SOFTWARE DESIGN DOCUMENT

**(SDD)**

**For**

**[Online Car Rental Management System]**

of the

**ON-BOARD AUTOMOBILE(OBA)**

CONTRACT NO:

**CASE (Man\_Doc\_006)**

CDRL SEQUENCE NO:

**CASE (Man\_Doc\_008)**

**Prepared for:**

**[Kereta Sewa Murah Mesra]**

**Generated By:**

|  |  |
| --- | --- |
| Team Members | ID Number |
| Sivapriyan A/L S Kummar | CB14094 |
| Wan Noorafirah binti Wan Ramli | CB14129 |
| Noraisyah binti Yaakub | CB14105 |
| Ruzdiana Aniera binti Ramli | CB14069 |
| Ahmad Solehin bin Sharudin | CB15003 |

**November 2016**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approval List Table** | | | | | | |
| Index | 01 | 02 | 03 | 04 | 05 | 06 |
| Writed by:  Name: | Date: | Date: | Date: | Date: | Date: | Date: |
| Verified by:  Quality Manager  Name: | Date: | Date: | Date: | Date: | Date: | Date: |
| Check by:  Configuration Manager  Name: | Date | Date: | Date: | Date: | Date: | Date: |
| Approved by:  Work Package Manager  Name: | Date | Date: | Date: | Date: | Date: | Date: |
| Authenticated by:  Project Manager  Name: | Date | Date: | Date: | Date: | Date: | Date: |

|  |  |
| --- | --- |
| **Revision History** | |
| **Revision** | **Description** |
| 01 | Detail design document |
| 02 | Correction List :   1. Item number 2. System ID no 3. Reference number 4. System overview 5. OCRMS architecture 6. Diagram each module (Register, Manage car info,Manage customer info, Manage booking, Generate) |
| 03 |  |
| 04 |  |
| 05 |  |
| 06 |  |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **1. SCOPE**...............................................................................................................  1.1. Identification ..................................................................................................  1.2. Overview of the System ….............................................................................  1.3. Overview of the Document …........................................................................ | 1  1  1  2 |
| **2. REFERENCED DOCUMENTS**...................................................................... 2.1. Reference Documents........................................................................ | 5  5 |
| **3. PRELIMINARY DESIGN**...............................................................................  3.1. System Overview ............................................................................................  3.1.1 System Architecture ...................................................................................  3.1.2 System States and Modes........................................................................... | 6  7  7  8 |
| **4. DETAILED DESIGN**.......................................................................................  4.1 Registration Subsystem............................................................................  4.1.1 Registration Package......................................................................  4.2 Manage car Subsystem.................................................................................  4.2.1 Manage Car Package..........................................................................  4.3 Manage Customer Info Subsystem..........................................................  4.3.1 Manage Customer Info Package........................................................  4.4 Manage Booking Subsystem…….................................................  4.4.1 Manage Booking Package…...........................................................  4.5 Generate Bill Subsystem…………………………………  4.5.1 Generate Bill Package….................................................................... | 13  13  14  18  19  23  24  26  27  40  41 |
| **5. DATA DICTIONARY……………………………………………………..** | 46 |
| **6. REQUIREMENTS TRACEABILITY………………………………………** | 48 |
| **7. ABBREVIATION and NOTES ……………………………………….** | 48 |

# SCOPE

## Identification

System name: Online Car Rental Management System

Abbreviation: OCRMS

System ID No.: OCRMS-V.02-2016

## Overview of the System

The objective of this project is to develop a web based computer system to support existing manual car rental management process.OCRMS will convert manual car rental management into a computerized system which is integrated with a database system. By this integration,the system is capable to insert,save, update, delete, retrieve records and data. This system can keep huge data organized, secured and may reduce the uses of lots of paper. This system is a web-based application which means that Graphic User Interfaces (GUI) are implemented using any web browser. By default it is an internet-based application whereas user need an internet connection to access this system. The users of this system are customer,employee and manager who have different level of authority and function access to the system .Plus, the outsider is able to access OCRMS home page as it is public.

The major activities of the system are supporting the management information of the car rental process which include the manipulating the specific data, which are booking,customers and cars details. By using this system, the customer can book car for rental online.This include modifying the booking details which include update and cancelling the booking.The employee can update car availability and info,view customer info and generate bill using this system.While the manager can use this system to view monthly rental reports.These system features makes the car rentaling process more efficient and less complicated while contributing to better management process.

**Purpose of the System:**

The purpose of this system is to support existing manual car rental management process. OCRMS will convert manual car rental management into a computerized system which is integrated with a database system. These document explained the purpose and features of the system, the interfaces that being used, what the system can do, the constraint under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

## Overview of the Document

This Software Requirement Specification(SRS) document have seven different part which is Scope ,Referenced Documents, Preliminary Design, Detailed Design, CSCI Data and CSCI Data Files, Requirements Traceability, Notes.

The purpose of this document is to outline the entire requirement that has been gathered for the Online Car Rental Management System. In the first chapter, the **scope** of this system have been stated. Then the overview for the overall system are stated where all the functions provided by the system are explained with details. The reference that we refer also stated as prove.

The second chapter is the **Referenced Documents**, which gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for a technical requirements specification in the third chapter.This chapter is more to explaining the system.It describes the user that will use the system, system function, the system constraints and the assumptions and dependencies. This to make sure that the client can understand the system flow and the weakness of system.OCRMS is accessible with any compatible browser such as Internet Explorer, Mozilla Firefox and Google Chrome.

Third chapter is **Preliminary Design**, it describes the software requirements to a level of more details that enables designers to design a system that satisfy those requirements and well-functioning as well. Besides that, it also explains every input and output of the system with all functions performed by the system in response.By this, client can get to know more about the product features, the interface requirements and the requirement traceability.

The fourth chapter, **Detailed Design**, of this documentation typically describes what is needed by the system user as well as requested properties of inputs and outputs. A functional specification is the more technical response to a matching requirements document, e.g. the Product Requirement Document “PRD”. Thus it picks up the results of the requirements analysis stage. On more complex systems multiple levels of functional specifications will typically nest to each other, e.g. on the system level, on the module level and on the level of technical details.

The fifth chapter, **Data dictionary,** of this documentation data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them. A first step in analyzing a system of objects with which users interact is to identify each object and its relationship to other objects. This process is called data modeling and results in a picture of object relationships.

The sixth chapter, **Specification module requirement**, of this document is basically a table that has been constructed with all of the unfamiliar abbreviations such as OCRMS which stands for Online Car Rental Management System so if users come across an abbreviation that they are not familiar with they can simply turn to the table and look up the word they are looking for.

The seven chapter, **Notes and abbreviation**, of this document is basically a table that has been constructed with all of the unfamiliar abbreviations such as OCRMS which stands for Online Car Rental Management System so if users come across an abbreviation that they are not familiar with they can simply turn to the table and look up the word they are looking for.

In general, this Software Design Document is divided into 7 sections as the following:

|  |  |
| --- | --- |
| **Chapter 1** | Describes the scope identification, system overview and the document overview. |
| **Chapter 2** | Referenced documents, government documents and non-government documents. |
| **Chapter 3** | Describes the preliminary design for the OCRMS. The preliminary design will identify OCRMS, Describes the OCRMS Architecture, static organization and dynamic organization. |
| **Chapter 4** | Describes the detail design. |
| **Chapter 5** | Describes the data dictionary |
| **Chapter 6** | Specification module requirement |
|  |  |
| **Chapter 7** | Notes and abbreviations. |

# 

# 2. REFERENCED DOCUMENTS

## 2.1. Reference Documents

This section consists of the contractual documents and non-contractual documents.

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **TYPE** | **TITLE** | **DESCRIPTION** |
| 1 | SDP (OCRMS-2016-Version 1.0) | Online Car Rental Management System | Software Development Plan |
| 2 | SRS (OCRMS-V.01-2016) | Online Car Rental Management System | Software Requirements Specification |

Figure 2.1: Reference Documents

# 3. PRELIMINARY DESIGN

## 3.1. System Overview

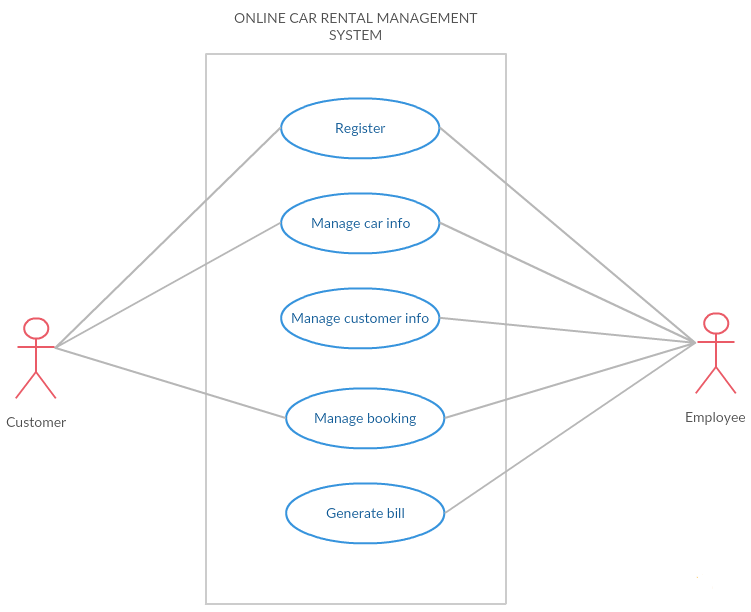


Figure 3.1 : System Design Overview

## 3.1.1 OCRMS Architecture

This paragraph identifies the internal organizational structure of the system. The relationship among system subsystem will be described.

## 3.1.1.1 Static Organization

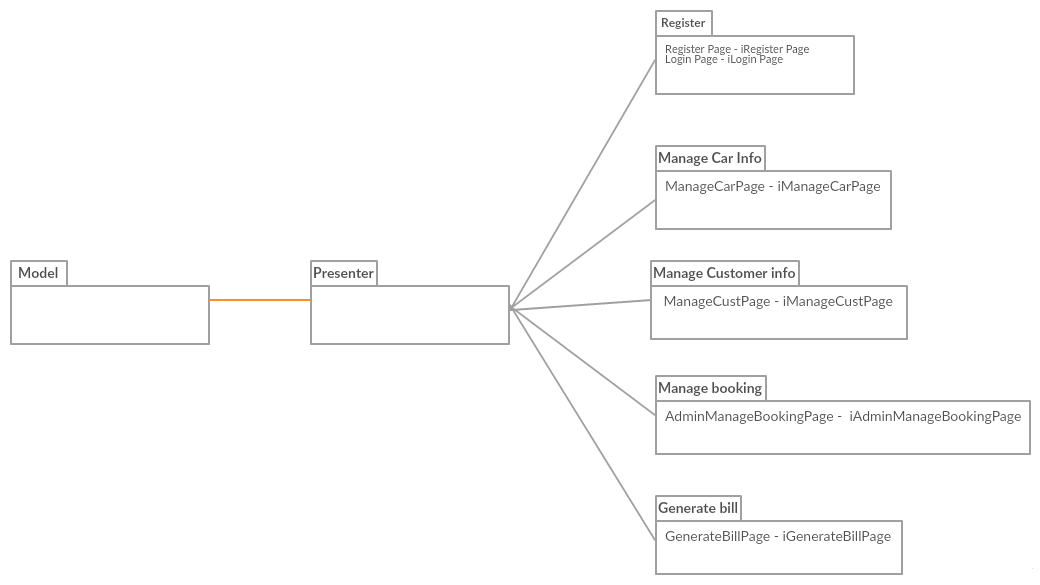


Figure 3.2 : Static Organization of [ Online Car Rental Management System]

**Model :**

1. Register

* Model store this attribute such as Customer\_Name, Phone\_Number, Address , License\_Number, Username , Password, Email.

1. Manage car info

* Model store this attribute such as Username, Password, Name, Address, Ic\_Number, Phone\_Number, Available\_Date, Date\_Borrowed, Car\_id, Car\_Name and No\_of\_Available.

1. Manage customer info

* Model store this attribute such as Customer\_Name, Email, Address,

Phone \_number, Date\_Borrowed, Car\_Id

1. Manage booking

* Model store this attribute such as username, password, Car\_ID, Car\_Name, Plate\_Num , Car\_History, Available\_Date , Time\_Borrowed , Date\_Borrowed, No\_of\_Car\_Available, Name, Address , License\_Num , Phone , Collect\_Date , Returned\_Date, id , Name, car\_avaibility , booking\_request , customer\_info

1. Generate bill

* Model store these attributes such as Employee\_ID , Password , Booking\_Details, Customer\_info, Date, Mfinancial\_summary, Dfinancial\_summary, Receipt .

**Presenter :**

1. Register

* Presenter implement all this function form model to view such as getCustomer\_Name, getPhone\_Number, getAddress , getLicense\_Number, getUsername , getPassword, getEmail.

1. Manage car info

* Presenter implement all this function form model to view such as getUsername, getPassword, getName, getAddress, getIc\_number, getPhone\_number, getAvailable\_Date, getDate\_Borrowed, getCar\_id, getCar\_Name, getNo\_of\_Car.

1. Manage customer info

* Presenter implement all this function form model to view such as getCustomer\_Name, getEmail, getAddress, getPhone \_number, getDate\_Borrowed, getCar\_Id

1. Manage booking

* Presenter implement all this function form model to view such as getUsername, getPassword, getCar\_ID, getCar\_Name, getPlate\_Num , getCar\_History, getAvailable\_Date , getTime\_Borrowed , getDate\_Borrowed, getNo\_of\_Car\_Available, getName, getAddress , getLicense\_Num , getPhone , getCollect\_Date , getReturned\_Date, getId , getCar\_avaibility , getBbooking\_request , getCustomer\_info

1. Generate bill

* Presenter implement all this function form model to view such as getEmployee\_ID , getPassword, getCustomer\_info, getBooking\_details, getFinancial\_summary, getReceipt .

**View :**

1. Register

* Register only have one interface (iRegister) which is implement frompresenter then display information to Register.
* Login only have one interface (iLogin) which is implement from presenter then display information to Login.

1. Manage car info

* Manage car info only have one interface (iManageCarPage) which is implement from presenter then display information to ManageCarPage.

1. Manage customer info

* Manage customer info only have one interface (iManageCustomerPage) which is implement from presenter then display information to ManageCustomerPage.

1. Manage booking

* Manage booking info only have one interface (iManageBookingPage) which is implement from presenter then display information to ManageBookingPage.

1. Generate bill

* Generate bill only have one interface (iGeneratebillpage) which is implement from presenter then display information to Generatebillpage.

This section describes the detail for each subsystem/package.

* 1. **Registration**

This subsystem is responsible to assists the customer to complete the registration procedure. The customer will register manually. Then, the employee will key in their information. Their detail will be recorded into the system and the ID will be generated for each and every new customer that have registered.

* + 1. Login
    2. Customer
    3. Register
  1. **Manage Car Info**

This subsystem is responsible to ensure available car so that there is no redundant booking on date and time.

* + 1. Car
    2. Employee
  1. **Manage Customer Info**

This subsystem is able to update the customer info if there is a change.

* + 1. Employee
    2. Customer
  1. **Manage Booking**

This subsystem is responsible for employee to manage booking based on customer requirements.

* + 1. Employee
    2. Customer
  1. **Generate Bill**

This subsystem is responsible to display receipt and financial summary.

* + 1. Employee
    2. Financial
    3. Receipt

**3.1.1.2. Dynamic Organization**

Figure 3.3 diagram shows components and their relationships between each other in System

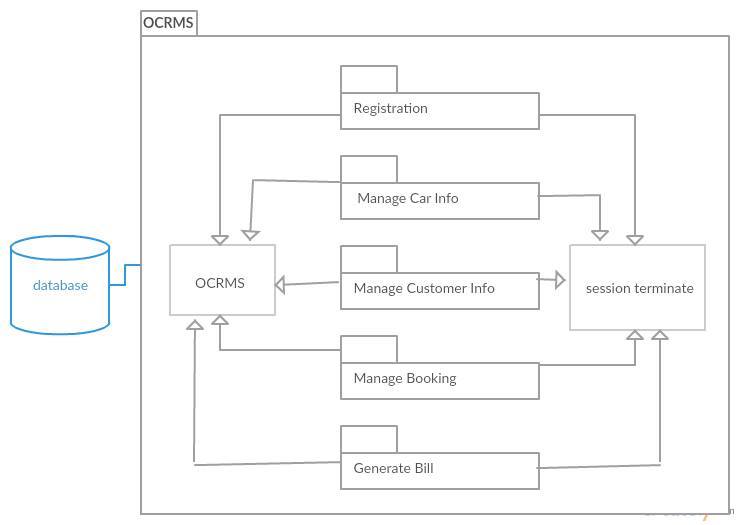


Figure 3.3 : Component Diagram of OCRMS

# 

# 4. DETAILED DESIGN

This section divided into the following paragraphs and subparagraphs to describe the detailed design.

## 4.1 Registration Subsystem (SDD-REQ-S01)

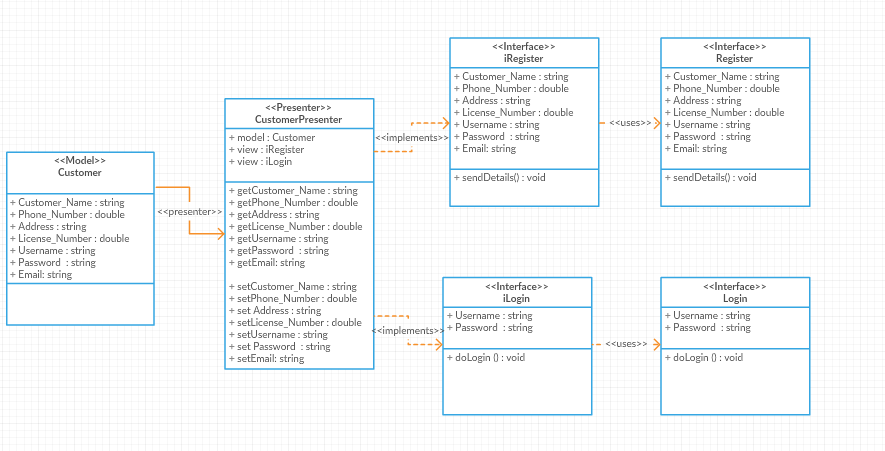


Figure 4.1: Registration Detail Design

**4.1.1 Model class design**

This subparagraph specifies the design of model class.

1. Login
2. Customer
3. Register

**4.1.1.1 Login class**

1. **Input/output data elements**

List of input and output data elements :-

Input : Username, Password

Output : verify username and password

1. **Algorithm**

* **Username**

|  |
| --- |
| Username = UN*(1)* ,  UI *(2),* UN *(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**   1. Read UN*(1)* 2. Display Y ( UN*(1)* ) 3. Repeat step 1 until 2 with UN *(2)*.......... UN*(n)*   **End** |

* **Password**

|  |
| --- |
| Password = PS*(1)* ,  PS *(2),* PS *(3)* ............ PS *(n)*  Display Y ; |
| **Begin :**   1. Read PS*(1)* 2. Display Y ( PS*(1)* ) 3. Repeat step 1 until 2 with PS *(2)*.......... PS*(n)*   **End** |

**4.1.1.2 Customer class**

1. **Input/output data elements**

List of input and output data elements :

Input : Customer\_Name, Phone\_Number, Address , License\_Number, Username , Password, Email.

Output : get customer information

* **Customer\_Name**

|  |
| --- |
| Password = CN*(1)* ,  CN *(2),* CN *(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**   1. Read CN*(1)* 2. Display Y ( CN*(1)* ) 3. Repeat step 1 until 2 with CN *(2)*.......... CN*(n)*   **End** |

* **Phone\_Number**

|  |
| --- |
| Password = PN*(1)* ,  PN *(2),* PN *(3)* ............ PN *(n)*  Display Y ; |
| **Begin :**   1. Read PN*(1)* 2. Display Y ( PN*(1)* ) 3. Repeat step 1 until 2 with PN *(2)*.......... PN*(n)*   **End** |

* **Address**

|  |
| --- |
| Password = AD*(1)* ,  AD *(2),* AD *(3)* ............ AD *(n)*  Display Y ; |
| **Begin :**   1. Read AD*(1)* 2. Display Y ( AD*(1)* ) 3. Repeat step 1 until 2 with AD*(2)*.......... AD*(n)*   **End** |

* **License\_Number**

|  |
| --- |
| Password = LN*(1)* ,  LN *(2),* LN *(3)* ............ LN *(n)*  Display Y ; |
| **Begin :**   1. Read LN*(1)* 2. Display Y ( LN*(1)* ) 3. Repeat step 1 until 2 with LN *(2)*.......... LN*(n)*   **End** |

* **Username**

|  |
| --- |
| Username = UN*(1)* ,  UI *(2),* UN *(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**   1. Read UN*(1)* 2. Display Y ( UN*(1)* ) 3. Repeat step 1 until 2 with UN *(2)*.......... UN*(n)*   **End** |

* **Password**

|  |
| --- |
| Password = PS*(1)* ,  PS *(2),* PS *(3)* ............ PS *(n)*  Display Y ; |
| **Begin :**   1. Read PS*(1)* 2. Display Y ( PS*(1)* ) 3. Repeat step 1 until 2 with PS *(2)*.......... PS*(n)*   **End** |

* **Email**

|  |
| --- |
| Password = E*(1)* ,  E *(2),* E *(3)* ............ E *(n)*  Display Y ; |
| **Begin :**   1. Read E*(1)* 2. Display Y ( E*(1)* ) 3. Repeat step 1 until 2 with E *(2)*.......... E*(n)*   **End** |

**4.1.1.3 Register class**

1. **Input/output data elements**

List of input and output data elements :

Input : Customer\_Name, Phone\_Number, Address , License\_Number, Username , Password, Email

Output : get customer information

* **Customer\_Name**

|  |
| --- |
| Password = CN*(1)* ,  CN *(2),* CN *(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**  1) Read CN*(1)*  2) Display Y ( CN*(1)* )  3) Repeat step 1 until 2 with CN *(2)*.......... CN*(n)*  **End** |

* **Phone\_Number**

|  |
| --- |
| Password = PN*(1)* ,  PN *(2),* PN *(3)* ............ PN *(n)*  Display Y ; |
| **Begin :**   1. Read PN*(1)* 2. Display Y ( PN*(1)* ) 3. Repeat step 1 until 2 with PN *(2)*.......... PN*(n)*   **End** |

* **Address**

|  |
| --- |
| Password = AD*(1)* ,  AD *(2),* AD *(3)* ............ AD *(n)*  Display Y ; |
| **Begin :**   1. Read AD*(1)* 2. Display Y ( AD*(1)* ) 3. Repeat step 1 until 2 with AD*(2)*.......... AD*(n)*   **End** |

* **License\_Number**

|  |
| --- |
| Password = LN*(1)* ,  LN *(2),* LN *(3)* ............ LN *(n)*  Display Y ; |
| **Begin :**   1. Read LN*(1)* 2. Display Y ( LN*(1)* ) 3. Repeat step 1 until 2 with LN *(2)*.......... LN*(n)*   **End** |

* **Username**

|  |
| --- |
| Username = UN*(1)* ,  UI *(2),* UN *(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**   1. Read UN*(1)* 2. Display Y ( UN*(1)* ) 3. Repeat step 1 until 2 with UN *(2)*.......... UN*(n)*   **End** |

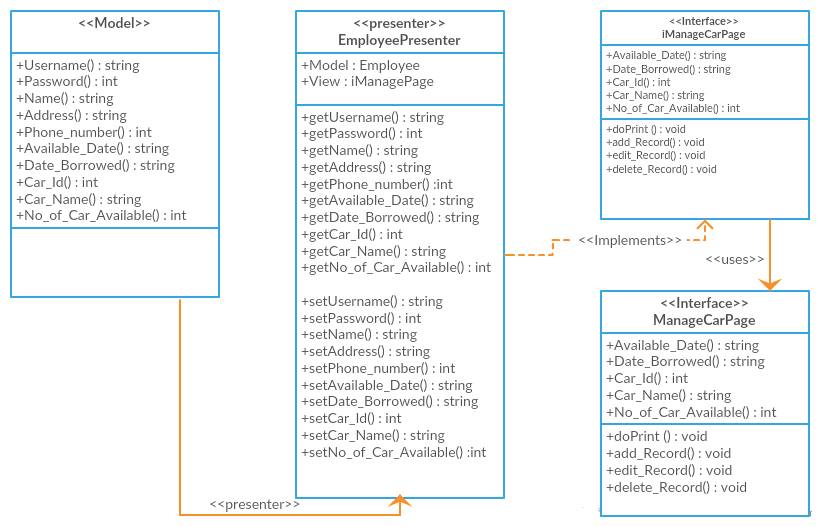
* **Password**

|  |
| --- |
| Password = PS*(1)* ,  PS *(2),* PS *(3)* ............ PS *(n)*  Display Y ; |
| **Begin :**   1. Read PS*(1)* 2. Display Y ( PS*(1)* ) 3. Repeat step 1 until 2 with PS *(2)*.......... PS*(n)*   **End** |

* **Email**

|  |
| --- |
| Password = E*(1)* ,  E *(2),* E *(3)* ............ E *(n)*  Display Y ; |
| **Begin :**   1. Read E*(1)* 2. Display Y ( E*(1)* ) 3. Repeat step 1 until 2 with E *(2)*.......... E*(n)*   **End** |

## 4.2 Manage Car Info Subsystem (SDD-REQ-S02)



**Figure 4.2 : Manage Car Info Detail Design**

**4.2.1 Model class design**

This subparagraph specifies the design of model class.

1. Car
2. Employee

**4.2.1.1 Car class**

1. **Input/output data elements**

List of input and output data elements :

Input :Car\_ID, Car\_Name, Plate\_Num,Car\_History, Available\_Date , Time\_Borrowed , Date\_Borrowed, No\_of\_Car\_Available

Output : get car information

**b) Algorithms**

The purpose of this class is to allow employee enter information of car into system.

Class type : Model class

Responsibility : To input information of car into system

Attributes : Car\_ID : Integer

Car\_Name : String

Plate\_Num : String

Car\_History : String

Available\_Date : String

Time\_Borrowed : String

Date\_ Borrowed : String

No\_of\_Car\_Available : Integer

1. Car\_ID

|  |
| --- |
| Car\_ID = CI*(1)* ,  CI *(2),* CI *(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI*(1)* 2. Display Y ( CI*(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

1. Car\_Name

|  |
| --- |
| Car\_Name = CN*(1)* , CN*(2),* CN*(3)* ............ CN *(n)*  Display Y ; |
| **Begin :**   1. Read CN*(1)* 2. Display Y ( CN*(1)* ) 3. Repeat step 1 until 2 with CN *(2)*.......... CN *(n)*   **End** |

1. Plate\_Num

|  |
| --- |
| Plate\_Num = PN*(1)* ,  PN *(2),* PN *(3)* ............ PN *(n)*  Display Y ; |
| **Begin :**   1. Read PN *(1)* 2. Display Y ( CN*(1)* ) 3. Repeat step 1 until 2 with PN *(2)*.......... PN*(n)*   **End** |

1. Car\_History

|  |
| --- |
| Car\_History = CH*(1)* ,  CH *(2),* CH *(3)* ............ CH *(n)*  Display Y ; |
| **Begin :**   1. Read CH *(1)* 2. Display Y (CH *(1)* ) 3. Repeat step 1 until 2 with CH *(2)*.......... CH *(n)*   **End** |

1. Available\_Date

|  |
| --- |
| Available\_Date= AD*(1)* ,  AD *(2),* AD*(3)* ............ AD *(n)*  Display Y ; |
| **Begin :**   1. Read AD *(1)* 2. Display Y (AD *(1)* ) 3. Repeat step 1 until 2 with AD*(2)*..........AD *(n)*   **End** |

1. Time\_Borrowed

|  |
| --- |
| Time\_Borrowed = TD(*1)* ,  TD*(2),* TD *(3)* ............ TD *(n)*  Display Y ; |
| **Begin :**   1. Read TD *(1)* 2. Display Y (TD *(1)* ) 3. Repeat step 1 until 2 with TD *(2)*.......... TD *(n)*   **End** |

1. Date\_ Borrowed

|  |
| --- |
| Date\_Borrowed = DB(*1)* ,  DB *(2),* DB *(3)* ............ DB *(n)*  Display Y ; |
| **Begin :**   1. Read DB *(1)* 2. Display Y ( DB *(1)* ) 3. Repeat step 1 until 2 with DB *(2)*.......... DB *(n)*   **End** |

1. No\_of\_Car\_Available

|  |
| --- |
| No\_of\_car\_available = CA(*1)* ,  CA *(2),* CA *(3)* ............ CA *(n)*  Display Y ; |
| **Begin :**   1. Read CA *(1)* 2. Display Y ( CA *(1)* ) 3. Repeat step 1 until 2 with CA *(2)*.......... CA *(n)*   **End** |

**4.2.1.2 Employee class**

1. **Input/output data elements**

List of input and output data elements :

Input : id , Name, Address, Phone

Output : get customer information

**b) Algorithms**

The purpose of this class is to allow staff to retrieve and store input information of customer into system.

Class type : Model class

Responsibility : To input information of customer into system

Attributes : ID : Integer

Name : String

Address : String

Phone : Double

1. ID

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID*(1)* 2. Display Y (ID*(1)*) 3. Repeat step 1 until 2 with ID*(2)*..........ID*(n)*   **End** |

1. Name

|  |
| --- |
| Name = N*(1)* , N *(2),* N*(3)* ............ N*(n*  Display Y ; |
| **Begin :**   1. Read N*(1)* 2. Display Y (N*(1)* ) 3. Repeat step 1 until 2 with N*(2)*..........N*(n)*   **End** |

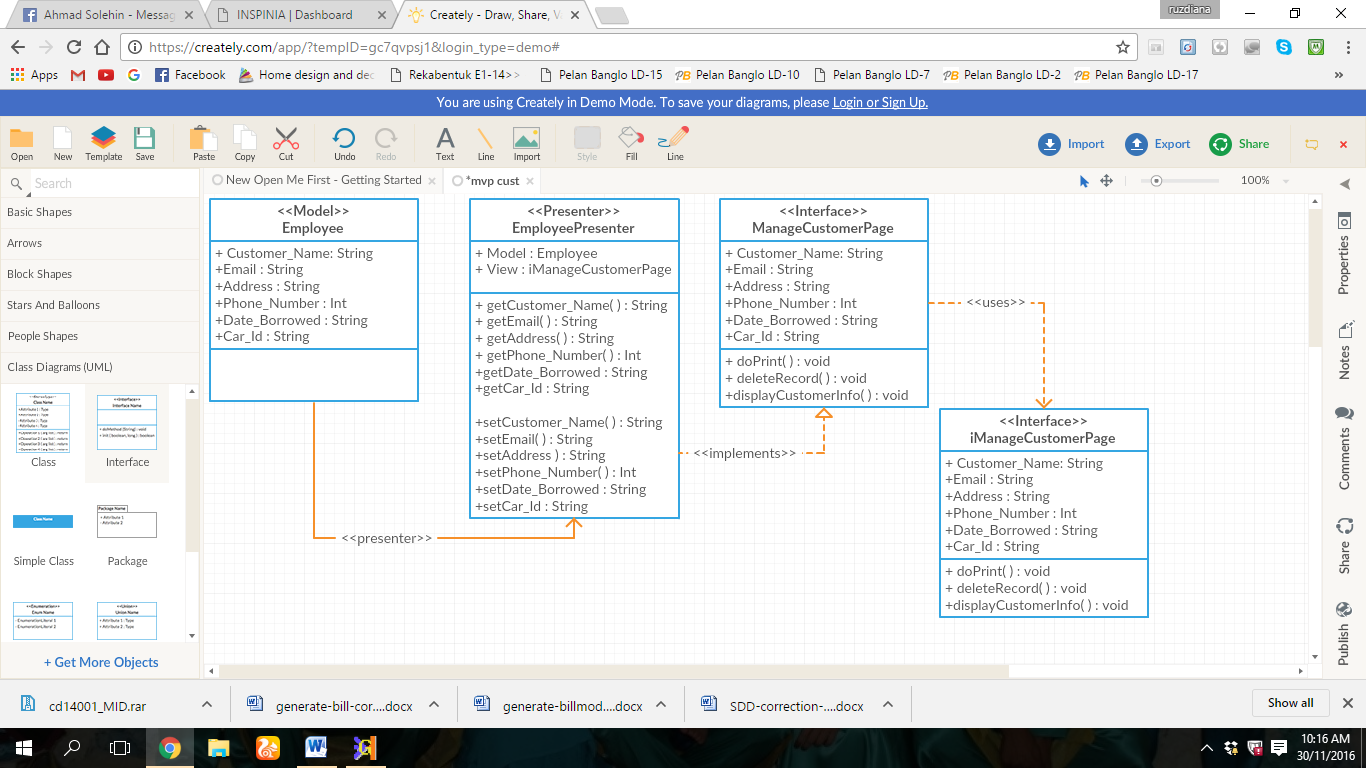
1. Address

|  |
| --- |
| Address = Ad*(1)* , Ad*(2),* Ad *(3)* ............Ad *(n)*  Display Y ; |
| **Begin :**   1. Read Ad *(1)* 2. Display Y (Ad*(1)* ) 3. Repeat step 1 until 2 with Ad*(2)*..........Ad*(n)*   **End** |

1. Phone

|  |
| --- |
| Phone = P*(1)* , P*(2)* , P*(3)* ............ P*(n)*  Display Y ; |
| **Begin :**   1. Read P *(1)* 2. Display Y(P*(1)*) 3. Repeat step 1 until 2 with P*(2)*..........P*(n)*   **End** |

## 4.3 Manage Customer Info Subsystem (SDD-REQ-S03)



**Figure 4.3 : Manage Customer Info Detail Design**

**4.3.1 Model class design**

This subparagraph specifies the design of model class.

1. Employee
   * + 1. **Employee class**
2. **Input/output data elements**

List of input and output data elements :

Input : Customer\_Name, Email, Address, Phone \_number, Date\_Borrowed, Car\_Id

Output : get customer information list

1. **Algorithms**

The purpose of this class is to allow employee to view and update information of customer into system.

Class type : Model class

Responsibility : To input information of customer into system

Attributes : Customer\_Name : String

Email : String

Address : String

Phone \_number : Integer

Date\_Borrowed : String

Car\_Id : String

1. Customer\_Name

|  |
| --- |
| Customer\_Name = CN*(1)* , CN *(2),* CN*(3)* ............ CN*(n)*  Display Y ; |
| **Begin :**   1. Read CN*(1)* 2. Display Y (CN*(1)* ) 3. Repeat step 1 until 2 with CN*(2)*..........CN*(n)*   **End** |

1. Email

|  |
| --- |
| Password = E*(1)* ,  E *(2),* E *(3)* ............ E *(n)*  Display Y ; |
| **Begin :**   1. Read E*(1)* 2. Display Y ( E*(1)* ) 3. Repeat step 1 until 2 with E *(2)*.......... E*(n)*   **End** |

1. Address

|  |
| --- |
| Address = Ad*(1)* , Ad*(2),* Ad *(3)* ............Ad *(n)*  Display Y ; |
| **Begin :**   1. Read Ad *(1)* 2. Display Y (Ad*(1)* ) 3. Repeat step 1 until 2 with Ad*(2)*..........Ad*(n)*   **End** |

1. Phone\_Number

|  |
| --- |
| Password = PN*(1)* ,  PN *(2),* PN *(3)* ............ PN *(n)*  Display Y ; |
| **Begin :**   1. Read PN*(1)* 2. Display Y ( PN*(1)* ) 3. Repeat step 1 until 2 with PN *(2)*.......... PN*(n)*   **End** |

1. Date\_ Borrowed

|  |
| --- |
| Date\_Borrowed = DB(*1)* ,  DB *(2),* DB *(3)* ............ DB *(n)*  Display Y ; |
| **Begin :**   1. Read DB *(1)* 2. Display Y ( DB *(1)* ) 3. Repeat step 1 until 2 with DB *(2)*.......... DB *(n)*   **End** |

1. Car\_ID

|  |
| --- |
| Car\_ID = CI*(1)* ,  CI *(2),* CI *(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI*(1)* 2. Display Y ( CI*(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

## 4.4 Manage Booking Subsystem (SDD-REQ-S04)

## 

**Figure 4.4 : Manage Booking Detail Design**

**4.4.1 Model class design**

This subparagraph specifies the design of model class.

1. Authentication
2. Car
3. Customer
4. Employee
   * + 1. **Authentication class**
5. **Input/output data elements**

List of input and output data elements :-

Input : username, password

Output : verify username and password

1. **Algorithms**

The purpose of this class is to store the username and password.

Class type : Model class

Responsibility : To verify username and password

Attributes : username : String

Password : String

1. Username

|  |
| --- |
| Username = U*(1)* , U *(2)* , U*(3)* ............ U*(n)*  Display Y ; |
| **Begin :**   1. Read U *(1)* 2. Display Y(U*(1)*) 3. Repeat step 1 until 2 with U*(2)*.......... U*(n)*   **End** |

1. Password

|  |
| --- |
| Password = Ps*(1)* , Ps*(2)* , Ps*(3)* ............ Ps*(n)*  Display Y ; |
| **Begin :**   1. Read Ps*(1)* 2. Display Y( Ps*(1)*) 3. Repeat step 1 until 2 with Ps *(2)*.......... Ps*(n)*   **End** |

* + - 1. **Car class**

1. **Input/output data elements**

List of input and output data elements :

Input :Car\_ID, Car\_Name, Plate\_Num , Car\_History, Available\_Date , Time\_Borrowed , Date\_Borrowed, No\_of\_Car\_Available

Output : get car information

**b) Algorithms**

The purpose of this class is to allow tuition staff to input information of teacher into system.

Class type : Model class

Responsibility : To input information of car into system

Attributes : Car\_ID : Integer

Car\_Name : String

Plate\_Num : String

Car\_History : String

Available\_Date : String

Time\_Borrowed : String

Date\_ Borrowed : String

No\_of\_Car\_Available : Integer

1. Car\_ID

|  |
| --- |
| Car\_ID = CI*(1)* ,  CI*(2),* CI*(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI*(1)* 2. Display Y ( CI*(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

1. Car\_Name

|  |
| --- |
| Car\_Name = CN*(1)* , CN*(2),* CN*(3)* ............ CN*(n)*  Display Y ; |
| **Begin :**   1. Read CN*(1)* 2. Display Y ( CN*(1)* ) 3. Repeat step 1 until 2 with CN *(2)*.......... CN*(n)*   **End** |

1. Plate\_Num

|  |
| --- |
| Plate\_Num = PN*(1)* ,  PN *(2),*PN *(3)* ............PN*(n)*  Display Y ; |
| **Begin :**   1. Read PN *(1)* 2. Display Y ( PN *(1)* ) 3. Repeat step 1 until 2 with PN *(2)*.......... PN*(n)*   **End** |

1. Car\_History

|  |
| --- |
| Car\_History = CH*(1)* ,  CH *(2),* CH *(3)* ............CH *(n)*  Display Y ; |
| **Begin :**   1. Read CH *(1)* 2. Display Y (CH *(1)*) 3. Repeat step 1 until 2 with CH *(2)*.......... CH *(n)*   **End** |

5) Available\_Date

|  |
| --- |
| Available\_Date= AD*(1)* ,  AD *(2),* AD*(3)* ............AD *(n)*  Display Y ; |
| **Begin :**   1. Read AD *(1)* 2. Display Y (AD *(1)*) 3. Repeat step 1 until 2 with AD*(2)*..........AD *(n)*   **End** |

6.Time\_Borrowed

|  |
| --- |
| Time\_Borrowed = TD(*1)* ,  TD*(2),* TD *(3)* ............ TD *(n)*  Display Y ; |
| **Begin :**   1. Read TD *(1)* 2. Display Y (TD*(1)*) 3. Repeat step 1 until 2 with TD *(2)*.......... TD*(n)*   **End** |

7) Date\_ Borrowed

|  |
| --- |
| Date\_Borrowed = DB(*1)* ,  DB *(2),* DB *(3)* ............ DB *(n)*  Display Y ; |
| **Begin :**   1. Read DB *(1)* 2. Display Y ( DB *(1)* ) 3. Repeat step 1 until 2 with DB *(2)*.......... DB *(n)*   **End** |

8) No\_of\_Car\_Available

|  |
| --- |
| No\_of\_car\_available = CA(*1)* ,  CA *(2),* CA *(3)* ............ CA *(n)*  Display Y ; |
| **Begin :**   1. Read CA *(1)* 2. Display Y ( CA *(1)* ) 3. Repeat step 1 until 2 with CA *(2)*.......... CA *(n)*   **End** |

* + - 1. **Customer class**

1. **Input/output data elements**

List of input and output data elements :

Input : id , Name, Address , License\_Num , Phone , Collect\_Date , Returned\_Date

Output : get customer information

1. **Algorithms**

The purpose of this class is to allow staff to retrieve and store input information of customer into system.

Class type : Model class

Responsibility : To input information of customer into system

Attribute : id : Integer

Name : String

Address : String

Phone : Double

License\_Num : String

Collect\_Date : String

Returned\_Date : String

1. ID

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID  *(1)* 2. Display Y (ID  *(1)* ) 3. Repeat step 1 until 2 with ID*(2)*.......... ID*(n)*   **End** |

1. Name

|  |
| --- |
| Name = N*(1)* , N *(2),* N *(3)* ............ N *(n)*  Display Y ; |
| **Begin :**   1. Read N  *(1)* 2. Display Y (N *(1)* ) 3. Repeat step 1 until 2 with N*(2)*.......... N*(n)*   **End** |

1. Address

|  |
| --- |
| Address = Ad*(1)* , Ad*(2),* Ad *(3)* ............ Ad *(n)*  Display Y ; |
| **Begin :**   1. Read Ad *(1)* 2. Display Y (Ad *(1)* ) 3. Repeat step 1 until 2 with Ad*(2)*.......... Ad*(n)*   **End** |

1. Phone

|  |
| --- |
| Phone = P*(1)* , P *(2)* , P*(3)* ............ P*(n)*  Display Y ; |
| **Begin :**   1. Read P *(1)* 2. Display Y(P*(1)*) 3. Repeat step 1 until 2 with P*(2)*.......... P*(n)*   **End** |

1. License\_Num

|  |
| --- |
| License\_Num = LN*(1)* , LN *(2)* , LN *(3)* ............ LN *(n)*  Display Y ; |
| **Begin :**   1. Read LN *(1)* 2. Display Y(LN*(1)*) 3. Repeat step 1 until 2 with LN*(2)*.......... LN*(n)*   **End** |

1. Collect\_Date

|  |
| --- |
| Collect\_Date = CD*(1)* , CD *(2)* , CD *(3)* ............ CD *(n)*  Display Y ; |
| **Begin :**   1. Read CD *(1)* 2. Display Y( CD *(1)*) 3. Repeat step 1 until 2 with CD *(2)*.......... CD *(n)*   **End** |

* + - 1. **Employee class**

1. **Input/output data elements**

List of input and output data elements :

Input : id , Name, car\_avaibility , booking\_request , customer\_info

Output : get employee information

**b) Algorithms**

The purpose of this class is to allow employee to manage booking request from customer

Class type : Model class

Responsibility : To input information of customer into system

Attributes : id : Integer

Name : String

No\_of\_car\_available : String

booking\_request : String

customer\_info : customer

1. Id

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID  *(1)* 2. Display Y (ID  *(1)* ) 3. Repeat step 1 until 2 with ID*(2)*.......... ID*(n)*   **End** |

1. **Name**

|  |
| --- |
| Name = N*(1)* , N *(2),* N *(3)* ............ N *(n)*  Display Y ; |
| **Begin :**   1. Read N  *(1)* 2. Display Y (N *(1)* ) 3. Repeat step 1 until 2 with N*(2)*.......... N*(n)*   **End** |

1. No\_of\_car\_available

|  |
| --- |
| No\_of\_car\_available = CA(*1)* ,  CA *(2),* CA *(3)* ............ CA *(n)*  Display Y ; |
| **Begin :**   1. Read CA *(1)* 2. Display Y (CA *(1)* ) 3. Repeat step 1 until 2 with CA *(2)*.......... CA *(n)*   **End** |

1. Booking\_Request

|  |
| --- |
| Booking\_Request = BR(*1)* ,  BR *(2),* BR *(3)* ............ BR *(n)*  Display Y ; |
| **Begin :**   1. Read BR *(1)* 2. Display Y ( BR *(1)* ) 3. Repeat step 1 until 2 with BR *(2)*.......... BR *(n)*   **End** |

1. Customer\_Info

|  |
| --- |
| Customer\_Info = CI(*1)* ,  CI *(2),* CI *(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI *(1)* 2. Display Y (CI *(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

**4.4.2 Presenter class design**

1. Car Presenter
2. Log Presenter
3. Employee Presenter
4. Customer Presenter

**4.4.2.1 Car Presenter class**

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Algorithms**

The purpose of this class is to display car list , search car list and get booing form request.

Class type : Presenter class

Responsibility : To display car list , search car list and get booing form request

Attributes : display\_Carlist : string

Search\_Car : string

Booking\_form : string

1. Display\_Carlist

|  |
| --- |
| Display\_Carlist = DC(*1)* ,  DC *(2),* DC *(3)* ............ DC *(n)*  Display Y ; |
| **Begin :**   1. Read DC *(1)* 2. Display Y (DC *(1)* ) 3. Repeat step 1 until 2 with DC *(2)*.......... DC *(n)*   **End** |

1. Search\_Car

|  |
| --- |
| Search\_Car = SC(*1)* , SC *(2),* SC *(3)* ............ SC *(n)*  Display Y ; |
| **Begin :**   1. Read SC *(1)* 2. Display Y ( SC *(1)* ) 3. Repeat step 1 until 2 with SC *(2)*.......... SC *(n)*   **End** |

1. Booking\_form

|  |
| --- |
| Booking\_Form = BF(*1)* ,  BF *(2),* BF *(3)* ............ BF *(n)*  Display Y ; |
| **Begin :**   1. Read BF *(1)* 2. Display Y ( BF *(1)* ) 3. Repeat step 1 until 2 with BF *(2)*.......... BF *(n)*   **End** |

**4.4.2.2 Login Presenter class**

**Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Algorithms**

The purpose of this class is to verify the username and password.

Class type : Presenter class

Responsibility : To control the login process

Attributes : none

Methods : verify( username: String, password : String)

1. Verify()

Responsibility : to verify username and password

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Verify(username, password);

END

* + - 1. **Employee Presenter class**

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Algorithms**

The purpose of this class is to display booking list, add , edit and delete record.

Class type : Presenter class

Responsibility : to display booking list, add , edit and delete record.

Attributes :

Methods : display\_bookinglist () : void

Delete\_record () :void

Add\_record () : void

Edit\_record () : void

1. display\_ bookinglist ()

Responsibility : allow employee to display booking information

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

display\_ bookinglist ();

END

1. Add\_record ()

Responsibility : allow employee to add a booking request

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Add\_record ();

END

1. Delete\_record ()

Responsibility : allow employee to delete a booking request

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Delete\_record ();

END

1. Edit \_record ()

Responsibility : allow employee to edit a booking request

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Edit \_record ();

END

**4.4.3 View class design**

1. Login page
2. Admin manage booking page

**4.4.3.1 Login page class**

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Algorithms**

The purpose of this class is to login into system.

Class type : view class

Responsibility : to create interface for login

Attributes : not applicable

Methods : doLogin()

1. doLogin()

Responsibility : to create interface for login

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

doLogin( username, password);

END

**4.4.3.2 Admin manage booking class**

1. **Input/output data elements**

List of input and output data elements :-

Input : none

Output : none

1. **Algorithms**

The purpose of this class is to manage booking .

Class type : view class

Responsibility : to create interface for add edit delete display page

Attributes : not applicable

The purpose of this class is to display booking list, add , edit and delete record.

Class type : view class

Responsibility : to display booking list, add , edit and delete record.

Attributes :

Methods : display\_bookinglist () : void

Delete\_record () :void

Add\_record () : void

Edit\_record () : void

1. display\_ bookinglist ()

Responsibility : none

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

display\_ bookinglist ();

END

1. Add\_record ()

Responsibility : none

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Add\_record ();

END

1. Delete\_record ()

Responsibility : none

Input parameter : none

Output parameter : none

Algorithm :

BEGIN

Delete\_record ();

END

1. Edit \_record ()

Responsibility : none

Input parameter : none

Output parameter : none

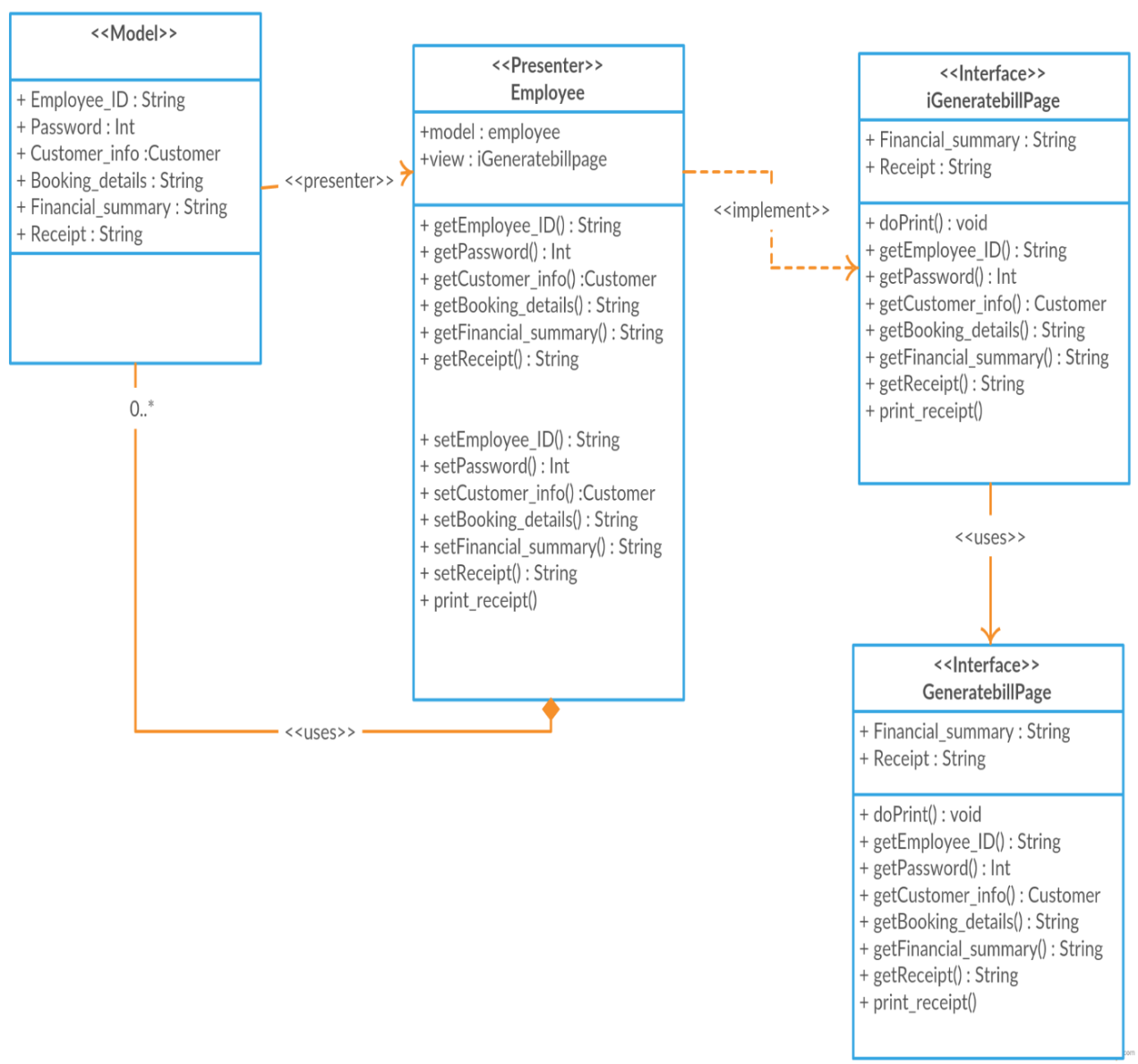
Algorithm :

BEGIN

Edit \_record ();

END

**4.5 Generate Bill Subsystem (SDD-REQ-S05)**



**Figure 4.5 : Generating Bill Detail Design**

**4.5.1 Model class design**

This subparagraph specifies the design of model class.

1. Employee
2. Receipt
3. Financial Summary
   * + 1. **Employee class**
4. **Input/output data elements**

List of input and output data elements :

Input : Employee\_ID, Password, Booking\_Details , Customer\_info

Output : get employee information

**b) Algorithms**

The purpose of this class is to allow employee to generate receipt and financial summary.

Class type : Model class

Responsibility : To generate receipt and input financial summary into system

Attributes : Employee\_ID : String

Password : Int

Booking\_details : String

Customer\_info : customer

**1) Employee\_ID**

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID  *(1)* 2. Display Y (ID  *(1)* 3. Repeat step 1 until 2 with ID*(2)*.......... ID*(n)*   **End** |

**2) Password**

|  |
| --- |
| Password = P*(1)* , P *(2),* P *(3)* ............ P *(n)*  Display Y ; |
| **Begin :**   1. Read P  *(1)* 2. Display Y (P *(1)* ) 3. Repeat step 1 until 2 with P*(2)*.......... P*(n)*   **End** |

**3) Booking\_Details**

|  |
| --- |
| Booking\_Details = BD(*1)* ,  BD*(2),* BD *(3)* ............ BD *(n)*  Display Y ; |
| **Begin :**   1. Read BD *(1)* 2. Display Y ( BD *(1)* ) 3. Repeat step 1 until 2 with BD *(2)*.......... BD *(n)*   **End** |

**4) Customer\_Info**

|  |
| --- |
| Customer\_Info = CI(*1)* ,  CI *(2),* CI *(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI *(1)* 2. Display Y (CI *(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

* + - 1. **Financial Summary class**

1. **Input/output data elements**

List of input and output data elements :-

Input : Employee\_ID,Booking\_details,Date

Output : get details about all financial costs

1. **Algorithms**

The purpose of this class is to calculate financial summary.

Class type : Model class

Responsibility : calculate financial summary

Attributes : Employee\_ID : String

Password : Int

Booking\_Details : String

Date : Date

**1) Employee\_ID**

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID  *(1)* 2. Display Y (ID  *(1)* ) 3. Repeat step 1 until 2 with ID*(2)*.......... ID*(n)*   **End** |

1. **Password**

|  |
| --- |
| Password = P*(1)* , P *(2),* P *(3)* ............ P *(n)*  Display Y ; |
| **Begin :**   1. Read P  *(1)* 2. Display Y (P *(1)* ) 3. Repeat step 1 until 2 with P*(2)*.......... P*(n)*   **End** |

**3) Booking\_Details**

|  |
| --- |
| Booking\_Details = BD(*1)* ,  BD*(2),* BD *(3)* ............ BD *(n)*  Display Y ; |
| **Begin :**   1. Read BD *(1)* 2. Display Y ( BD *(1)* ) 3. Repeat step 1 until 2 with BD *(2)*.......... BD *(n)*   **End** |

**4) Date**

|  |
| --- |
| Date = D(*1)* ,  D *(2),* D *(3)* ............ D *(n)*  Display Y ; |
| **Begin :**   1. Read D *(1)* 2. Display Y ( D *(1)* ) 3. Repeat step 1 until 2 with D *(2)*.......... D *(n)*   **End** |

* + - 1. **Receipt class**

1. **Input/output data elements**

List of input and output data elements :

Input : Employee\_id,Booking\_Details,Customer\_info

Output : create receipt

**b) Algorithms**

The purpose of this class is to create receipt

Class type : Model class

Responsibility : To input information and create receipt

Attributes : Employee\_ID : String

Booking\_Details : String

Customer\_info : String

1. **Employee\_ID**

|  |
| --- |
| ID = ID *(1)* , ID*(2),* ID *(3)* ............ ID *(n)*  Display Y ; |
| **Begin :**   1. Read ID  *(1)* 2. Display Y (ID  *(1)* ) 3. Repeat step 1 until 2 with ID*(2)*.......... ID*(n)*   **End** |

**2) Booking\_Details**

|  |
| --- |
| Booking\_Details = BD(*1)* ,  BD*(2),* BD *(3)* ............ BD *(n)*  Display Y ; |
| **Begin :**   1. Read BD *(1)* 2. Display Y ( BD *(1)* ) 3. Repeat step 1 until 2 with BD *(2)*.......... BD *(n)*   **End** |

**3) Customer\_Info**

|  |
| --- |
| Customer\_Info = CI(*1)* ,  CI *(2),* CI *(3)* ............ CI *(n)*  Display Y ; |
| **Begin :**   1. Read CI *(1)* 2. Display Y (CI *(1)* ) 3. Repeat step 1 until 2 with CI *(2)*.......... CI *(n)*   **End** |

# 5.0 DATA DICTIONARY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Key | Reference | Description | |
| Employee\_ID | varchar(8) | PK |  | A unique ID to identify each Employee | |
| Customer\_ID | varchar(10) |  |  | A unique ID to identify each Customer | |
| Username | varchar(20) |  |  | Contain username for employee and customer | |
| Password | varchar(10) |  |  | Contain password for employee and customer | |
| Name | varchar(20) |  |  | | Contain name for employee and customer |
| Phone\_Number | int(15) |  |  | A field that indicate customer and employee phone number | |
| Address | varchar(50) |  |  | A field that indicate customer and employee address | |
| License\_Number | varchar(20) |  |  | A field that indicate customer license | |
| Car\_ID | varchar(10) |  |  | Contain car\_id | |
| Car\_Name | varchar(20) |  |  | Contain car\_name | |
| Plate\_Num | varchar(10) |  |  | A field that indicate car plate number | |
| Car\_History | varchar(20) |  |  | A field that indicate customer | |
| Available\_Date | varchar(10) |  |  | A field that indicate available date car | |
| Time\_Borrowed | varchar(10) |  |  | A field that indicate time | |
| Date\_Borrowed | varchar(20) |  |  | A field that indicate date | |
| No\_of\_Car\_Available |  |  |  | Contain number of car available | |
| Collect\_Data | varchar(20) |  |  | A field that indicate customer booking | |
| Return\_Date | varchar(20) |  |  | A field that indicate customer car return date | |
| Car\_Availibity | varchar(20) |  |  | Contain car\_availibity details | |
| Booking request | double(10) |  |  | Contain customer booking request | |
| Customer\_Info | double(10) |  |  | Contain customer\_info data | |
| Rent | double(10) |  |  | Amount rent | |
| MaintenanceCost | double(10) |  |  | Cost for car maintenance | |
| Other\_Cost | double(10) |  |  | Cost spend for other use | |
| EmployeeSalary | double(10) |  |  | Salary given to employee | |
| ReceiptNO | int(10) |  |  | Contain receiptNo details | |
| ReceiptDetails | varchar(70) |  |  | Contain receipt details | |
| FinancialDetails | varchar(100) |  |  | Contain financial details | |
| BookingDetails | varchar(50) |  |  | Contain booking details | |
| ReportReference | varchar(100) |  |  | Contain report reference number | |
| Date | varchar(10) |  |  | Date of the report created | |
| TotalAmount | varchar(10) |  |  | A field that indicate total amount of packaged | |
| ReportID | varchar(10) | PK |  | A field that indicate report ID | |
| Status\_Available | varchar(8) |  |  | A field that indicate the status available of the ca | |

**6.0 SPECIFIC MODULE REQUIREMENTS**

|  |  |
| --- | --- |
| SDD-REQ-S01 | Customer should be allowed to register themselves to use the system by choosing a unique id and password during the registration process. |
| SDD-REQ-S02 | Employee should be able to update car information based on customer required |
| SDD-REQ-S03 | Employee should be able to update castomer information based on customer required |
| SDD-REQ-S04 | Employee should be able to manage booking based on customer required |
| SDD-REQ-S05 | Employee should be able to update generate bill based on booking details |

**7.0 ABBREVIATION**

* UMP Universiti Malaysia Pahang
* OBA On Board Automobile
* OOAD Object Oriented Analysis Design
* SDD Software Development Document
* SDP Software Development Plan
* SRS Software Requirement Specification
* UML Unified Modelling Language