SCSJ2203: Software Engineering

System Documentation

UTM ALIS System

Version 3.0

03 February 2021

School of Computing, Faculty of Engineering

Prepared by: <ROLLNEY>

Revision Page

a. Overview

The content of current document contains the Overall Description, the specific requirement and system features that includes Use Case Diagram, Use Case Specification with extra details using Activity Diagram and also Sequence Diagram.

b. Target Audience

- Stakeholders
- Business Analyst
- Programmers

c. Project Team Members

Name	Task
Arif Masyhur Bin Mohamed Hatta (B19EC0007) (Team Leader)	• Decision table for UC012 (Make Reservation) • Update Sequence Diagram
Nur Atiqah Binti Mohd Fua'ad (B19EC0032)	• State diagram for UC012 (Make Reservation)
Mohamad Faiz Hakimi Bin Ishak (A18CS0107)	• Activity diagram for UC004 (Register Activity) • Activity diagram for UC012 (Make Reservation)
Norhasliana Amiza Binti Abdullah (A18CS0178)	• Decision table for UC004 (Register Activity) • Update Sequence Diagram

d. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
Version 3.0	Norhasliana Amiza Binti Abdullah	Completed SDD Version 3.0	02/02/2021

Table of Contents

1. Introduction

1.1 Purpose

This Software Document describes the detailed requirements of “UTM ALIS” web-based application. The aim for the development of the method will be demonstrated in this document. The system constraints, interface and interactions with other external applications will explained here. This document is specifically meant to proposed acceptance by the UTM Alumni as a guide for the development of the first version of the system for the development team.

1.2 Scope

Generally, this web-based application, UTM ALIS bring benefit to UTM Alumni and Non-Alumni which is this application will help them finding their industrial partner based on what their applying for jobs or their research. Through this application, they also can make donation anytime and anywhere using DuitNow or Credit Card. Using this application also, they can reserve an available facility. The “UTM ALIS” should be directly found in a website using any platform in mobile phone, personal computer and others.

System Admin provide an information about statistics by extracting data through UTMSPACE and SRAD. When the user opens the website, there is also news for them that they can read. System Admin uses UTM ALIS to administer the system and store all the user information accurate and securely.

Hence, the internet connection is needed when the user wants to access into the UTM ALIS. All the information will be store in database that located on web-server. The application has the capability of representing detailed information about statistics and leaderboard of UTM ALIS.

1.3 Definitions, Acronyms and Abbreviation

No.	Acronyms and Abbreviation	Definitions
1.	ALIS	Alumni Integrated System
2.	Alumni	A former student, especially a male one, of particular school, college or university
3.	UTM	University Teknologi Malaysia
4.	SRAD	An existing UTM student management system called Student Recruitment and Admission Division
5.	UTMSPACE	An existing UTM student management system.
6.	DuitNow	An e-wallet application that enables you to transfer funds to your recipient's DuitNow ID instead of their bank account number.

1.4 References

I. IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998

II. System documentation: Part 3 (version 1.0 ed.). (2019). UTM ALIS system.

III. Wan Kadir, Wan Mohd Nasir. “Lab 3 - Step 5: Presentation Layer Implementation.” *YouTube*, YouTube, 29 Nov. 2020, www.youtube.com/watch?v=arn7si1FSP0.

IV. Wan Kadir, Wan Mohd Nasir. “Lab 3 - Step 4: Business Link Layer Implementation.” *YouTube*, YouTube, 29 Nov. 2020, www.youtube.com/watch?v=tM2ldiI0EcI.

V. Wan Kadir, Wan Mohd Nasir. “Lab 3 - Step 3: Data Link Layer / Middleware Layer Implementation.” *YouTube*, YouTube, 29 Nov. 2020, www.youtube.com/watch?v=8b1S1Zrdk8M.

VI. Wan Kadir, Wan Mohd Nasir. “Lab 3 - Step 2: Create a VS Project & Add the Generated Source Codes.” *YouTube*, YouTube, 29 Nov. 2020, www.youtube.com/watch?v=xS1Qlf4pnSc.

VII. Wan Kadir, Wan Mohd Nasir. “Lab 3 - Step 1: Code & Database Engineering.” *YouTube*, YouTube, 29 Nov. 2020, www.youtube.com/watch?v=RM5KPGa2Cgc.

1.5 Overview

This document describes provides an insight of the overall product with the description. Next, it also describes product perspective, the types of interfaces such as user interface, hardware interface and etc. Requirements specification in detailed terms and a description of the external system interfaces and the system features are provided in this document. It is also describing the UTM ALIS's subsystem by providing the Use Case Diagram, Use Case Specification, Sequence Diagram and Activity Diagram. Performance requirements, design constraints, software system attributes, System Architectural Design, Detailed Description of Components and other related specification are also included.

2. Specific Requirements

2.1 External Interface Requirements

2.1.1 User Interfaces

Because the system is a web-based system, thus the user interface will be revolving around web pages. The main web pages as listed below:

- ●Home Page
- ●Make Donation Page
- ●Activity Registration Page
- ●Facility Reservation Page

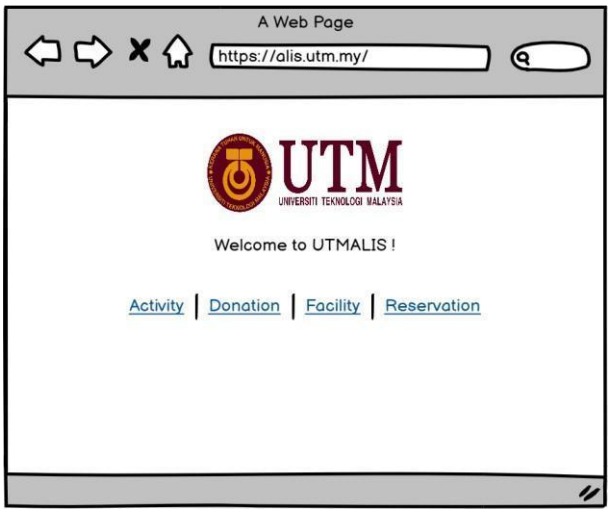


Figure 2.1.1.1: Prototype Interface of Home Page

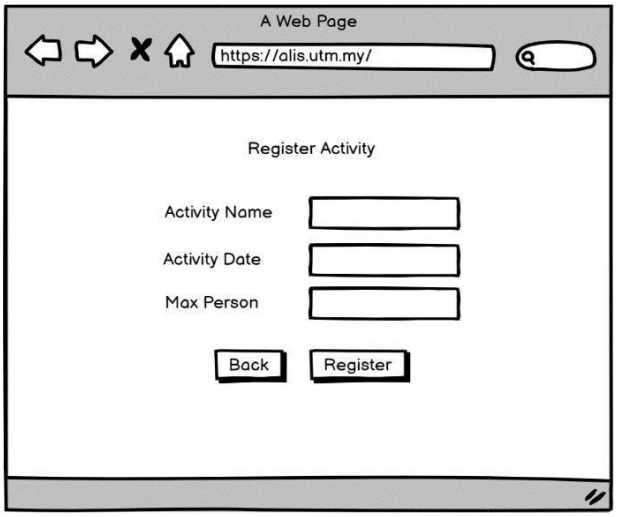


Figure 2.1.1.2: Prototype Interface of Activity Registration Page

Based on Figure 2.1.1.1 and Figure 2.1.1.2, it shows as examples of the interfaces where there will be button and hyperlink that linked to other webpage form. The user will interact using GUI and computer peripherals such as mouse and keyboard. The user interface for the software shall be compatible to any browser such as Internet Explorer, Mozilla or Netscape Navigator by which user can access to the system.

2.1.2 Hardware Interfaces

The hardware needed to run this system can be divided in two categories. One is for computers and the other is mobile. For the system to operate smoothly, the computer should at least have this minimum requirement specification:

i. 2GB RAM
ii. 10 GB of free space
iii. Intel Pentium G3258 or equivalent
iv. Connected to Monitor
v. Connected to Keyboard
vi. Connected to Mouse
vii. Internet capable

On mobile, the minimum requirement specification should be:

i. 2GB RAM
ii. 4GB of free space
iii. Octa-core 1.2 GHz Cortex-A53 or equivalent
iv. Wi-Fi 802.11 b/g/n or HSPA 42.2/5.76 Mbps, LTE Cat4 150/50 Mbps capable

2.1.3 Software Interfaces

The UTM ALIS system shall interact with the following systems:

DuitNow	This interface is made for allowing the UTM Alumni to make donations. The system will support an interface with the DuitNow.
Operating System	Windows 10 or previous version. This OS is chosen for its best support and user-friendliness which helps the developers to develop the system without any complications.
Web Browser	As the system is a web-based system, it should be able to interact with browsers to provide its service.
UTMSPACE	To import or export data to this interface
SRAD	To import or export data to this interface

2.1.4 Communication Interfaces

- The users can connect to UTM ALIS services under the standard http protocol.
- Since the system is web-based, it will use port 80
- Port 80 is the port that the server "listens to" or expects to receive from a Web client, assuming that the default was taken when the server was configured or set up.
- As users may download from the system, TCP will also be implemented.

2.2 System Features

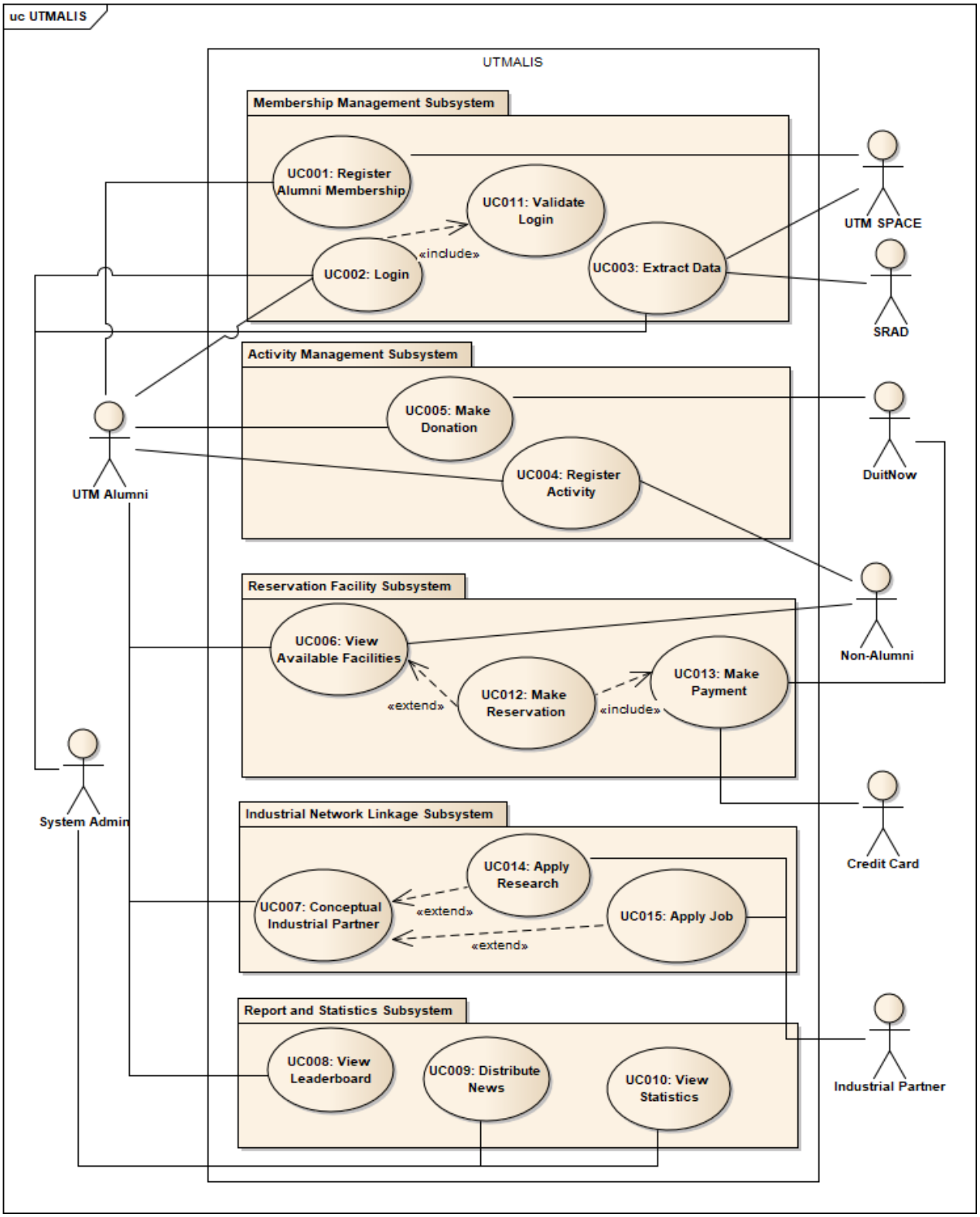


Figure 2.2.1: Use Case Diagram for <UTMALIS>

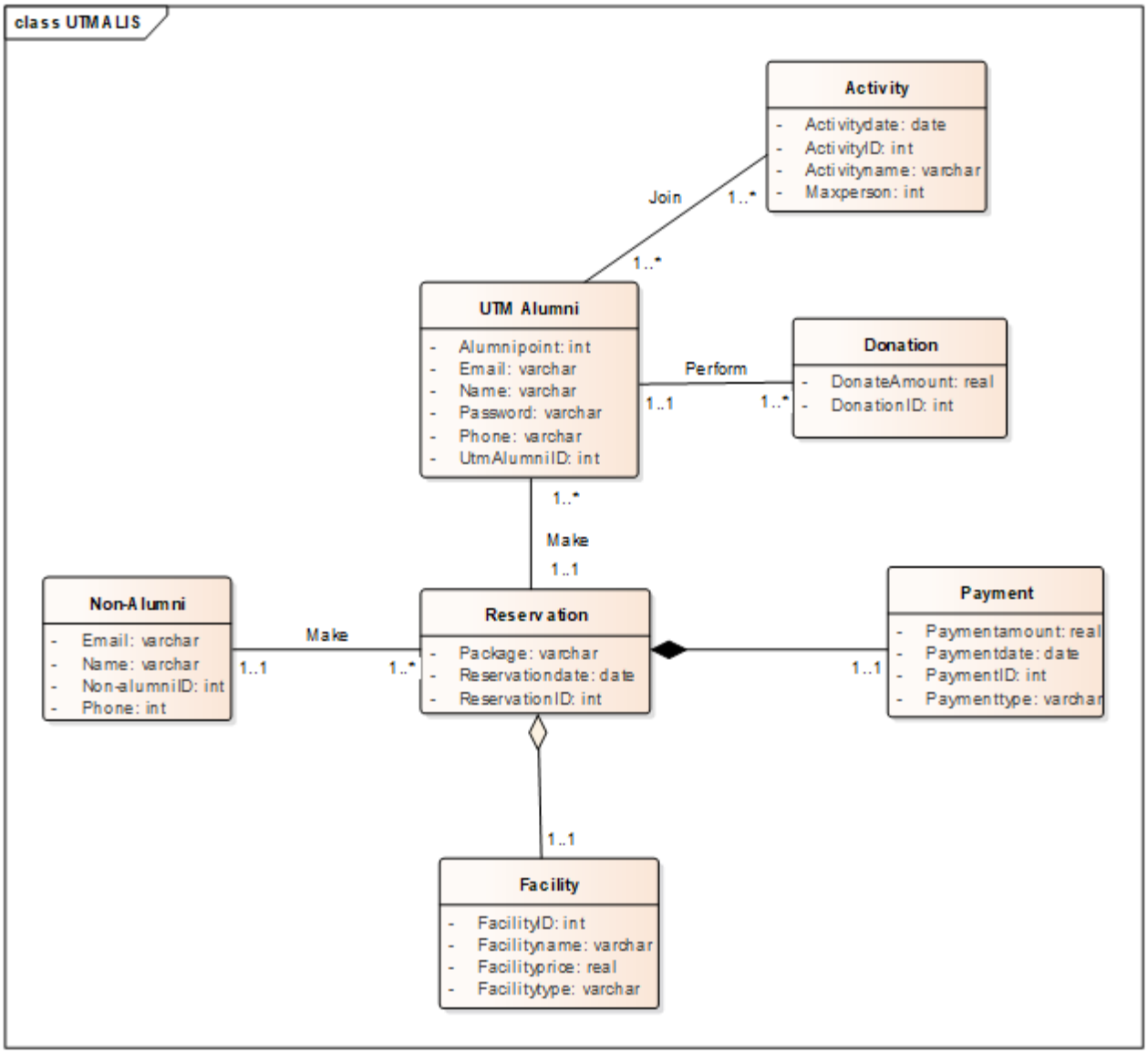


Figure 2.2.2: Domain Model for <UTMALIS>

2.2.1 UC004: Use Case <Register Activity>

Table 2.1: Use Case Description for <Register Activity>

Use case: <Register Activity>
ID: UC004
Actors: 1. UTM Alumni 2. Non-Alumni
Pre-conditions: 1. There is active connection to UTM ALIS 2. User has logged in to the system
Flow of Events: 1. The use case starts when the UTM Alumni/Non-Alumni enters the register activity page. 2. The ALIS system displays the available list of activities. 3. The user chooses the available activity from the displayed list by clicking the name of activity. 4. The ALIS system displays the chosen activity details. 5. The UTM Alumni/Non-Alumni must fill in the form that the system displayed. 6. The UTM Alumni/Non-Alumni submit the form by clicking the “Submit” button. 7. If the form is successful submitted 7.1. ALIS system displays the message “Your registration is successful”. 7.2. If the user == UTM Alumni 7.2.1 ALIS system calculates the Alumni Points and updates them. 7.2.2 The use case ends. 7.3. Else 7.3.1 The use case ends. 8. Else 8.1. Exception 1 is executed.

Post-conditions: Successful compilation 1.1 The activity log has been updated. 1.2 UTM Alumni log has been updated.
Exception Flow: Incomplete form 1.1 The system displays an error message “The form is Incomplete!” 1.2 Use case resume at normal flow 5. UTM Alumni can opt out of the page anytime.

2.2.2 UC005: Use Case <Make Donation>

Table 2.2: Use Case Description for <Make Donation>

Use case: < Make Donation >
ID: UC005
Actors: 1. UTM Alumni 2. DuitNow
Pre-conditions: 1. There is an active connection to UTMALIS. 2. The user has logged in to the system. 3. The user has a DuitNow account.
Flow of Events: 1. The use case starts when UTM Alumni enter the donation page. 2. UTM Alumni clicks the “Make Donation” button to start the donation process. 3. The system prompts UTM Alumni to enter an amount of donation. 4. UTM Alumni enters the amount of the donation. 5. UTM Alumni clicks the “Donate” button. 6. The system initiates the DuitNow system. 7. The system sends the amount to DuitNow system 8. The DuitNow system displays QR code. 9. UTM Alumni scans QR code using his/her phone. 10. UTM Alumni clicks the “Confirm” button to pay the total amount due. 11. If balance is insufficient 11.1 Exception flow 1 is executed. 12. Else 12.1 “DuitNow” Records the transactions. 12.2 “DuitNow” sends a successful message “Transaction completed”. 12.3 The system displays successful message “Thank you for Making

Reservation”

13. The system calculates Alumni Point
14. The system updates Alumni Point

Post-conditions:

1. UTM Alumni success makes donations.
2. UTMALIS will receive the donation from the Alumni.
3. The system log has been updated.

Exception Flow:

1. Insufficient balance

   1.1 DuitNow displays error message “Insufficient balance”

   1.2 Use case ends

2.2.3 UC006: Use Case <Available Facilities>

Table 2.3: Use Case Description for <Available Facilities>

Use case: <Available Facilities>
ID: UC006
Actors: 1. UTM Alumni 2. Non-Alumni
Pre-conditions: 1. There is active connection to UTM ALIS 2. User has logged in to the system
Flow of Events: 1. The use case starts when UTM Alumni/Non-Alumni enters the view facility page. 2. ALIS system shows a list of facilities available to be rented. 3. UTM Alumni/Non-Alumni choose a facility by clicking on the name of the facility. 4. ALIS system displays the details of that particular facility. 5. UTM Alumni/Non-Alumni click the “Rent This Facility” button to make a reservation. 6. Use case UC011: Make reservation is performed. 7. The use case ends.
Alternative flow *n*: 1. The use can opt out of the page at any time
Post-conditions: ● The UTM Alumni/Non Alumni successfully viewed the facilities.

2.2.4 UC012: Use Case <Make Reservation>

Table 2.4: Use Case Description for <Make Reservation>

Use case: <Make Reservation>
ID: UC012
Actors: 1. UTM Alumni 2. Non-Alumni 3. DuitNow System 4. Credit card
Pre-conditions: 1. There is active connection to UTM ALIS 2. User has logged in to the system
Flow of Events: 1. The use case starts when the ALIS system shows the available date to be rented. 2. UTM Alumni/Non Alumni choose starting date by clicking on the date. 3. UTM Alumni/Non Alumni choose starting time by clicking on the displayed time. 4. UTM Alumni/Non Alumni choose the end date by clicking on the date. 5. UTM Alumni/Non Alumni choose end time by clicking on the displayed time. 6. ALIS system displays facility reservation forms to be filled. 7. UTM Alumni/Non Alumni enter his/her personal details. 8. UTM Alumni/Non Alumni enter total number of pax. 9. UTM Alumni/Non Alumni clicks the “next” button to proceed. 10. ALIS system display available package. 11. UTM Alumni/Non Alumni select a package by clicking on the available package. 12. ALIS system calculates total payment and deposit to be paid. 13. ALIS System display the total payment and deposit amount.

14. UTM Alumni/Non Alumni click “Proceed to Payment” button to do the payment.

15. If UTM Alumni/Non Alumni click the “Pay with Credit Card” button.

    15.1 Alternate flow 1 is executed.

16. Else

    16.1 Alternate flow 2 is executed.

17. ALIS system records the transaction.

18. The ALIS system records the reservation.

19. If UTM Alumni

    19.1 ALIS System calculates Alumni Points.

    19.2 ALIS System updates Alumni Points.

20. ALIS sends an email to UTM Alumni/Non Alumni about the complete reservation details.

21. Use case ends.

Alternative Flow:

1. Pay with Credit Card
   1.1 The credit card system displays the total amount and deposit due.
   1.2 UTM Alumni/Non Alumni enters credit card information.
   1.3 If information is invalid
      1.3.1 The credit card system displays error message
      1.3.2 The use case ends with indication of failure
   1.4 Else
      1.4.1 Credit card system records the transaction.
      1.4.2 ALIS System displays a successful message “Thank you for Making Reservation”.
2. Pay with “DuitNow”
   2.1 ALIS System initiates “DuitNow” Application.
   2.2 “DuitNow” displays QR code to be scanned.
   2.3 UTM Alumni/Non Alumni scans the QR code using his/her phone.
   2.4 UTM Alumni/Non Alumni clicks “Confirm” button to pay the total amount

Due.

2.5 If balance is insufficient

2.5.1 Exception flow 1 is executed.

2.6 Else

2.6.1 “DuitNow” Records the transactions.

2.6.2 “DuitNow” displays a successful message “Transaction completed”.

2.6.3 ALIS System displays a successful message “Thank you for Making Reservation”.

2.6.4 Use case resumes at Normal Flow 4.

Post-conditions: Successful Compilation

- The UTM Alumni/Non Alumni successfully reserved facilities.
- System log has been updated
- Alumni Points have been updated.
- Available facilities have been updated.

Exception Flow:

1. Insufficient balance

1.1 DuitNow displays an error message “Insufficient balance”.

1.2 Use case ends.

2.3 Performance Requirements

The system should be able to initiate DuitNow and Credit Card System within 10 seconds of request.

- The user shall be able to learn the system within 5 minutes of using.
- The system shall use icons to depict commonly known activity.
- The system shall be able to produce statistics within 5 seconds of request time.
- The system will not face any slowdown operation even after 1000 user that
- used the system simultaneously.

2.4 Design Constraints

The system shall only be develop using web-based programming language

- The system will only take a maximum 20 MB of memory usage.
- The system will only take a maximum 50 MB of storage usage.

2.5 Other Requirements

- The system shall be available 99% of the time
- The system shall encrypt every username and password
- The system will have less than 2 maintenances per month

3. System Architectural Design

3.1 Architecture Style and Rationale

We chose a layered architecture pattern as our architectural design pattern. Layered architecture is used to divide the system into different layers to show the component of every layer of the system. In our proposed system, we decided to implement three-layer architecture into our system. Using this architecture style, our proposed system is divided into 3 different layers which are application layer, domain layer and data access layer. Application layer is a layer that holds interfaces that the user or client will interact with the system. Domain layer is the layer that holds the business logic or features that will represent the business model of the system. Data access layer is the layer that is responsible to access the data from the database that is required by the system. It also stores and saves the new data into the database system. By using this architectural style, development on the system is much easier to be carried out because we can develop according to the layers which will not affect the system entirely. Not only that, it is more suitable for a group of software engineers working on layered patterns as their tasks can be distributed based on layers.

3.2 Component Model

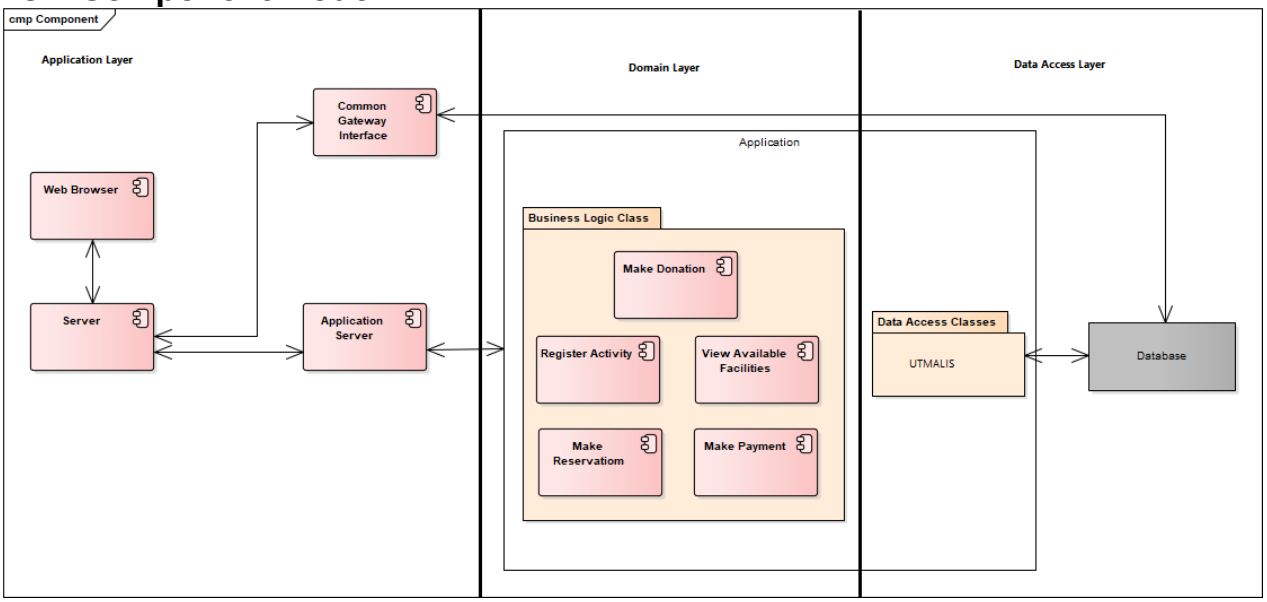


Figure 3.2: Component Diagram of <UTMALIS>

4. Detailed Description of Components

4.1 Complete Package Diagram

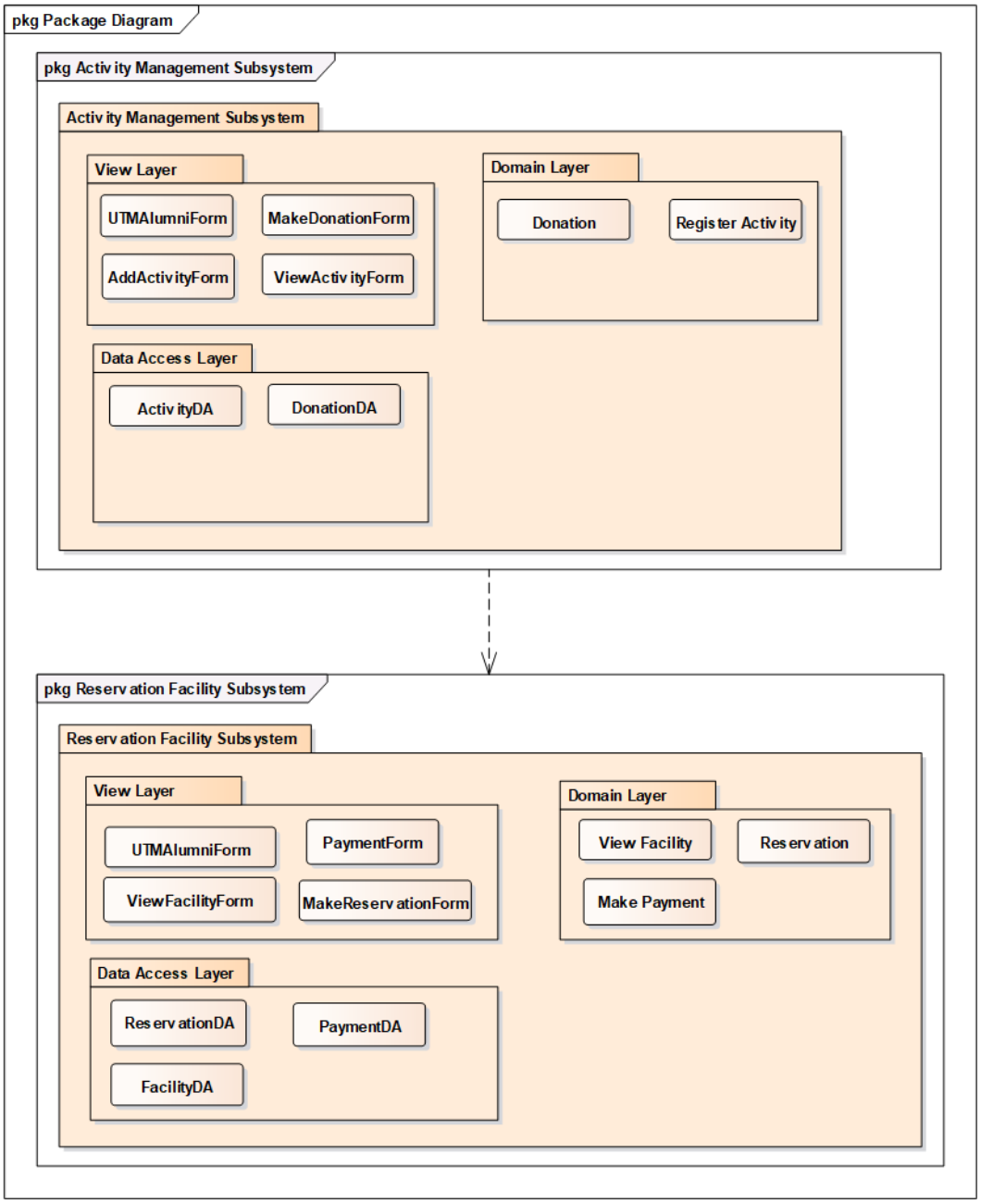


Figure 4.1: Package Diagram for <UTMALIS>

4.2 Detailed Description

In UTMALIS system, we choose two subsystems which are activity management subsystem and reservation facilities subsystem. These two subsystems can be structured into three layers that consists of view layer, domain layer and data access layer. View layer is the layer which handles user interface and receives user actions meanwhile domain layers is in charge of collecting entity objects and business logic that is related to the system. Next, data access layer is the layer which consists of database related to the system.

The activity management subsystem involves two use case which are Register Activity (UC005) and Make Donation (UC006). The subsystem would provide register activity interface to the user in the system. There is form for user to add the activity and make donation. The user also can update and delete the registered activity and the donation record. Besides, the subsystem also allow user to view their registered activity and the donation record by accessing data from the database.

For reservation facility subsystem, it is consisting of two use cases which are View Available Facilities (UC006) and Make Reservation (UC012). The subsystem will provide an interface for each use cases. The user can view the available facilities before make any reservation through this subsystem. All the data of available facilities are retrieved from the database. Hence, to make any reservation, the user has to fill in the form that have been provided. the submitted data will update in database and also in the reservation record. The user can update, delete and clear the record.

4.2.1 P001: <Activity Management> Subsystem

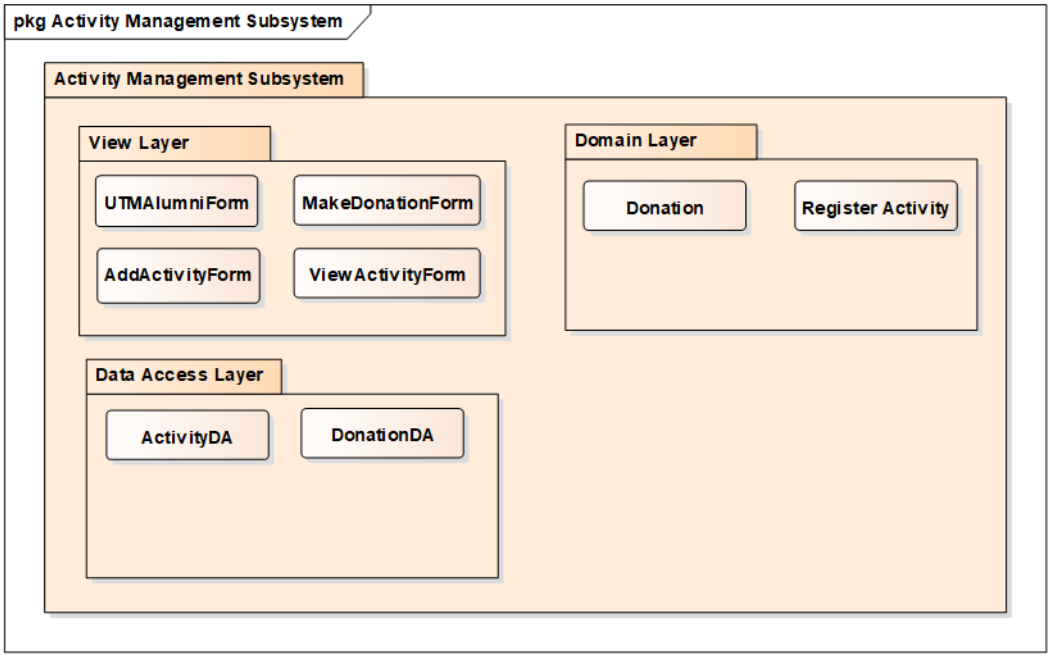


Figure 4.2.1: Package diagram for <Activity Management> Subsystem

4.2.1.1 Class Diagram for <Activity Management> Subsystem

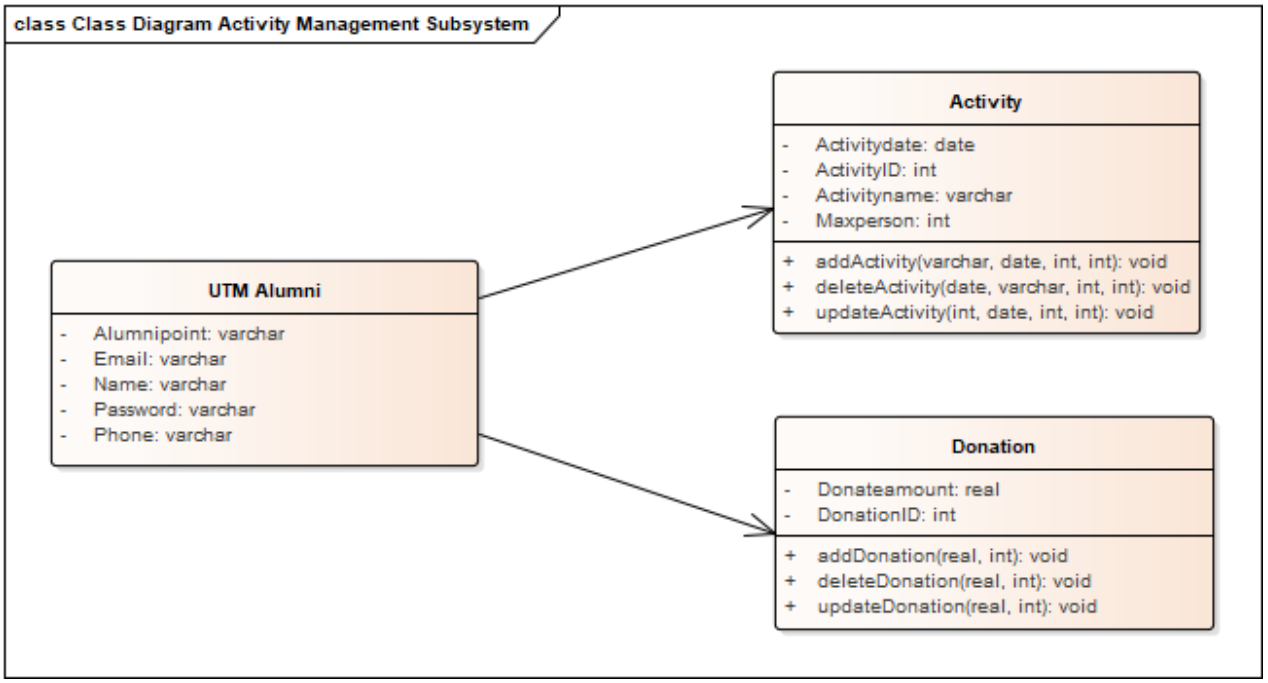


Figure 4.3: Class diagram for <Activity Management> Subsystem

Entity Name	Register Activity
Method Name	Display
Input	-
Output	Display register activity page
Algorithm	1. Start 2. Display homepage UTMALIS 3. Choose “Activity” 4. Display the activity page 5. End

Entity Name	Register Activity
Method Name	Register an activity
Input	Activitydate, ActivityID, Activityname, Maxperson
Output	Your registration activity is successful
Algorithm	1. Start 2. Insert activity ID 3. Choose activity date 4. Insert activity name 5. Insert the number of maximum people can join the activity 6. Click on “Add activity” button 7. End

Entity Name	Register Activity
Method Name	Update, Delete Activity
Input	-
Output	Update the activity page
Algorithm	1. Start 2. If the user wants to update the activity 3. Click on “Update Activity” button 4. Else the user wants to delete activity 5. Click on “Delete Activity” 6. End

Entity Name	Register Activity
Method Name	View registered activity
Input	-
Output	Display the registered activity
Algorithm	1. Start 2. Display homepage UTMALIS 3. Choose “View activity” 4. Display all the registered activity 5. If the user wants to register an activity 6. Click on “Manage Activity” button 7. End

Entity Name	Make Donation
Method Name	Display
Input	-
Output	Display makes donation page
Algorithm	1. Start 2. Display homepage UTMALIS 3. Choose “Make Donation” 4. Display “Make Donation” page 5. End

Entity Name	Make Donation
Method Name	Make donation
Input	Donateamount, DonationID
Output	Donation Successful
Algorithm	1. Start 2. Insert donation ID 3. Insert donation amount 4. Click on “Make donation” button 5. End

Entity Name	Make Donation
Method Name	Update, Delete Donation
Input	-
Output	Update the donation page
Algorithm	1. Start 2. If user want to update the donation 3. Click on “Update Donation” button 4. Else the user wants to delete donation 5. Click on “Delete Donation” button

	6. End
Entity Name	Make Donation
Method Name	View Donation Record
Input	-
Output	Display the donation record
Algorithm	8. Start 9. Display homepage UTMALIS 10. Choose “View Donation” 11. Display all the donation record 12. If the user wants to make donation 13. Click on “Donation Form” 14. End

4.2.1.1 Design Pattern for Class Activity (Abstract Factory Design Pattern)

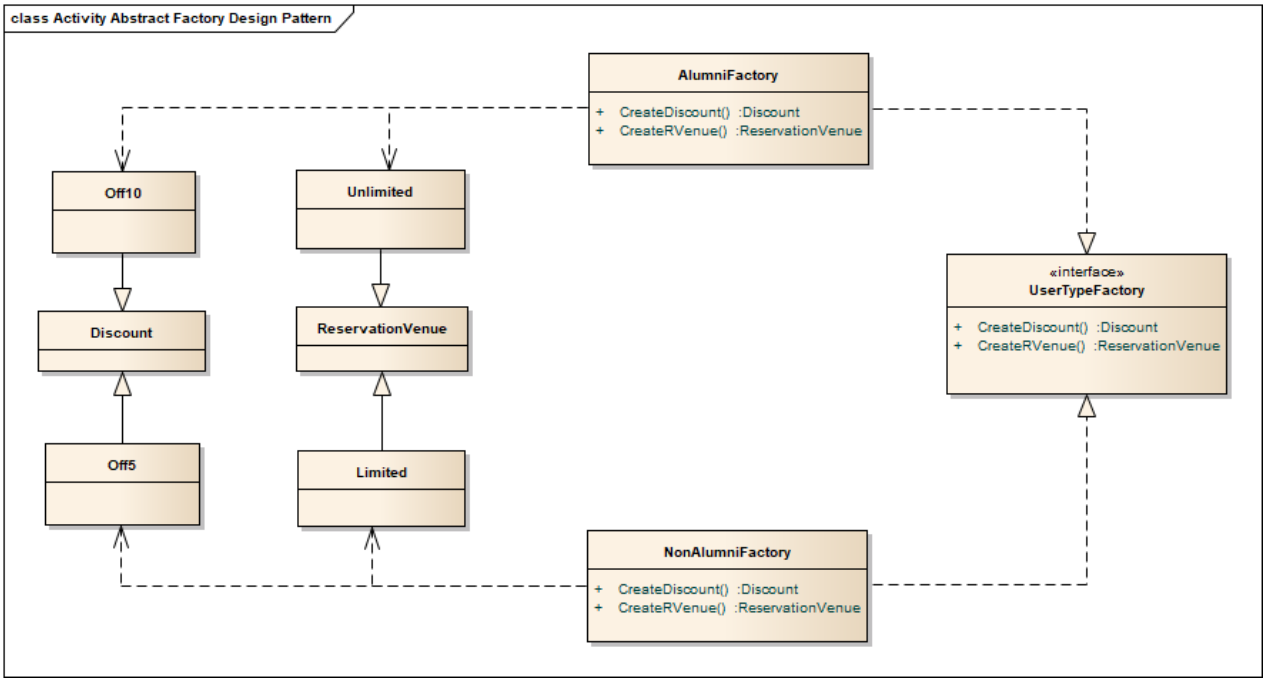


Figure 4.2.1.1: Class Diagram for Activity Class

The abstract factory pattern consists of an AbstractFactory, ConcreteFactory, AbstractProduct, ConcreteProduct and Client. It is providing interfaces for creating families of related or dependent objects without specifying their concrete classes. Besides, the client software creates a concrete implementation of the abstract factory, then it is using the generic interfaces to create the concrete objects that are part of the family of objects. However, the client does not know which concrete objects it gets from each of these concrete factories since it uses only the generic interfaces of their product.

For Activity class, we use this pattern (Abstract Factory Pattern) by defining a UserTypeFactory as an abstract factory that declares an interface for operations that create the objects. It does not instantiate the product object directly. Instead, there are two concrete factories which are AlumniFactory and NonAlumniFactory that implements the operations declared in the abstract factory (UserTypeFactory) for creating concrete products (Off10, Off5, Limited, Unlimited). Discount and ReservationVenue are two product objects that are created by the corresponding concrete factories (AlumniFactory

and NonAlumniFactory) and implement the abstract factory (UserTypeFactory) interface. Concrete factory (AlumniFactory and NonAlumniFactory) is inheriting from abstract factory (UserTypeFactory).

4.2.1.2 Design Pattern for Class Donation (Observer Design Pattern)

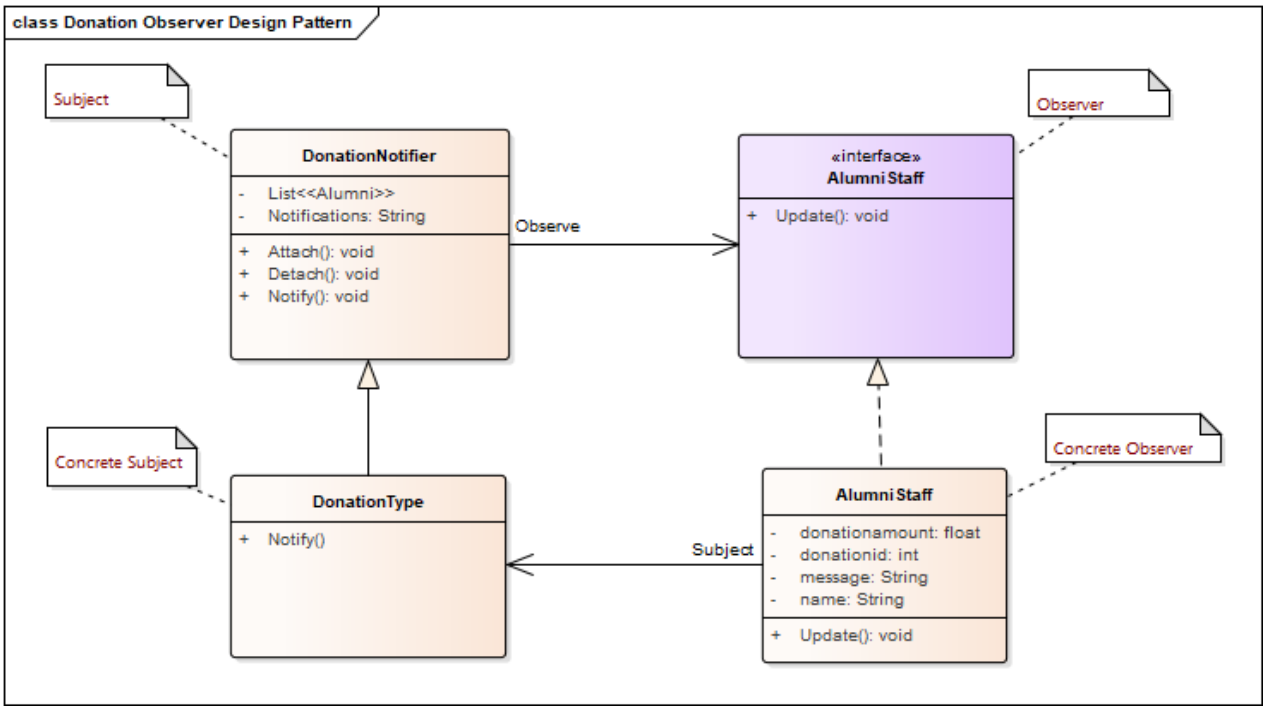


Figure 4.2.1.2: Class Diagram for Donation Class

Observer design pattern is an object behavioral pattern that standardizes the operation between objects that interoperate using a one-to-many relationship which an object, named the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods. In this case, an observer design pattern has been chosen to be implemented in this Donation class so that the Alumni will receive a confirmed notification message that they have successfully donated the donation and the amount of alumni points that they will receive after the donation.

Using this approach, Concrete Observer (Alumni Staff) examine notification information using the update parameter and act accordingly. Upon receiving change notification, concrete objects call a method of concrete subject to retrieve the details of the notified change. Since the behavior for attaching, detaching and notifying observers is the same, the Subject (DonationType) base can specify and implement the methods, which are inherited by all other concrete subjects.

First, design the subject (DonationNotifier) interface and implement code for attaching,

detaching, and notifying observer (Alumni Staff) objects. Second, the classes that manage information of interest to Observer (Alumni Staff) inherits from the subject class created in the first step. Third, design the Observer (Alumni Staff) interface which includes an abstract update interface method. Fourth, the Observer (Alumni Staff) in the system implements the observer interface which requires implementing the update method. Lastly, at run time, create the observer (Alumni Staff) and attach it to the subject (DonationNotifier).

4.2.1.3 Limited Entry Decision Table (LEDT) <<Register Activity>>

Condition	1	2	3	4
The user is an UTM Alumni	T	T	F	F
The user is a Non UTM Alumni	T	F	T	F
Action				
Unlimited Reservation Venue		X		
Limited Reservation Venue			X	
Discount 10%		X		
Discount 5%			X	
No Discount	X			X

4.2.1.4 Sequence Diagram

a) SD004: Sequence diagram for Activity Management

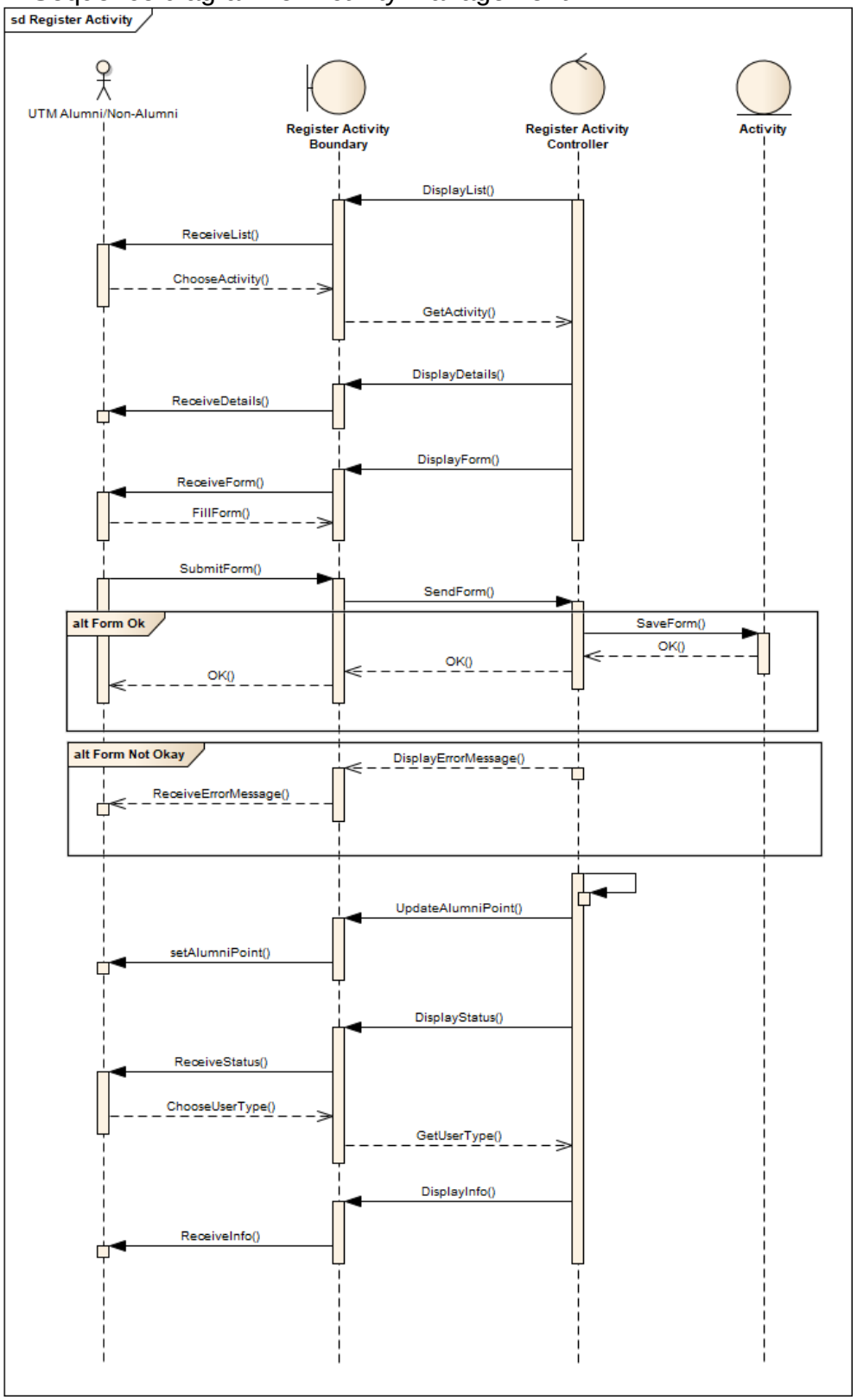


Figure 4.2.1.4.1: Sequence Diagram for <Activity Management>

b) SD005: Sequence diagram for Make Donation

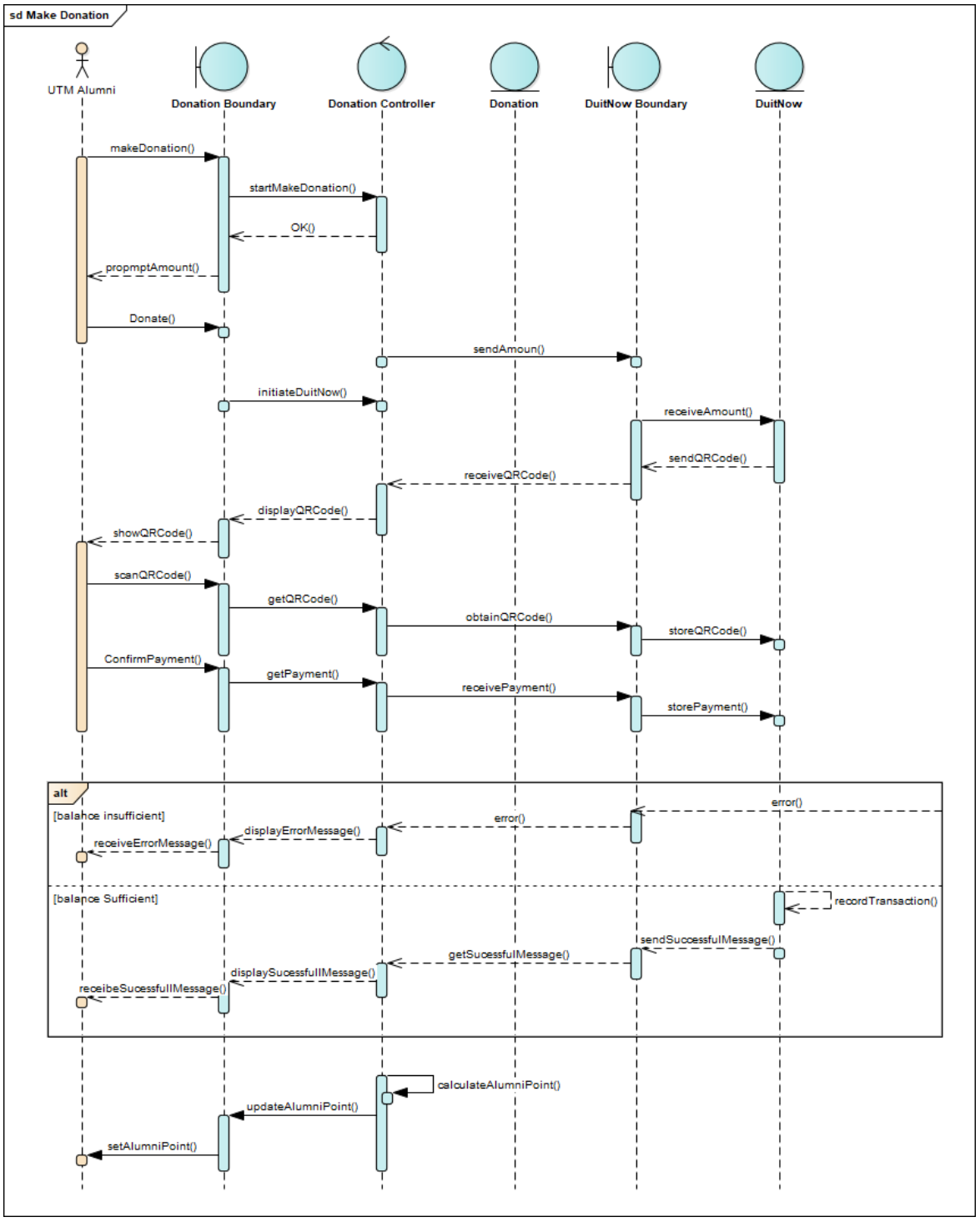


Figure 4.2.1.4.2: Sequence Diagram for <Make Donation>

4.2.1.5 Activity Diagram for Make Donation

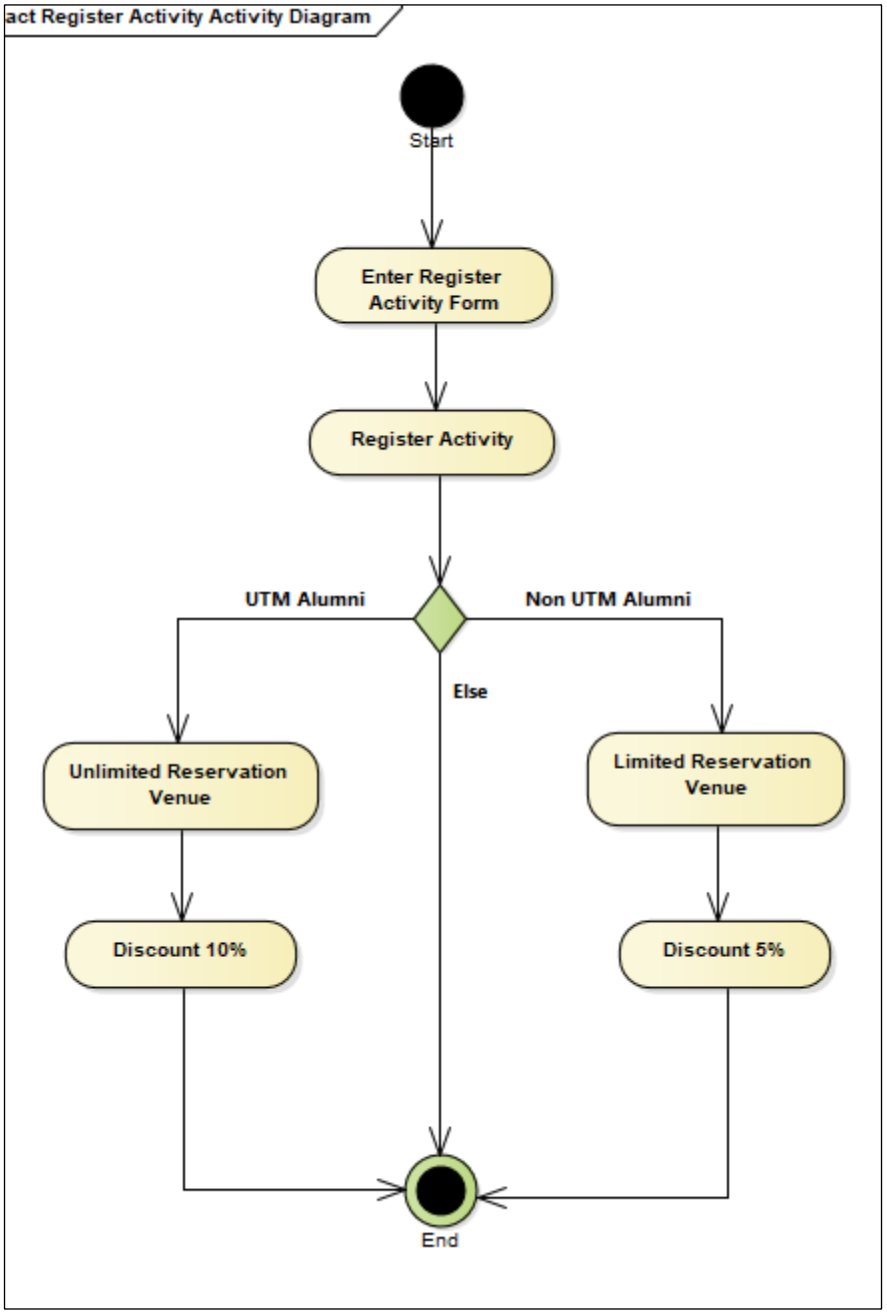


Figure 4.2.1.5: Sequence Diagram for <Make Donation>

4.2.2 P002: <Reservation Facility> Subsystem

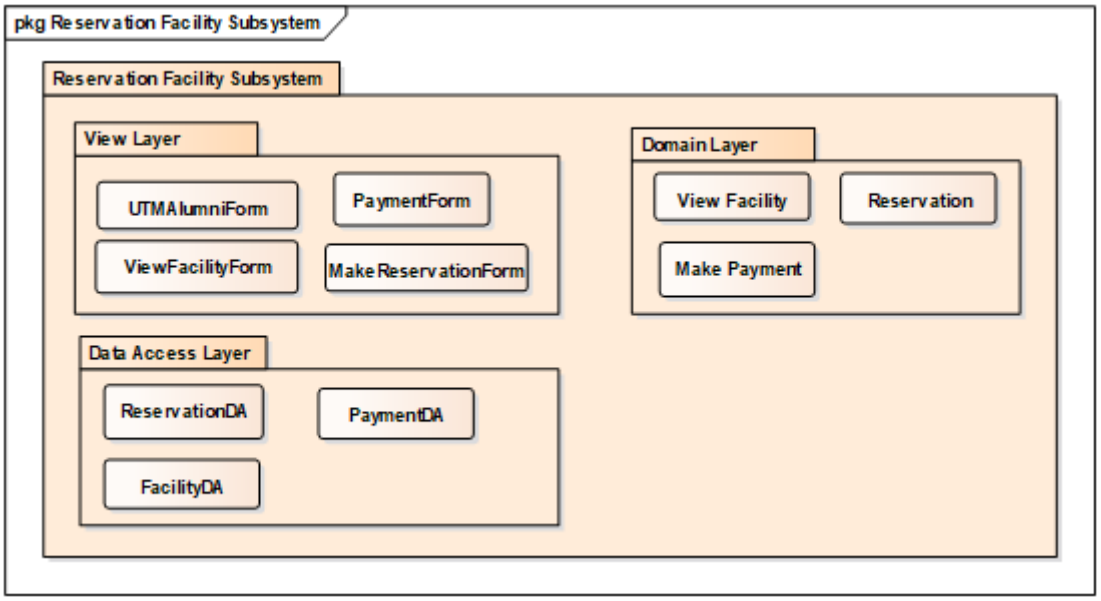


Figure 4.2.2: Package diagram for <Reservation Facility> Subsystem

4.2.2.1 Class Diagram for <Reservation Facility> Subsystem

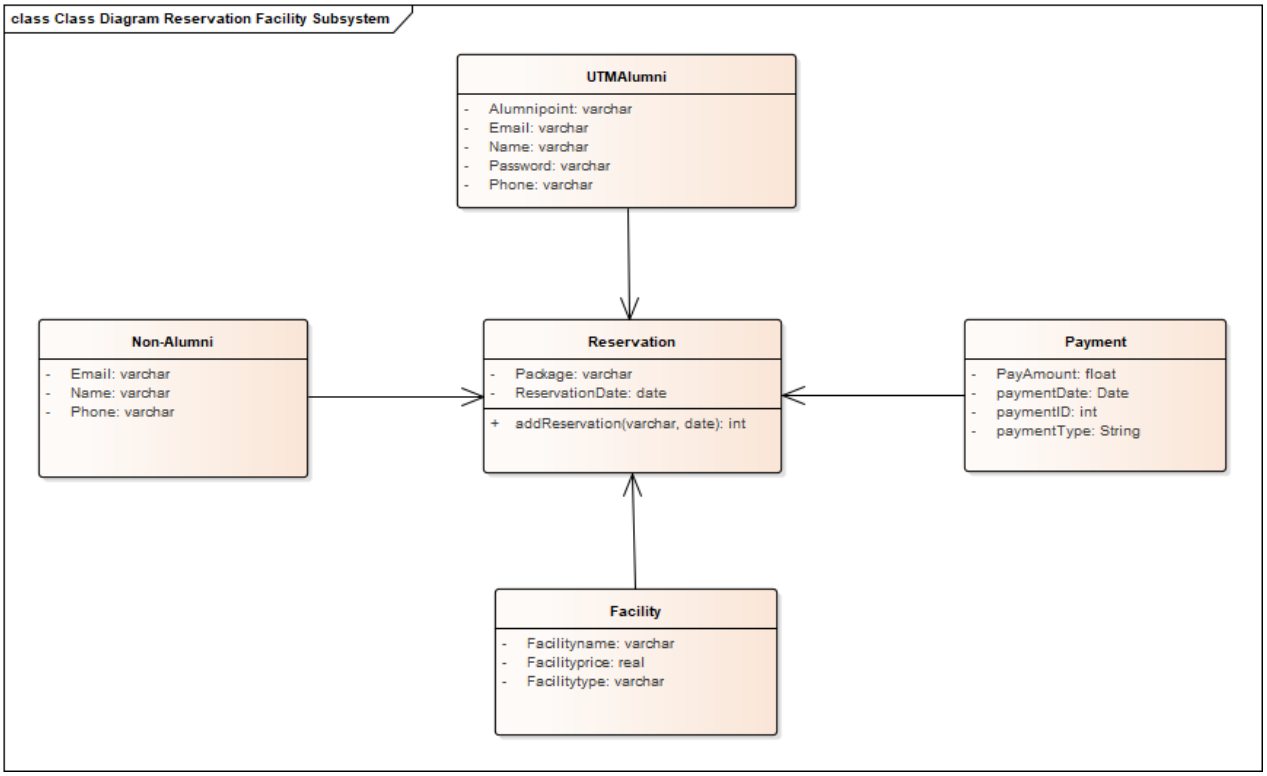


Figure 4.2.2.1: Class Diagram for <Reservation Facility> Subsystem

Entity Name	Available Facilities
Method Name	Display
Input	-
Output	Display available facilities page
Algorithm	1. Start 2. Display homepage UTMALIS 3. Choose “View Facilities” 4. Display the page 5. If the user wants to make reservation 6. Click on “Make reservation” button 7. End

Entity Name	Make Reservation
Method Name	Make reservation
Input	-
Output	Display makes reservation page
Algorithm	1. Start 2. Choose “Make Reservation” in home page 3. Display “Make Reservation” page 4. End

Entity Name	Make Reservation
Method Name	Make reservation
Input	Package, ReservationID, ReservationDate
Output	The reservation is successful
Algorithm	1. Start 2. Insert package 3. Insert reservation date 4. Insert reservation ID 5. Click in “Add Reservation” button 6. End

Entity Name	Make Reservation
Method Name	Make Payment
Input	ReservationID, paymentID, paymentType, paymentDate, PaymentAmount
Output	Payment successful
Algorithm	1. Start 2. Insert reservation ID 3. Insert payment ID 4. Choose payment type 5. Insert payment date 6. Insert payment amount 7. Click on “Make Payment” 8. End

4.2.2.2 State Machine Diagram for Make Reservation Class

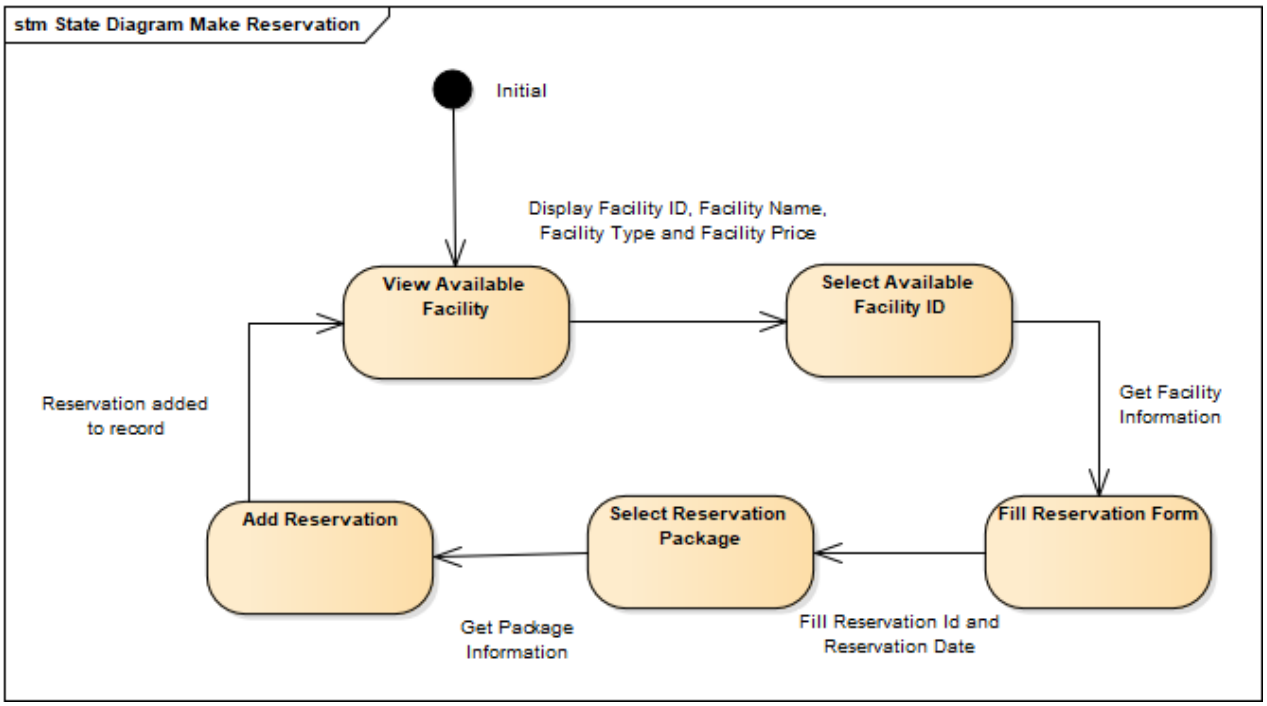


Figure 4.2.2.2: State Machine Diagram for Make Reservation Class

4.2.2.3 Design Pattern for Class Reservation (Factory Design Pattern)

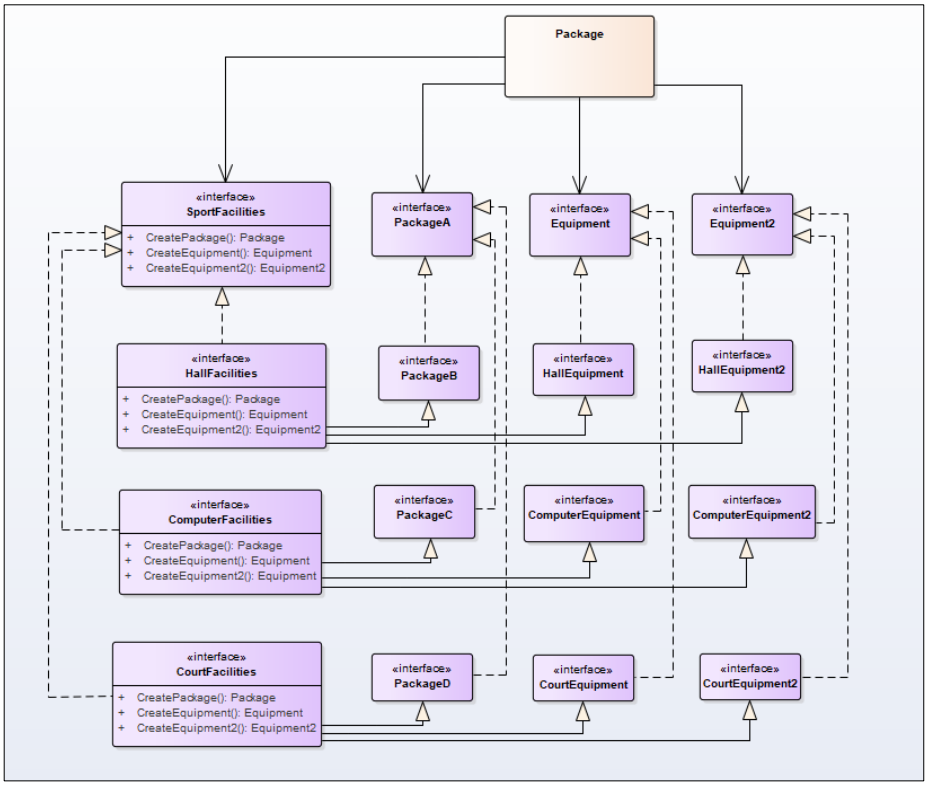


Figure 4.2.2.3: Class Diagram for Registration Class

Abstract factory method is an object-creational design pattern intended to manage and encapsulate the creation of a set of objects that conceptually belong together and that represent a specific family of products. It provides an interface for creating families of related objects without specifying their concrete classes.

For this use case we use the abstract factory method in choosing the package used in the facilities, the Package knows about creator and product interfaces that allow it to vary behavior without changing client code.

On the left is the creator classes for the facilities and on the right side is the product classes where the first one uses the package of sport facilities, so it only stores the equipment for the sport. In the hall it stores the equipment uses in a hall but have the same type name but different content. The user can choose the package they want according to the facilities they choose to reserve on the certain date.

4.2.2.4 Limited Entry Decision Table (LEDT) << Reservation Subsystem>>

Get Reserve Discount	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**	**D5**	**D6**
Package Type is	A	A	B	B	C	C	D	D	None	None
Duration time more than 5 hours	T	F	T	F	T	F	T	F	T	F
Action										
Package Discount	10%	10%	10%	10%	10%	10%	10%	10%	0%	0%
Reserve Duration Discount	5%	0%	5%	0%	5%	0%	0%	0%	5%	0%

4.2.2.5 Design Pattern for Class Payment (Observer Design Pattern)

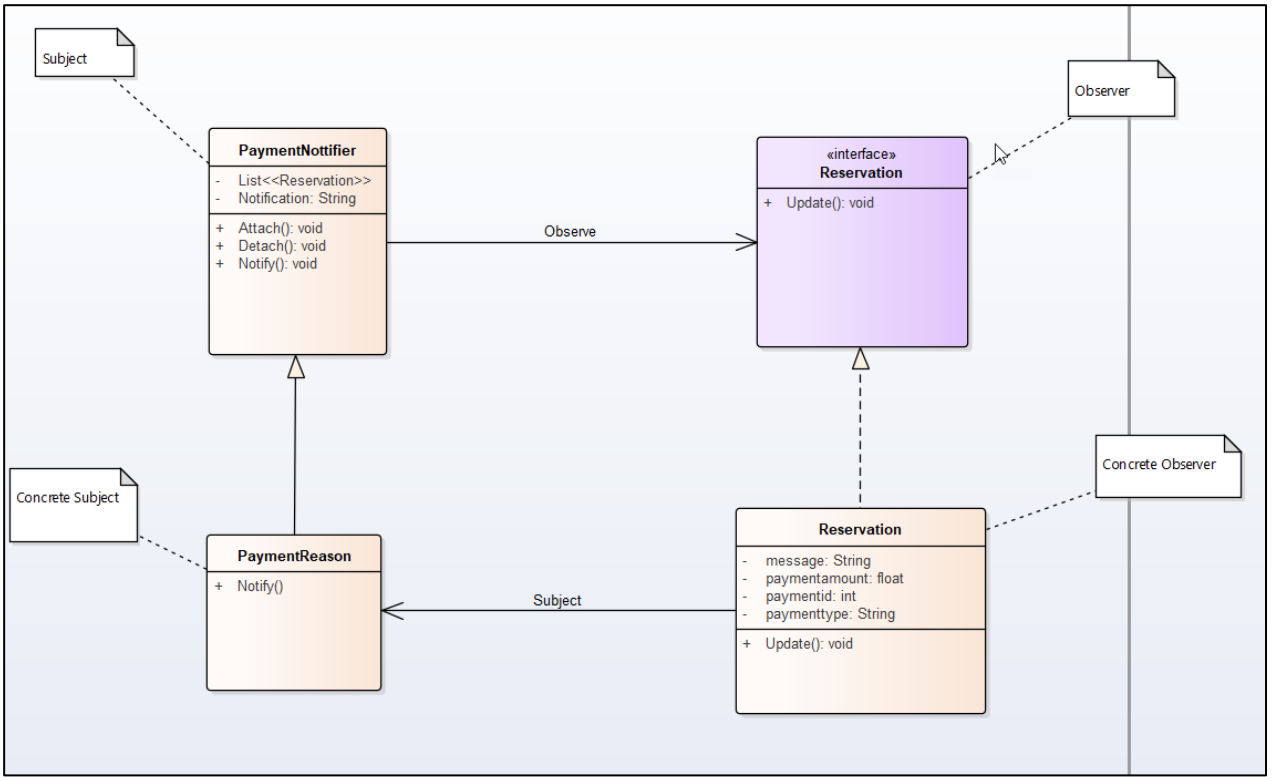


Figure 4.2.2.5: Class Diagram for Registration Class

For the registration class, observer design pattern has been chosen to be implemented in this Donation class so that the any reservation made can be notify when the payment has been paid. For this approach the concrete observer (Reservation) will examine the notification information by using the update parameter. When the concrete object receives the notification, it will call a method to retrieve the notification details. Since the behavior for attaching, detaching and notifying observes is the same, the Subject (PaymentReason) base can specify and implement the methods, which are inherited by all other concrete subjects.

At first, we need to design the subject (PaymentNotifier) interface and implement the attach, detach and notify observer (Reservation) object code. Then, the observer (Reservation) will inherit the information from the subject (PaymentNotifier). After that, the interface for the observer needs to be design which need to include the abstract interface method. Next, the Observer (Reservation) in the system implement the observer interface which requires implementing the update method. Finally, at run time, create the observer (Reservation) and attach it to the subject (PaymentNotifier).

4.2.2.6 Sequence Diagram

a) SD006: Sequence diagram for View Available Facilities

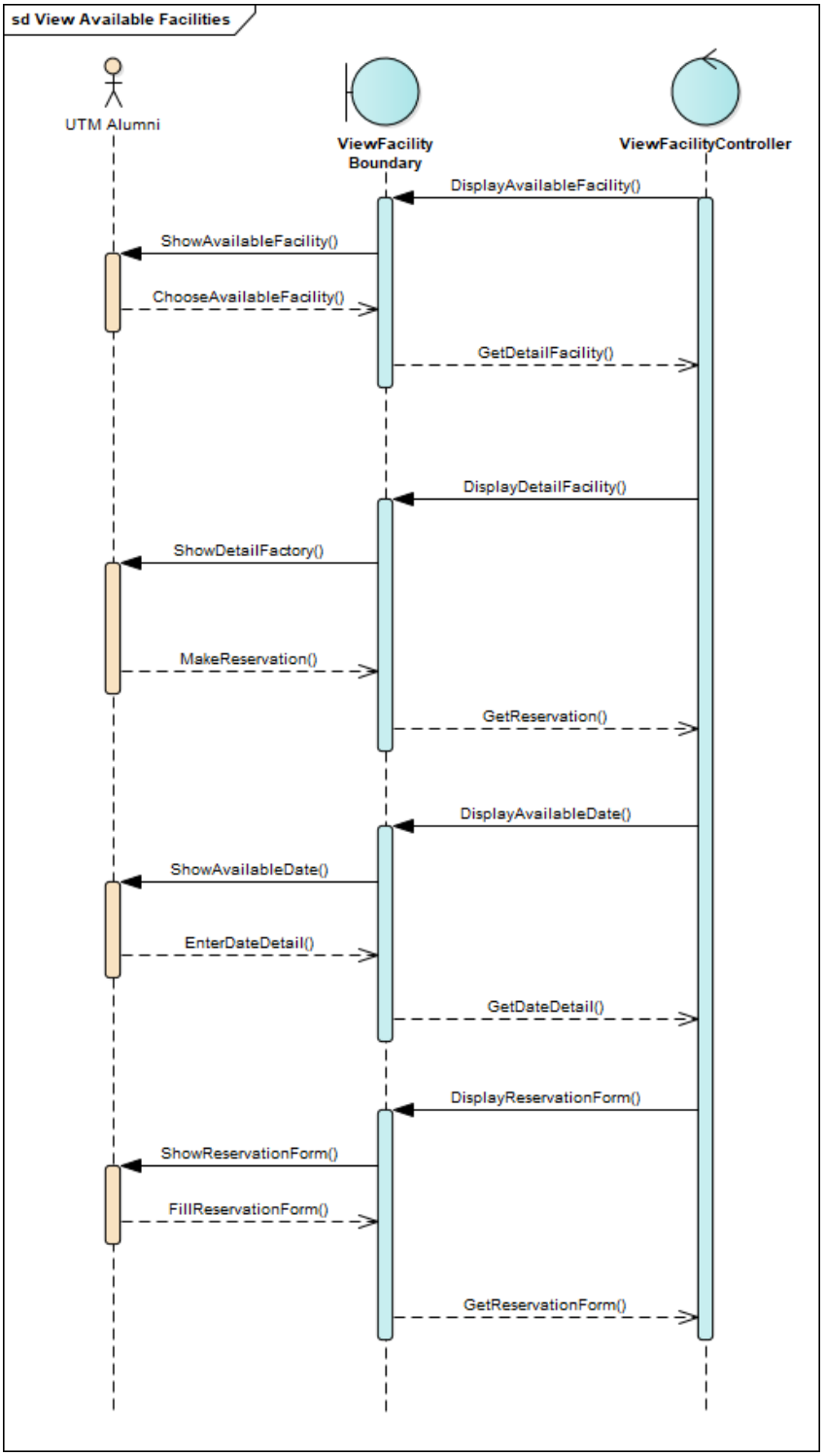
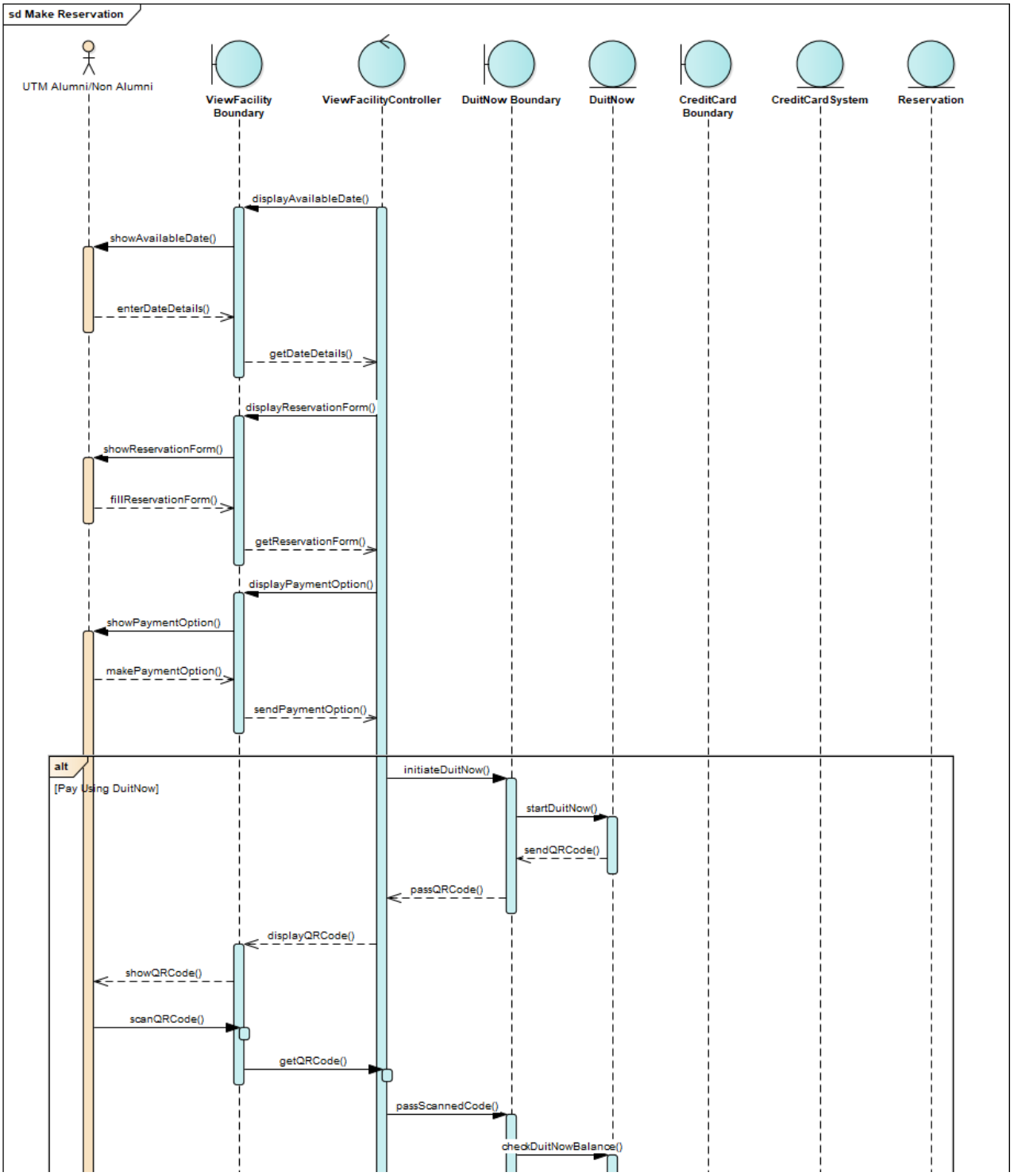


Figure 4.2.2.6.1 : Sequence Diagram for <View Available Facilities>

b) SD012: Sequence diagram for Make Reservation



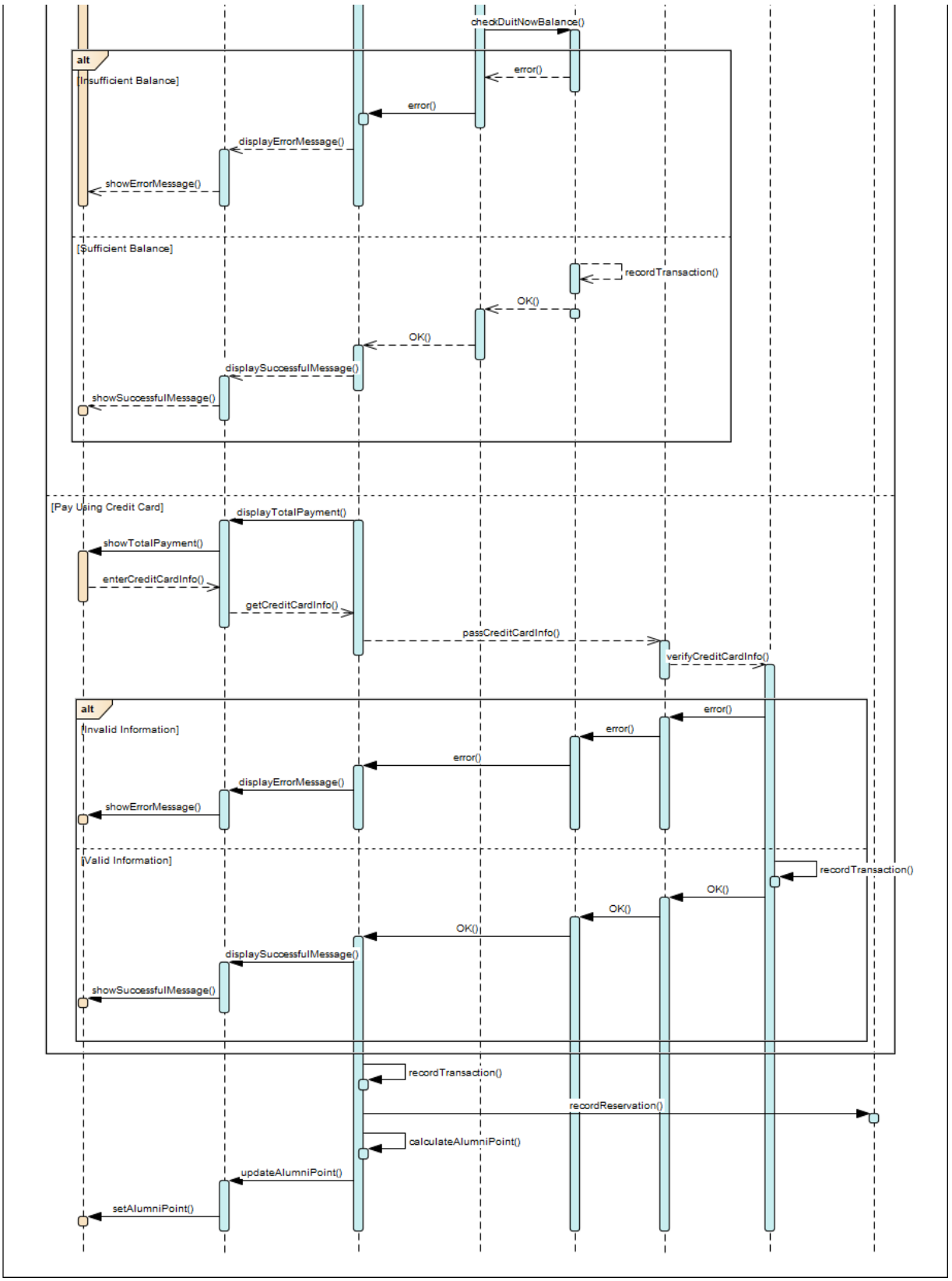


Figure 4.2.2.6.2: Sequence Diagram for <Make Reservation>

4.2.2.6 Activity Diagram for Make Reservation

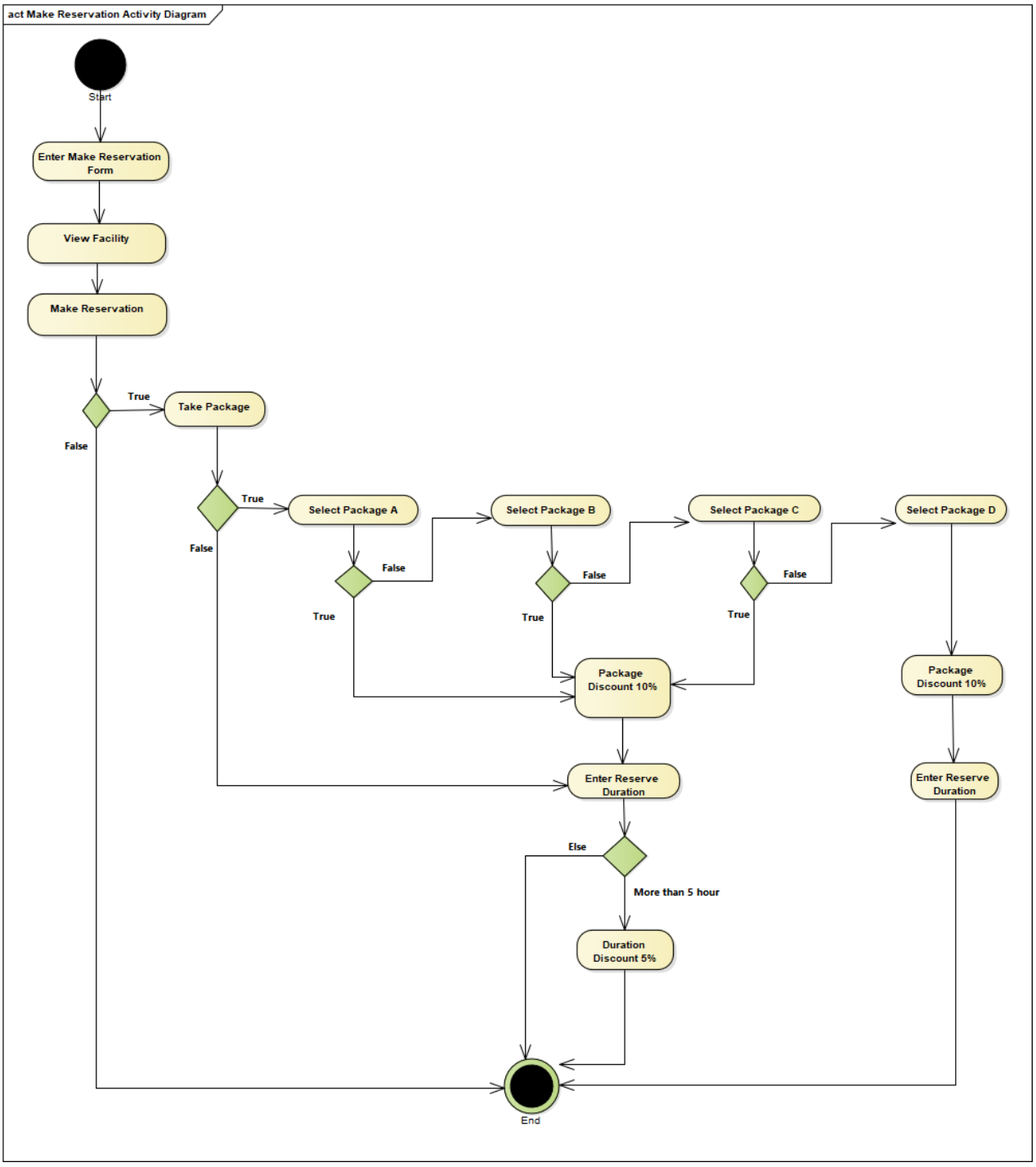


Figure 4.2.1.6: Activity Diagram for <Make Reservation>

5. Data Design

5.1 Data Description

The major data or systems entities are stored into a relational database that comprises 7 entities.

Entity Name	Description
Activity	The activity that user can join and register
Donation	The money that are given to help people or organization
Facility	Available facility that user can make reservation
Payment	Payment for the reservation facilities made by the user
Reservation	Reservation on available facilities
Non-alumni	The user who can use the system
UtmAlumni	The user who can use the system

5.2 Data Dictionary

5.2.1 Entity: <UTM Alumni>

Attribute Name	Type	Description
UTMAlumniID	int	ID for user from UTM graduate students
Alumnipoint	int	Collect and can be used for discount later.
Email	varchar	Mail address use by the Alumni UTM
Name	varchar	Insert the name of user
Password	varchar	Password registers by the Alumni
Phone	varchar	Phone number used by Alumni user

5.2.2 Entity: <Non-Alumni>

Attribute Name	Type	Description
Non-alumniID	varchar	ID from the IC used for Non-Alumni
Email	varchar	Mail address use by the Non-Alumni
Name	varchar	Name of user Non-Alumni
Phone	varchar	Phone number used by Non-Alumni

5.2.3 Entity: <Activity>

Attribute Name	Type	Description
ActivityID	int	Every activity is assigned with ID
Activitydate	date	Date they choose to have the activity
Activityname	varchar	The name of activity held
Maxperson	int	The number of activity participants

5.2.4 Entity: <Donation>

Attribute Name	Type	Description
DonationID	int	ID given after every donation made
DonateAmount	real	The amount user chooses to donate

5.2.5 Entity: <Facility>

Attribute Name	Type	Description
FacilityID	int	The ID for every facility can be used for easier reservation.
Facilityname	varchar	Name of facility can be reserve
Facilitytype	varchar	Describe suitable activity following its name
Facilityprice	real	The price needs to pay for reservation.

5.2.6 Entity: <Reservation>

Attribute Name	Type	Description
ReservationID	int	ID create for every reservation made so there will be no redundant
Package	varchar	Package prepared with the facility such as equipment
Reservationdate	date	Date chosen to make the reservation of facility

5.2.7 Entity: <Payment>

Attribute Name	Type	Description
PaymentID	int	ID for the receipt
Paymentype	varchar	Can be paid using either credit card or FPX
Paymentdate	date	The date payment was made
Paymentamount	real	Amount paid for the facility reservation

6. User Interface Design

6.1 Overview of User Interface

UTM Alumni Integrated System (ALIS) is a system specifically designed for all UTM alumni that registered in the system. The system contained various features and functionalities such as register activity, make donation, facilities reservation and payment. These functions are classified into two subsystems which are Activity Management Subsystem and Reservation Facility Subsystem.

For Activity Management Subsystem, users will be directed to the main page of UTMALIS and they can choose to register activity, view activity, make donation or view donation. For register activity, user will fill in the required details in the form page and once they have submitted their registration they can view the record on view activity form. For make donation, user will fill in the required details in the form page and once they have submitted their donation they can view the record on view donation form.

For Reservation Facility Subsystem, user can choose to view facilities, make reservation and make payment. For view facilities, user can view the list and details of available facilities and if they wish to make a reservation they can go to reservation page and fill in the required details in the form page and once they have submitted the reservation, they can make their payment on payment page.

The interface of the system consists of interactive design where all the icon in this system have caption that make the user easy to use it. It can decrease user confusion while using the system. besides, as a user-friendly system, the interface is not overly complex, we providing the quick access to common features. In simple words, we keep it simple and clean. All the tools and options are well organized in this system.

6.2 Screen Images

6.2.1 Main Home Page of UTMALIS

Figure 6.2.1.1: User Interface of UTMALIS Home Page

6.2.2 Activity Management Subsystem

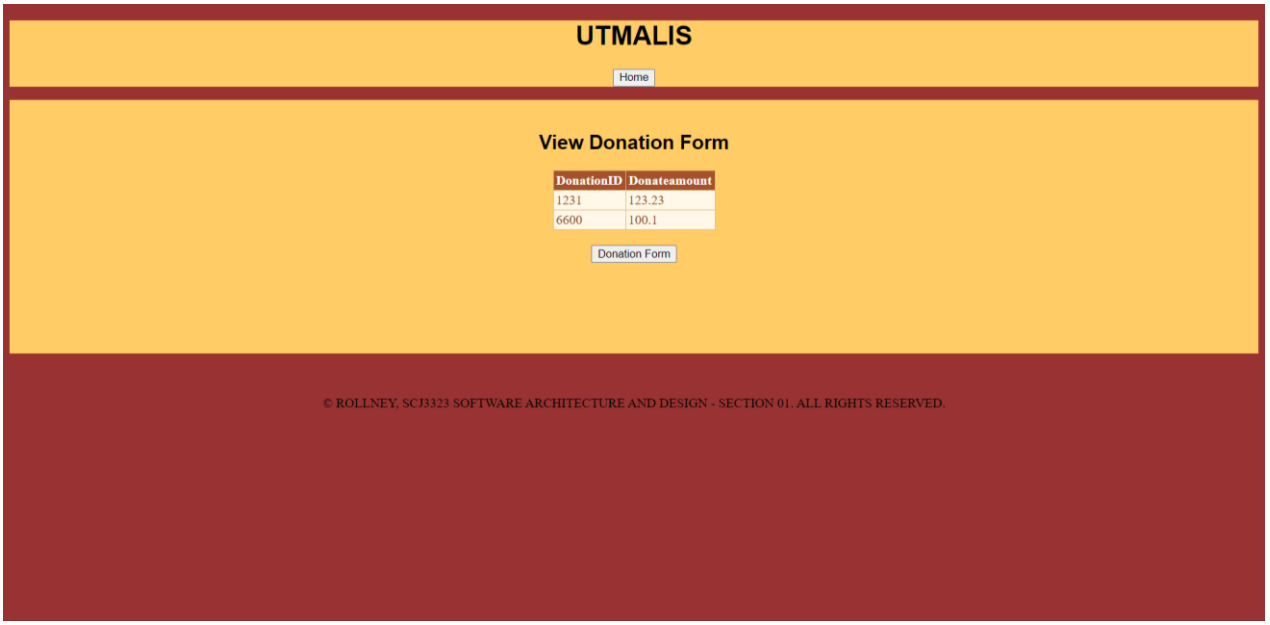


Figure 6.2.2.1: User Interface of View Donation Page

Figure 6.2.2.2: User Interface of Donation Page

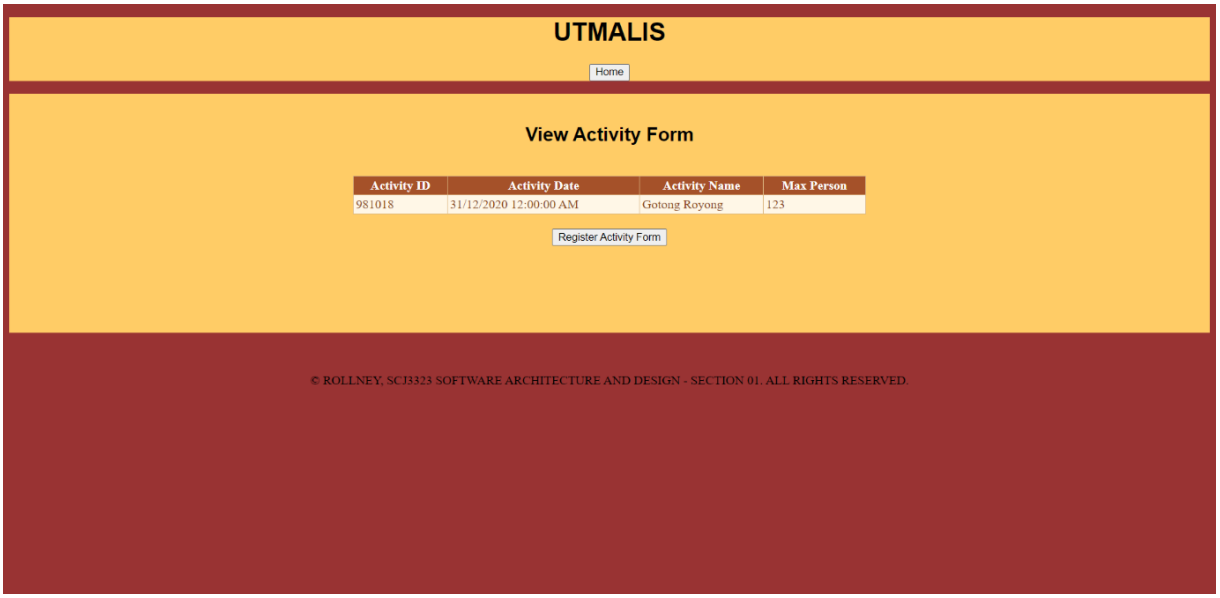


Figure 6.2.2.3: User Interface of View Activity Page

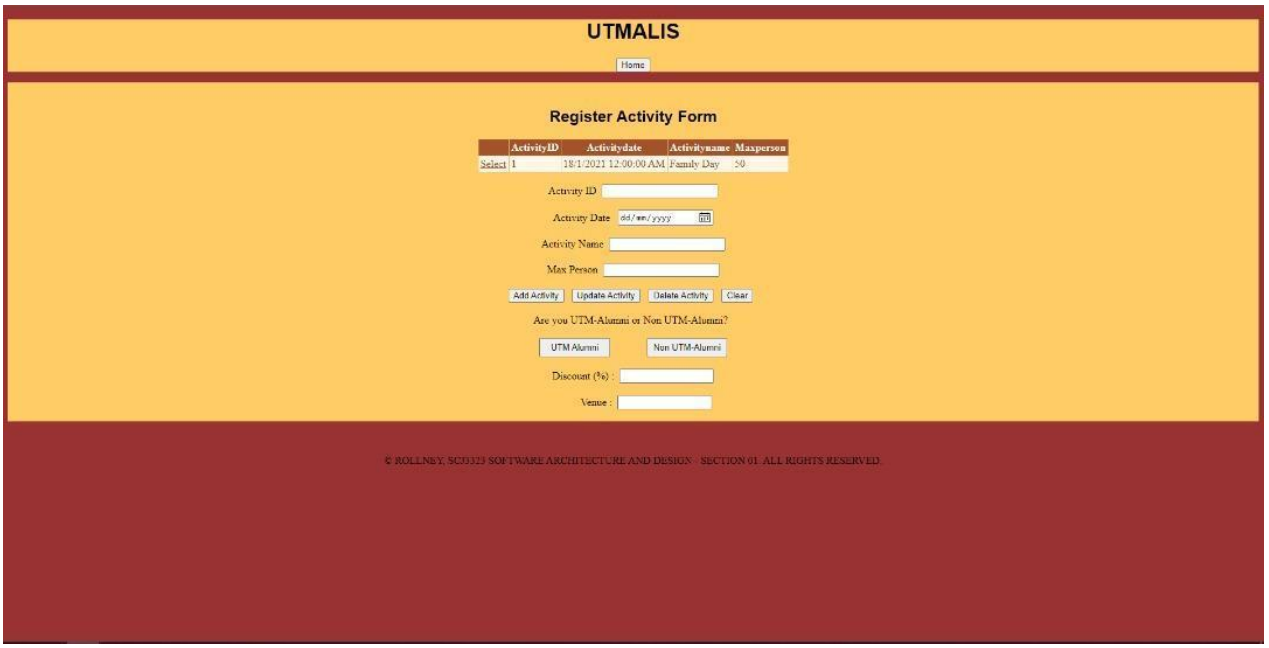


Figure 6.2.3.1: User Interface of Register Activity Page

6.2.3 Reservation Facility Subsystem

Figure 6.2.3.1: User Interface of View Facilities Page

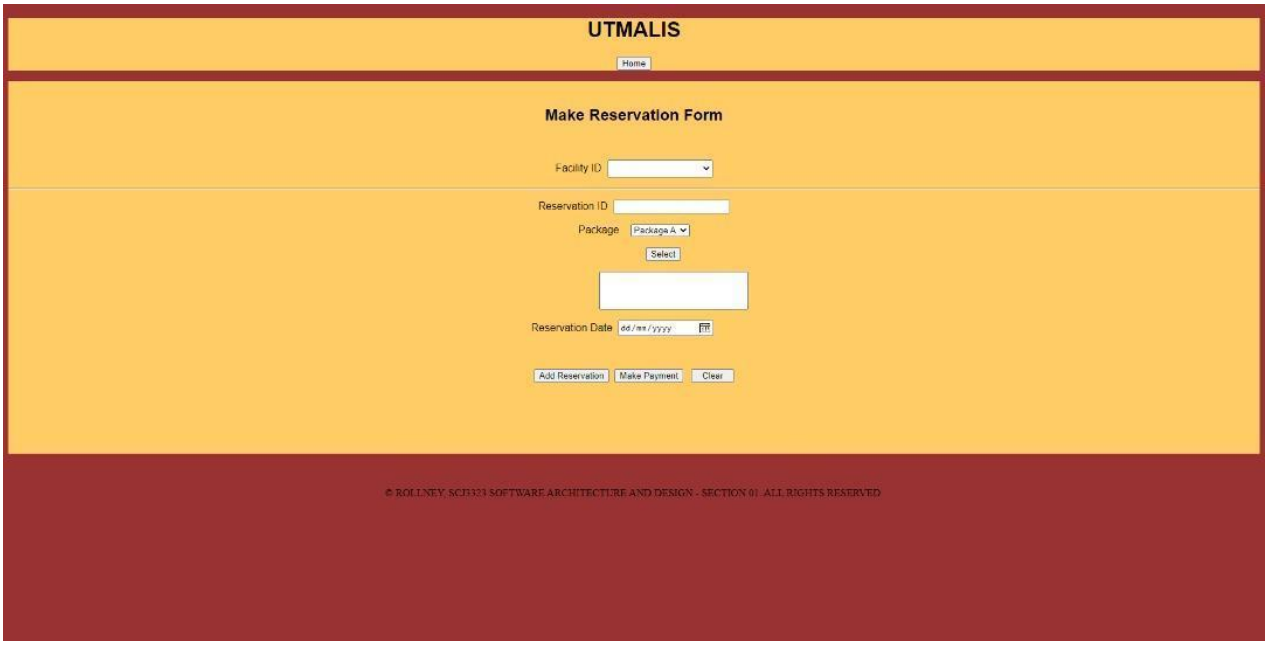


Figure 6.2.3.2: User Interface of Make Reservation Page

Figure 6.2.3.3: User Interface of Make Payment Page

7. Requirements Matrix

The sequence diagrams for each use case vs. corresponding classes are as below.

	UTM Alumni	Non-UTM Alumni	Activity	Donation	Reservation	Facility	Payment
P001, UC004, SD004	X		X				
P001, UC005, SD005	X			X			
P002, UC006, SD006	X	X			X	X	X
P002, UC0012, SD012	X	X			X	X	X