message “Vui lòng nhập giá gói từ 50.000-10.000.000”. | | 3 | Credit plan name is already existed in the system. | System displays warning message “Tên gói không được trùng, vui lòng chọn tên gói khác.”. |   **Relationships: N/A**  **Business Rules:**   * New credit plan detail will be updated to the system with input information. No change on plan status. * Credit plan name is unique to make sure no plan is duplicated. | | | |

Table 31: < Manager > Edit credit plan

#### < Manager > Delete credit plan

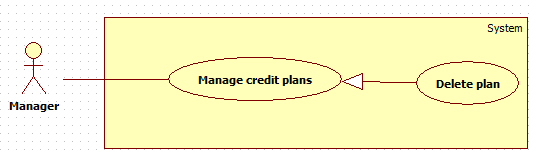


Figure 28: < Manager > Delete credit plan

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS20** | | | |
| **Use Case No.** | GBTS20 | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete plan | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager.   **Summary:**   * This use case allows Manager to delete a credit plan.   **Goal:**   * Manager can delete a credit plan from the system to make it unavailable to buy in any situation.   **Triggers:**   * Manager sends “Delete plan” command of one plan on plan list.   **Preconditions:**   * Actor logged in the system with role “Manager”. * There is at least one credit plan in the system with status “True”   **Post Conditions:**   * **Success:** Credit plan is deleted from the system. * **Fail:** System displays error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Delete plan” command of one plan on plan list. | System displays confirmation message: “Bạn có chắc chắn muốn xóa?” | | 2 | Manager sends “Yes” command.  [Alternative 1] | System deletes target plan from the system and displays success message: “Xóa thành công”; |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends cancel command. | System return to main view. |   **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * Plans which have status “False” will be unavailable for being bought by Passenger. * If a plan has not been related to any ticket, system delete this plan completely, otherwise, system sets its status to “False”. | | | |

Table 32: < Manager > Delete credit plan

#### < Manager > Get income report

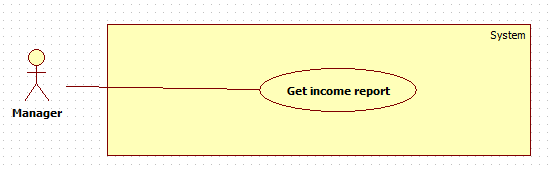


Figure 29: < Manager > Get income report

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS21** | | | |
| **Use Case No.** | GBTS21 | **Use Case Version** | 1.0 |
| **Use Case Name** | Get income report | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager   **Summary:**   * This use case allows Manager can get outcome report.   **Goal:**   * Manager can view total income from selling ticket in a specific date range and also the details list of bought tickets.   **Triggers:**   * Manager sends “Get income report” command.   **Preconditions:**   * Actor logged in system with role “Manager”.   **Post Conditions:**   * **Success:** Income report is showed. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Get income report” command. | System shows “Get income report” view with following information:   * “Từ ngày”: date input, default date is 1st of current month, required. * “Đến ngày”: date input, default date is the current date, required. * “Get report” command. | | 2 | Manager inputs date for both two field. |  | | 3 | Manager sends “Get report” command. | System verifies input information, gets income report based on input date range. System displays report as a table, each row has following information:   * “Ngày giờ”: label * “Thẻ thanh toán”: label * “Khách hàng”: label * “Số tiền”: label   [Exception 1,2,3,4]  [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | Syste Response | | 1 | If there is no data to show. | System shows a message “Không có dữ liệu trong khoảng thời gian đã chọn”. |   **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Từ ngày”: System displays warning message “Vui lòng chọn ngày bắt đầu”. * “Đến ngày”: System displays warning message “Vui lòng chọn ngày kết thúc”. | | 2 | Manager inputs start date greater than end date. | System displays error message: “Ngày bắt đầu phải nhỏ hơn ngày kết thúc.” | | 3 | Manager inputs date range over 30 days. | System displays error message: “Khoảng thời gian không vượt quá 30 ngày.” | | 4 | Manager inputs end date greater than current date. | System displays error message: “Ngày bắt đầu phải nhỏ hơn ngày kết thúc.” |   **Relationships:** N/A  **Business Rules:**   * Start date and end date format are “dd/mm/yyyy”. * End date must not be earlier than start date. * Total income would be calculated from all bought tickets match input date range. * End date must not greater than current date. * Date range for getting report must be within 30 days. * System allows Manager to search and sort report on all fields. | | | |

Table 33: < Manager > Get income report

#### < Manager > Create promotion

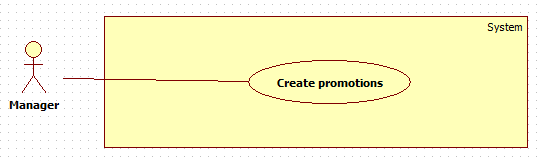


Figure 30: < Manager > Create promotion

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS22** | | | |
| **Use Case No.** | GBTS22 | **Use Case Version** | 1.0 |
| **Use Case Name** | Create promotions | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Manager   **Summary:**   * This use case allows Manager to create promotion and offer for suggesting Passenger.   **Goal:**   * Manager can create promotion in the system. * Passenger will receive latest promotion from the system when the Auto Handler sends it to them.   **Triggers:**   * Manager sends “Create promotions” command.   **Preconditions:**   * Actor logged in system with role “Manager”.   **Post Conditions:**   * **Success:** New promotion is added to the system. * **Fail:** Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Manager sends “Create promotion” command. | System shows “Create promotion ” view with following information:   * “Tên quảng cáo”: free text input, length 6-250, required. * “Nội dung”: html input area, required. * “Save” command. | | 2 | Manager inputs information. |  | | 3 | Manager sends “Save” command. | System verifies input information. Create new promotion with input information and add to system.  [Exception 1,2] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1,2]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Manager does not input required fields. | System notices that user need to input all required field.   * “Tên quảng cáo”: System displays warning message “Vui lòng nhập quảng cáo”. * “Nội dung”: System displays warning message “Vui lòng nhập nội dung”. | | 2 | Manager inputs some fields with wrong format | System notices that user need to input all required field.   * “Tên quảng cáo”: System displays warning message “Vui lòng nhập tên quảng cáo từ 6-250 ký tự”. |   **Relationships:** N/A  **Business Rules:**   * New promotion will be created with input information and its status is “True”. * After Auto Handler sends promotion successfully, the status will be change to “False”. | | | |

Table 34: < Manager > Create promotion

#### Admin

#### < Admin> Overview use case



Figure 31: < Admin> Overview use case

#### < Admin> Search account



Figure 32: < Admin> Search account

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS23** | | | |
| **Use Case No.** | GBTS23 | **Use Case Version** | 1.1 |
| **Use Case Name** | Search account | | |
| **Author** | Doan Minh Duc | | |
| **Date** | September 13, 2016 | **Priority** | High |
| **Actor:**   * Administrator.   **Summary:**   * This use case allows admin to search for accounts.   **Goal:**   * Admin can quickly navigate and view the information of accounts by phone number.   **Triggers:**   * Admin sends Search command.   **Preconditions:**   * Actor logged in the system with role “Administrator”.   **Post Conditions:**   * **Success:** System shows the result. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin inputs the phone number then sends Search command. | System displays matched account in a table with these following information for each account:  +“Số điện thoại”: label  +“Vai trò”: label  +”Họ tên”: label  +”Trạng thái”: label  [Alternative 1] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Admin input a none-existed phone number | System shows no results matched. |   **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * Each account has a unique phone number. * System display top 10 account on each page. | | | |

Table 35: < Admin> Search account

#### < Admin> Add account

Figure 33: < Admin> Add account

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS24** | | | |
| **Use Case No.** | GBTS24 | **Use Case Version** | 1.1 |
| **Use Case Name** | Add account | | |
| **Author** | Doan Minh Duc | | |
| **Date** | September 13, 2016 | **Priority** | High |
| **Actor:**   * Administrator.   **Summary:**   * This use case allows admin to create an account in the system.   **Goal:**   * Admin can create new account with specific role such as Admin, Staff, Manager or Passenger.   **Triggers:**   * Admin sends creating account command.   **Preconditions:**   * User logged in the system with the role “Administrator”.   **Post Conditions:**   * **Success:** New account is inserted into the system. * **Fail:** System displays appropriate error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin sends creating account command. | System list information of account in system and components:   * “Số điện thoại”: number input, required, length 10-15 * “Vai trò”: select one of the options such as “Quản trị”, “Quản lý”, “Hành khách”, “Nhân viên” ,required. * “Họ tên”: free text input, length 3-80. * Register command. * Cancel command. | | 2 | Admin fills in information for each account.  [Alternative 1, 2] | System validates input information. | | 3 | Admin sends Register command. | System displays a successful message: “Tạo tài khoản thành công”  [Exception 2] |   **Alternative Scenario:**  [Alternative 1]   |  |  |  | | --- | --- | --- | | 1 | Admin clicks on Cancel button to cancel the register process. | System redirects to admin’s main page. |   [Alternative 2]   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin inputs invalid data. | System displays an error message and asks admin to fill the form again. |   **Exception:**  [Exception 2]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Admin inputs an existed phone number. | System displays an error message “Số điện thoại đã tồn tại trong hệ thống”. |   **Relationships: N/A**  **Business Rules:**   * Phone number is unique on each account. * “Vai trò” is the account’s role with 4 options, “Admin”, “Quản lý”,”Nhân viên”, “Hành khách”. “Admin” will act as an Administrator role in the system. “Quản lý” will act as Manager in the system. “Nhân viên” will act as a Staff role in the system. “Hành khách” will act as a Passenger role in the system. * New account will be created with the status “Đang hoạt động”. * The auto-generated password will be sent to registered phone number. | | | |

Table 36: < Admin> Add account

#### < Admin> Activate/Deactivate account



Figure 34: < Admin> Edit account

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS25** | | | |
| **Use Case No.** | GBTS25 | **Use Case Version** | 1.2 |
| **Use Case Name** | Activate/Deactivate account | | |
| **Author** | Doan Minh Duc | | |
| **Date** | September 13, 2016 | **Priority** | High |
| **Actor:**   * Administrator.   **Summary:**   * This use case allows admin to activate or deactivate an account.   **Goal:**   * The status of an account is updated in the system.   **Triggers:**   * Admin sends Activate/Deactivate account command.   **Preconditions:**   * User logged in the system with the role “Administrator”. * The account must be available in the database.   **Post Conditions:**   * **Success:** Anaccount’s status is updated in the database. * **Fail:** System displays appropriate error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin sends “Activate/Deactivate account” command on a specific account | System activates/deactivates the account |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * An account’s status can be “Đang hoạt động” or “Đã khóa”. “Đang hoạt động” means the account is activated, it can be used to access the system. “Đã khóa” means the account is disabled, it cannot be used to access the system. | | | |

Table 37: < Admin> Edit account

#### Auto Handler

#### < Auto Handler > Overview use case



Figure 36: < Auto Handler > Overview use case

#### < Auto Handler > Parse bus route

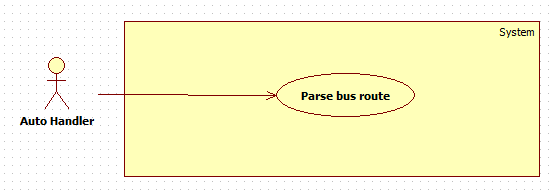


Figure 37: < Auto Handler > Parse bus route

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS27** | | | |
| **Use Case No.** | GBTS27 | **Use Case Version** | 1.0 |
| **Use Case Name** | Parse bus route | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Auto Handler   **Summary:**   * This use case allows Auto Handler to get newest bus route list.   **Goal:**   * Auto Handler can get newest bus route list for adapting changes of bus route in real life.   **Triggers:**   * The current time hit configured time.   **Preconditions:**   * Configured time has been set.   **Post Conditions:**   * **Success:** Newest bus route list is updated to the system. * **Fail:** Write error log.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Auto Handler checks the current time. If the current time hit configured time. Auto Handler sends “Parse bus route” command. | System parses and get newest route list and updates to database.  Write to log file.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Auto Handler is interrupted | No promotion will be sent. Error details will be tracked in a log file. |   **Relationships:** N/A  **Business Rules:**   * System will parse route list from <http://www.buyttphcm.com.vn/TTLT.aspx> * Sent log file structure:   File name: Promotions.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Promotion Name | Result | |  |  |  |  | |  |  |  |  |  * Error log file structure:   File name: AutoHandler\_Error.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Auto Handler Name | Error Details | |  |  |  |  | |  |  |  |  | | | | |

Table 39: < Auto Handler > Parse bus route

#### < Auto Handler > Suggest promotion

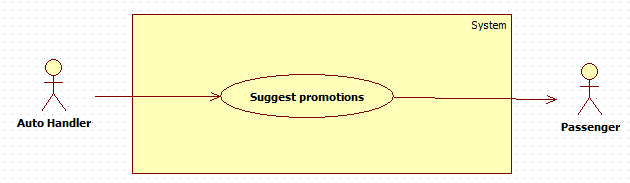


Figure 38: < Auto Handler > Suggest promotion

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS28** | | | |
| **Use Case No.** | GBTS28 | **Use Case Version** | 1.0 |
| **Use Case Name** | Suggest promotions | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Auto Handler   **Summary:**   * This use case allows Auto Handler to get latest promotions then suggest to Passenger.   **Goal:**   * Passenger will receive latest promotions and offers from system such as credit plans sale off, free ticket, etc.   **Triggers:**   * The current time hit configured time.   **Preconditions:**   * Configured time has been set. * There is at least one promotion with status “True”.   **Post Conditions:**   * **Success:** Passenger receives latest promotion and offer. * **Fail:** Write error log.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Auto Handler checks the current time. If the current time hit configured time. Auto Handler send “Suggest promotions” command. | System gets latest promotions and sends to all passengers. Change related promotion and offer status to “False”.  Write to log file.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Auto Handler is interrupted | No promotion will be sent. Error details will be tracked in a log file. |   **Relationships:** N/A  **Business Rules:**   * The promotion which will be sent must have status “True”. * After Auto Handler sends promotion & offer successfully, the status will be change to “False”. * If the current time doesn’t hit configured time or no promotion have status “True”, do nothing. * If the current time hit configured time. Auto Handler get latest promotion which have status “True” then send to passengers. * Promotion will be send from 7pm – 9pm for not harassing passengers. * Sent log file structure:   File name: Promotions.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Promotion Name | Result | |  |  |  |  | |  |  |  |  |  * Error log file structure:   File name: AutoHandler\_Error.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Auto Handler Name | Error Details | |  |  |  |  | |  |  |  |  | | | | |

Table 40: < Auto Handler > Suggest promotion

#### < Auto Handler > Send notification

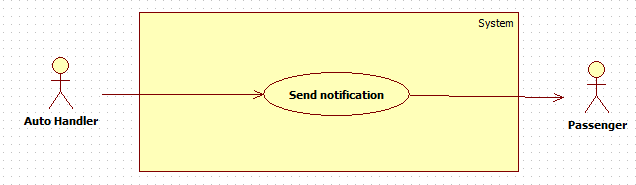


Figure 39: < Auto Handler > Send notification

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS29** | | | |
| **Use Case No.** | GBTS29 | **Use Case Version** | 1.0 |
| **Use Case Name** | Send notification | | |
| **Author** | HoangDN | | |
| **Date** | 13/09/2016 | **Priority** | Normal |
| **Actor:**   * Auto Handler   **Summary:**   * This use case allows Auto Handler to notify Passenger.   **Goal:**   * Passenger will receive notifications from system when their card’s balance is running out.   **Triggers:**   * Auto Handler sends “Send notification” command.   **Preconditions:**   * Passenger has recently bought a ticket. * Passenger’s card balance is running out.   **Post Conditions:**   * **Success:** Passenger receives notification from system. * **Fail:** Write error log.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Auto Handler is waiting for message. If it get any message from system about one card’s balance is running out. Auto Handler sends “Send notification” command. | System sends notification to Passenger: “Thẻ {Card ID} của bạn sắp hết tiền, vui lòng nạp thêm tiền vào thẻ!”  Write to log file.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Auto Handler is interrupted | No notification will be sent. Error details will be tracked in a log file. |   **Relationships:** N/A  **Business Rules:**   * Card’s balance is running out when the balance is less than 5.000đ * System notifies Passenger one time after they bought a ticket if their card’s balance is running out. * Sent log file structure:   File name: Notifications.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Card ID | Result | |  |  |  |  | |  |  |  |  |  * Error log file structure:   File name: AutoHandler\_Error.log   |  |  |  |  | | --- | --- | --- | --- | | No | Time | Auto Handler Name | Error Details | |  |  |  |  | |  |  |  |  | | | | |

Table 41: < Auto Handler > Send notification

#### Mobile Application

#### Emulator

#### < Emulator > Overview use case



Figure 40: < Emulator > Overview use case

#### < Emulator > Write card

Figure 41: < Emulator > Write card

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS30** | | | |
| **Use Case No.** | GBTS30 | **Use Case Version** | 1.1 |
| **Use Case Name** | Write NFC card | | |
| **Author** | Doan Minh Duc | | |
| **Date** | September 13, 2016 | **Priority** | High |
| **Actor:**   * Emulator.   **Summary:**   * This use case allows emulator to insert new card into the system to use the system and to be managed.   **Goal:**   * New UID in inserted into the system, available for using or activating.   **Triggers:**   * Emulator send write card command.   **Preconditions:**   * Valid card.   **Post Conditions:**   * **Success:** Emulator displays a successful message. * **Fail:** Emulator displays an error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Emulator reads the card ID then send write card command | System checks the existence of the UID in the database  Systems insert the card ID into the system then display successful message along with an appropriate sound.  [Exception 1] |   **Alternative Scenario: N/A**  **Exceptions:**  [Exception 1]   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Card ID is already existed in the system | System displays an error message along with an appropriate sound. |   **Relationships: N/A**  **Business Rules:**   * Card ID is unique on each card. * Card ID is inserted into the system with the default status “Chưa kích hoạt”. * Only written card can be used in the system. * Emulator’s screen turns green and plays a single beep sound to indicate a success. * Emulator’s screen turns red and plays a triple beep sound to indicate an error. | | | |

Table 42: < Emulator > Write card

#### < Emulator > Verify ticket

Figure 42: < Emulator > Verify ticket

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – GBTS31** | | | |
| **Use Case No.** | GBTS31 | **Use Case Version** | 1.1 |
| **Use Case Name** | Verify ticket | | |
| **Author** | Doan Minh Duc | | |
| **Date** | September 13, 2016 | **Priority** | High |
| **Actor:**   * Emulator.   **Summary:**   * This use case allows emulator to verify passenger’s card.   **Goal:**   * Verify card by checking the balance then decide to grant access or not.   **Triggers:**   * Passenger taps his/her NFC card on the emulator.   **Preconditions:**   * N/A   **Post Conditions:**   * **Success:** Emulator charges card’s holder for boarding. * **Fail:** Emulator display an error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Emulator reads the ID of the card then send verify ticket command. | System checks the balance mapped with the UID.  +If the balance is equal or more than the cost fare of the bus route, the system will charge the card then display a successful message.  +If the balance is less than the cost fare of the bus route, the system will display an error message. |   **Alternative Scenario:** **N/A**  **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * Card ID is unique on each card. * If the passenger forgets his/her card, he/she can buy a single trip card on the bus or pay in cash. * Emulator’s screen turns green and plays a single beep sound indicated a passenger can pass. * Emulator’s screen turns red and plays a triple beep sound indicated a passenger cannot pass. * Auto Handler pushes a notification to remind passenger to top-up when the money left in the card is less than 5.000 VND. | | | |

Table 43: < Emulator > Verify ticket

## Software System Attribute



### Usability

* The Android application for emulator requires 10 minutes training for staff.
* The Android application for passenger will take 10-30 minutes to get used to the mobile system completely

### Reliability

* The number of sending notification failure is 1 time per 1000 notifications.
* Timer tasks run at configured time with 100% execution rate.

### Availability

* System is available 24 hours per day and 7 days per week.
* System should take at most 5 hours per month for backup or repairing.

### Security

* Privacy: Each role of user has a specific permission to interact with system.
* System requires SMS verification for all users.
* System always checks authorization and authenticated before doing anything.
* Only admin can grant permission to other roles.

### Maintenanability

* The system is divided into separated modules.
* The code is easy to maintain and upgrade.

### Portability

* The web application is running on Windows Server 2008 or above.
* Mobile application for passenger and emulator runs on Android API greater than 4.1.
* Providing easy installation.

### Performance

* Web appication handles the task within 10 seconds.
* System response time of SMS message depend on telecomunications infrastructurs and server.
* Emulator reads data from NFC card in less than 5 second within the distance between 0 centimeter and 1 centimeter in the condition with no obstacle.

## Conceptual Diagram





Figure 43: Conceptual Diagram

**Data Dictionary:**

|  |  |
| --- | --- |
| Entity Name | Description |
| User | Contains all properties about user. |
| Passenger | Contains all properties about passenger info. |
| Admin | Contains all properties about admin info. |
| Staff | Contains all properties about staff info. |
| Manager | Contains all properties about manager info. |
| Card | Contains all properties about NFC card. |
| Ticket Type | Contains all properties about ticket type for based on bus route, each ticket type has diffirent price. |
| Bus Route | Contains all properties about bus route in Ho Chi Minh City. |
| Ticket | Contains all properties about ticket to specify which card buy ticket belongs to which ticket type on which bus route. |
| Credit Plan | Contains all info about credit plan which will be chosen while adding credit to card. |
| Payment Transaction | Contains all info about transaction when adding credit to card via cash, credit plan or scratch card. |
| Promotion | Contains all info about promotion. |
| Scratch Card | Contains all info about scratch card. |
| Offer Subscription | Contains all info about offer subscription. |
| User Subscription | Contains all info about user subscription to specify who subscribe which offer subscription. |

Table 44: Conceptual Diagram Data Dictionary

# Report No. 4 Software Design Description

## Design Overview

* 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: gives the interface design description, which includes component interface, web application interface, and mobile application design.
* Section 6: describe a fully attributed Entity Relationship Diagram.
* Section 7: describe the algorithms that apply in the system.

## System Architecture Design

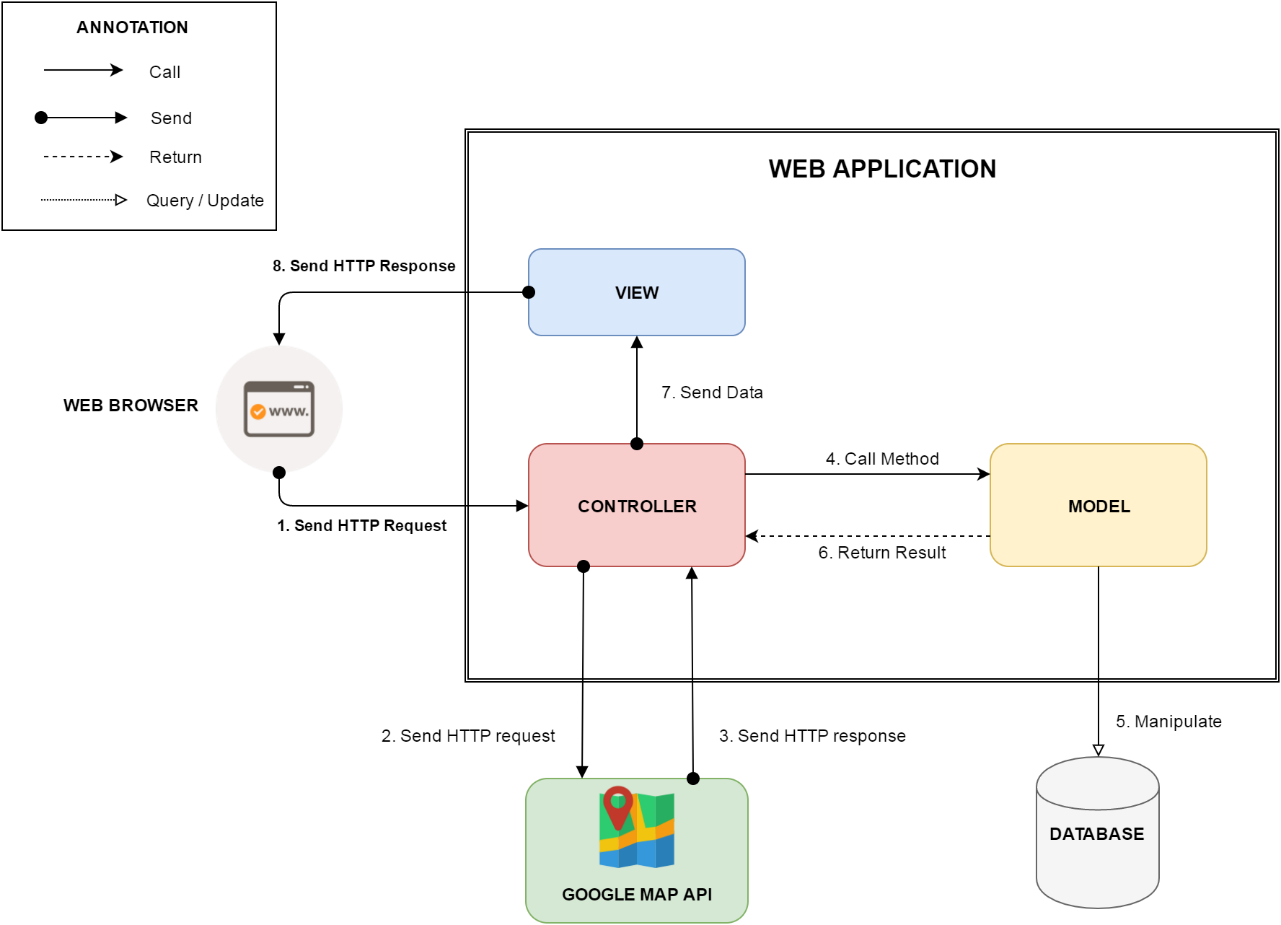


Figure 44: Web Application Architecture

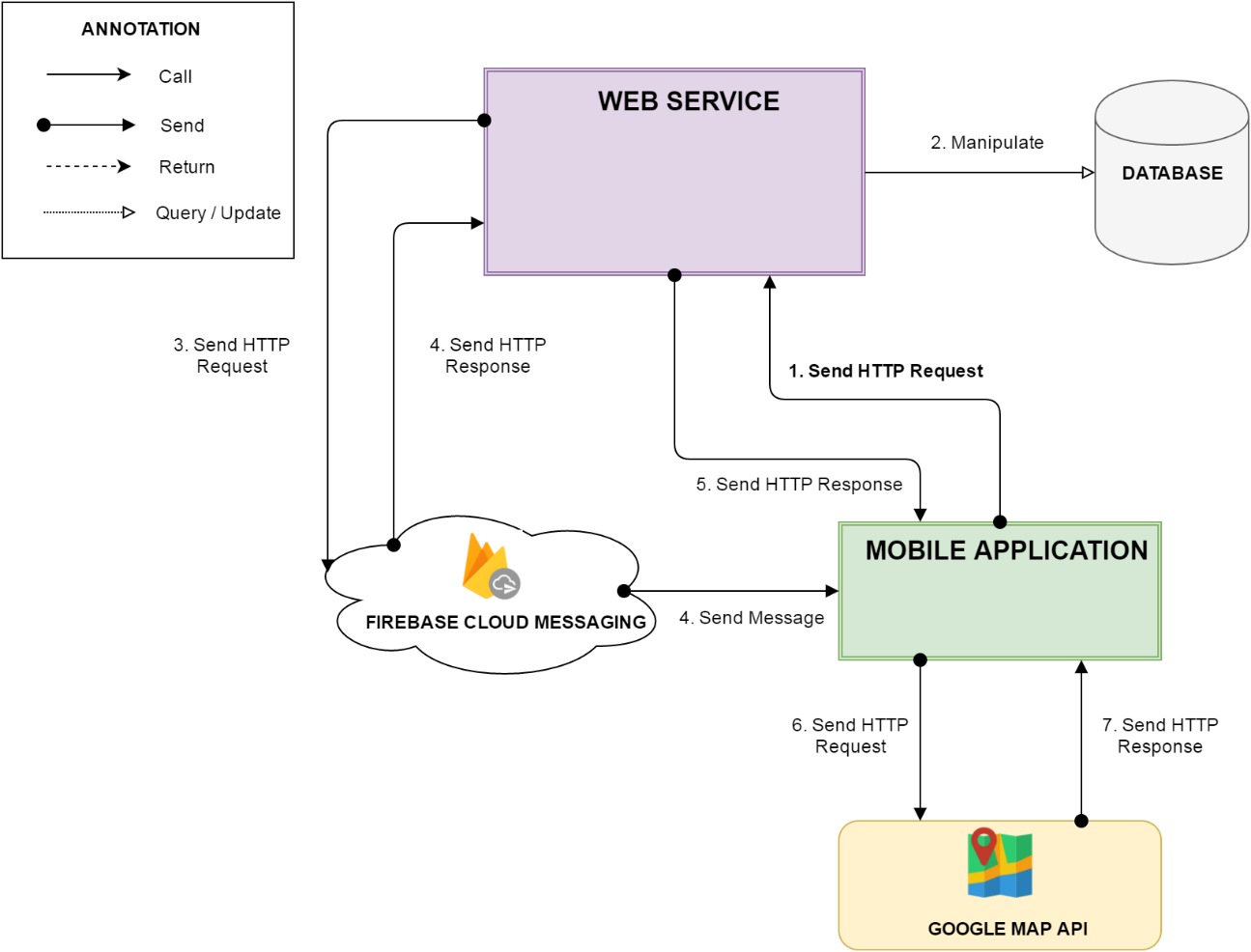


Figure 44: Mobile Application Architecture

### Web Application Architecture Description

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

* Web application contains Web services 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.
* We can organize the code better for maintainability, extensibility, reusability so we can expand the scope to other kind of illnesses such as flu, fever…
* In scope of 3-members team, MVC architecture makes it easier to split the big project into small modules and make it easier to assign each module for members in our team.

### Android Application Architecture Description

In our Android application, the application is developed under MVC architecture style. We choose this architecture for Android application because of following advantages:

* We can organize the code better for maintainability, extensibility and reusability.

## Component Diagram



Figure 45: Component Diagram

|  |  |
| --- | --- |
| Component Dictionary: Describes components | |
| Mobile application | Mobile application package |
| Web service | Provide API for mobile application to interact with the system. |
| Service | Common component is used to handle system’s business operations. |
| Admin Component | Component to handler admin activities in the system |
| Passenger Component | Component to handler passenger activities in the system |
| Staff Component | Component to handler staff activities in the system |
| Manager Component | Component to handler manager activities in the system |
| Repository | Component is used to handle interaction between the system and database. |
| Auto Handler Component | Component is used to handle scheduler in the system |
| Model | Entity framework |
| Google Firebase Service | Handle push notification with Google Firebase Message |

Table 45: Component Dictionary

## Detailed Description



### Class Diagram



Figure 46: Class Diagram

|  |  |  |
| --- | --- | --- |
| **Class Dictionary: Describe Classes** | | |
| **Class name** | **Mapping column with conceptual diagram** | **Description** |
| Subscription | Subscription | Contains the information of subscription. |
| User Subscription | Subscription | Contain the subscription information of user who subscribe an offer. |
| User | User | Contains the information of user. |
| Card | Card | Contains the information of card. |
| Ticket | Ticket | Contains the information of ticket. |
| Ticket Type | Ticket Type | Contains the information of ticket type. |
| Bus Route | Bus Route | Contains the information of bus route. |
| Credit Plan | Credit Plan | Contain the the information of credit plan which is used to add credit to card in the system. |
| Payment Transaction | N/A | Contains transaction history of card. |
| Promotion | Promotion | Contains the information of promotion. |
| Scratch Card | Scratch Card | Contains the information of scratch card. |

Table 46: Describe class

### Class diagram explanation

* + 1. Ticket Type

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a ticket type |
| Name | String | Private | Name of the ticket type |
| Price | Integer | Private | Price of the ticket type |
| Description | String | Private | Description of the ticket type |
| Status | Integer | Private | Status of the ticket type |

Table 47: Ticket Type Attributes

**Methods**

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

Table 48: Ticket Type Methods

* + 1. Ticket

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a ticket ticket |
| CardID | Integer | Private | Unique identififer of the NFC card |
| TicketTypeId | Integer | Private | Unique identifier of the ticket type |
| BusRouteId | Integer | Private | Unique identifier of the bus route |
| Total | Integer | Private | The ticket price |
| BoughtDated | DateTime | Private | The date time that ticket is bought |
| IsNoCard | Boolean | Private | Ticket is bought by card or cash |

Table 49: Ticket Attributes

**Methods**

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

Table 50: Ticket Methods

1. Bus Route

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a bus route |
| Code | String | Private | Unique code of the bus route |
| Name | String | Private | Name of the bus route |

Table 51: Bus Routes Attributes

**Methods**

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

Table 52: Bus Routes Methods

1. Card

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of an card |
| UniqueIdentifier | String | Private | Manufacture UID NFC card |
| CardName | String | Private | Name of the NFC card |
| RegistrationDate | DateTime | Private | Date of registration of the NFC card |
| Balance | Double | Private | Balance of the NFC card |
| Status | Integer | Private | Status of the NFC card |
| UserId | Integer | Private | Indentifier of the NFC’card holder |
| DataVersion | Long | Private | The version of data on card |

Table 53: Card Attributes

**Methods**

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

Table 54: Card Methods

1. Payment Transaction

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a payment transaction |
| CardId | Integer | Private | Identifier of the NFC card |
| CreditPlanId | Date | Private | Identifier of the credit plan |
| TransactionId | String | Private | Identifier of the payment transaction |
| Total | Double | Private | Total cost of the payment transaction |
| PaymentDate | DateTime | Private | Date of payment |

Table 55: Payment Transaction Attribute

**Methods**

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

Table 56: Payment Transaction Methods

1. User

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a user |
| PhoneNumber | String | Private | User’s phone number |
| Password | String | Private | User account’s password |
| Fullname | String | Private | User’s fullname |
| Status | Integer | Private | User’s statatus |
| RoleId | Integer | Private | User’s role identifier |
| DeviceToken | String | Private | Firebase notification token of a passenger’s mobile. |
| MinBalance | Integer | Private | Minimum balance of passenger’s cards that the system will notify them. |

Table 57: User Attributes

**Methods**

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

Table 58: User Methods

1. Promotion

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a promotion |
| Name | String | Private | Name of the promotion |
| Description | String | Private | Description of the promotion |
| Status | Integer | Private | Status of the promotion |
| CreatedDate | DateTime | Private | Date of creation |
| ExpiredDate | DateTime | Private | Date of expiration |

Table 59: Promotion Attributes

**Methods**

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

Table 60: Methods Methods

1. Credit Plan

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a credit plan |
| Name | String | Private | Name of the credit plan |
| Price | Integer | Private | Price of the credit plan |
| Status | Integer | Private | Status of the credit plan |

Table 61: Credit Plan Attributes

**Methods**

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

Table 62: Credit Plan Methods

1. Role

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a role |
| Name | String | Private | Name of the role |

Table 63: Role Attributes

**Methods**

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

Table 64: Role Methods

1. Scratch Card

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a scratch card |
| Code | String | Private | Code of a scratch card |
| Price | Integer | Private | Price of of a scratch card |
| Status | Integer | Private | Status of a scratch card |
| ExpiredDate | DateTime | Private | Date of expiration |

Table 65: Scratch Card Attributes

**Methods**

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

Table 66: Scratch Card Methods

1. Offer Subscription

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of an offer subscription |
| Name | String | Private | Name of an offer subscription |
| Code | String | Private | Code of an offer subscription |
| Price | Integer | Private | Price of an offer subscription |
| DiscountPercent | Integer | Private | Percentage of discount on each ticket of an offer subscription |
| TicketNumber | Integer | Private | Number discounted ticket of an offer subscription |
| Status | Integer | Private | Status of an offer subscription |

Table 67: Offer Subscription Attributes

**Methods**

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

Table 68: Offer Subscription Methods

1. User Subscription

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | Integer | Private | Unique identifier of a user subscription |
| UserId | Integer | Private | Identifier of the user |
| SubscriptionId | Integer | Private | Identifier of the offer subscription |
| ExpiredDate | DateTime | Private | Date of expiration |
| IsActive | Boolean | Private | Auto extend the subscription or not |
| TicketRemaining | Integer | Private | Number discounted ticket remaining. |

Table 69: User Subscription Attributes

**Methods**

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

Table 70: User Subscription Methods

### Interactive Diagram

* + 1. **Passenger**
       1. Add credit

Summary: This diagram shows process of passenger toping-up his/her card.

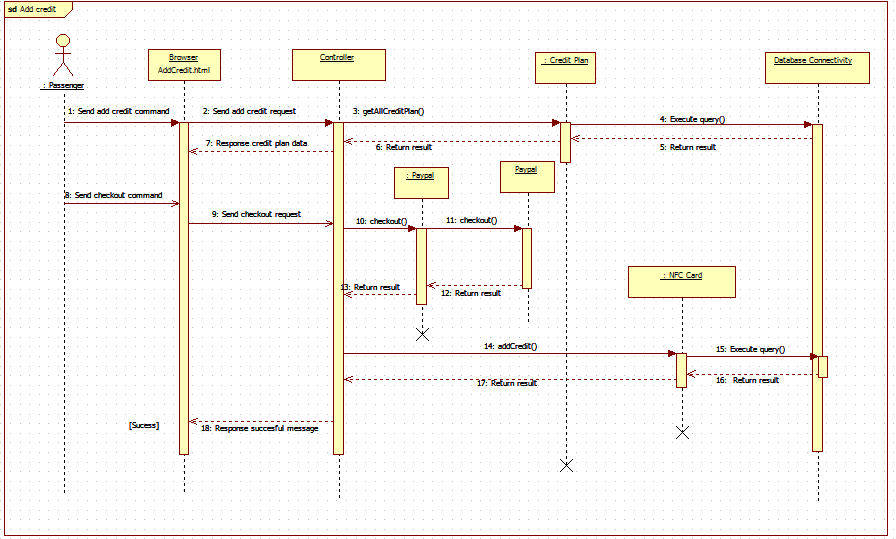


Figure 47: <Sequence Diagram> Add Credit

* + - 1. Get Cards

Summary: This diagram shows process of passenger getting his/her card information.

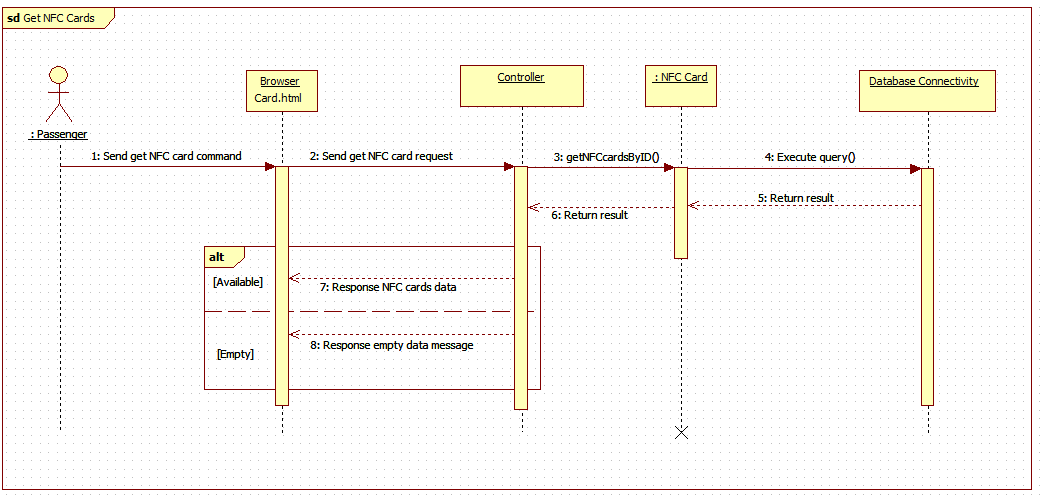


Figure 48: <Sequence Diagram> Get Cards

* + - 1. Find bus

Summary: This diagram shows process of passenger finding a bus’s information

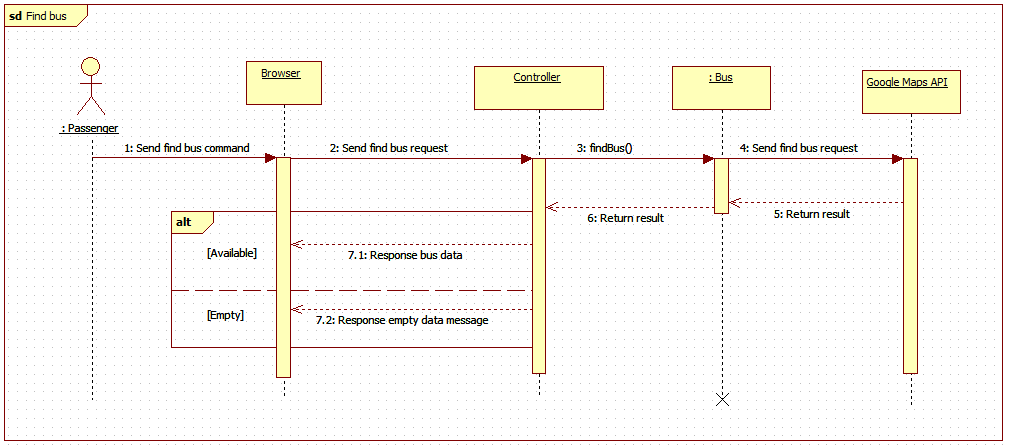


Figure 49: <Sequence Diagram> Find bus

* + - 1. Get outcome report

Summary: This diagram shows process of passenger getting his/her outcome report.

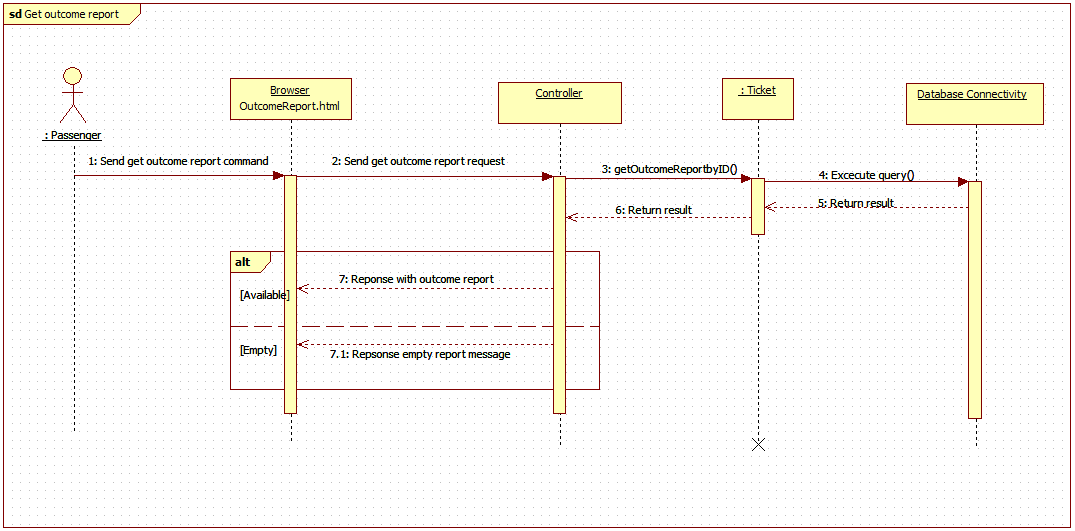


Figure 49: <Sequence Diagram> Get Income Report

* + 1. **Staff**

1. Edit Passenger

Summary: This diagram shows process of staff editing passenger information.

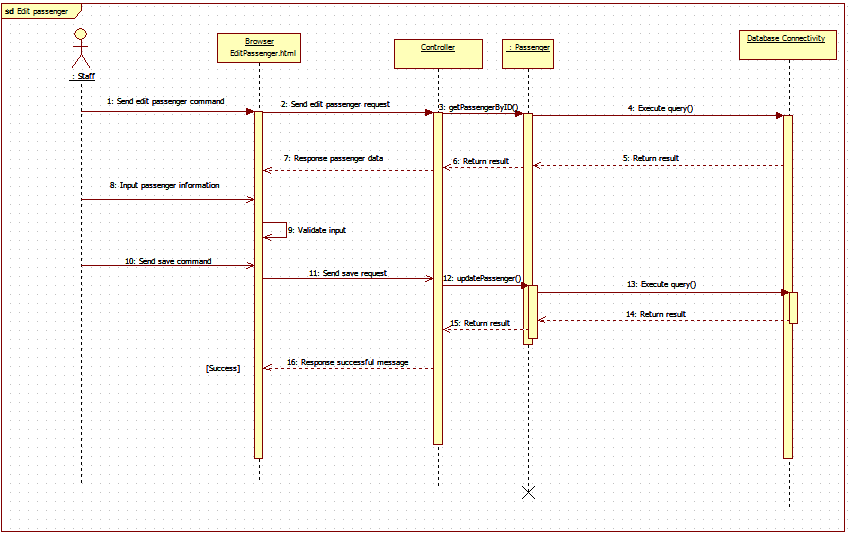


Figure 50: <Sequence Diagram> Edit Passenger

1. Activate/Deactivate Card

Summary: This diagram shows process of staff activating or deactivating passenger’s card

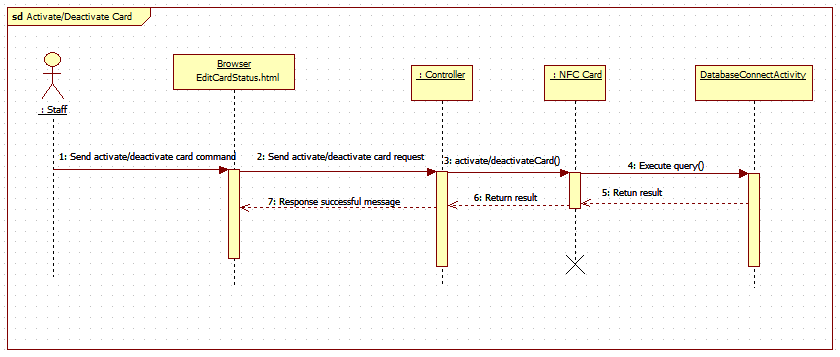


Figure 51: <Sequence Diagram> Activate/Deactivate Card

1. Add Card

Summary: This diagram shows process of staff adding new card

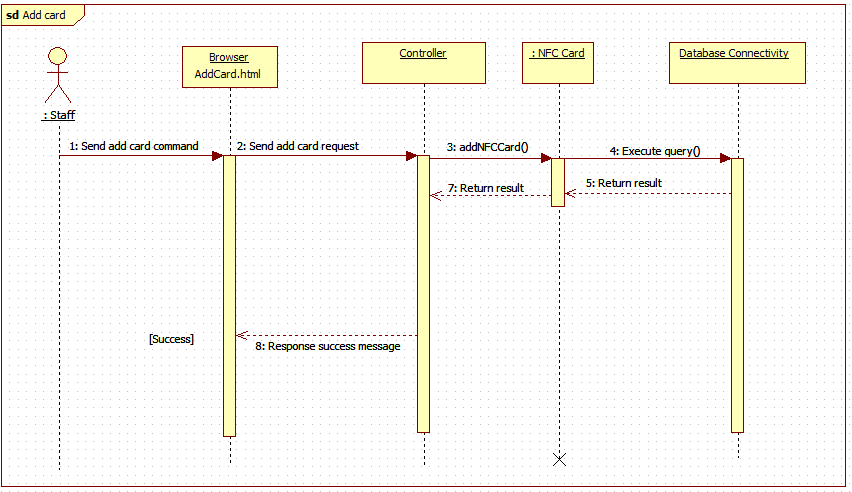


Figure 52: <Sequence Diagram>Add Card

* + 1. Manager

1. Get Income Report

Summary: This diagram shows process of manager getting income report

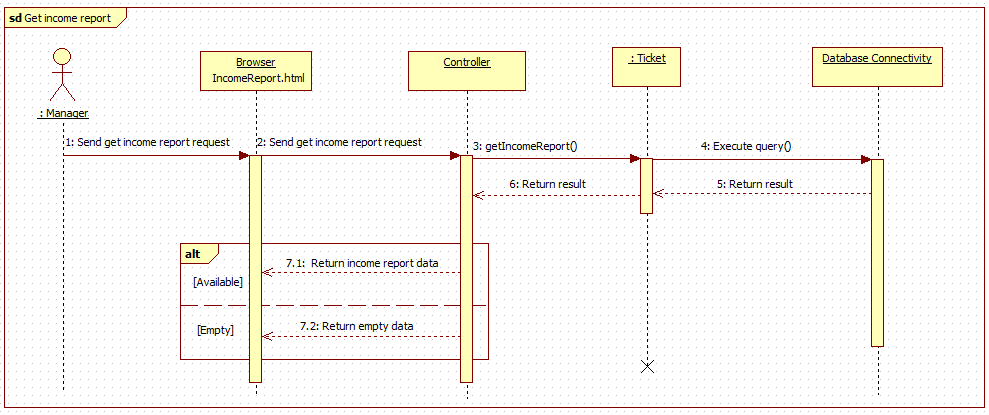


Figure 53: <Sequence Diagram>Get Outcome Report

1. Create promotion

Summary: This diagram shows process of manger creating a promotion

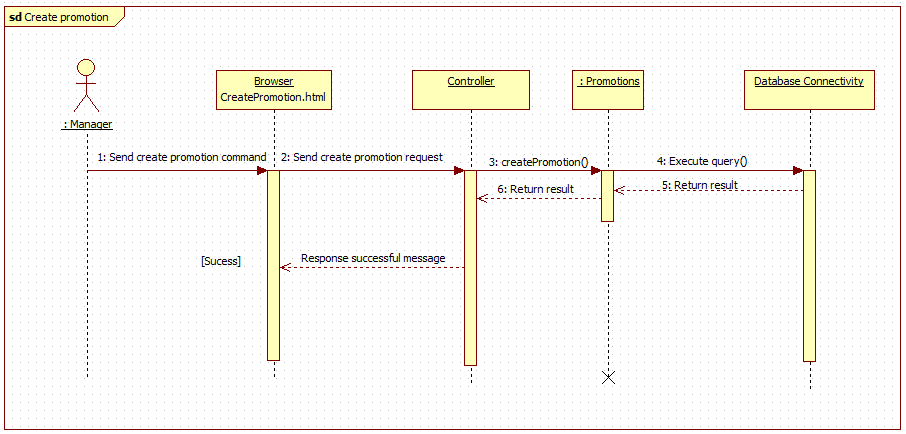


Figure 54: <Sequence Diagram>Create Promotion

1. **Mobile Application**
   * + 1. Add credit

Summary: This diagram shows process of passenger toping-up his/her card via mobile application.

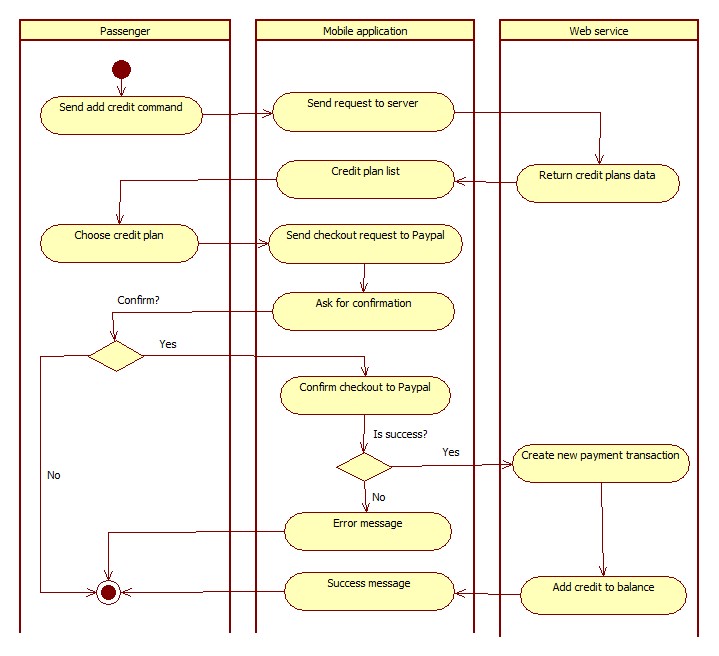


Figure 55: <Activity Diagram> Add Credit

* + - 1. Find Bus

Summary: This diagram shows process of passenger finding a bus route information via mobile application.

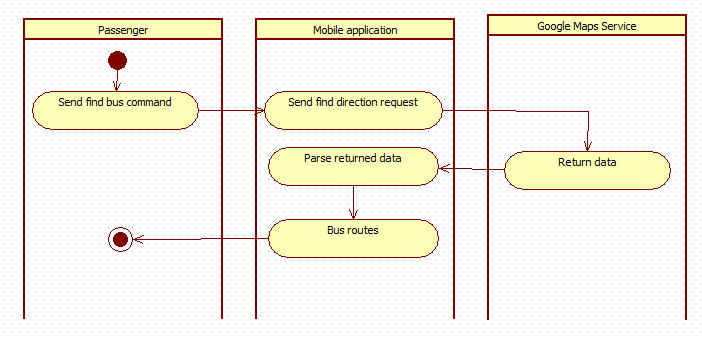


Figure 56: <Activity Diagram>Find Bus

* + - 1. Get Outcome Report

Summary: This diagram shows process of passenger getting his/her outcome report via mobile application.

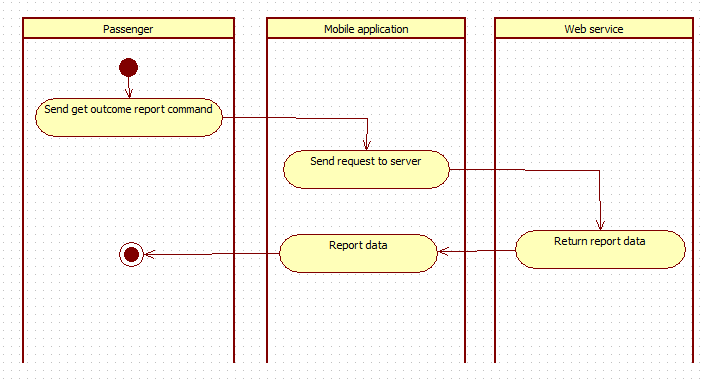


Figure 57: <Activity Diagram> Get Outcome Report

1. Send Notification

Summary: This diagram shows process of sending notification from system to passenger’s mobile phone.



Figure 58: <Activity Diagram> Send Notification

## User Interface

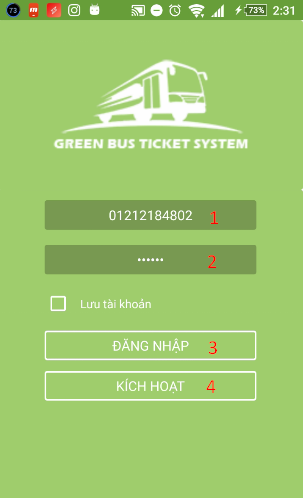
5.1 Component Interface

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Signature | Description | Input | Output | Output format | Exception |
| /Api/PushOfflineData | Push offline ticket from emulator to server after connection lost. | key: string  cardId: string  ticketTypeId: int  routeCode: string  boughtDate: string | Json string of message object | Json | JsonException |
| /Api/GetBusStop | Get all bus stops of a bus route. | key: string  routeCode: string | Json string of message object | Json | JsonException |
| /Api/AddCard | Add new NFC card to the system | key: string  cardId: string | Json string of medical record object | Json | JsonException |
| /Api/ChangeCardName | Change name of a card | key: string  cardId: string  name: string | Json string of message object | Json | JsonException |
| /Api/Topup | Add credit to card by scratch card | key: string  cardId: string  code: string | Json string of pet object | Json | JsonException |
| /Api/GetCardInfo | Get information of a card | key: string  cardId: string | Json string of pet object | Json | JsonException |
| /Api/AddCardBalance | Add credit to card by Paypal | key: string  cardId: string  creditPlanId: int  transactionId: string | Json string of pet object | Json | JsonException |
| /Api/GetProfile | Get profile of an account | key: string  phone: string | Json string of pet object | Json | JsonException |
| /Api/GetAllCreditPlan | Get all credit plans information | key: string | Json string of pet object | Json | JsonException |
| /Api/GetAllTicketType | Get all ticket type information | key: string | Json string of pet object | Json | JsonException |
| /Api/ GetBusRouteByCode | Get a bus route details information by route code | key: string  routeCode: string | Json string of pet object | Json | JsonException |
| /Api/ GetAllBusRoutes | Get all bus routes information | key: string | Json string of pet object | Json | JsonException |
| /Api/ SellTicket | Sell ticket to emulator by card | key: string  cardId: string  ticketTypeId: int  routeCode: string  currentBalance: long  dataVersion: long | Json string of pet object | Json | JsonException |
| /Api/ RequestSellTicketToken | Get current server token | key: string | Json string of pet object | Json | JsonException |
| /Api/ SellTicketWithToken | Sell ticket to emulator by phone, server token is required | key: string  cardId: string  ticketTypeId: int  routeCode: string  currentBalance: long  dataVersion: long  token: string | Json string of pet object | Json | JsonException |
| /Api/ Login | Login to the system | key: string  phone: string  password: string | Json string of pet object | Json | JsonException |
| /Api/ UpdateProfile | Update account profile | key: string  phone: string  password: string  fullname: string  minBalance: int | Json string of pet object | Json | JsonException |
| /Api/ ActivateAccountByApp | Activate account by mobile application | key: string  phone: string  cardId: string | Json string of pet object | Json | JsonException |
| /Api/ GetAllCard | Get card list of an account | key: string  phone: string | Json string of pet object | Json | JsonException |
| /Api/ GetReport | Get outcome report | key: string  phone: string  beginDate: string  endDate: string | Json string of pet object | Json | JsonException |
| /Api/ RegisterNotificationToken | Register device token to get notification | key: string  phone: string  token: string | Json string of pet object | Json | JsonException |
| /Api/ GetPromotion | Get current available promotions | key: string | Json string of pet object | Json | JsonException |

Table 71: Component Interface

### 5.2 Mobile Application Design

#### 5.2.1 Login



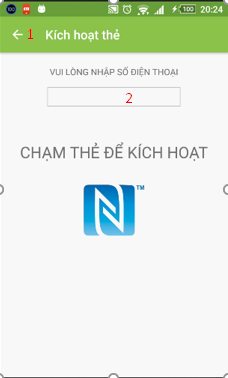
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control type | Data type |
| 1 | Phone number | Allow user to input their registered phone number. | No | Yes | Number input | String |
| 2 | Password | Allow user to input their registered password | No | Yes | Text input | String |

**Fields**

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 3 | Login | Perform login function | Yes | Open view main screen or display error messages |
| 4 | Activate | Allow user active account by phone number. | Yes | Open view active screen |

5.3.2 Active Card



**Fileds**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control type | Data type |
| 1 | Phone number | Allow user to input their registered phone number. | No | Yes | Text input | String |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 1 | Back | Navigate back 1 screen | No | Open previous screen in the activity stack |

**Button/Hyperlinks**

‘

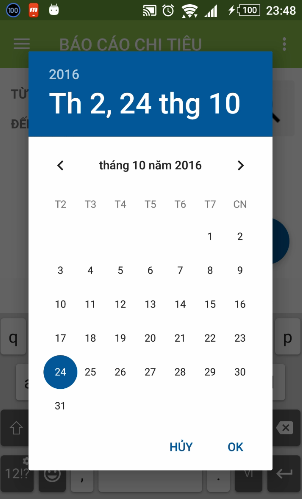
5.3.3 Main screen



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 1 | Navigation Drawer | Navigation Drawer | No | Open/Close Navigation Drawer |
| 2 | Menu item | Let user choose a function | No | Open fast function |
| 3 | Search | Open search screen | No | Open search screen |

5.3.4 View Report



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control type | Data type |
| 1 | Date Start | Allow user to input date start. | No | Yes | Text input | Datetime |
| 2 | Date End | Allow user to input date end. | No | Yes | Text input | Datetime |

**Fileds**

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 3 | View report | View report of card | No | Open/Close Navigation Drawer |
| 4 | Navigation Drawer | Navigation Drawer | No | Open/Close Navigation Drawer |

#### 5.3.5 Manage Account



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Field Name | Description | Read only | Mandatory | Control type | Data type |
| 2 | Full Name | Allow user input name. | No | Yes | Text Input | String |
| 3 | Old password | Allow user input password. | No | Yes | Text | String |
| 4 | New password | Allow user input password | No | Yes | Text | String |
| 5 | Confirm password | Allow user input password | No | Yes | Text | String |
| 6 | Notification | Allow user min balance to get notification. | No | Yes | Text | Number |

**Fields**

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Function | Description | Validation | Outcome |
| 1 | Navigation Drawer | Navigation Drawer | No | Open/Close Navigation Drawer |
| 7 | Save | Send request to server | No | Change data on this screen |

## Database design

6.1 Entity Relationship Diagram

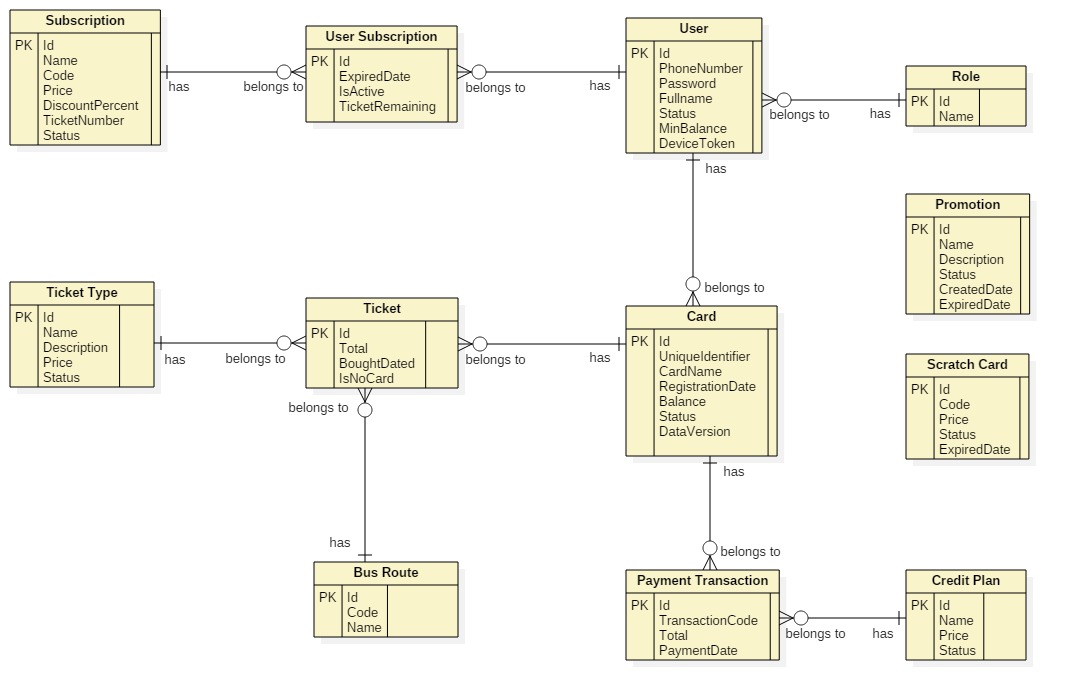


Figure 59: <Database design> Entity Relation Diagram

6.2 Entity Dictionary

|  |  |
| --- | --- |
| Entity Name | Description |
| User | Contains all properties about user. |
| Passenger | Contains all properties about passenger info. |
| Admin | Contains all properties about admin info. |
| Staff | Contains all properties about staff info. |
| Manager | Contains all properties about manager info. |
| Card | Contains all properties about NFC card. |
| Ticket Type | Contains all properties about ticket type for based on bus route, each ticket type has diffirent price. |
| Bus Route | Contains all properties about bus route in Ho Chi Minh City. |
| Ticket | Contains all properties about ticket to specify which card buy ticket belongs to which ticket type on which bus route. |
| Credit Plan | Contains all info about credit plan which will be chosen while adding credit to card. |
| Payment Transaction | Contains all info about transaction when adding credit to card via cash, credit plan or scratch card. |
| Promotion | Contains all info about promotion. |
| Scratch Card | Contains all info about scratch card. |
| Subscription | Contains all info about offer subscription. |
| User Subscription | Contains all info about user subscription to specify who subscribe which offer subscription. |

Table 72: Entity Dictionary

## Algorithms

7.1 AES Encryption

7.1.1 Definition

AES (acronym of Advanced Encryption Standard) is a symmetric encryption algorithm. The algorithm was developed by two Belgian cryptographer Joan Daemen and Vincent Rijmen. AES was designed to be efficient in both hardware and software, and supports a block length of 128 bits and key lengths of 128, 192, and 256 bits.

7.1.2 Define Problem

For convevient purposes of allowing passenger to buy ticket in case of no internet connection from emulator to server, we save card’s balance and data version to NFC card. We need to encrypt these information to make sure no one can change the original information.

7.1.3 Solution

In order to secure these informations, we use AES Encryption Algorithm to encrypt informations before saving to NFC card.

Algorithm reference: <http://howtodoinjava.com/security/java-aes-encryption-example/>

7.2 Token Authentication

7.2.1 Definition

Token authentication is a technique to prove that the passengers is who they claim to be. Requests which have the right token will be processed.

7.2.2 Define problem

When the passengers use mobile application to buy ticket instead of their cards. Mobile application sends some informations to emulator for verification, we have to make sure that these informations only valid in a specific time for buying ticket, anyone who can capture these informations will not be able to use it and buy ticket later.

7.2.3 Solution

Token we use in this solution is a GUID. GUIDs are stored as 128-bit values, and are displayed as 32 hexadecimal digits with groups separated by hyphens.

GUID Reference: <https://en.wikipedia.org/wiki/Globally_unique_identifier>

To solve this problem, we should follow these steps:

* Server creates a token if there is no token on the server.
* Server change the token frequently (every 30 minutes).
* When mobile application is used to buy ticket, it asks the token from server.
* Then mobile application sends a request along with this token to emulator.
* Emulator sends to server. Server verifies the received token, if it matchs current server token, request will be processed.

1. Complexity: O(n)
2. Flowchart

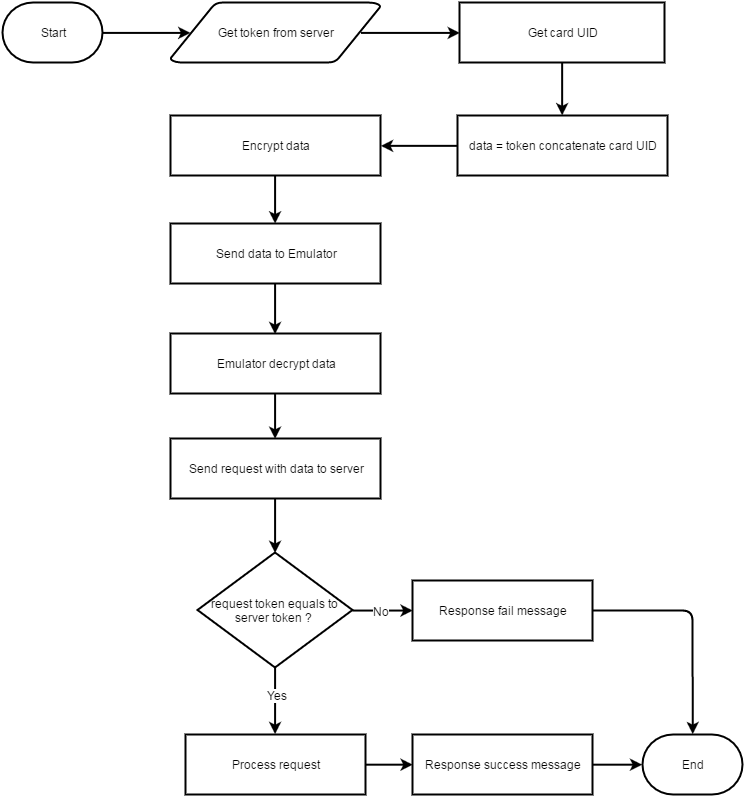


Figure 60: <Algorithms> Entity Relation Diagram

1. **Daily Usage Prediction**
2. Definition

Daily Usage Prediction is the way to predict how much a passenger need to spend for buying ticket in the next day base on the average bought tickets in the past.

1. Define Problem

Passenger sometimes forget to add credit to their cards. Each passenger has a diffirent travel demand. We should notify them in the morning if the remaining balance in their card is not enough for spending in the next day.

1. Solution

To solve this problem, we should follow these steps:

* Everyday, at a specific time, Auto Handler will filter the current passenger list to get passengers who have already installed mobile application, bought at least 30 tickets before and the last bought ticket must not exceed 7 days from the current date.
* Calculate the average spending for each customer and on each card of them:

**Average spending** =

**Total Spending**: Total money spent on a card in last 30 days.

**Number of dates**: Number of distinct dates which have at least one ticket was bought in last 30 days.

* If the current card’s balance less than the average spending above, a notification will be send to passenger’s mobile for reminding them.

1. Complexity: O(n2)
2. Flowchart

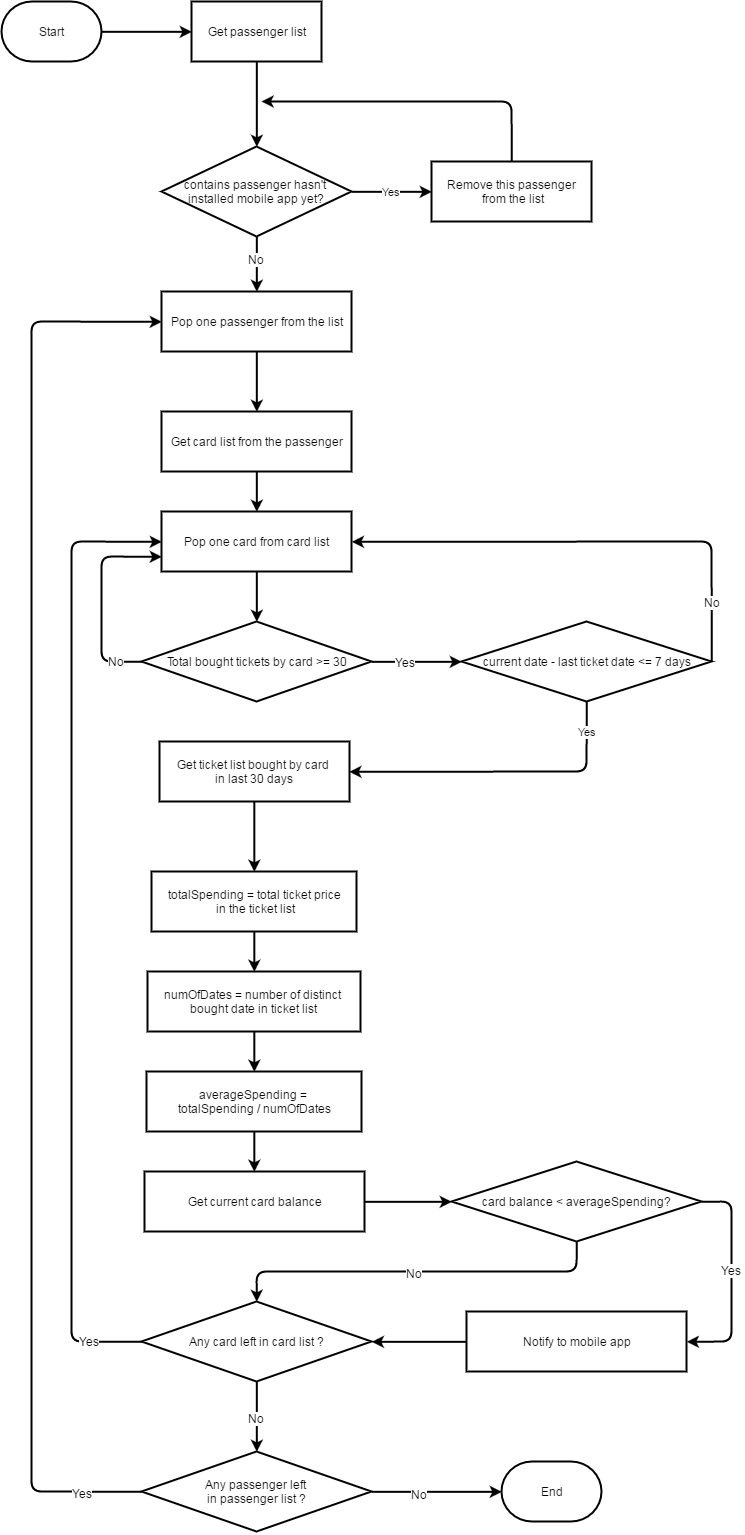


Figure 61: <Algorithms> Daily Usage Prediction

# Report No. 5 System Implementation & Test

## Introduction

### Overview

This section describes the approach and methodologies used by group to plan, organize and manage the testing of GBTS 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 GBTS system based on the core flow.
* Method: black-box testing
* Technique: check list

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

* Integration testing would be performed by all member of team and approved by team leader.

System testing is focused on assessing the system’s reliability. This process is concerned with finding errors that result from unanticipated interactions between components and component interface problems.

## Database Relationship Diagram

### Physical Diagram

### Data Dictionary



Figure 62: <Database Relation Diagram> Physical Diagram

|  |  |  |
| --- | --- | --- |
| Data dictionary: describe content of all tables | | |
| Table Name | **Mapping column with Conceptual diagram** | **Description** |
| User | User | Contain the user information. |
| Role | N/A | Contain the role information. |
| BusRoute | BusRoute | Contain the bus route information |
| Card | Card | Contain the card information |
| Credit Plan | Credit Plan | Contain the credit plan which is used to add credit to card in the system. |
| Offer Subscription | Subscription | Contain the offer subscription information. |
| Payment Transaction | Payment Transaction | Contain the payment transaction information. |
| Promotion | Promotion | Contain the promotion information. |
| ScratchCard | ScratchCard | Contain the scratch card information. |
| Ticket | Ticket | Contain the bought ticket information for a card on a bus route or via cash. |
| Ticket Type | Ticket Type | Contain the type of ticket information |
| Token | N/A | Contain the server token information for verifying ticket bought by phone. |
| UserSubscription | Subscription | Contain the subscription information of user who subscribe an offer. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Attributes | Description | Domain | Null |
| BusRoute | ID{PK} | Unique identifier of a bus route | Integer | No |
| Code | Unique code for each bus route | String | No |
| Name | Name of a bus route | String | No |
| Card | Id{PK} | Unique identifier of a card | Integer | No |
| UniqueIdentifier | Manufacturer ID of a card | String | No |
| CardName | Name of a card | String | Yes |
| RegistrationDate | Registration date of a card | Date Time | No |
| Balance | Balance of a card | Float | No |
| Status | Status of a card | String | No |
| UserId | Unique identifier of an user | Integer | Yes |
| DataVersion | Data version of a card | Big integer | Yes |
| CreditPlan | Id{PK} | Unique identifier of a credit plan | Integer | No |
| Name | Name of a credit plan | String | No |
| Price | Price of a credit plan | Integer | No |
| Status | Status of a credit plan | Integer | No |
| PaymentTransaction | Id{PK} | Unique identifier of a payment transaction | Integer | No |
| CardId | Unique identifier of a card | Integer | No |
| CreditPlanId | Unique identifier of a credit plan | Integer | No |
| TransactionCode | Unique code of a transaction | String | No |
| Total | Total payment amount of a transaction | Integer | No |
| PaymentDate | Date and time of a transaction | Date Time | No |
| Promotion | Id{PK} | Unique identifier of a promotion | Integer | No |
| Name | Name of a promotion | String | No |
| Status | Status of a description | Integer | No |
| CreateDate | Created date of a promotion | Date Time | No |
| ExpireDate | Expired date of a promotion | Date Time | No |
| Role | Id{PK} | Unique identifier of a role | Integer | No |
| Name | Name of a role | String | No |
| ScratchCards | Id{PK} | Unique identifier of a scratch card | Integer | No |
| Code | Code of a scratch card | String | No |
| Price | Price of a scratch card | Integer | No |
| Status | Status of a scratch card | Integer | No |
| ExpiredDate | Expired date of a scratch card | Date Time | No |
| Subscription | Id | Unique identifier of a subscription | Integer | No |
| Name | Name of a subscription | String | No |
| Code | Code of a subscription | String | No |
| Price | Price of a subscription | Integer | No |
| DiscountPercent | Discount percent to ticket price of a subscription | Float | No |
| TicketNumber | Number of ticket can be use in a subscription | Integer | No |
| Status | Status of a subscription | Integer | No |
| Ticket | Id{PK} | Unique identifier of a ticket | Integer | No |
| CardId | Unique identifier of a card | Integer | No |
| BusRouteId | Unique identifier of a bus route | Integer | No |
| Total | Total payment amount for a ticket | Integer | No |
| BoughtDated | Date of ticket bought | Date Time | No |
| IsNoCard | Bought ticket using phone | Boolean | No |
| TicketType | Id{PK} | Unique identifier of a ticket | Integer | No |
| Name | Name of a ticket type | String | No |
| Description | Description of a ticket type | String | Yes |
| Price | Price of a ticket type | Integer | No |
| Status | Status of a ticket type | Integer | No |
| User | UserId{PK} | Unique identifier of a user | Integer | No |
| PhoneNumber | Phone number of a user’s account | String | No |
| Password | Password for a user’s account | String | No |
| Fullname | Fullname of a user | String | Yes |
| Status | Status of a user | Integer | No |
| RoleId | Unique identifier of a user’s role | Integer | No |
| DeviceToken | Device token of user’s device | String | Yes |
| MinBalance | Minimum balance to get notification | Integer | Yes |
| UserSubScription | Id | Unique identifier of a user subscription | Integer | No |
| UserId | Unique identifier of a user | Integer | No |
| SubscriptionId | Unique identifier of a subscription | Integer | No |
| ExpiredDate | Expired date of user subscription | Date Time | No |
| IsActive | Activity of user subscription | Boolean | No |
| TicketRemaining | Number of ticket remain of user subscription | Integer | No |

Table 73: Data dictionary: Describe content of all tables

## Performance Measures

## Web Application Loading Speed

* + 1. Definition

This section tests the speed of loading some page of the web application.

* + 1. Test Environment
    - Server: Shared Windows Hosting on server:
      * Operating System: Windows Server 2012 R2
      * Network: 100 Mbps
    1. Test Cases

|  |  |  |
| --- | --- | --- |
| Test case | Role | Page |
| 1 | Unauthorize User | * Login |
| 2 | Passenger | * Get card list |
| 3 | Passenger | * Get outcome report |
| 4 | Staff | * Publish scratch card |
| 5 | Manager | * Get income report |

* + 1. Test Results

The test is run 100 times. Each time we run all the listed test cases and record the average time of web services responding.

|  |  |
| --- | --- |
| Test case | Average Loading Time (second) |
| 1 | 1.15 |
| 2 | 1.38 |
| 3 | 1.96 |
| 4 | 2.16 |
| 5 | 2.23 |

Test report: <https://github.com/dnhoang/GBTS/blob/master/Documents/Test_Report/GBTS_Performance_Test_Report.xlsx>

### Mobile API Speed

1. Definition

This section shows the testing speed results of some mobile application functions when connection to server through API

1. Test Environment
   * + Server: Shared Windows Hosting on server:
       - Operating System: Windows Server 2012 R2
       - Network: 100 Mbps
2. Test Cases

|  |  |  |
| --- | --- | --- |
| Test case | Role | Function |
| 1 | Unauthorize User | * Login |
| 2 | Passenger | * Get card list |
| 3 | Passenger | * Get outcome report |
| 4 | Passenger | * Topup card |

1. Test Results

The test is run 100 times. Each time we run all the listed test cases and record the average time of web services responding.

|  |  |
| --- | --- |
| Test case | Average Response Time (second) |
| 1 | 1.44 |
| 2 | 1.95 |
| 3 | 2.05 |
| 4 | 1.34 |

Test report: <https://github.com/dnhoang/GBTS/blob/master/Documents/Test_Report/GBTS_Performance_Test_Report.xlsx>

## Test Plans

### Features to be tested

* 1. **Manager**:
     + Add/update credit plans
     + Add/update ticket types
     + Add/update promotion packages
     + Add/delete promotions
  2. **Staff**:
     + Add/update passengers’ cards
     + Update passengers’ information
     + Publish scratch cards
  3. **Passenger**:
     + Top up card
     + Update card’s name
  4. **Admin**:
     + Insert account
     + Activate/deactivate account
  5. **Auto handler**:
     + Predict usage
     + Generate token
     + Crawl bus route data
     + Auto extend subcription
  6. Login, logout, update profile
  7. **Mobile application**:
     + Login, logout
     + Top up card
     + Add credit
     + Activate card
     + Update passenger’s profile
  8. **Emulator application**:
     + Verify ticket
     + Top up card
     + Publish card

### Features not to be tested

1. **Manager**:

* View income report

1. **Passenger**:

* View cards’ information
* Search bus routes by bus code
* Find buses by lacations

1. **Mobile** **application**

* View cards’ information
* View outcome report

## System Testing Test Case

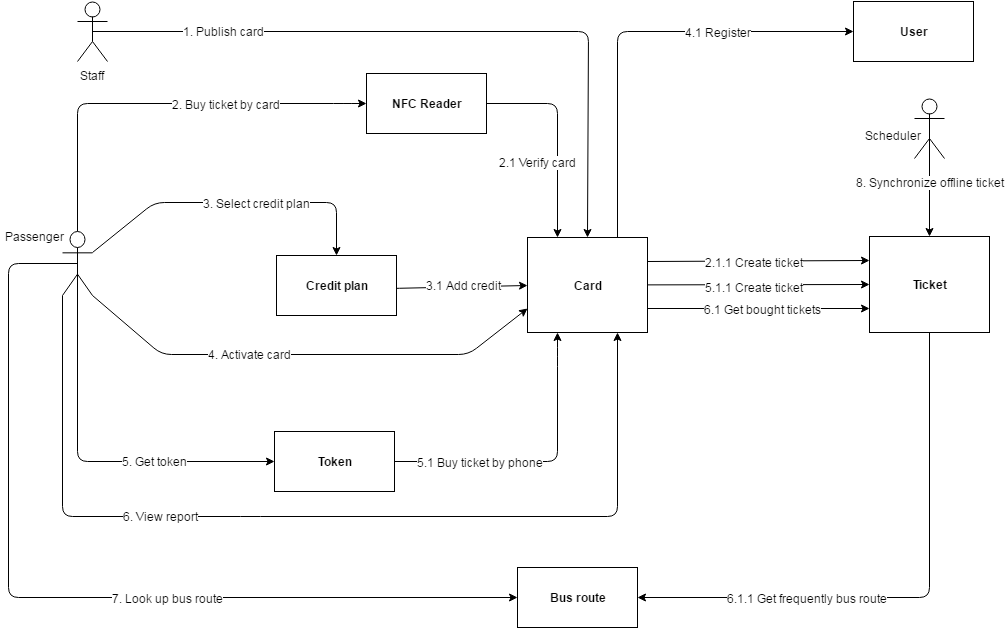


Figure 63: <Testcase> System Test

### Web Application Test Cases

* + - * 1. <Unauthorized User> Login

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| LG01 | Login with valid phone number and password | 1. Go to login page 2. Input valid phone and password 3. Press “Đăng nhập” button | Show target main page. |  | Pass | From 20/9/16  To 12/12/16 |
| LG02 | Login with invalid phone number or password | 1. Go to login page 2. Input invalid phone and password 3. Press “Đăng nhập” button | Show message “Sai số điện thoại hoặc mật khẩu” |  | Pass | From 20/9/16  To 12/12/16 |
| LG03 | Login with blocked account | 1. Go to login page 2. Input blocked account phone number and password 3. Press “Đăng nhập” button | Show message “Tài khoản đã bị khóa” |  | Pass | From 20/9/16  To 12/12/16 |

Table 75: <Unauthorized> Login

* + - * 1. <Manager> Add card

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| AC01 | Add card successful | 1. Go to card page 2. Press “Thêm thẻ mới” button 3. Tap new card on the emulator 4. Input initial card balance 5. Press “Lưu” button | Updated card page with new promotion added |  | Pass | From 1/10/16  To 12/12/16 |

Table 76: <Manager> Add Card

* + - * 1. <Staff> Activate/Deactivate card

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| AD01 | Activate card successful | 1. Go to card page 2. Press “Mở khóa thẻ” button | “Mở khóa thẻ” button change to “Khóa thẻ” button |  | Pass | From 1/10/16  To 12/12/16 |
| AC02 | Deactivate card successful | 1. Go to card page 2. Press “Mở khóa thẻ” button | “Khóa thẻ” button change to “Mở khóa thẻ” button |  | Pass | From 1/10/16  To 12/12/16 |

Table 77: <Staff> Add Card

* + - * 1. <Passenger> Add credit

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| CR01 | Add credit successfully | 1. Go to card page 2. Press “Nạp tiền” button 3. Select a credit plan 4. Input valid PayPal account 5. Press “Continue” button | Updated “Thẻ của bạn” panel with updated balance |  | Pass | From 25/9/16  To 12/12/16 |
| CR02 | Add credit using invalid PayPal account | 1. Go to card page 2. Press “Nạp tiền” button 3. Select a credit plan 4. Input invalid PayPal account 5. Press “Continue” button | Show error message “Check your email address and password and try again.” |  | Pass | From 25/9/16  To 12/12/16 |

Table 78: <Passenger> Add Credit

### Web Application Test Cases

* + - * 1. <Passenger> Login

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| LI01 | Login successfully | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button | Show main screen |  | Pass | From 1/10/16  To 12/12/16 |
| LI02 | Login with invalid phone number or password | 1. Open “GBTS” application 2. Input invalid phone number and password 3. Press “Login” button | Show message “Sai điện thoại hoặc mật khẩu” |  | Pass | From 1/10/16  To 12/12/16 |
| LI03 | Login with blocked account | 1. Open “GBTS” application 2. Input with blocked account 3. Press “Login” button | Show message “Tài khoản đã bị khóa” |  | Pass | From 1/10/16  To 12/12/16 |

Table 79: <Passenger> Login

* + - * 1. <Passenger> Add credit

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| CR01 | Add credit successful | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Select card panel 5. Choose a card 6. Press “Nạp tiền” button 7. Select credit package 8. Press “Thanh toán” button 9. Press “Pay with” PayPal” button 10. Input valid email and password 11. Press “Login” button 12. Press “Pay with PayPal” button 13. Press “Pay” button | Updated “Thẻ của bạn” panel with updated balance |  | Pass | From 5/10/16  To 12/12/16 |
|  | Add credit using wrong PayPal account | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Select card panel 5. Choose a card 6. Press “Nạp tiền” button 7. Select credit package 8. Press “Thanh toán” button 9. Press “Pay with” PayPal” button 10. Input invalid account 11. Press “Login” button 12. Press “Pay with PayPal” button 13. Press “Pay” button | Show error message “Check your email address and password and try again.” |  | Pass | From 5/10/16  To 12/12/16 |

Table 80: <Passenger> Add Credit

* + - * 1. <Passenger> Buy ticket by phone

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| PT01 | Buy ticket by phone successfully | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Tap phone on the emulator | Show successful message |  | Pass | From 7/11/16  To 12/12/16 |

Table 81: <Passenger> Buy ticket by phone

* + - * 1. <Passenger> Top up card

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| TU01 | Top up card successfully | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Select card panel 5. Choose a card 6. Press “Nạp tiền” button 7. Press “Nạp tiền bằng thẻ cào” button 8. Input valid scratch card’s code 9. Press “Nạp” button | Show main screen |  | Pass | From 10/10/16  To 12/12/16 |
| TU02 | Top up card using wrong code | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Select card panel 5. Choose a card 6. Press “Nạp tiền” button 7. Press “Nạp tiền bằng thẻ cào” button 8. Input valid scratch card’s code 9. Press “Nạp” button | Show message “Mã nạp tiền không hợp lệ hoặc hết hạn” |  | Pass | From 10/10/16  To 12/12/16 |
| TU03 | Top up card without a code | 1. Open “GBTS” application 2. Input valid phone number and password 3. Press “Login” button 4. Select card panel 5. Choose a card 6. Press “Nạp tiền” button 7. Press “Nạp tiền bằng thẻ cào” button 8. Leave scratch card’s code space blank 9. Press “Nạp” button | Show message “Mã nạp tiền không hợp lệ hoặc hết hạn” |  | Pass | From 10/10/16  To 12/12/16 |

Table 82: <Passenger> Top up card

* + - * 1. <Emulator> Verify ticket

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| TU01 | Verify ticket successfully | 1. Tap GBTS card on the emulator | Show success screen |  | Pass | From 20/9/16  To 12/12/16 |

Table 82: <Emulator> Verify ticket

* + - * 1. <Emulator> Verify mobile ticket

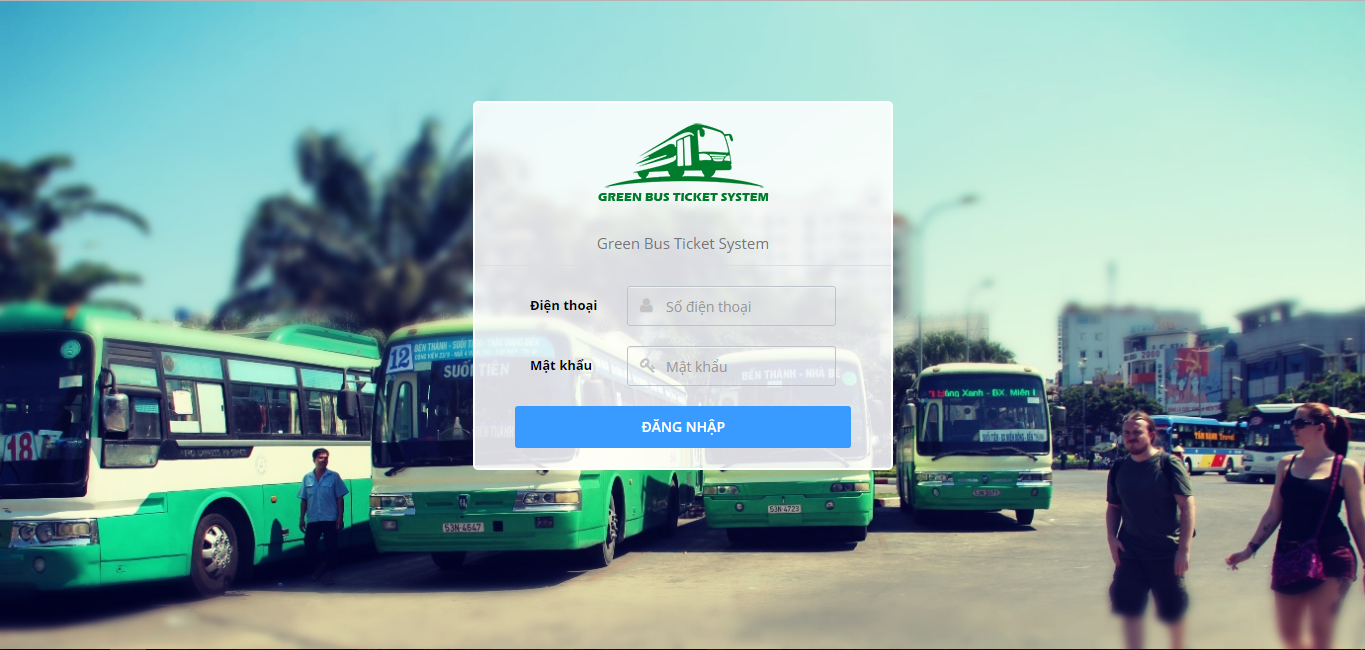
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Test Case Description | Test case procedure | Expected output | Inter-test case dependence | Result | Test Date |
| TU01 | Verify ticket with token successfully | 1. Open GBTS application on passenger’s phone 2. Tap passenger’s phone on the emulator to confirm | Show success screen |  | Pass | From 1/10/16  To 12/12/16 |

Table 83: <Emulator> Verify mobile ticket

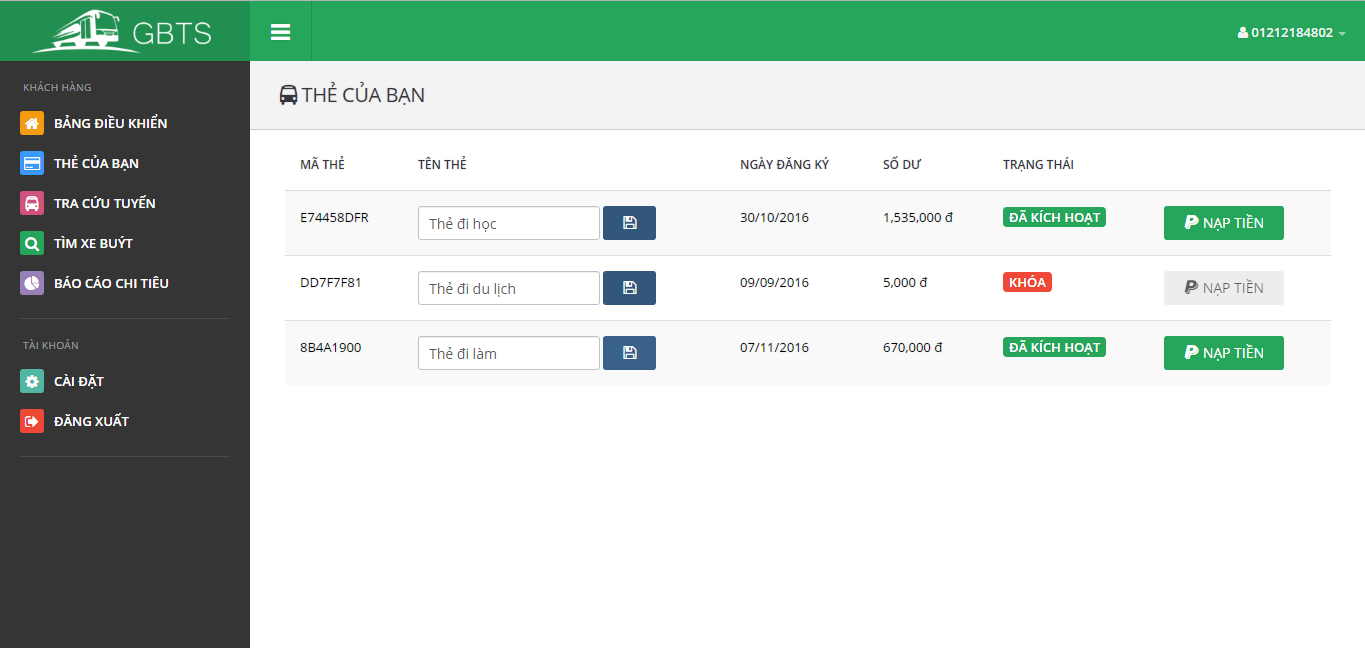
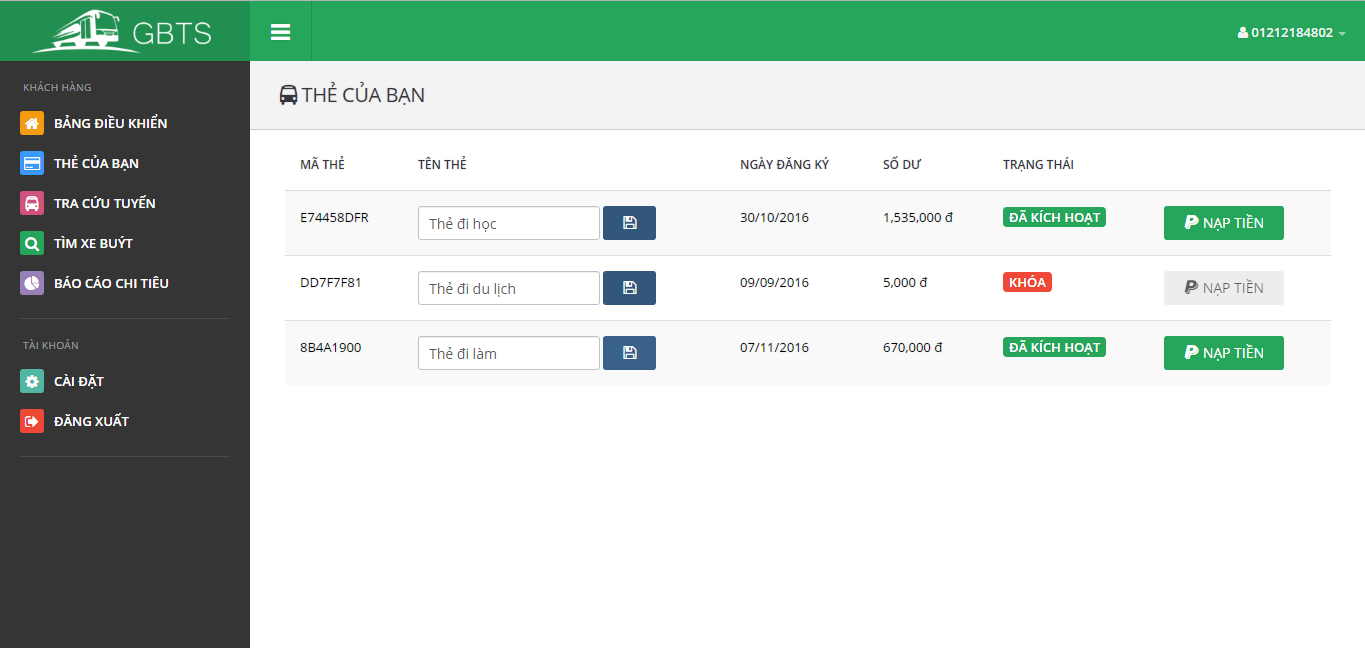
# Software User’s Manual

### Web application

1.1 <Unauthorized user>Login



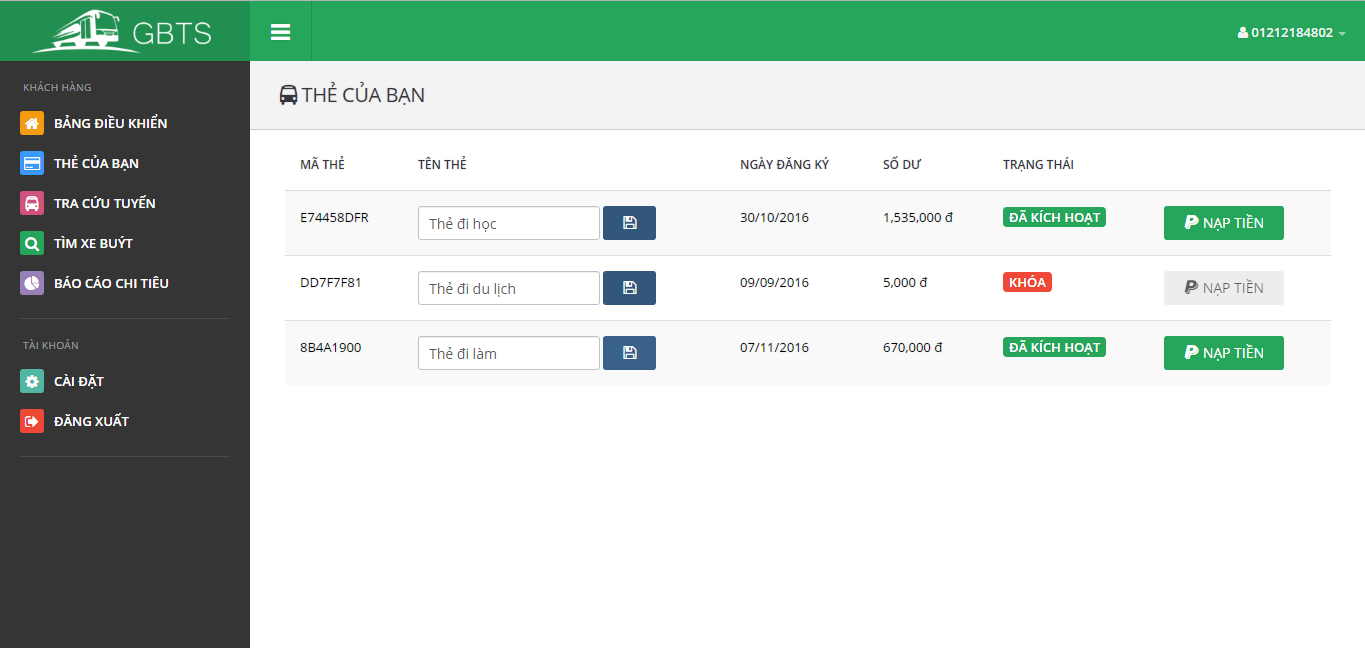
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “ĐĂNG NHẬP” button |

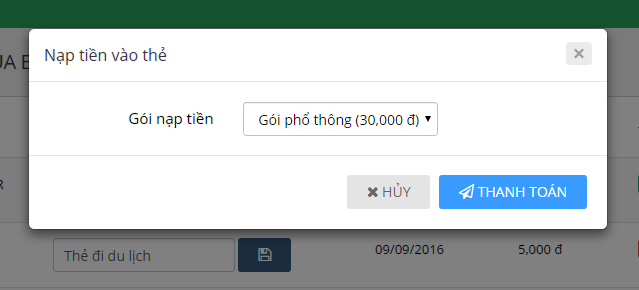
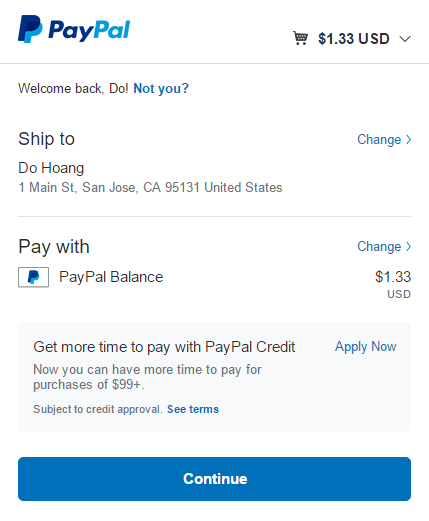
1.2 <Passenger> Get NFC cards

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Thẻ của bạn” button |

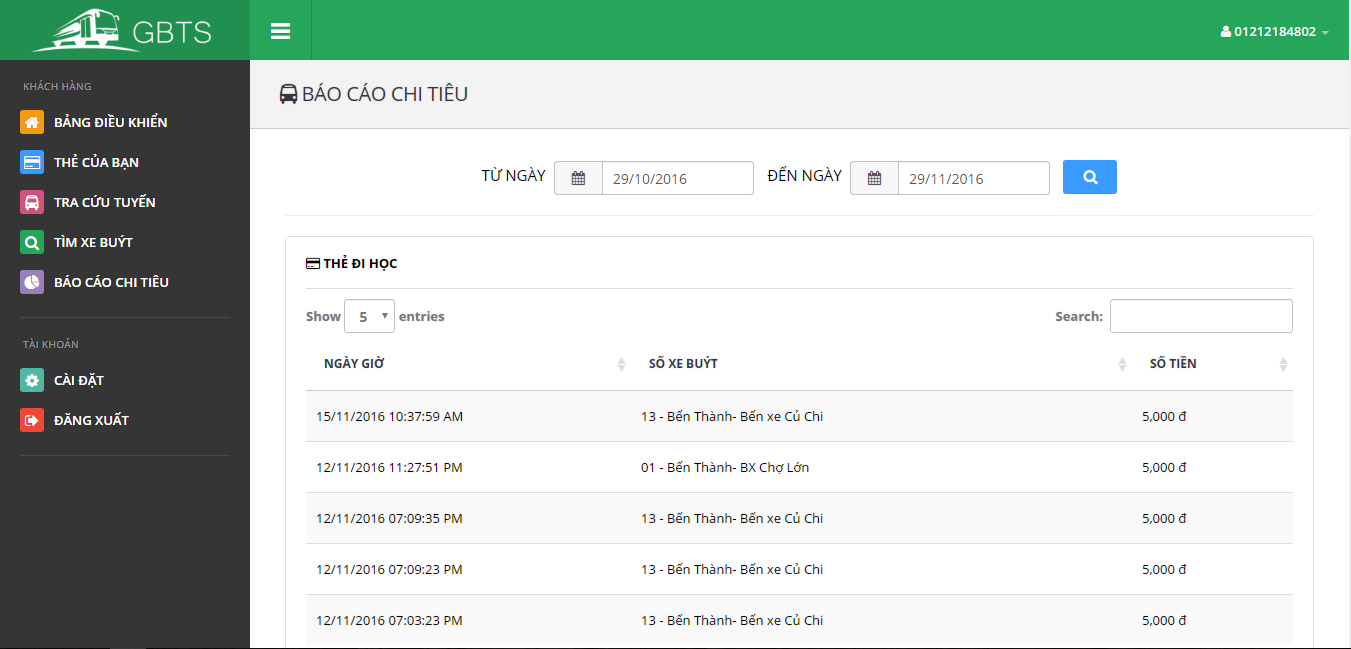
1.3 <Passenger> Edit card name

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Thẻ của bạn” button |
| 2 | Input new card name |
| 3 | Click save button |

<Passenger> Add credit

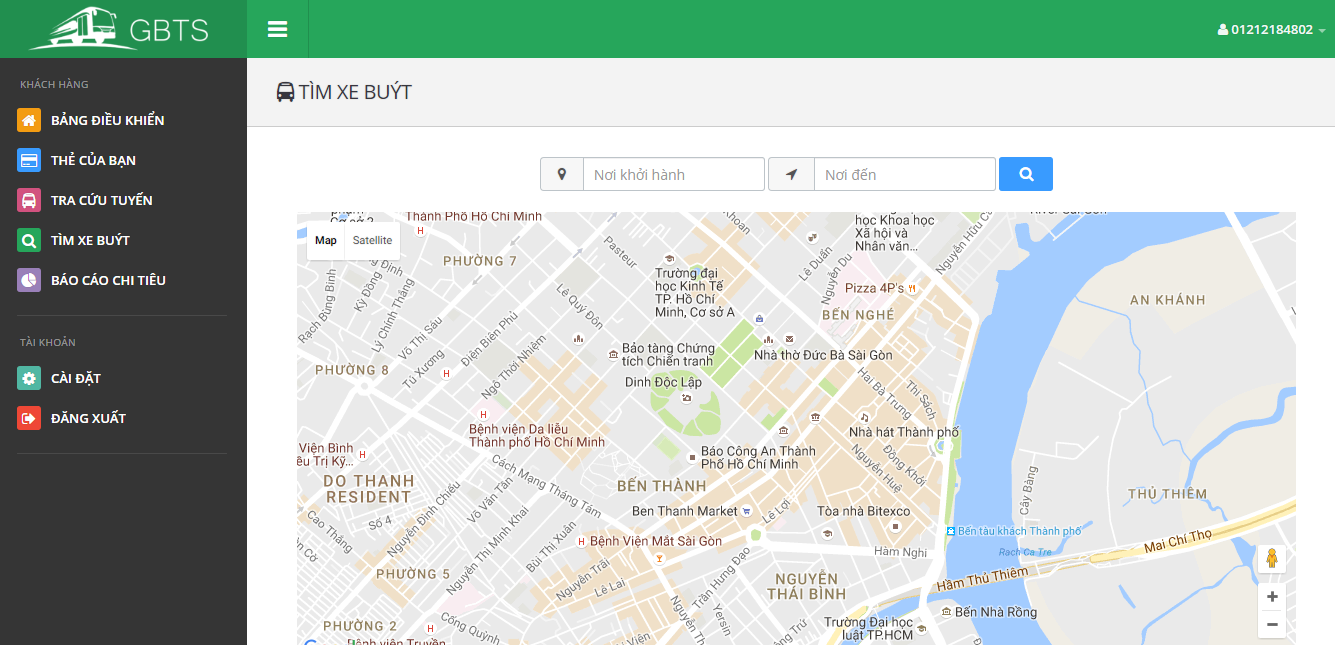


|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Thẻ của bạn” button |
| 2 | Click “Nạp tiền” button |
| 3 | Select a credit plan |
| 4 | Click “Thanh toán” button |
| 5 | Confirm the payment by clicking “Continue” |

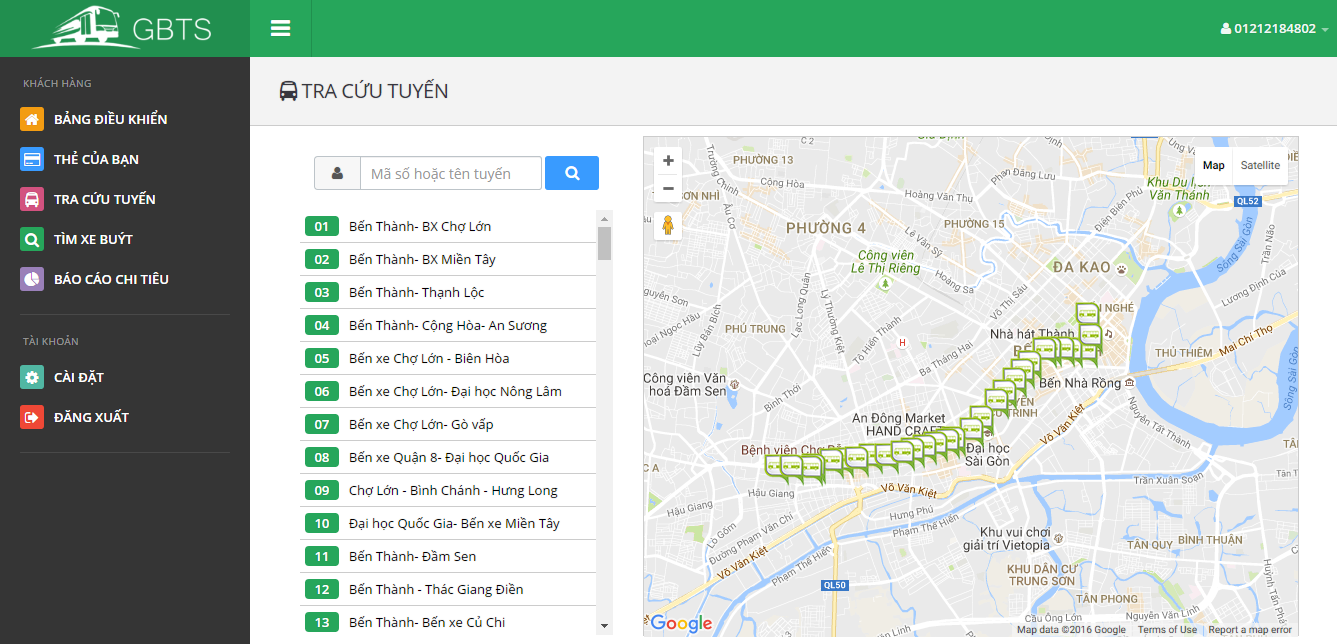
1.4 <Passenger> Get outcome report

|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Báo cáo chi tiêu” button |

1.5 <Passenger> Look up bus route

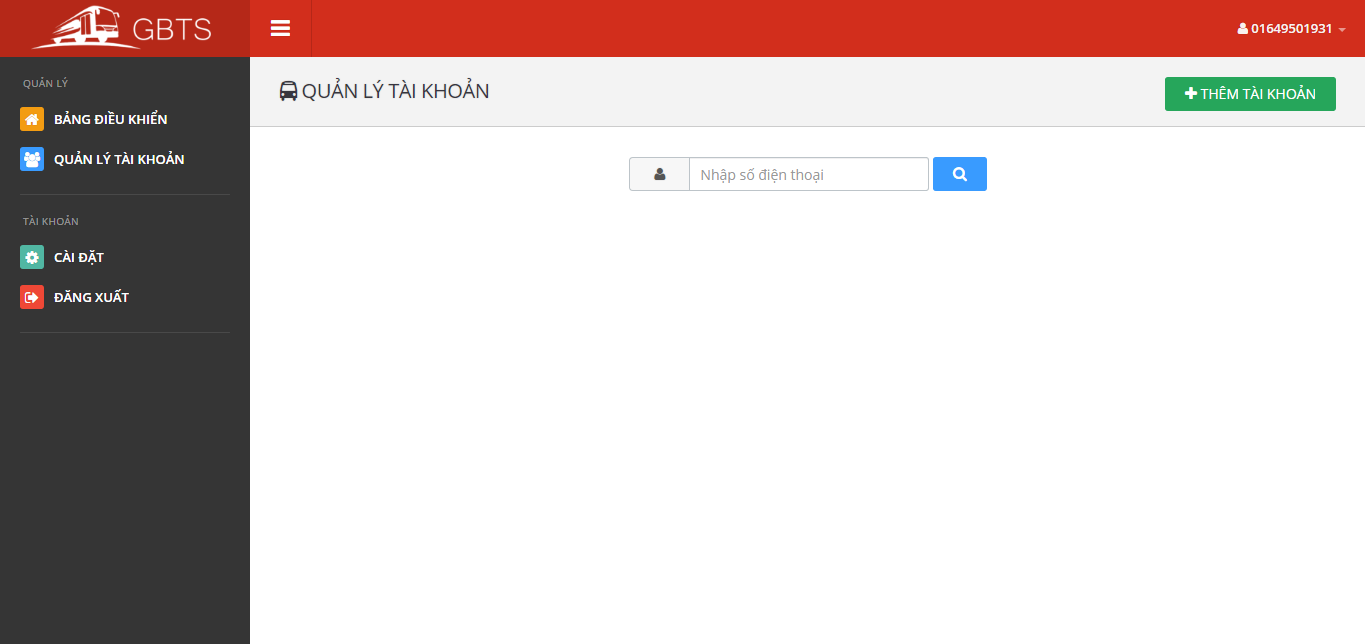


|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Tìm xe buýt” button |
| 2 | Input “Nơi khởi hành” and “Nơi đến” |
| 3 | Click search button |

1.6 <Passenger> Find bus

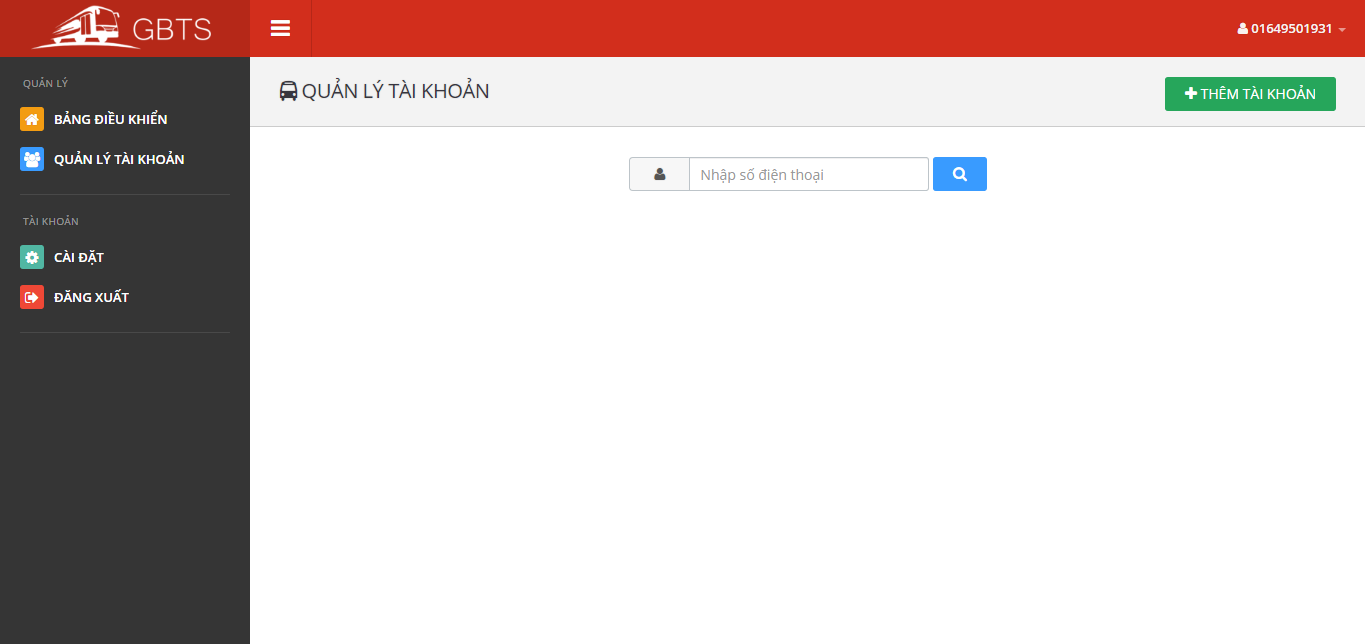
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Tra cứu tuyến” button |
| 2 | Choose a bus route in a list or input bus route name or code into “Mã số tuyến hoặc tên tuyến” |

1.7 <Administrator> Search account



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý tài khoản” button |
| 2 | Input a phone number |
| 3 | Click search button |

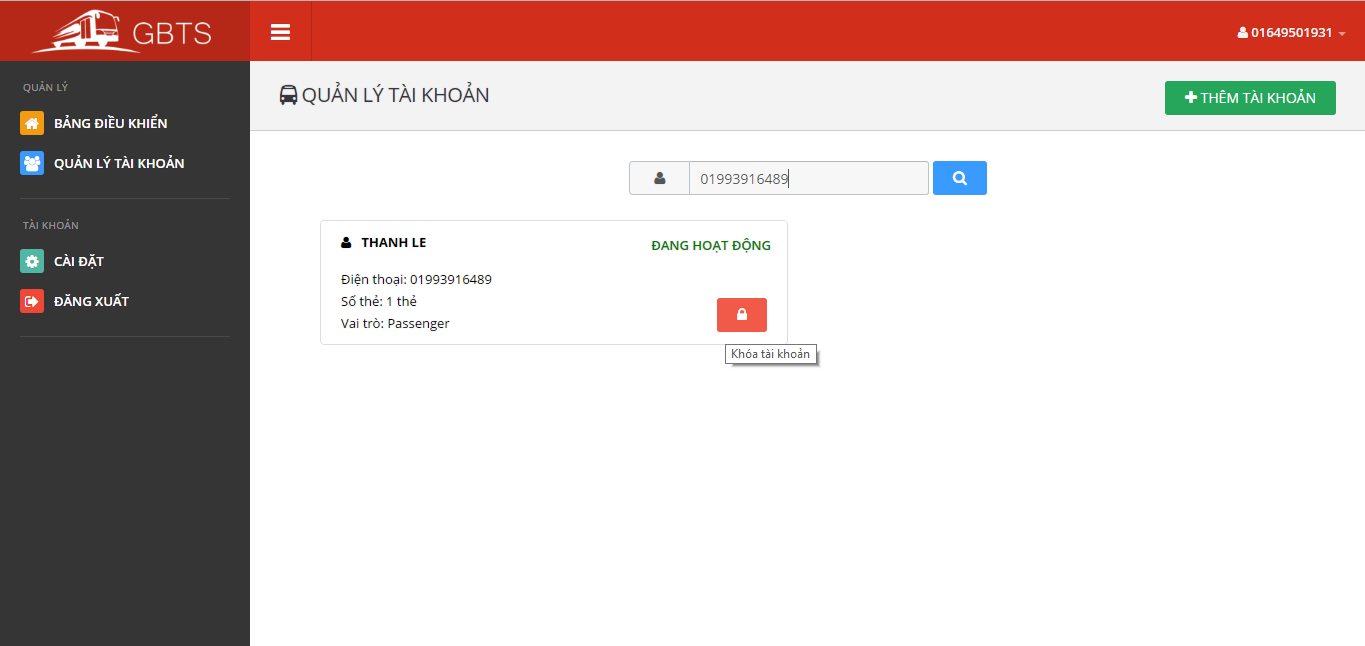
1.8 <Administrator> Add account

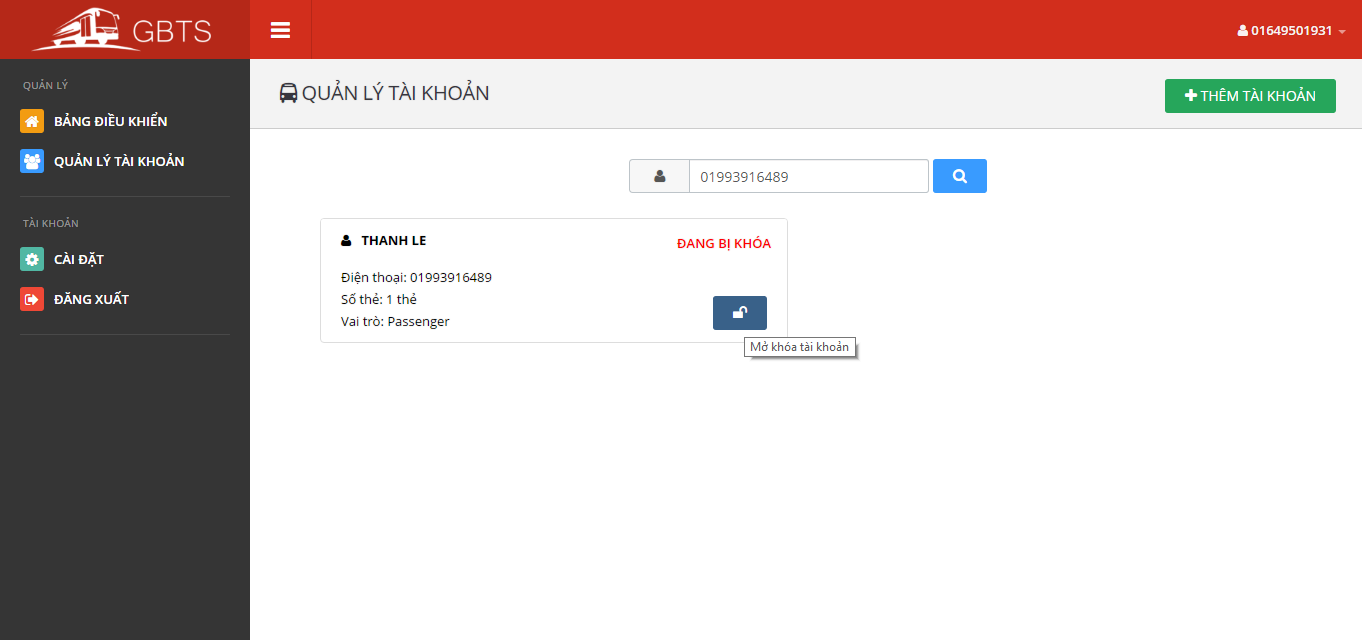




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý tài khoản” button |
| 2 | Click Thêm tài khoản” button |
| 3 | Input phone number and select one of the roles |
| 4 | Click “Lưu” button |

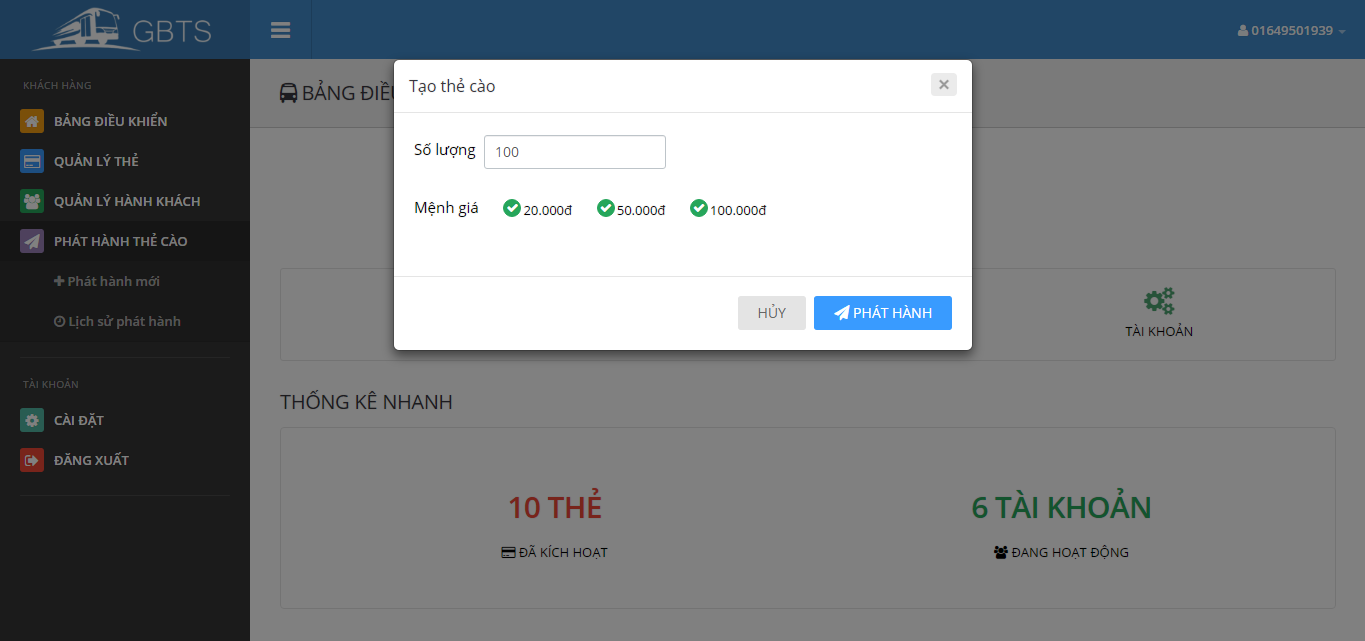
1.9 <Administrator> Activate/Deactivate account





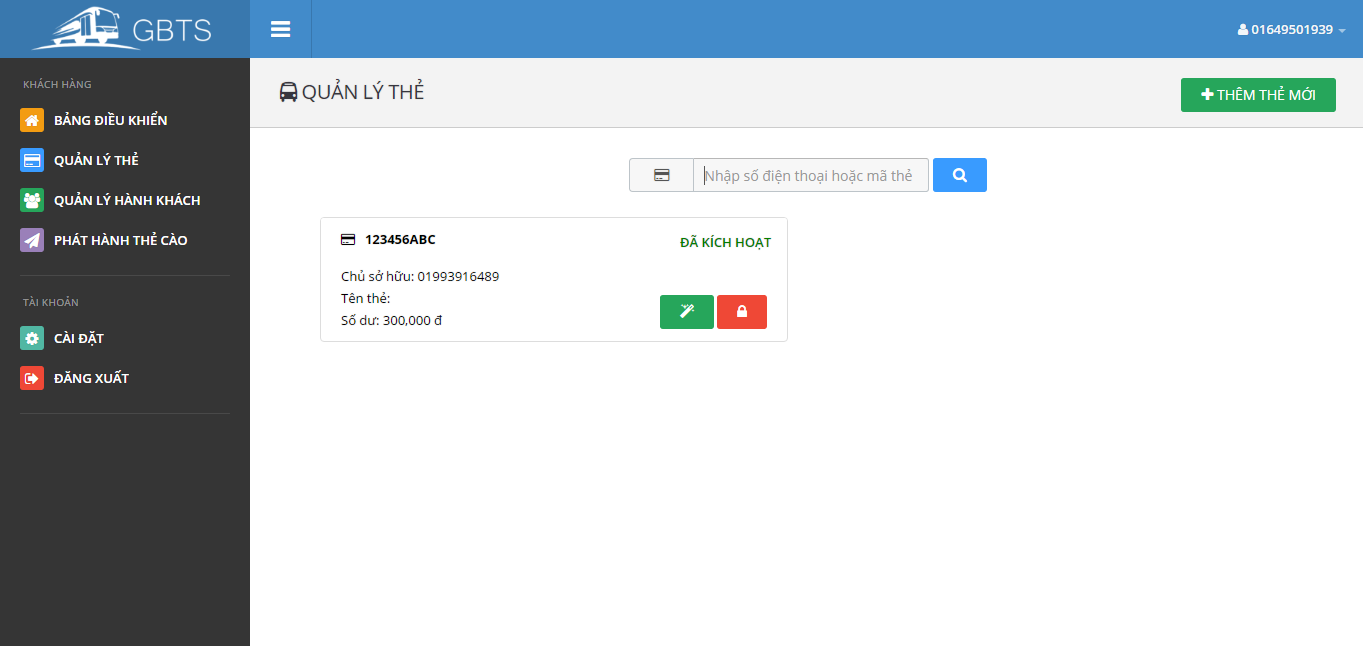
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý tài khoản” button |
| 2 | Input a phone number |
| 3 | Click “Khóa tài khoản” or “Mở khóa tài khoản” button |

1.10 <Staff> Publish scratch cards



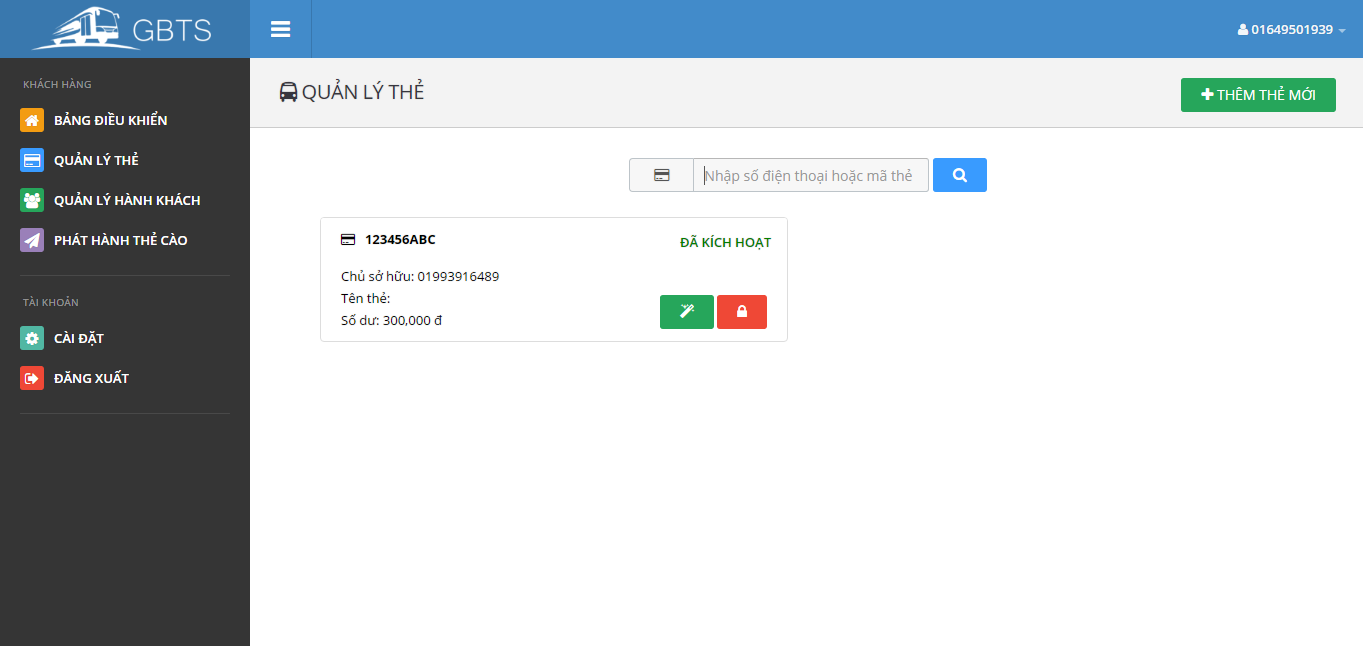
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Phát hành thẻ cào” button |
| 2 | Click “Phát hành mới” button |
| 3 | Input all information |
| 4 | Click “Phát hành” button |

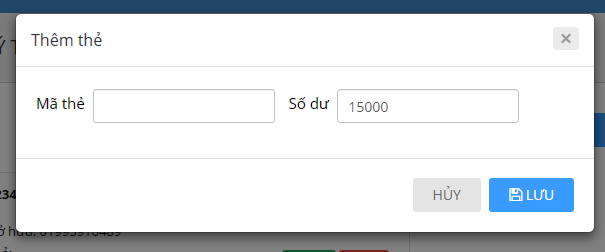
1.11 <Staff> Search card



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý thẻ” button |
| 2 | Input card ID |
| 4 | Click search button |

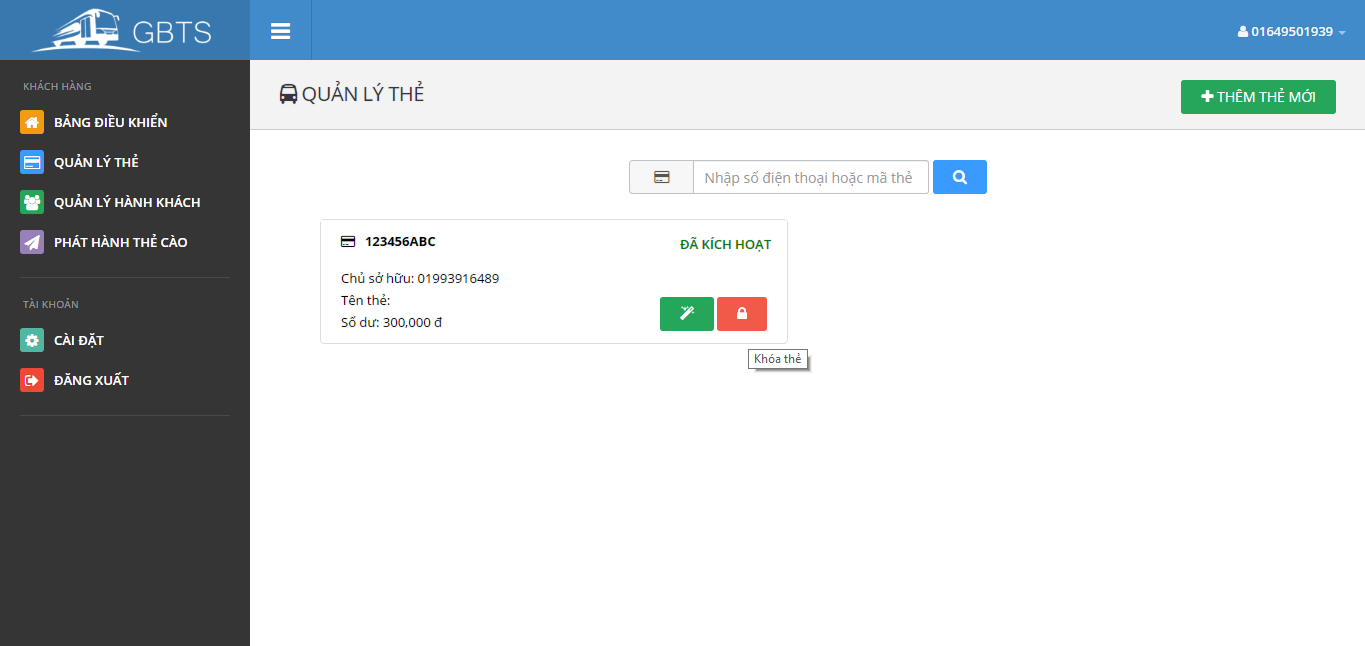
1.12 <Staff> Add card

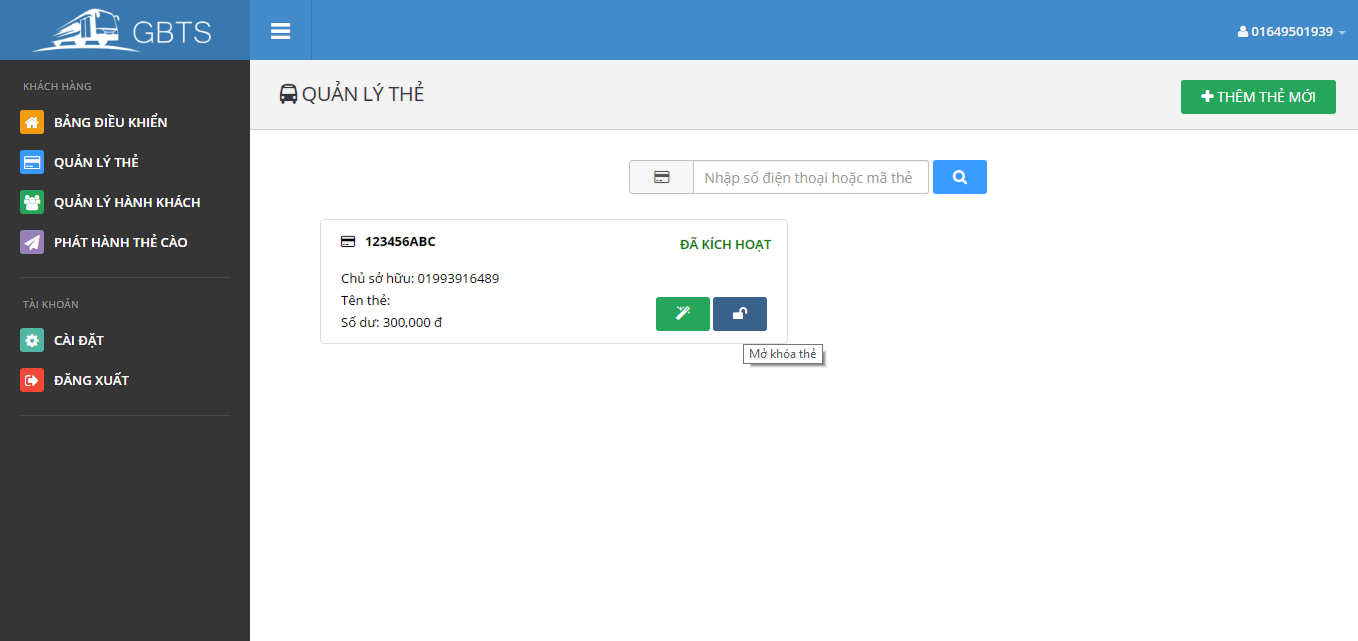




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý thẻ” button |
| 2 | Click “Thêm thẻ mới” button |
| 3 | Tap new NFC card on the NFC reader or enter card ID by hand |
| 4 | Input initial balance |
| 5 | Click “Lưu” button |

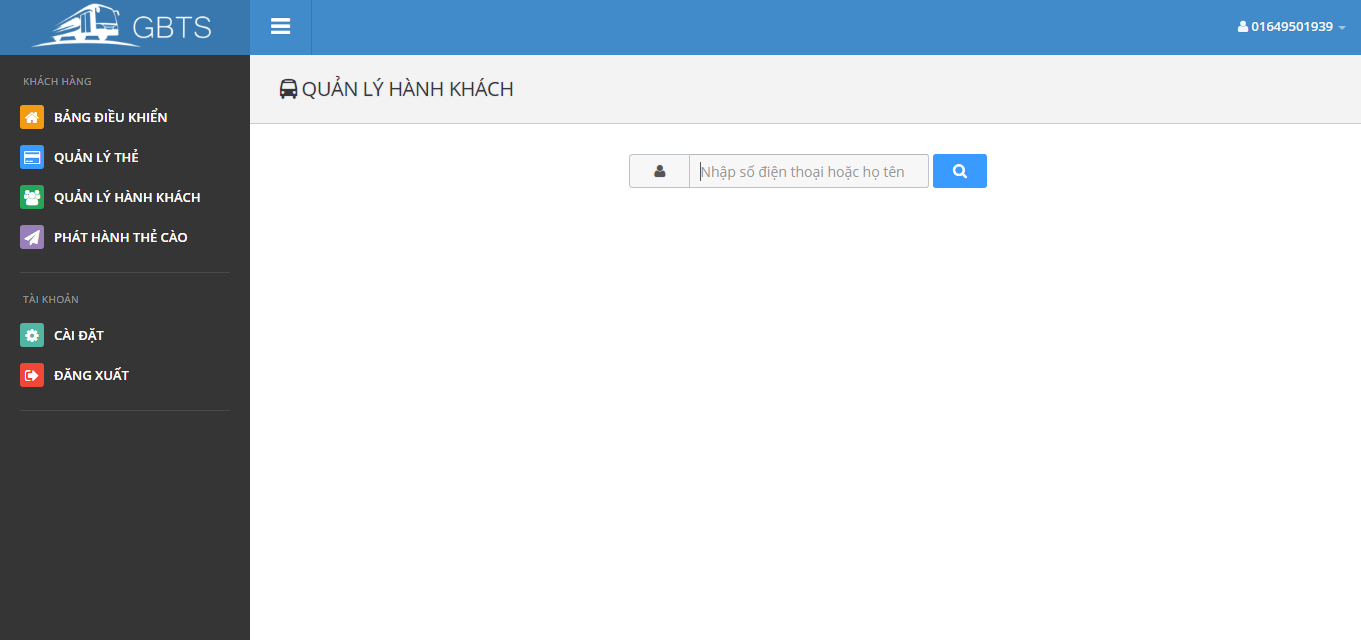
1.13 <Staff> Activate/Deactivate card





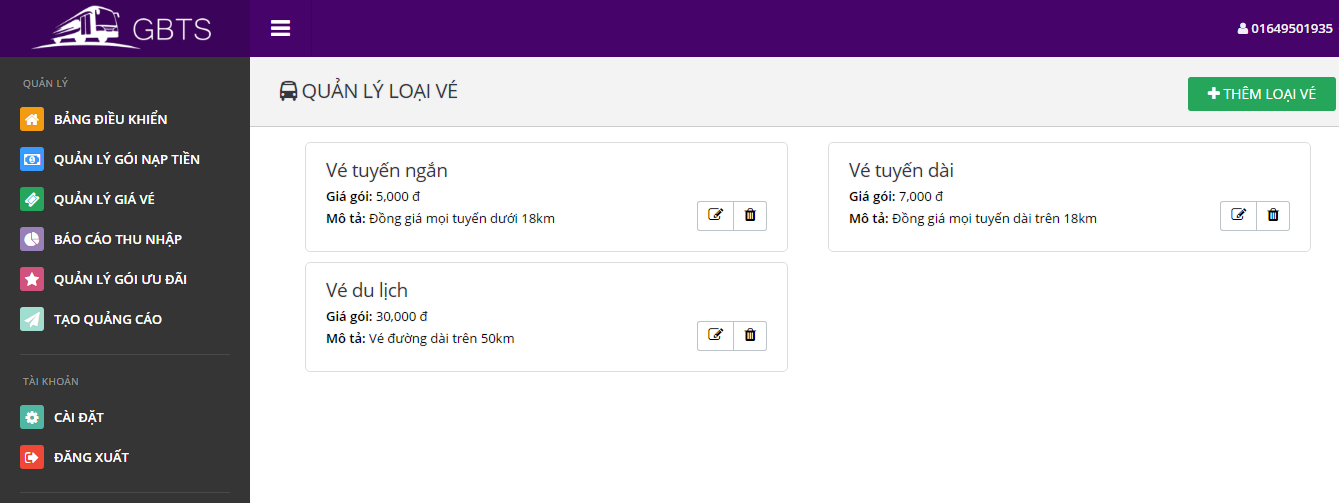
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý thẻ” button |
| 2 | Input card ID or a phone number |
| 3 | Click “Khóa thẻ” button or “Mở khóa thẻ” button |

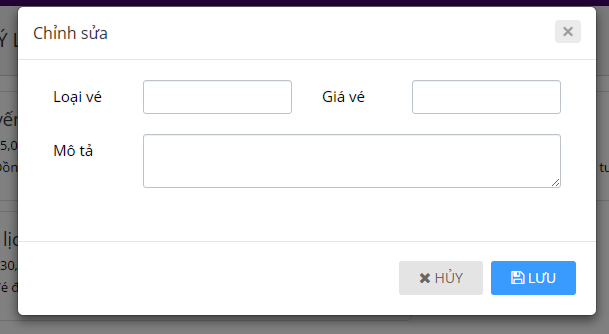
1.14 <Staff> Search passenger



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý hành khác” button |
| 2 | Input phone number or name |
| 3 | Click search button |

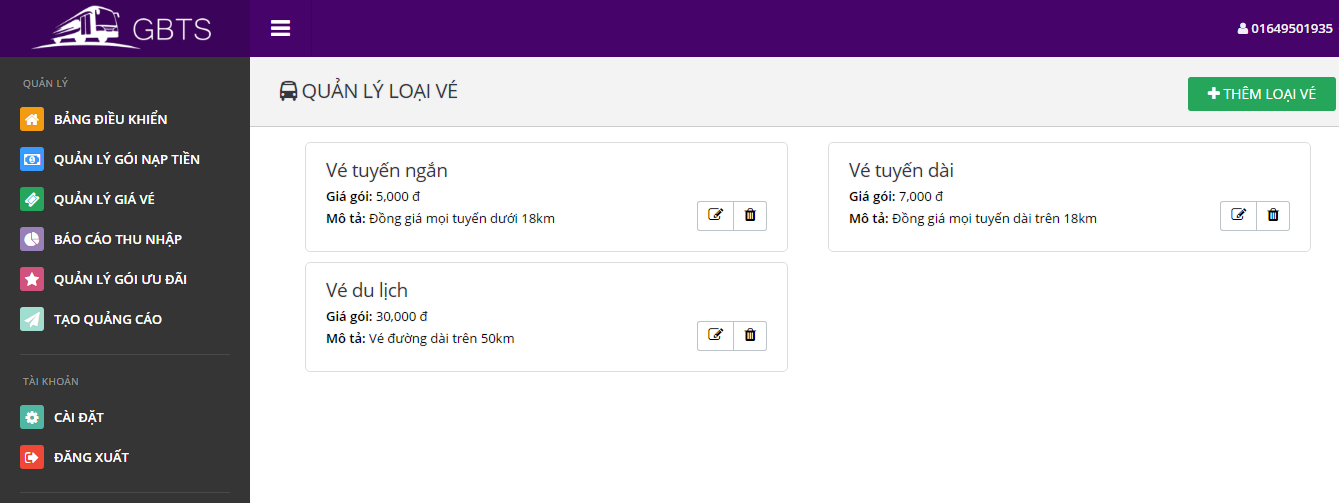
1.15 <Manager> Add ticket type

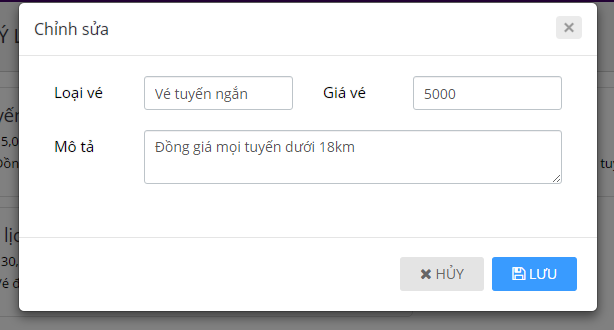




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý giá vé” button |
| 2 | Input phone number or name |
| 3 | Click “Lưu” button |

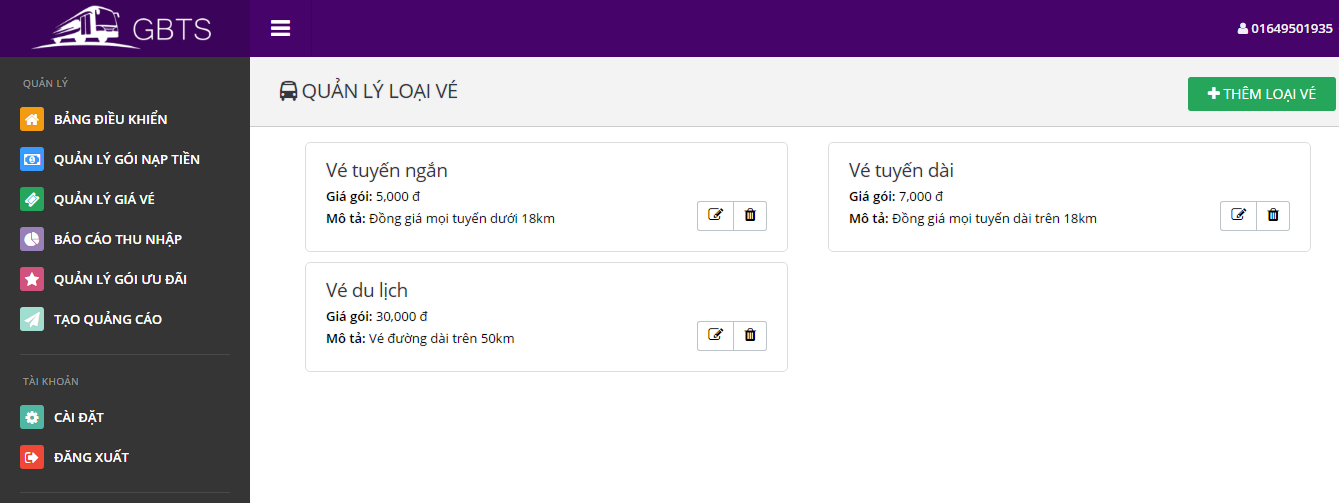
1.16 <Manager> Edit ticket type

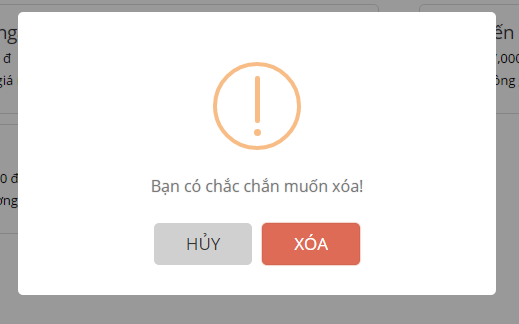




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý giá vé” button |
| 2 | Click edit button on a ticket type |
| 3 | Click “Lưu” button |

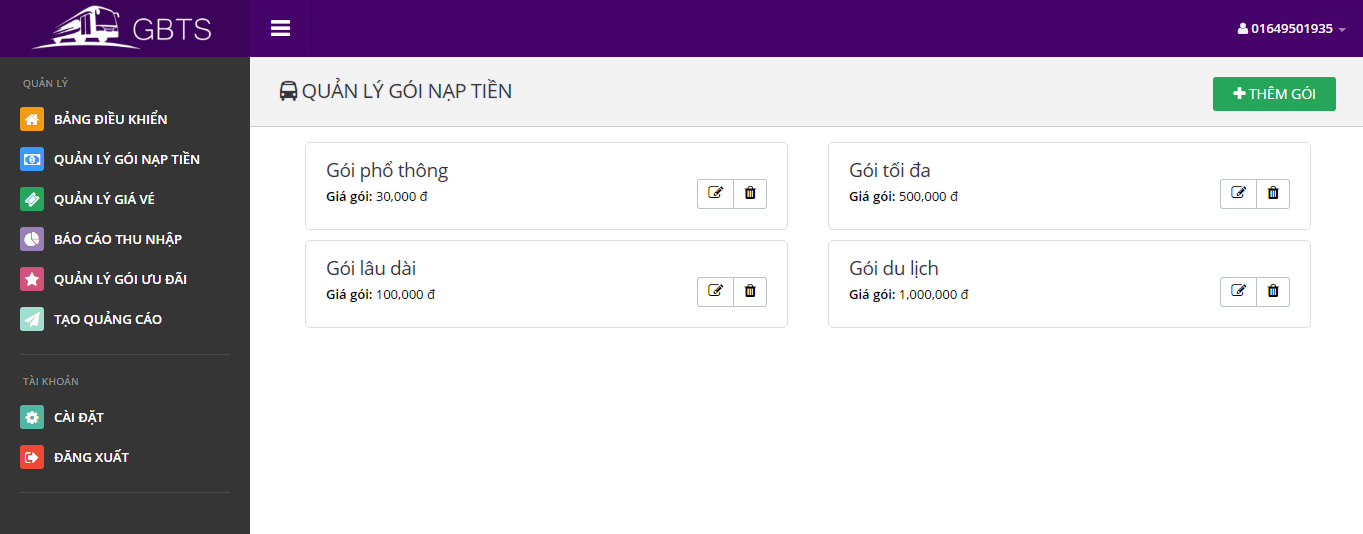
1.17 <Manager> Delete ticket type

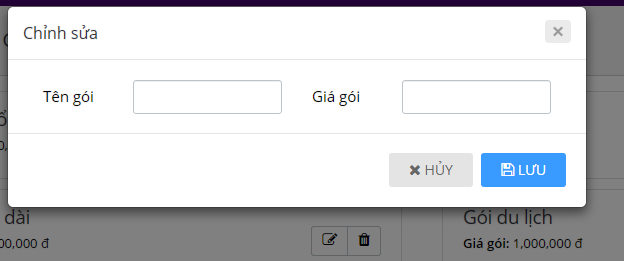




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý giá vé” button |
| 2 | Click delete button on a ticket type |
| 3 | Confirm by clicking “Xóa” button |

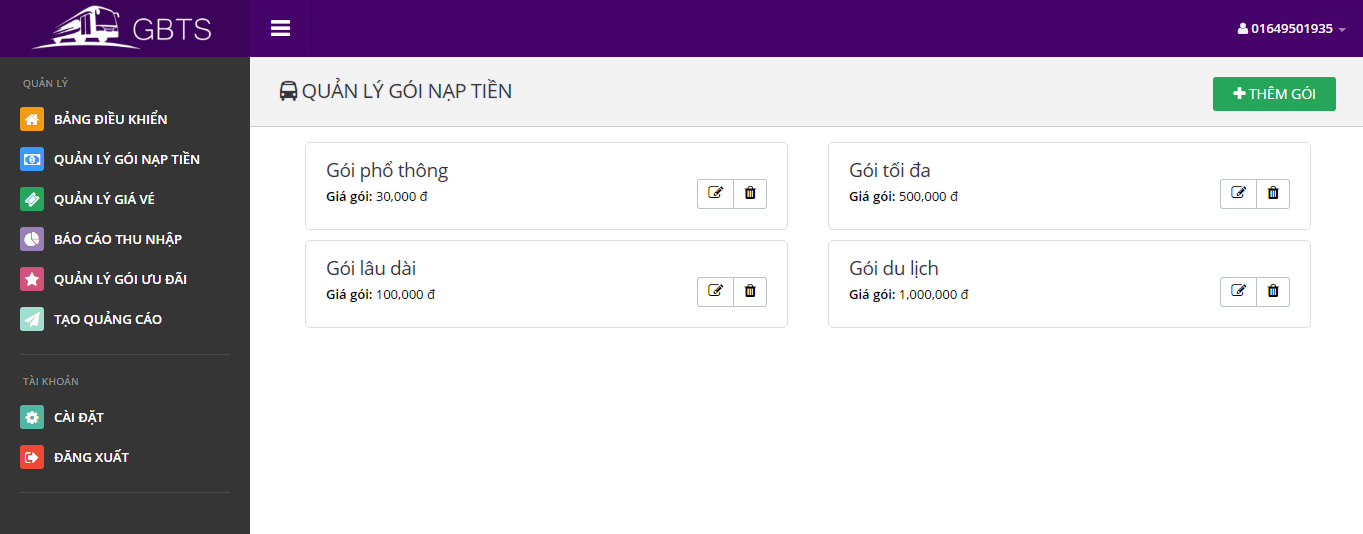
1.18 <Manager> Add credit plan

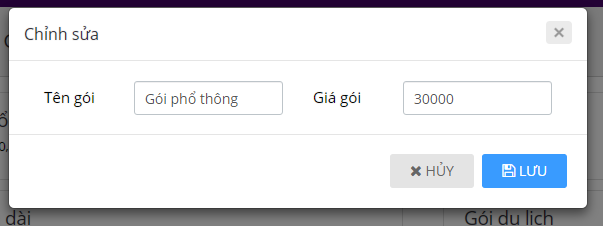




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói nạp tiền” button |
| 2 | Click “Thêm gói” button |
| 3 | Input new credit plan information |
| 4 | Click “Lưu” button |

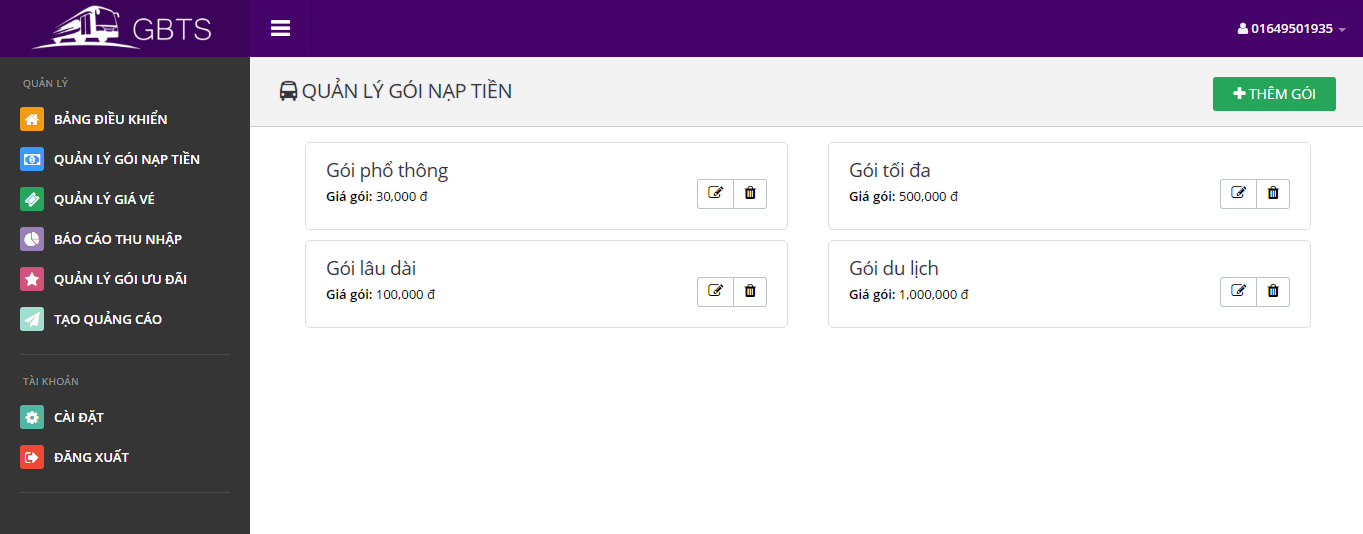
1.19 <Manager> Edit credit plan





|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói nạp tiền” button |
| 2 | Click edit button on a credit plan |
| 3 | Edit credit plan information |
| 4 | Click “Lưu” button |

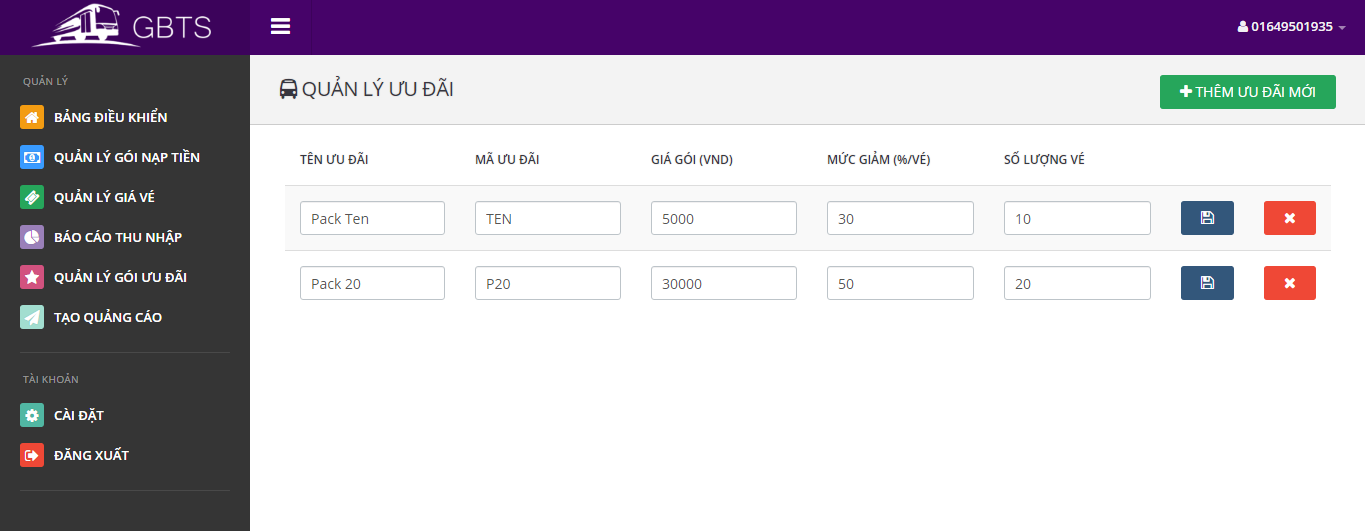
1.20 <Manager> Delete credit plan

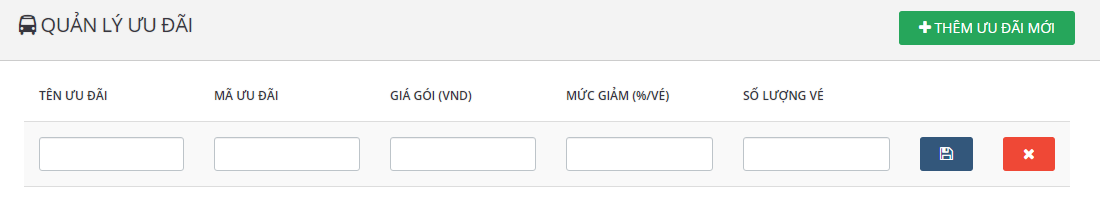




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói nạp tiền” button |
| 2 | Click delete button on a credit plan |
| 3 | Confirm by clicking “Xóa” button |

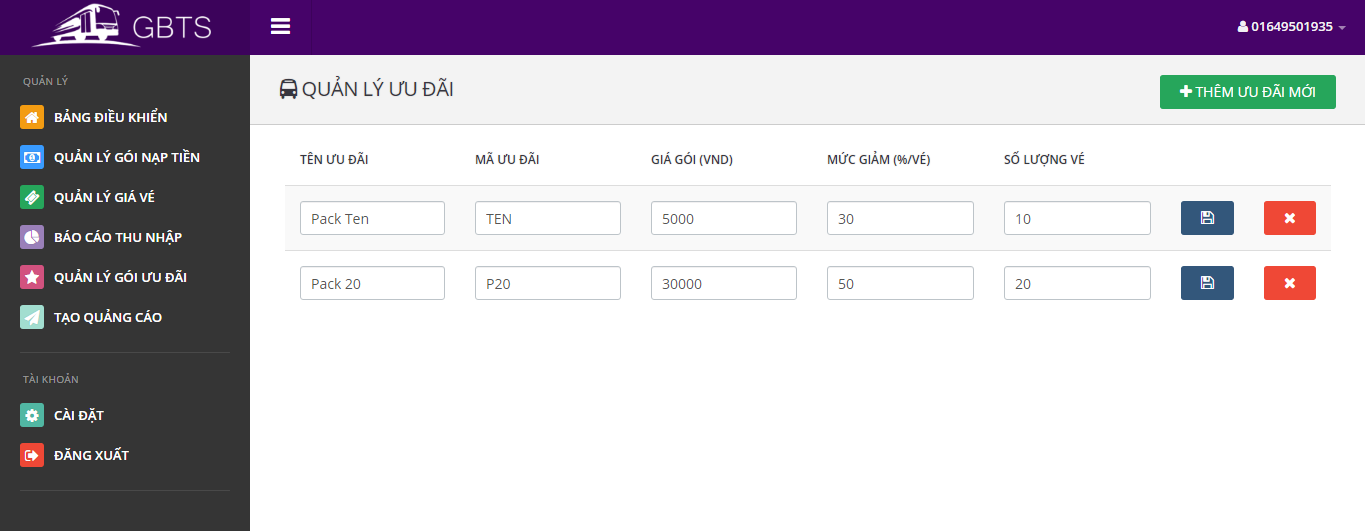
1.21 <Manager> Add subscription





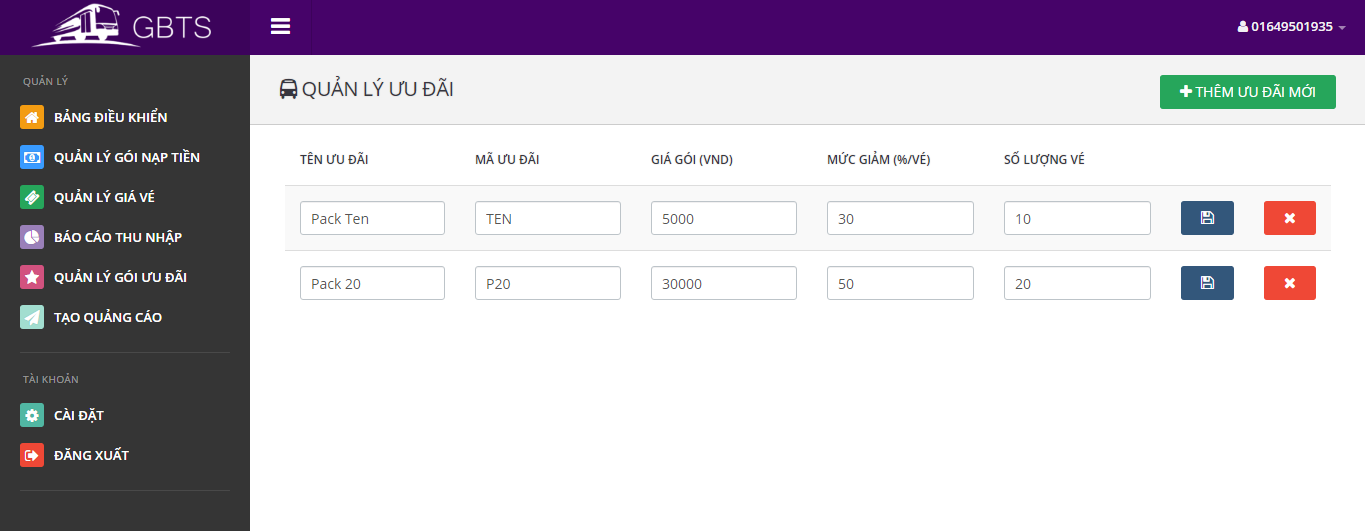
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói ưu đãi” button |
| 2 | Click “Thêm ưu đãi mới” button |
| 3 | Enter new promotion package information |
| 4 | Click save button |

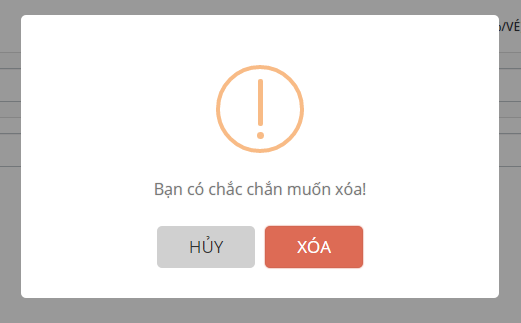
1.22 <Manager> Edit subscription



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói ưu đãi” button |
| 3 | Edit promotion package information |
| 4 | Click save button |

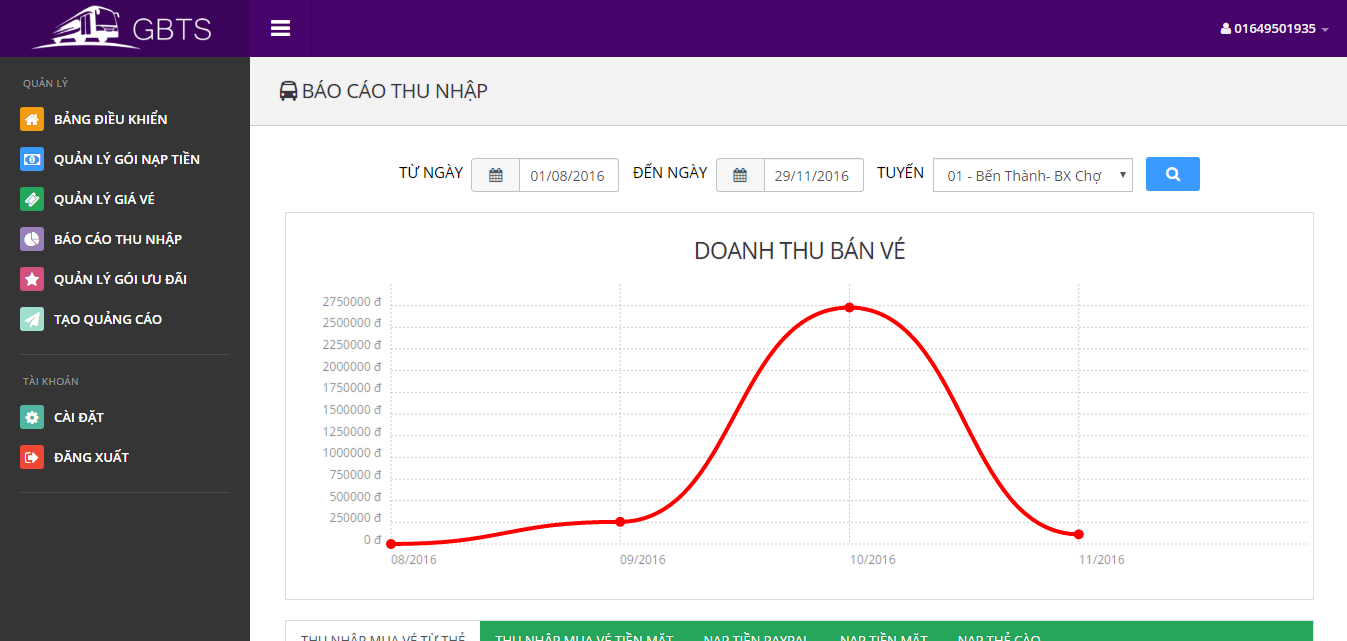
1.23 <Manager> Delete subscription





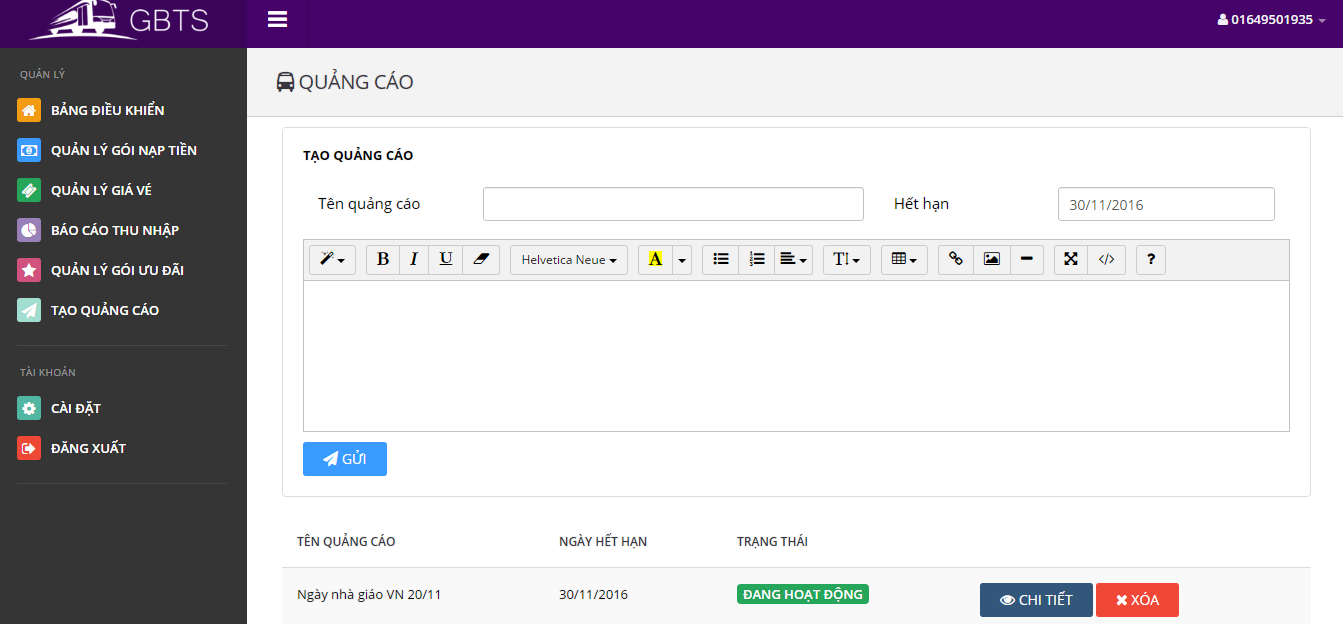
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Quản lý gói ưu đãi” button |
| 3 | Click delete button on a promotion package |
| 4 | Confirm by click “Xóa” button |

1.24 <Manager> Get income report



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Báo cáo thu nhập” button |
| 3 | Input query information |
| 4 | Click search button |

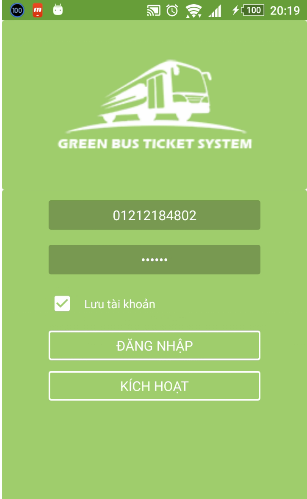
1.25 <Manager> Create promotion



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “Tạo quảng cáo” button |
| 3 | Input new advertisement information |
| 4 | Click “Gửi” button |

### Mobile application

#### 2.1 <Passenger> View Login

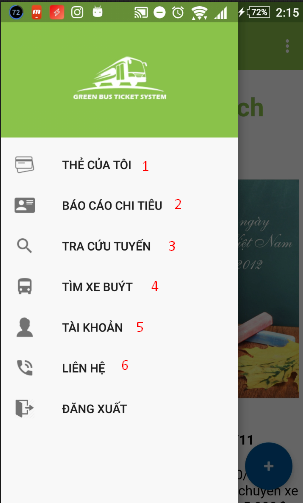


|  |  |
| --- | --- |
| Step | Description |
| 1 | Tab “ĐĂNG NHẬP” |
| 2 | Tab “KÍCH HOẠT “ |

#### 2.2 <Passenger> View Active Card

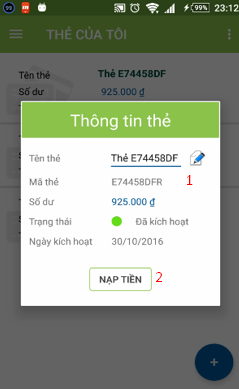
|  |  |
| --- | --- |
| Step | Description |
| 1 | Input your phone number, tap and hold to Active Card |

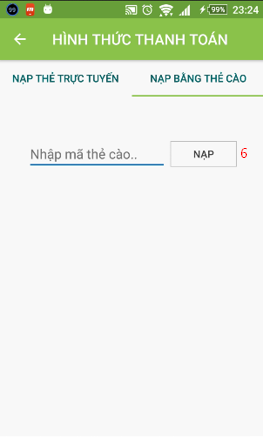
#### 2.3 <Passenger> Function Overview



|  |  |
| --- | --- |
| Step | Description |
| 1 | Tab “THẺ CỦA TÔI”. |
| 2 | Tab “BÁO CÁO CHI TIÊU” |
| 3 | Tab “TRA CỨU TUYẾN” |
| 4 | Tab “TÌM XE BUÝT” |
| 5 | Tab “TÀI KHOẢN” |
| 6 | Tab “LIÊN HỆ “ |

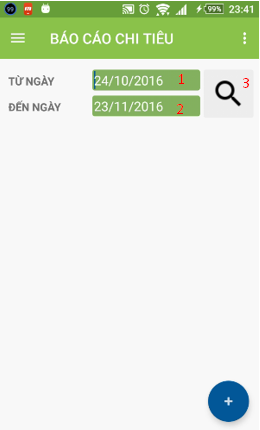
#### 2.4 <Passenger> THẺ CỦA TÔI

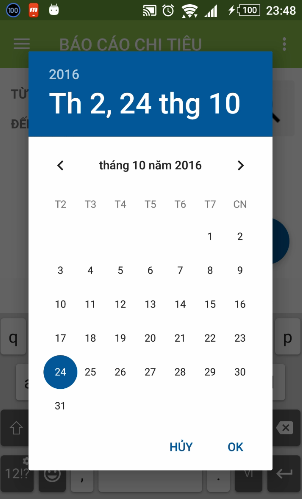




|  |  |
| --- | --- |
| Step | Description |
| 1 | Click one item of list card on fragment to view details and purchase. |
| 2 | Tab “ICON” to edit Name card. |
| 3 | Tab “NẠP THẺ TRỰC TUYẾN” purchase to recharge. |
| 4 | Tab “NẠP BẰNG THẺ CÀO”. |
| 5 | Click one item of list credit plan on fragment to recharge. |
| 6 | Tab “NẠP” to recharge. |

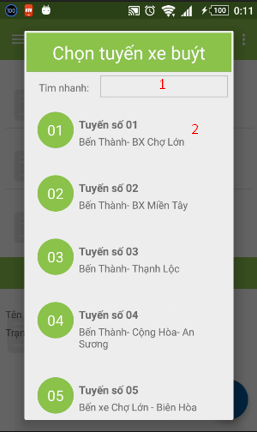
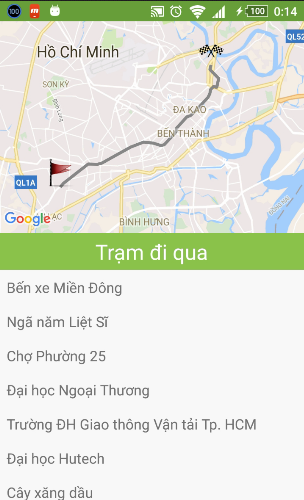
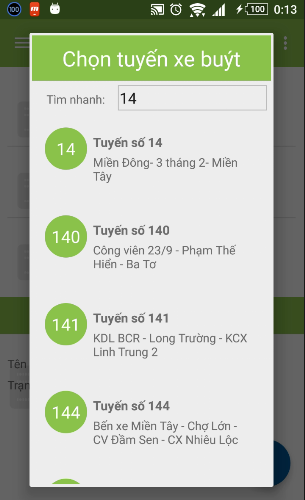
#### 2.5 <Passenger> BÁO CÁO CHI TIÊU





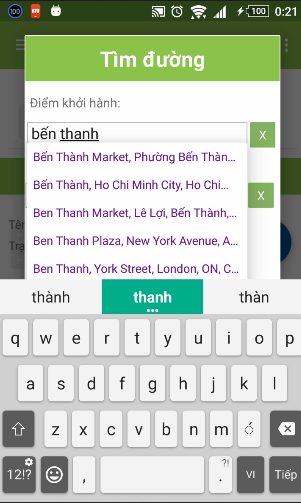
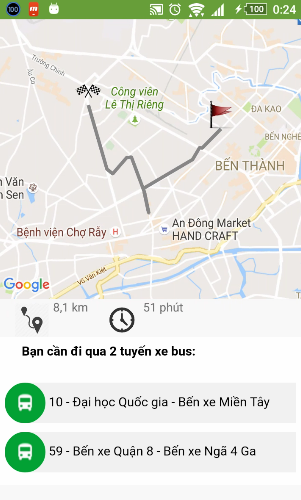
|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “TỪ NGÀY” to choose date. |
| 2 | Click “ĐẾN NGÀY” to choose date. |
| 3 | Click “ICON SEARCH” to get report. |

#### 2.6 <Passenger> TRA CỨU TUYẾN



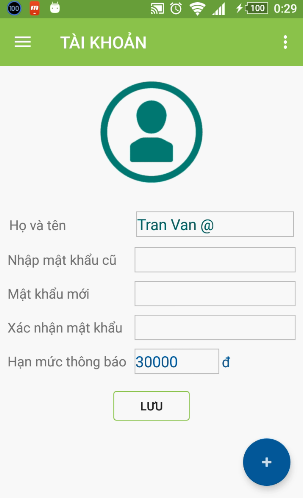
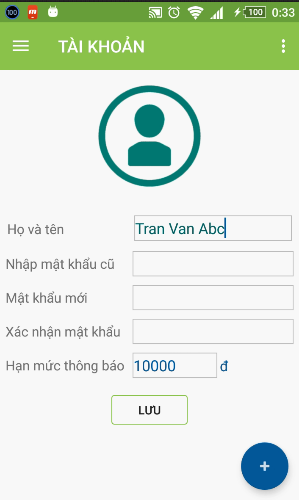
|  |  |
| --- | --- |
| Step | Description |
| 1 | Filler to fine bus routes |
| 2 | Click one item of list bus routes to find direction and bus routes. |

#### 2.7 <Passenger> TÌM XE BUÝT



|  |  |
| --- | --- |
| Step | Description |
| 1 | Choose origin direction |
| 2 | Choose destination direction |
| 3 | Tab “TÌM KIẾM” to find bus routes |

#### 2.8 <Passenger > TÀI KHOẢN



|  |  |
| --- | --- |
| Step | Description |
| 1 | Click “LƯU” to change profile. |

# Appendix

1. UML Documentation of IBM <http://www.ibm.com/developerworks/rational/library/769.html?ca=drs->
2. AES encryption

[http://](http://www.ibm.com/developerworks/rational/library/769.html?ca=drs-)www.java2s.com/Code/Java/Security/EncryptionanddecryptionwithAESECBPKCS7Padding.htm

1. Android Developer Guide – Application Fundamentals

<https://developer.android.com/guide/components/fundamentals.html>

1. FPT Vietnamese Accentizer

[http://doc.openfpt.vn/services/vnaccent/documentation.html#/README](http://doc.openfpt.vn/services/vnaccent/documentation.html%23/README)

1. Google Firebase Api

<https://firebase.google.com/docs/reference/>