Online Bus Ticket System

System Design

V:1.0

08.12.2017

Didar Damla Turanlı

Hatice Dilara Turanlı

Ayşenur Bayram

Anıl Aydıner

Prepared for

SE301 Software Engineering



Table of Contents

[1. Introduction 1](#_Toc433996772)

[1.1. Purpose of the System 1](#_Toc433996773)

[1.2. Design Goals 1](#_Toc433996774)

[1.3. Definitions, Acronyms, and Abbreviations 2](#_Toc433996775)

[1.4. References 2](#_Toc433996776)

[2. Current Software Architecture 2](#_Toc433996777)

[3. Proposed Software Architecture 2](#_Toc433996778)

[3.1. Overview 2](#_Toc433996779)

[3.2. System Decomposition 3](#_Toc433996780)

[3.3. Hardware Software Mapping 5](#_Toc433996781)

[3.4. Persistent Data Management 5](#_Toc433996782)

[3.5. Access Control and Security 6](#_Toc433996783)

[3.6. Global Software Control 7](#_Toc433996784)

[3.7. Boundary Conditions 7](#_Toc433996785)

[4. Subsystem Services 9](#_Toc433996786)

[5. References 12](#_Toc433996787)

SYSTEM DESIGN DOCUMENT

# Introduction

Design is the abstraction of the solution of a problem in general terms and it is also a visual transfer of a developed application.Successful completion of the design allows to save the time spent in the application part.

## Purpose of the System

The OBTS system’s main purpose is to provide quality,fast and secure service to users.The system has some usefull features that can help user to buy ticket and booking easily.Users can buy tickets with just a few clicks.Users do not have to register to buy tickets,but register operation is necessary to win a gift ticket,delete ticket and booking.Furthermore,system is completely free.

## Design Goals

This system should be a system that encourages users to make fast and reliable transactions.

* **Security**

One of the most important non-functional requirements of this system is system security.

* **Reliability**

The system has to continue online operations without error.This developed system should be highly reliable and all information belonging to user must be retained.

* **End User**

Users entering this system as visitor can not make reservation and win gift tickets that registered users can make.Therefore,visitors only view voyage and buy ticket.

* **Performance**

The system must have a robust structure for storing and managing the data that is entered by users.

* **Flexibility**

System requires internet and available web browser with the device.

* **Implementation**

System can be modified and readable.The number of errors must be minimized

## Definitions, Acronyms, and Abbreviations

**PHP**: Hypertext Preprocessor

**SDD**: System Design Document

**MVC**: Model View Control

**OBTS:**Online Bus ticket system

## References

References to existing systems, etc.

# Current Software Architecture

Nowadays,many companies have their own online bus ticket systems.OBTS system has two different login part for admin and registered user.In this system other one user is visitor.Registered user do some operations such as view vacation,buy ticket,delete ticket and win a gift ticket.Visitor only buy a ticket and view vacation on the OBTS system.This system is being made by using PHP.Also,we use PHPMYADMİN to create database.

# Proposed Software Architecture

In OBTS system,our proposed model is Model-View-Controller(MVC).This architectural system supply all the goals and requirements.In our system,entity object and data will be in model and controller can be called bridge.Thats mean,communication and interaction between model and view provides by controller.

## Overview

The OBTS system is user friendly system and all user can use the system easily.OBTS system has many subsystems;

**User Interface:** It provides services for all users(visitor,admin and registered user)for common interfaces like a login form,password change form,personal form and etc.

**Admin Interface:**It provides services to display admins’ forms.For example,addBus form,deleteBus form and addVoyage form.

**Visitor Interface:** It provides services for visitor,such as view vacation subsystem.

**RegisteredUser Interface:** It provides services for registered user,such as booking,buy ticket subsystem.

**AddVoyage subsytem:** It provides services for admin to add voyage.

**DeleteVoyage subsytem:** It provides services for admin to delete voyage.

**ViewVoyage subsytem:** It provides services for visitor and registered user to view voyage.

**AddBus subsytem:** It provides services for admin to add bus.

**DeleteBus subsytem:** It provides services for admin to delete bus.

**Payment subsytem:** It provides services for visitor and registered user to buy ticket.

**Register subsytem:** It provides services for visitor to register the system

**Login subsytem:** It provides services for admin and registered user to login the system.

## System Decomposition

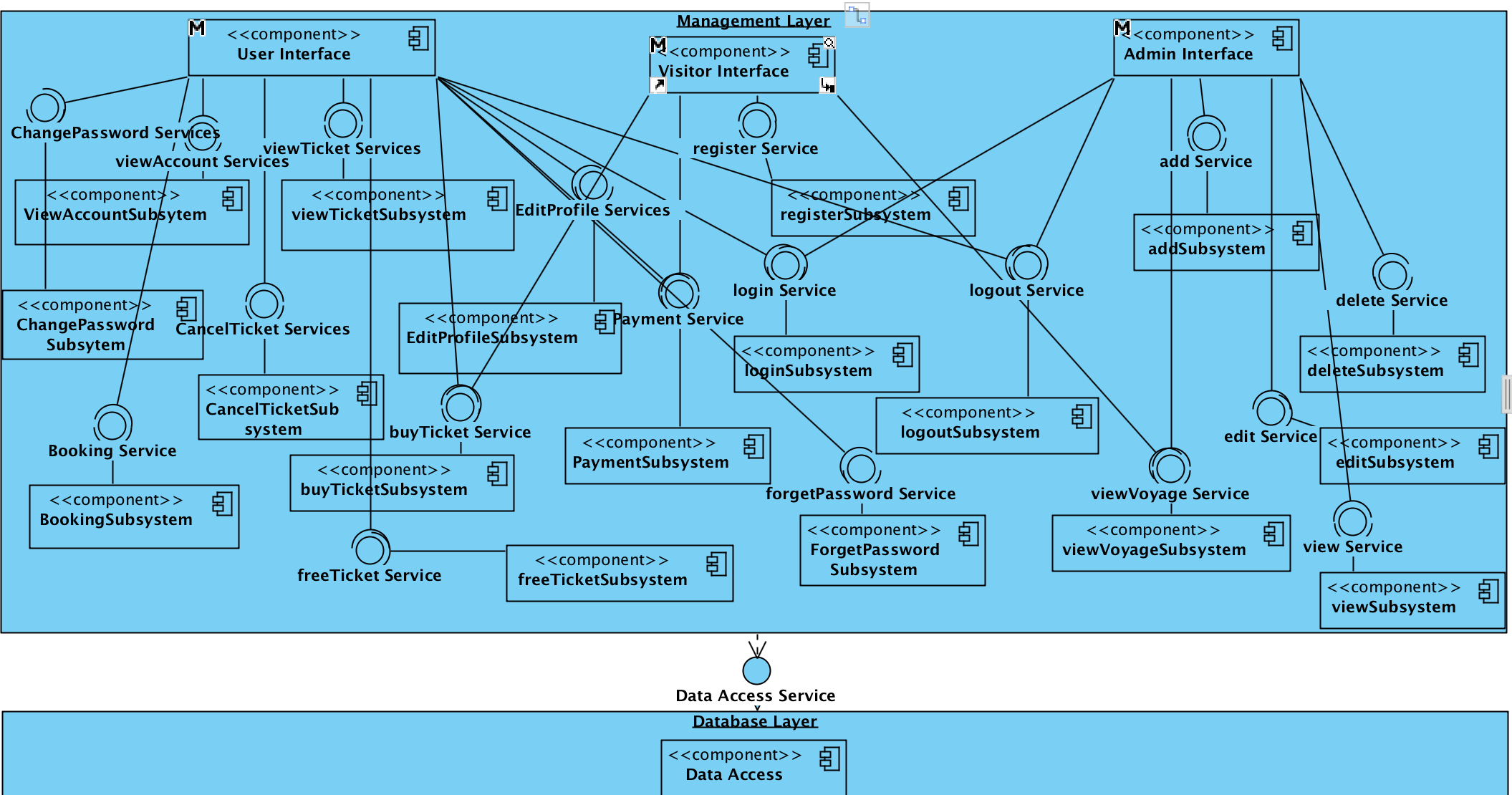
****

Figure1: Coupling view of Subsystem Decomposition

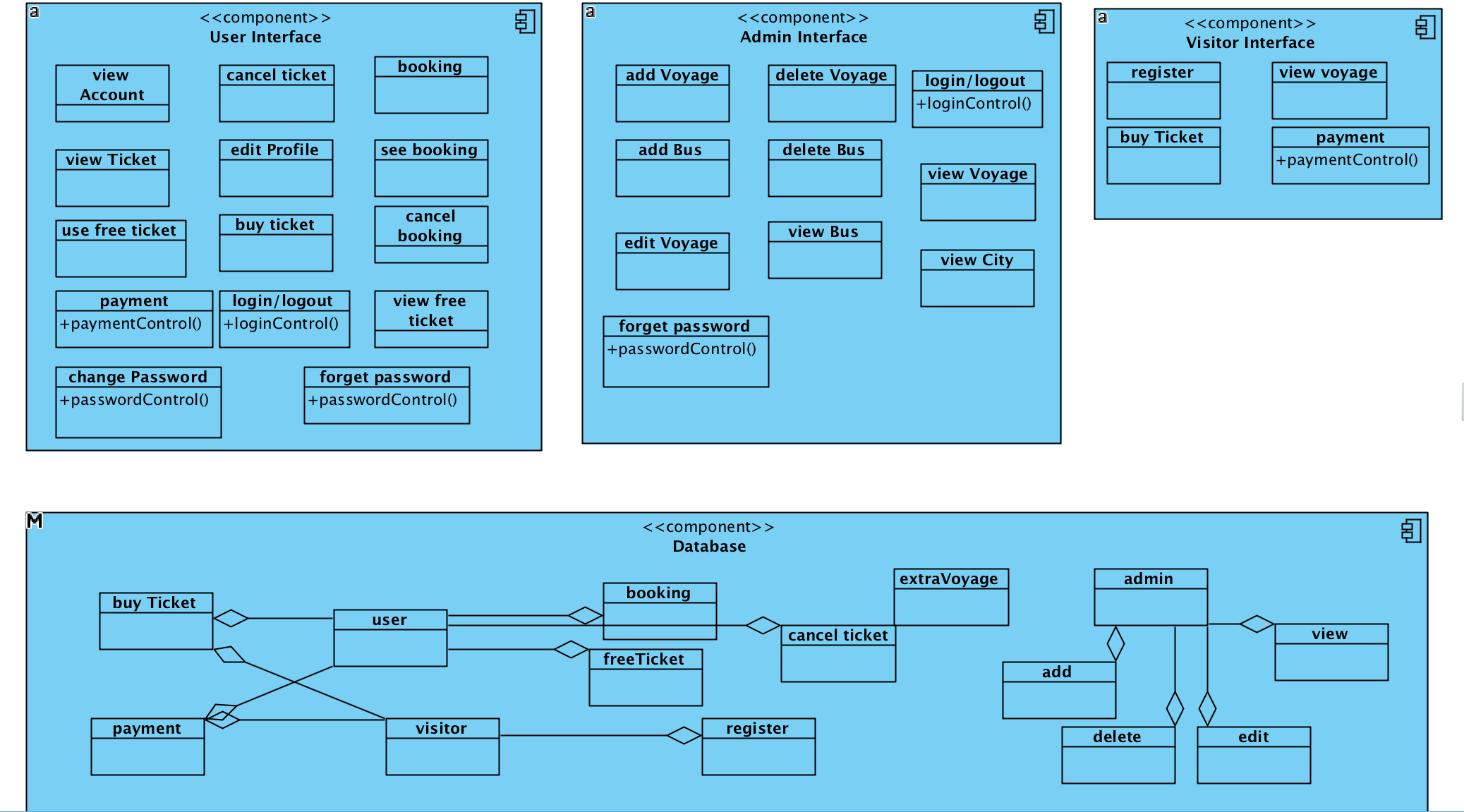
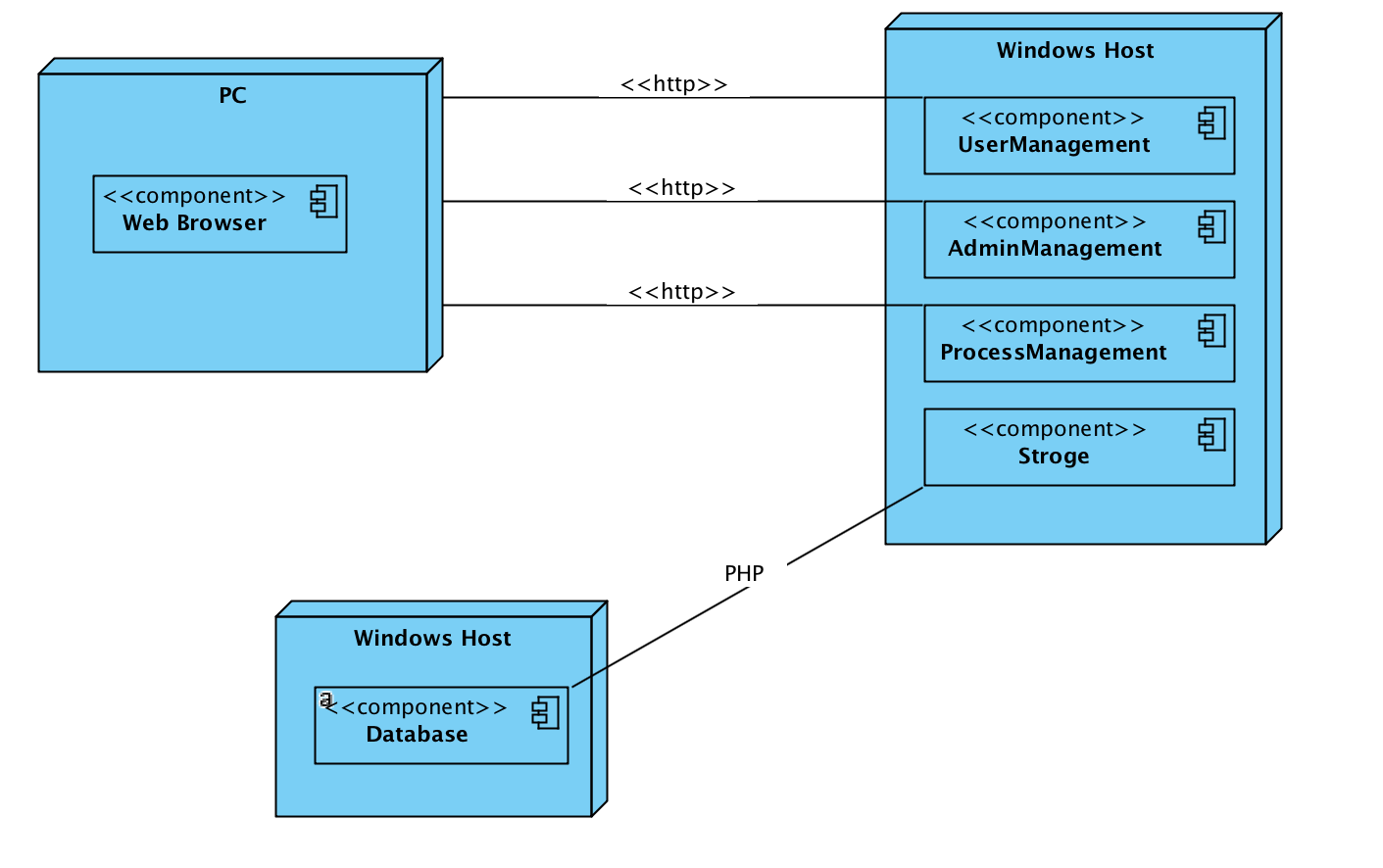
****

Figure 2: Cohesion view of Subsystem Decomposition

## Hardware Software Mapping



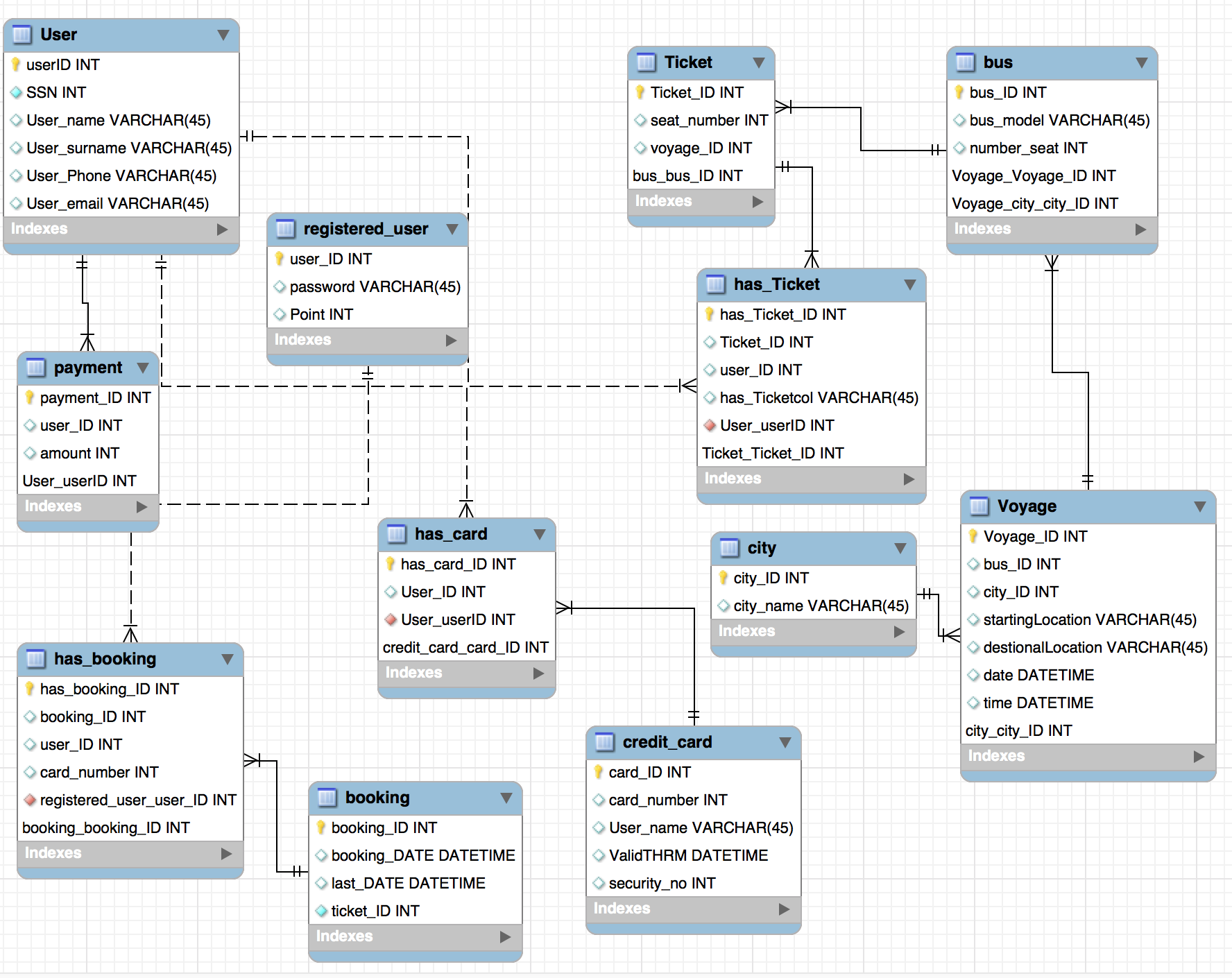
## Persistent Data Management

In Online Bus Ticket system, we decide to store persistent data in a database. The persistent datas are; Users of the system (Visitor, Registered user, Admin),Ticket data which may be recorded a ticket\_id, seat\_no, voyage\_id. There are relation between user and ticket. . For each user there are some ticket. Voyage is stored in the system as well . There are relation between voyage and bus. For each voyage there are some bus.Also a different relation about voyage is between voyage and city. For each voyage there are some bus. . Bus is stored in the system as well. There are relation between ticket and bus. For each bus there are some ticket.

We store booking in RegisteredUser data. For each registereduser there are some booking. guests’ rate/comment data is stored in the system as well.

These are the persistent data that are need to be stored in database so that the data can outlive a single execution of the system. In OBTS, because of its advantages and suitability for the OBTS system, we decided to use relational database for the data management infrastructure. Some of the advantages of choosing a relational database ;Ease of use, Abstraction, Security,Flexibility, Precision,,Data Independence, Data Manipulation Language.

Here Online Bus Ticket’s ER diagram:



## Access Control and Security

This system is a multi-user system which contains two users and these users are visitors and registered users.The system provides different interfaces according to the functions of these users in the system. First, the administrator will connect to the system through the login interface and perform some operations such as adding, deleting and editing etc.. Only admin access the database and all datas that related to the system will be manually entered into the database. All users who will use the system will register themselves on the system. The registration process is carried out by user interfaces and the relevant data is entered into the registration forms. All entered data will be saved in the database and data will be exported from this database. Users who want to take advantage of all the privileges the system has are required to register with the system and perform their operations after logging into the system. Users will login to the system with their ssn numbers and passwords. During registration, field filling does not require access to the database, while completion of the process requires the data to be written to the database, which requires read and write access to the database. In that case, the required database fields will be blocked and simultaneous access of multiple users will be denied.Also,in this database system, users'

ssn numbers and passwords will be stored in the user table. When this information is entered by the registered user, the information is checked by authentication interfaces and if the user information is correct, it is directed to its own page.

## Global Software Control

OBTS is thread safety. In other words, our system OBTS is multi-threaded program. The reason we have optimized BTS for multi-threading is that OBTS should provide services for many users at the same time, and also some functions of OBTS should be synchronized. The systems’ dynamic control distributed among different controllers such that each page delegates some responsibility to other pages. In addition, threads are more intuitive, they currently introduce many problems during debugging and testing. For instance, in the mentioned design, remaining seat count and number in a bus is being refreshed when any reservation is completed or cancelled. Therefore, the user accesses to the most recent data every time.

## Boundary Conditions

OBTS is activated by the system admin calling the ‘Initialize System’ use case. Once the initialization of the system is completed, the users are register onto system. Moreover, the system admin activates the server by calling the ‘Start Server’ use case. Furthermore, when the system is brought from non-initialized state to steady-state, the system is opened and free for the system users (User, Visitor and Admin) to login and perform their tasks. While the users of the system are online and perform their tasks to accomplish, there might be some errors occurring during the transaction processes of the tasks that performed by the users.

The mentioned errors could be originated from attempting to login with invalid credentials, trying to update the personal information (which are necessary to be filled) with empty information. All these exceptions are get caught by the system and handled properly.

For instance, for attempting to login with invalid credentials error, the use case name ‘Invalid Credentials’ is called to handle the exception. When the ‘Invalid Credentials’ use case is invoked, the system has already checked the user’s ssn number and password that if the user ssn and password matches with the ssn and password in the database of the system. The system realizes that they do not match with each other, then in the process of the ‘Invalid Credentials’ use case, the user, who tries to login, is being informed with a proper message indicates that “SSN or Password is Incorrect”. Thus, the system lets user to know that the credentials that the user uses to login in not correct so that the user can realize the situation and try again with the correct one. For trying to update the personal information (which are necessary to be filled) with empty information exception, the exception occurs when the current logged user tries to update his/her personal information but supply empty fields which are not supposed to be empty. Then OBTS performs a checking to make sure that the necessary information fields are given and the information fields are proper to be updated. However, the system realizes that the necessary fields were not supplied, so the use case name ‘Missing Fields’ is invoked to handle the exception. When the ‘Missing Fields’ use case is invoked, a proper message (the message indicates the necessary fields which are not supposed to be empty, but left empty by the user) is display to the logged user to fill the necessary fields to accomplish the update task.

When the system is terminated by system admin invoking the ‘Shutdown Server’ use case. All users that currently logged in to the system will be disconnected. This situation occurs for instance; the system admin closes the system for a maintenance before the registration day(s). No user can login to the system until the system is initialized by the admin again.

The use cases mentioned above, namely ‘Start Server’, ‘Shutdown Server’ can be seen below:

|  |
| --- |
| *Use case name:* InitializeSystem |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* 1. Upon successful login, the Admin executes the commands to register the client and guest information. |
| *Entry Condition:* The Admin logs into the database server that the OBTS uses. |
| *Exit Condition:* Initialization of the system is completed and the all user and course information of the university are registered onto the database server. |

|  |
| --- |
| *Use case name:* StartServer |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* 1. Upon successful login, the Admin executes the command that is used for the start the server. |
| *Entry Condition:* The Admin logs into the server machine that the OBTS services built on. |
| *Exit Condition:* OBTS services are available and waits for connections from user of the BTS. |

|  |
| --- |
| *Use case name:* ShutdownServer |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* 1. Upon successful login, the Admin executes the command that is used for the shut down the server. |
| *Entry Condition:* The Admin logs into the server machine that the OBTS services built on. |
| *Exit Condition:* OBTS services are unavailable and connections which were already activated are disabled. |

# Subsystem Services

During the subsystem decomposition of *bus ticket reservation / sale* , we divide the system into smaller subsystems with strong coherence. The different subsystems should have a loose coupling.

The subsystem separation shows the entities of the following subsystems:

* + User management subsystems
  + Admin management subsystems
  + Process management subsystems
  + Database subsystems  
      
      
    **User management subsystems**

This subsystem responsible for managing different users of the system by taking care of login information of different users. It manages the username and password of all users of the system.

*Operations provided by this subsystem are:*

* + Login()
  + Logout()

**Admin Management Subsystem**

This subsystem responsible for managing user accounts and ticket information. It provides function for opening an account, updating an account and closing an account. Admin is the actor who communicates with this subsystem. This subsystem uses user management subsystems for authenticating the admin, user their information and voyages information.

*The operations provided by admin management subsystems are:*

* + Login()
  + Add Voyage()
  + DeleteVoyage()
  + DeleteBus()
  + forgetPassword()
  + EditVoyage()
  + viewBus()
  + viewVoyage()
  + viewCity()
  + Logout()

**Process Management Subsystems**

This subsystem is responsible for managing the process. This provides all functions for managing ticket details, booking, free ticket , payment and other things.

*User performed by this subsystem are:*

* + Login()
  + viewAccount()
  + viewTicket()
  + changePassword()
  + cancelTicket()
  + editProfile()
  + booking()
  + seeBooking()
  + cancelBooking()
  + buyTicket()
  + payment()
  + viewFreeTicket()
  + useFreeTicket()
  + forgetPassword()
  + Logout()

*Visitor performed by this subsystem are:*

* + Register()
  + viewVoyage()
  + buyTicket()
  + payment()

**Database Subsystems**

The database subsystem will be implemented by relational database management system used to store admin’s data ,visitor’s data and user’s data.

**Admin Interface Subsystems**

This subsystem responsible for managing user and visitor information. It provides function for ticket information. Admin is the actor who communicates with this subsystem. This subsystem uses user management subsystems for authenticating the admin user information.

**User Interface Subsystems**

This subsystem in charge of managing the process. This ensure for edit profile, booking, payment, free ticket, ticket, view account, ticket and booking, cancel ticket and booking.

**Visitor Interface Subsystems**

This subsystem in charge of managing the process. This ensure for register , view voyage, buying ticket, payment.

**Database Subsystems**

The database subsystem will be implemented by relational database management system used to store admin’s data, user’s data and visitor’s data.

# References

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.
2. Lecture presentations of the course (the presentations were provided by the Instructor who offered the Software Engineering course).
3. Kamilkoc.com