|  |
| --- |
| **FPT-aptech computer education** |
| eProject Document |
| ONLINE BUS TICKET RESERVATION SYSTEM |
|  |
| |  |  | | --- | --- | | **Group 2** | | | **Group Member** | BaHL\_C00106<Eproject\_Code>  ThanhDV\_C00115<Eproject\_Code>  HungPV\_RollNo<Eproject\_Code>  TrungBD\_RollNo<Eproject\_Code>  DatLQ\_RollNo<Eproject\_Code> | | **Instructor** | ThiDK | | **Batch** | C0809G | | **Semester** | 3|4 | |
|  |

|  |
| --- |
| - Ha noi, 10/2010 - |

**THE TABLE OF CONTENTS**

# Introduction

*<Introduction to this document, how it is organize, what it contains>*

# Problem Definition

## Problem Abstraction

*< Write down the problem abstract. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## The Current System

*< Describe the current system\situation. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## The Proposed System

*< Describe the system under developed. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## Boundaries of the System

*< List the scope\boundaries of the System-under-developed. This can be paraphrase from the Customer’s Requirement Statement sent from India>*

## Development Environment

*<Describe the environment for system development. Include software and hardware requirements>*

# Requirements and Business Flow

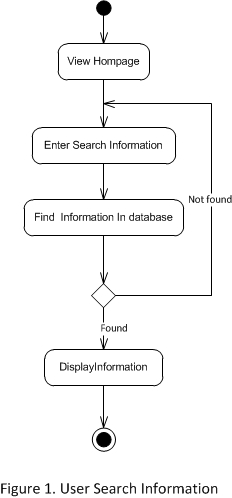
*<In this section you will provide the requirements you developed against. This section should be useful, so only provide enough information (requirements, diagrams) to help with the implementation of the software. Focus on the “non-obvious” aspects. If you are dealing with a business process, activity diagrams are good. Making a set of requirements that is compact, simple and covers all the key aspects of the software can. Don’t do the work if you don’t need it!>*

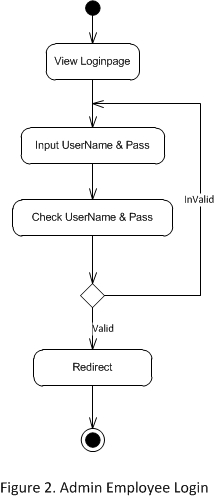
## Customer Requirement Specification

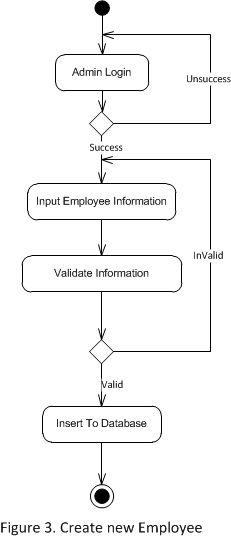
*<Summarize the customer requirement into the compact form>*

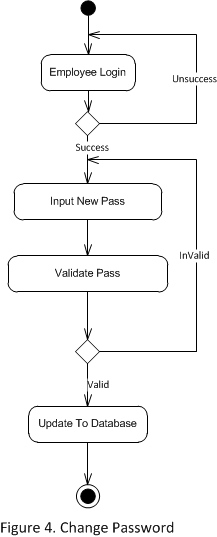
## Activity Diagram

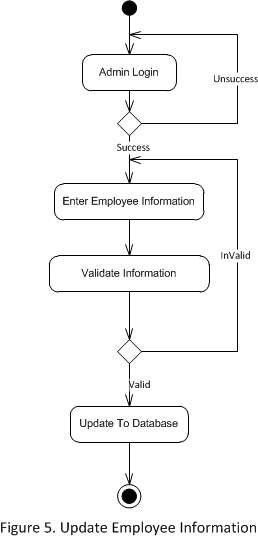
Thay doi



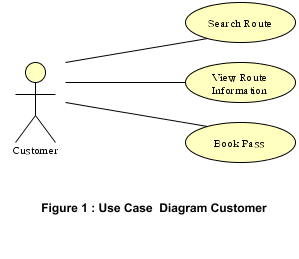


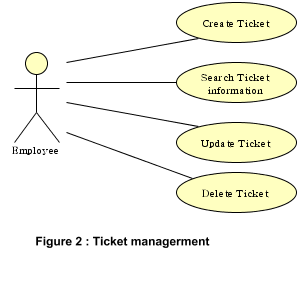


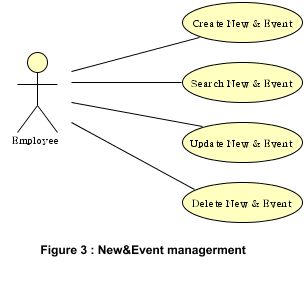


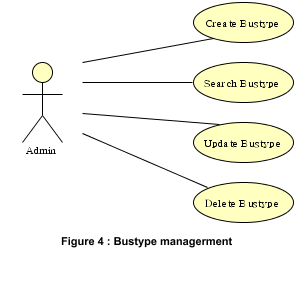


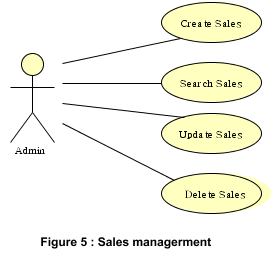
## Use Case Diagram

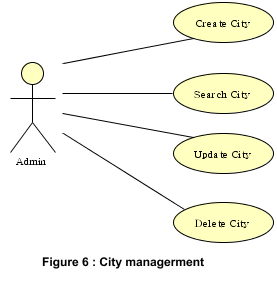


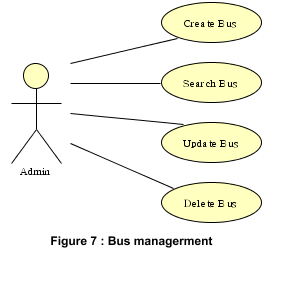


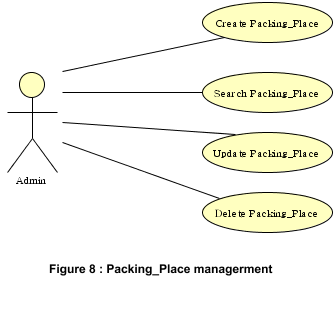


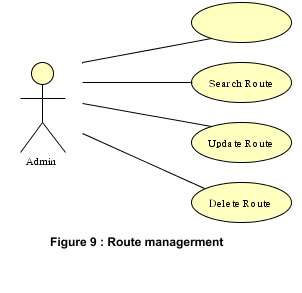


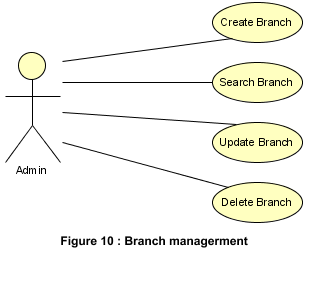


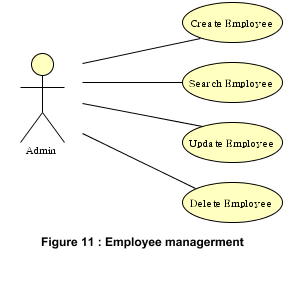












## Use Case Specification

*<Write down all non-trivial use cases. This should reflect what you get when your team does the system analysis. Use the template to write the detailed specification for use cases>*

*<Use case temlpate:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | <UC001> | **Use-case Version** | | <1.0> |
| **Use-case Name** | <Name> | | | |
| **Author** | <Members> | | | |
| **Date** | Dd/mm/yyyy | **Priority** | <High\Normal\Low> | |
| **Actor:**  <Lit all actors>  **Summary:**  *<Briefly describe the use case>*  **Goal:**  *<Briefly describe the goal of use case>*  **Triggers**  *<What leads this use case?>*  **Preconditions:**  *<List the required pre-conditions for this use case>*  **Post Conditions:**  *<List the required post-condition for this use case>*  **Main Success Scenario:**  *<List the main steps for this use case to reach the goal successfully >*    **Alternative Scenario:**  *<List the other steps for this use case to reach the goal in some alternatives condition >*  **Exceptions:**  *<list the exceptions of this use case>*  **Relationships:**  *<List the relationships that use case relates to>*  **Business Rules:**  *<Any concern about the business>* | | | | |

## Other Concerns<Optional>

*<You can list here all other concerns about the business or the requirements if needed>*

# Design

*[This section shows design of the system. This could be a part of the Developers Manual]*

## System Architecture

*<Explain and present the architecture of the system using texts or diagams>*

## Class Diagram

*<Provide class diagrams for the project>*

## Class Diagram Explanation

*<Provide brief explanation about the class diagram above. You do not need to explain “obvious” parts of your class diagram. For example, I know what a “Login” class is. Don’t say “The login class was created to store login information.”>*

## Sequence Diagram (Optional)

*<for important and complex interactions, protocols or algorithms, sequence diagrams should be drawn for clearing the details and supporting the system implementation. This section is optional>*

## Collaboration Diagram (Optional)

*<for important and complex interactions, collaboration diagrams should be drawn for clearing the details and supporting the system implementation. This section is optional>*

## State Diagram (Optional)

*<put all state diagrams here>*

## Entity Relationship Diagram

*<Provide the ERD Diagram for the system here. If your team uses file or in-memory storage facility instead of database, replace this section by ‘Data Structures’>*

## Database Design

*<Provide the detailed database design for the system here. If your team uses file or in-memory storage facility instead of database, remove this section, use the ‘Data Structures’ section >*

## Algorithms (optional)

*<Provide the detailed description about algorithms used in the system. You can use Flow Chart or Activity Diagram to represent algorithms. Focus on the important and complex algorithms>*

## Others (optional)

*<Any design concerns or diagrams can be put here>*

# System Prototype

*<Put the system prototype or mock UI here. Focus on* ***important forms*** *and the* ***screen flows*** *between forms. If you use RAD and .NET, prototyping really help you to reach your development goals quickly>*

# Management and Project Planning

## Management Approach

*< Briefly describe the management approach that your team selects. Is your team self-managed or managed by one leader? How do you assign tasks to team members? How often do you meet? What do you do during meeting? Etc.>*

## Project Plan

*<The detailed project plan is put here. You can use WBS Excel sheet, Sprint Backlog (see Sprint Backlog\_Template.xls), Task sheet, Gantt chart, etc. to present your team’s plan. You can capture the Gantt chart in PMS if you use it to plan your project>*

## Task Sheet

*<Write down the tasks in Task Sheet-compatible format, this Task Sheet works as the activity report of the projec or the plan of the project (not recommended); see eProject Guide for detailed Task Sheet>*

## Meeting Minutes (Optional)

*<Put all minutes of your team meetings here>*

# Checklists

## Check List of Validation

*< Put the checklist here; describe how it is used and the resulted checklist>*

## Submission Checklist

*< Put the checklist here; describe how it is used and the resulted checklist>*

# Screenshots

*<Capture some intuitive and main screens of the software and put them here>*

# Coding Convention

*<Provide the coding convention for your team. If you simply want to use the existing code standard(s) such as ‘Java Code Convention’, you can refer to it\them by name or URL>*

# Other Concerns<Optional>

*<If you have any other information you want to add to this document, place it here. This could include thoughts on the eProject, improvements, etc.>*

# Appendix

## Glossary [Optional]

*<Place all definitions or abbreviation used in this document >*

## References [Optional]

*<Place all referenced materials used in this document >*

## Others<Optional>