

# Informatics College Pokhara



Title: Perfect GYM Club

Software Engineering

CS5002NP

Coursework 2

**Submitted By:**

Student Name: Kushal Gurung

London Met ID: 18029022

Group: L2C3

Date: 04-Jun-2020

Words Count: 4359

**Submitted To:**

Mr. Sandeep Gurung

Module Leader

Software Engineering

## Table of Contents

Introduction .....	1
1. Gantt chart.....	2
1.1 Short Description of Project Development .....	2
1.2 Gantt chart.....	3
2. Use Case Diagram .....	4
2.1 Short Description of Use Case Diagram .....	4
2.2 Use Case Diagram .....	5
2.3 High Level Use Case Description .....	6
2.4 Expanded Use Case Description .....	10
3. Collaboration Diagram and Sequence Diagram .....	12
3.1 Collaboration Diagram .....	12
3.1.1 Steps involved in Collaboration diagram.....	12
3.2 Sequence Diagram.....	15
3.2.1 Steps involved in Sequence Diagram .....	15
4. Class Diagram.....	19
4.1 Steps involved in Class Diagram .....	19
5. Progressing forward from current development phase .....	23
6. Prototype.....	26
Conclusion .....	36
References .....	37

## List of Figures

Figure 1: Gantt chart .....	3
Figure 2: Perfect GYM Club's Use Case Diagram .....	5
Figure 3: Object of Domain Classes .....	12
Figure 4: Addition of Control Object .....	12
Figure 5: Addition of Boundary Object .....	13
Figure 6: Addition of an Actor.....	13
Figure 7: Addition of Associations.....	13
Figure 8: Perfect GYM Club's Collaboration Diagram.....	14
Figure 9: Domain Classes.....	15
Figure 10: Addition of Control Object Lifeline.....	15
Figure 11: Addition of Boundary Object Lifeline .....	16
Figure 12: Addition of Actor Lifeline .....	16
Figure 13: Addition of Messages.....	17
Figure 14: Drawing Domain class object lifeline.....	17
Figure 15: Perfect GYM Club's Sequence Diagram.....	18
Figure 16: Domain Classes.....	20
Figure 17: Addition of Associations.....	21
Figure 18: Perfect GYM Club's Class Diagram .....	22
Figure 19: Prototype of Login form.....	26
Figure 20: Prototype of Customer Registration Form.....	27
Figure 21: Prototype of Customer Report Form .....	28
Figure 22: Prototype of Customer Deregistration Form .....	29
Figure 23: Prototype of Fee Payment Form.....	30
Figure 24: Prototype of Daily Task Form .....	31
Figure 25: Prototype of To Do List Form.....	32
Figure 26: Prototype of Check Payment Form .....	33
Figure 27: Prototype of Staff Registration Form.....	34
Figure 28: Prototype of Staff Deregistration Form.....	35

## List of Tables

Table 1: Register Customer high level use case description .....	6
Table 2: Pay fee high level use case description .....	6
Table 3: Check Payment high level use case description .....	7
Table 4: Generate Customer Report high level use case description .....	7
Table 5: De-register Customer high level use case description .....	7
Table 6: Register Staff high level use case description.....	8
Table 7: De-register Staff high level use case description .....	8
Table 8: Define Daily Task high level use case description .....	8
Table 9: Create To Do List high level use case description .....	9
Table 10: Login high level use case description .....	9
Table 11: Register Customer expanded use case description .....	10
Table 12: Create To Do List expanded use case description.....	11
Table 13: Finding Unique Domain Classes .....	19

## Introduction

This was the second coursework of Software Engineering module which was assigned to us on 20<sup>th</sup> week. The coursework was based on the development of a computerized information system for Perfect GYM Club to cope with the difficulties that are being faced, while maintaining the records of customers and their payment details. For this project, Rational Unified Process along with the Unified Modelling Language were used to carry out the project and fulfil the computerized system demand of the Perfect GYM Club.

Rational Unified Process (RUP) is a software development process which belongs to Rational, a category of IBM. It's main purpose in Software Engineering is to develop high quality software with a foreseeable budget and time frame. RUP has four life cycle phases, which includes Inception, Elaboration, Construction, and Transition.

Inception is the phase where the project's basic idea and structure is determined. In this phase, the team will decide whether the project is worth pursuing at all, based on the risk management, cost and time estimation and the scheduling resources. In Elaboration, the requirements and necessary architecture of the project are analysed. This phase provides stability of the project, and also evaluates whether the vision of the project is stable or not. In Construction phase, the application features are designed, coded and tested. Whereas in Transition phase, the finished product is released and handed over to customers. This phase also includes the handling of all post-release support, bug fixes, and patches until the end-user is satisfied (Powell-Morse, 2017).

Similarly, Unified Modeling Language (UML) is a standardized modelling language that consists of a combined set of diagrams to visualize a way of developing the system. It contains many structure and behavioral diagrams out of which, Class diagram, Use Case diagram, Sequence diagram and Communication diagram were drawn in our project to design a computerized information system (Wright, 2014). I believe that the computerized system that is developed in this project will be able to protect and secure the gym's information, and also enhance the standard and quality of the gym to maintain the records and payment details of their customer.

## **1. Gantt chart**

Gantt chart is a horizontal view of scheduled tasks that represents a project development showing all the activities performed in the provided time zone. It contains list of tasks, time duration required for specific tasks, and start and end date of tasks (Gantt, 2020). In my project, Gantt chart was developed with the help of draw.io software. It is one of the most popular tool that is used for drawing Unified Modelling Language (UML) and other different types of diagrams.

### **1.1 Short Description of Project Development**

For the development of the project, the phases of Rational Unified Process (RUP) will be followed. The Inception phase will start from 19<sup>th</sup> of March and is estimated to be finished on 24<sup>th</sup> of April where risk management, planning, prototype development and other necessary activities will be done. Similarly, in Elaboration phase, problem domain will be analysed and then, structure and behavioral diagrams will be drawn. The diagrams will include Class diagram, Use Case diagram, Sequence diagram and Communication diagram. Elaboration phase is estimated to last up to 28 days from 23<sup>rd</sup> of March to 20<sup>th</sup> of April. Likewise, the system will be built and tested in the Construction phase from 3<sup>rd</sup> of April until 24<sup>th</sup> of April. Similarly, in the Transition phase, training and analysis of user's review will be conducted from 12<sup>th</sup> of April until 30<sup>th</sup> of April. The documentation of the project will continue along with these four phases of RUP methodology from 3<sup>rd</sup> of April until 30<sup>th</sup> of April.

These all are the estimated time duration and may be changed according to the changes in requirement that may arise during the development of the project, since we are following RUP methodology. The figure of Gantt chart for the project is shown in the figure below.

## 1.2 Gantt chart

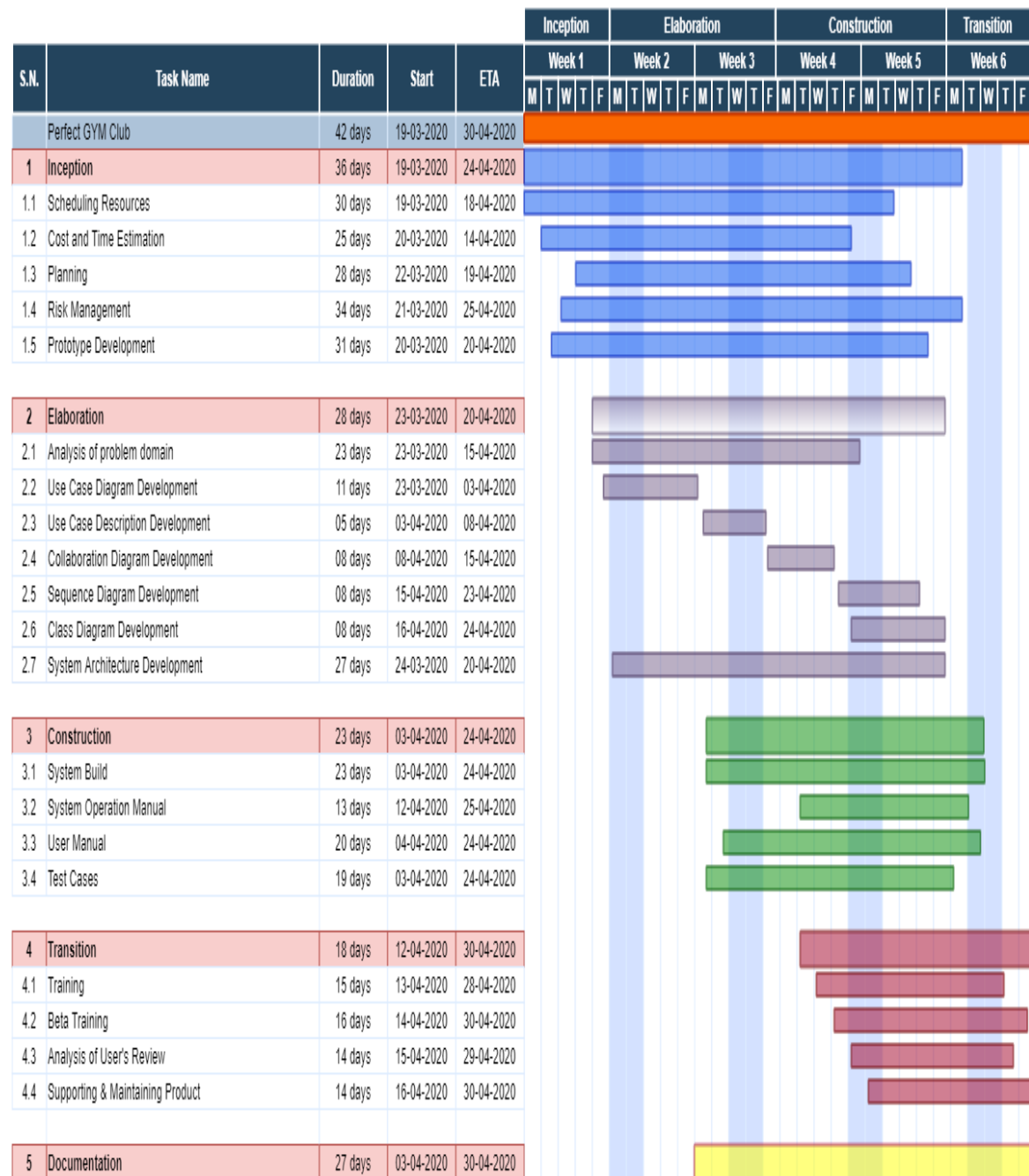


Figure 1: Gantt chart

## **2. Use Case Diagram**

### **2.1 Short Description of Use Case Diagram**

A Use Case is a behavioral Unified Modeling Language diagram, which is used for modelling the functions and features of a system with the help of actors and their relationship with use cases. It is developed for obtaining the outside look of the system. There are four elements in the Use Case Diagram: System, Use Case, Actor and Relationship.

In Use Case Diagram, System represents the project that we are developing. It includes all the use cases of the system. Likewise, use cases are the task or the processes that can be performed by the system. Similarly, actor are responsible for performing any specific use cases in the system. Actor is of two types: primary and secondary actor. Lastly, relationships are used to show relationship between an actor and use cases. Relationship is also of two types: Include and Extend relationship. These relationships are used to show relationships between base use cases and included or extended use cases. Included use case is the use case that will execute itself when the base use case executes, whereas extended use case may or may not execute depending on the extension point when the base use case is executed (uml-diagrams, 2020).

In Use Case Diagram, System is represented by a rectangular shape, Use Case is represented by an oval shape, Actor is represented by a stick-figure and relationship is represented by a straight line. The Use Case Diagram of Perfect GYM Club is shown in the figure below.



## 2.2 Use Case Diagram

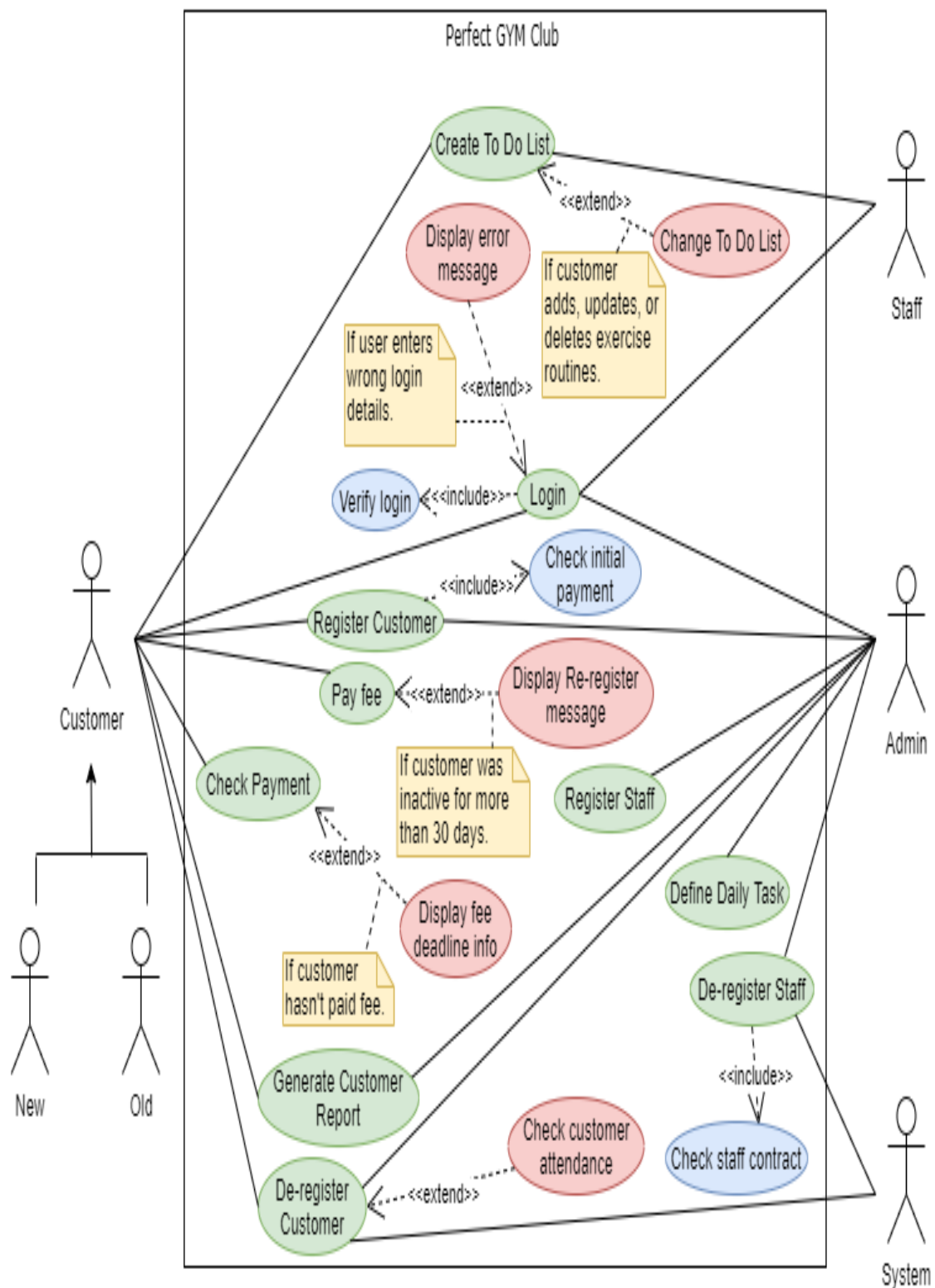


Figure 2: Perfect GYM Club's Use Case Diagram

### 2.3 High Level Use Case Description

High Level Use Case description is a short and non-detailed type of use case description, which provides short description of each required process. It contains use case, actor and description of the use case (Britton & Doake, 2005). In high level use case description, use case determines the name of the use case that is going to be described, actor is anyone who calls the use case, and description is the element where use cases are described.

Here are the high level use case description of all the base use cases of Perfect GYM Club:

#### i. High Level Use Case Description number: 1

<b>Use Case:</b>	Register Customer
<b>Actor:</b>	Admin, Customer
<b>Description:</b>	Customer provide their registration details to the system. Then, the customer pays the initial fee required for the registration. Then, he/she will receive their GYM joining form. Customers can also register themselves manually through admin, which is a secondary actor.

*Table 1: Register Customer high level use case description*

#### ii. High Level Use Case Description number: 2

<b>Use Case:</b>	Pay fee
<b>Actor:</b>	Customer
<b>Description:</b>	Customer provides his/her ID details to the system, pays the monthly fee, and receives confirmation of the payment, which is recorded in the system.

*Table 2: Pay fee high level use case description*

**iii. High Level Use Case Description number: 3**

<b>Use Case:</b>	Check Payment
<b>Actor:</b>	Customer
<b>Description:</b>	Customer provides his/her ID details to the system and checks if it is time to pay their monthly fee.

*Table 3: Check Payment high level use case description***iv. High Level Use Case Description number: 4**

<b>Use Case:</b>	Generate Customer Report
<b>Actor:</b>	Customer, Admin
<b>Description:</b>	Admin obtains the customer's details and update their report in the system, in case if it is necessary. Customer can login and see their report in the system.

*Table 4: Generate Customer Report high level use case description***v. High Level Use Case Description number: 5**

<b>Use Case:</b>	De-register Customer
<b>Actor:</b>	Customer, Admin, System
<b>Description:</b>	Customer informs the admin if she/he decides to leave the gym. Then, the admin will then confirm the customer's deregistration. Also, system will automatically deregister a customer, if she/he has been remained inactive in gym for more than a month.

*Table 5: De-register Customer high level use case description*

**vi. High Level Use Case Description number: 6**

<b>Use Case:</b>	Register Staff
<b>Actor:</b>	Admin
<b>Description:</b>	Admin obtains and verifies the staff's registration form details. Then, admin enters the staff's registration details in the system. Then, the system will confirm the staff's registration, and issues a unique staff gym ID number.

*Table 6: Register Staff high level use case description***vii. High Level Use Case Description number: 7**

<b>Use Case:</b>	De-register Staff
<b>Actor:</b>	Admin, System
<b>Description:</b>	After the staff's contract expires, admin will offer 10 days of buffer time to extend their contract. If the contract is not extended, the system automatically deregisters the staff. If In case, the staff leaves the gym before termination of their contract, admin will deregister staff and restrict him/her to use the system.

*Table 7: De-register Staff high level use case description***viii. High Level Use Case Description number: 8**

<b>Use Case:</b>	Define Daily Task
<b>Actor:</b>	Admin
<b>Description:</b>	Admin defines all set of task that can be carried out in the gym.

*Table 8: Define Daily Task high level use case description*

**ix. High Level Use Case Description number:9**

<b>Use Case:</b>	Create To Do List
<b>Actor:</b>	Customer, Staff
<b>Description:</b>	Customers select a list of exercises in the system and creates a To Do list for themselves. If customer wants, she/he can make changes in their To Do List by adding, updating or by deleting exercise routines. If customer have no idea on types of exercises, staff will help to create a To Do List for the customer.

*Table 9: Create To Do List high level use case description***x. High Level Use Case Description number: 10**

<b>Use Case:</b>	Login
<b>Actor:</b>	Customer, Staff, Admin
<b>Description:</b>	The user provides their login details and enters into the Perfect GYM Club system. After login, they can use the system's features.

*Table 10: Login high level use case description*

## 2.4 Expanded Use Case Description

Expanded Use Case Description is a more detailed Use Case Description. It shows the step-by-step actions of actor and the response of the system in typical course of events. In typical course of events, each step is given a number, and the new step starts from the next line with a new number. In typical course of events, the steps are taken positively by assuming that no any exception cases will occur. If there are any exception cases, then we can use alternative courses where we can note down the step where we think the exception might occur.

- **Expanded Use Case Description number: 1**

**Use Case:** Register Customer

**Actor:** Customer

**Description:** Customer provide their registration details to the system. Then the customer pays the initial fee required for the registration. Then, he/she will receive their GYM joining form.

**Typical Course of Events:**

Actor Action	System Response
1. The new customer uses the on-screen registration form to enter their personal details into the system.	
	2. Displays registration fee details.
3. Pays registration fee.	
	4. Records customer's registration fee payment.
	5. Provides a printable GYM joining form, which will contain the customer's new login details and gym joining details.
6. Prints the GYM joining form.	

*Table 11: Register Customer expanded use case description*

**Alternative Courses:**

**Line 3:** The customer does not have enough money to pay for their registration fee. Use Case ends.

- **Expanded Use Case Description number: 2**

**Use Case:** Create To Do List

**Actor:** Customer

**Description:** Customers select a list of exercises in the system and creates a To Do list for themselves. If customer wants, she/he can make changes in their To Do list by adding, updating or by deleting exercise routines.

**Typical Course of Events:**

Actor Action	System Response
1. Customer provides login details to the system.	
	2. Displays a menu of system features which are available for customer users.
3. Selects the To Do List menu option.	
	4. Displays a list of exercises available in the Perfect GYM Club.
5. Adds a list of required exercises as per their choice.	
	6. Displays confirmation message.

*Table 12: Create To Do List expanded use case description*

**Alternative Courses:**

**Line 1:** Customer provides incorrect login details. Use Case ends.

### 3. Collaboration Diagram and Sequence Diagram

#### 3.1 Collaboration Diagram

A Communication Diagram is a behavioral UML diagram that shows the interaction between objects and particular use cases. A collaboration diagram contains actors, boundary object, controller object, domain classes, associations and messages. Collaboration Diagram is also called as Communication Diagram.

##### 3.1.1 Steps involved in Collaboration diagram

In my case, a Collaboration Diagram is drawn for the Register Customer Expanded Use Case Description. Here are seven the steps that were involved during the development of Collaboration diagram.

##### a. Finding the Domain Classes

The Domain Classes that were found from Register Customer Expanded Use Case Description are: Registration, Fee and Customer.

##### b. Drawing an object of the Domain Classes

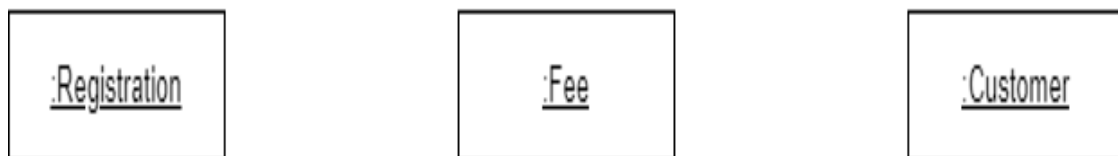


Figure 3: Object of Domain Classes

Domain Classes are represented by rectangle shapes. The name of domain classes are underlined and colon symbol “:” is placed before their names.

##### c. Adding a Control Object

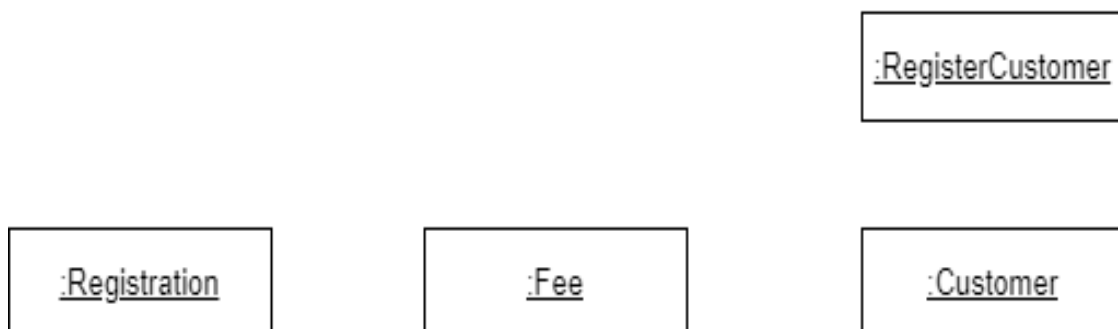


Figure 4: Addition of Control Object



The name of Control Object is kept according to the name of expanded use case description for which we are constructing the collaboration diagram. In the above figure, :RegisterCustomer is the Control Object.

#### d. Adding a Boundary Object

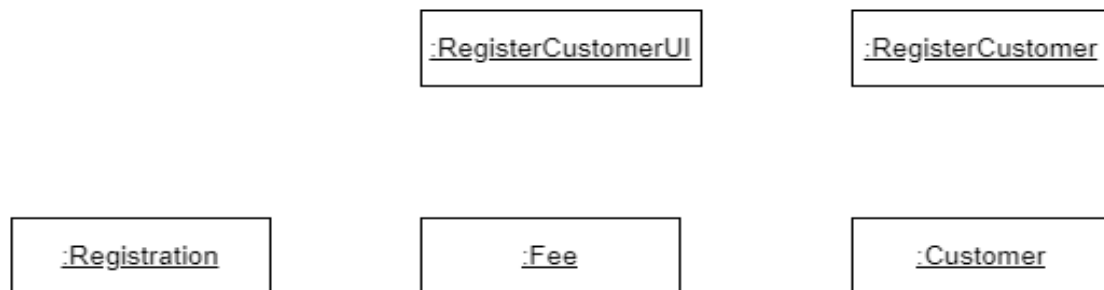


Figure 5: Addition of Boundary Object

The name of Boundary Object is same as Control Object, but there is addition of word “UI”. In the above figure, :RegisterCustomerUI is the Boundary Object.

#### e. Drawing an Actor

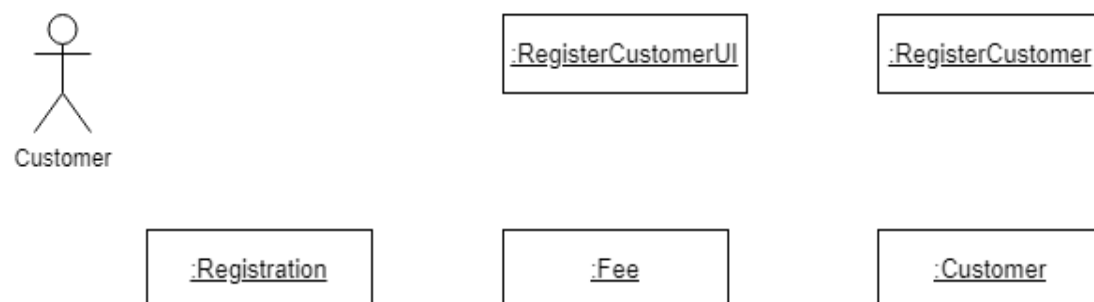


Figure 6: Addition of an Actor

Similar to Use Case Diagram, the Actor is represented with stick-figure. The name of actor is given Customer as shown in the above figure.

#### f. Adding Associations

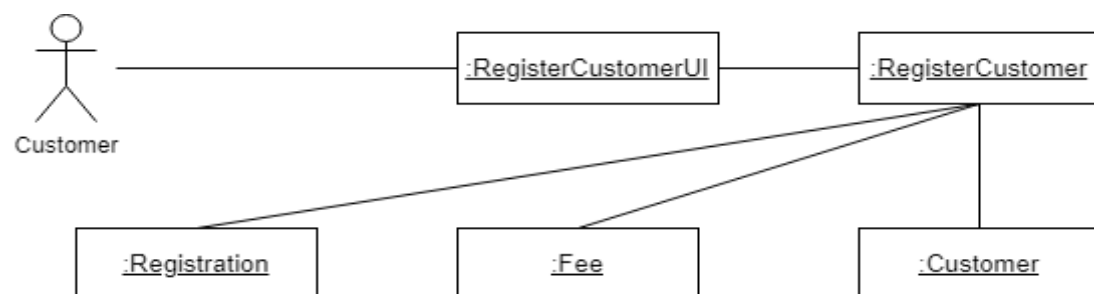


Figure 7: Addition of Associations

Associations are drawn to send messages among domain classes, control object, boundary object and actor. In Collaboration diagram, an actor can only be associated with boundary object.

### g. Adding Messages

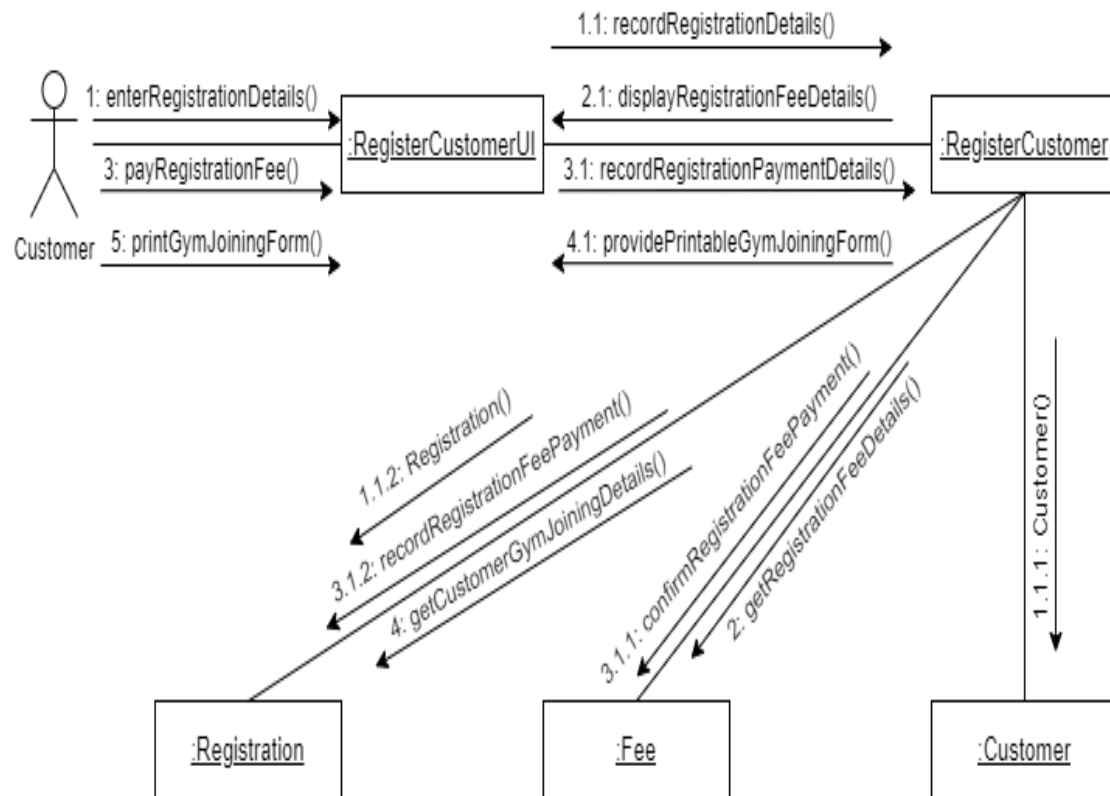


Figure 8: Perfect GYM Club's Collaboration Diagram

This is the final step for developing a collaboration diagram. In this step, messages are added to show what particular information are being transferred among actor, boundary object, controller object and domain classes. After the addition of messages, construction of Collaboration Diagram is completed.

The Collaboration Diagram of the Register Customer expanded use case description is shown in the above figure.

## 3.2 Sequence Diagram

Sequence diagram is a behavioral UML diagram that is used to show the interactions among actor, boundary object, control object and domain classes in terms of exchanging messages along with the timeline. Sequence diagram also contains many elements such as: actor, boundary object, control object, lifeline, activations, messages, focus of control, and so on. Sequence Diagram is also known as Event Diagram.

### 3.2.1 Steps involved in Sequence Diagram

#### a. Find the domain classes

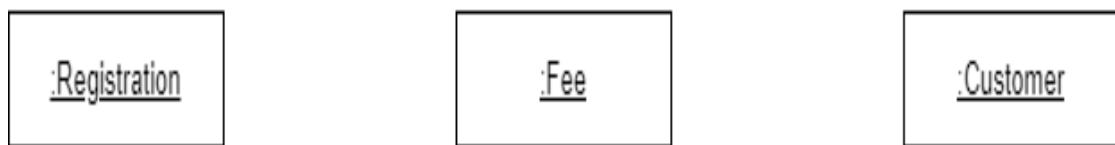


Figure 9: Domain Classes

Registration, Fee and Customer are the domain classes as shown in the above figure. The name of domain classes are underlined and colon symbol ":" is placed before their names.

#### b. Adding Control Object Lifeline

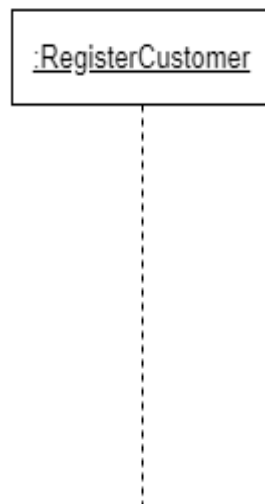
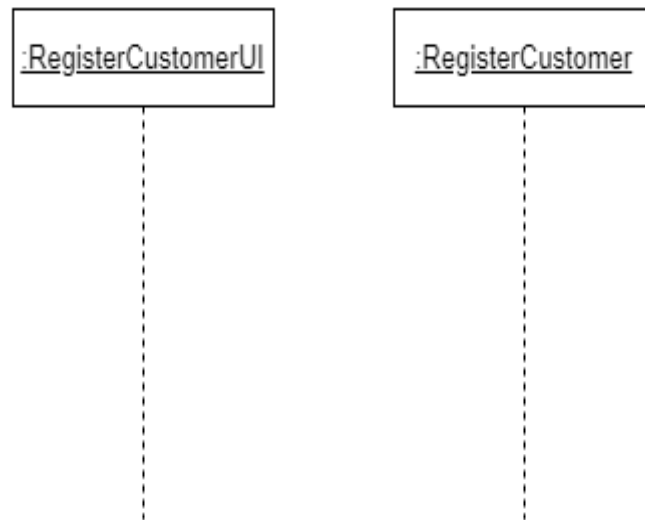


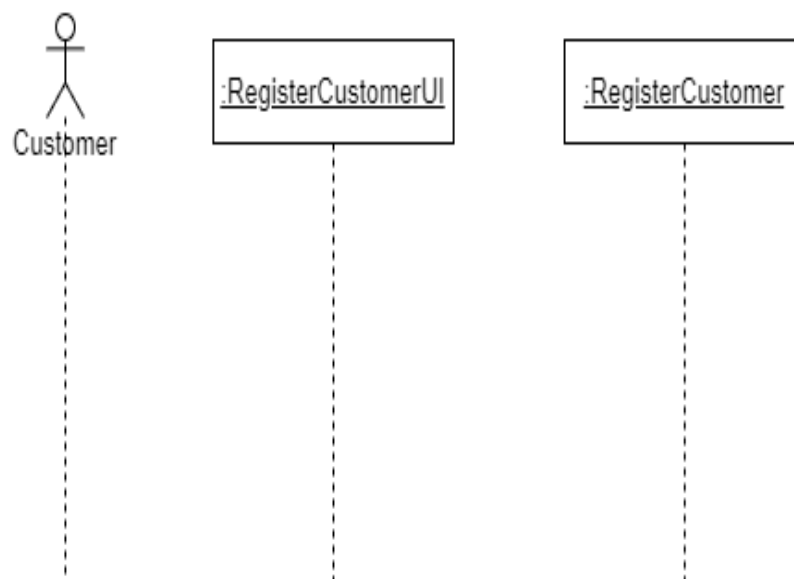
Figure 10: Addition of Control Object Lifeline

Here, :RegisterCustomer with the vertical dotted line is the control object lifeline as shown in the above figure.

**c. Adding Boundary Object Lifeline**

*Figure 11: Addition of Boundary Object Lifeline*

In this figure, :RegisterCustomerUI with a vertical dotted line is the boundary object lifeline.

**d. Adding Actor Lifeline**

*Figure 12: Addition of Actor Lifeline*

Similarly, Customer is the actor in the above figure. The stick-figure with the vertical dotted line is called Actor Lifeline.

### e. Adding Messages

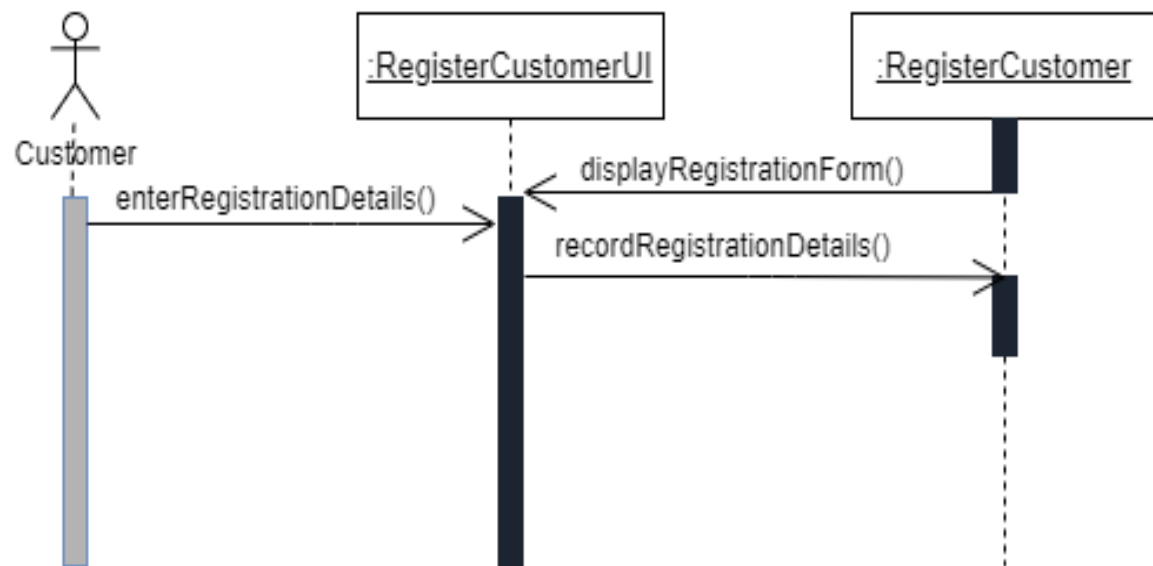


Figure 13: Addition of Messages

Here, messages are added to show what particular information are being transferred among the objects and actor.

### f. Drawing Domain class object lifeline

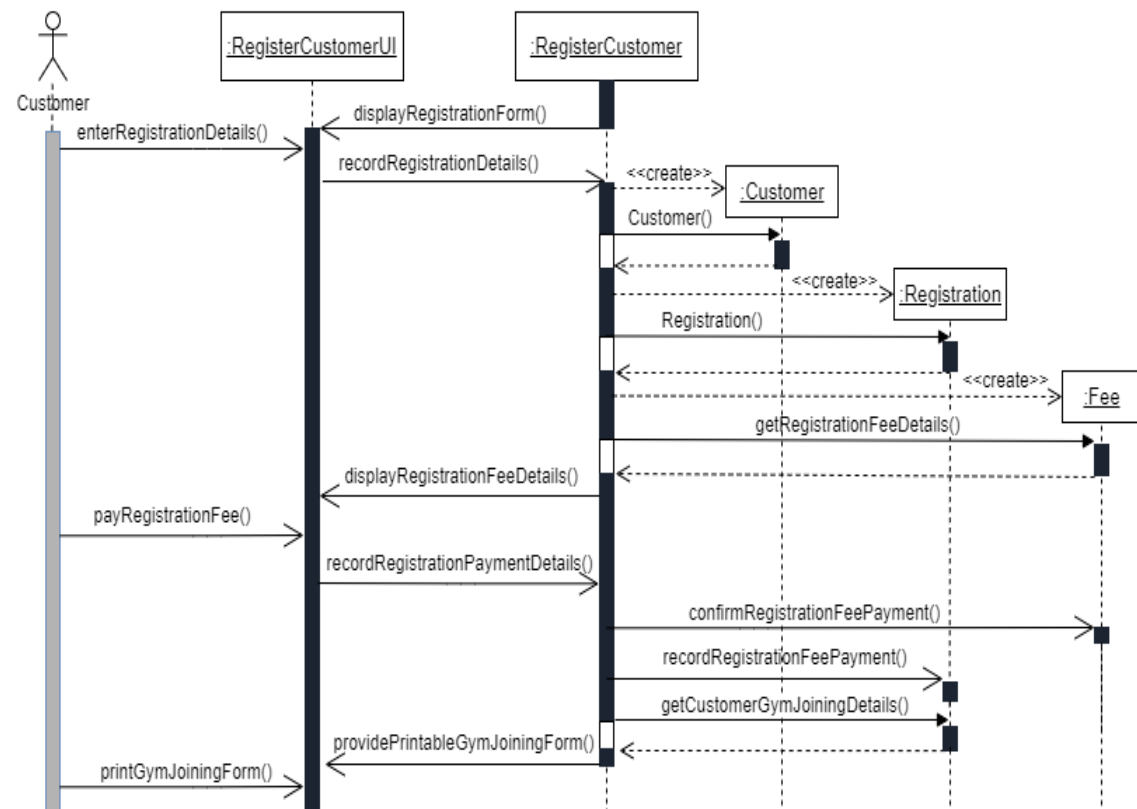
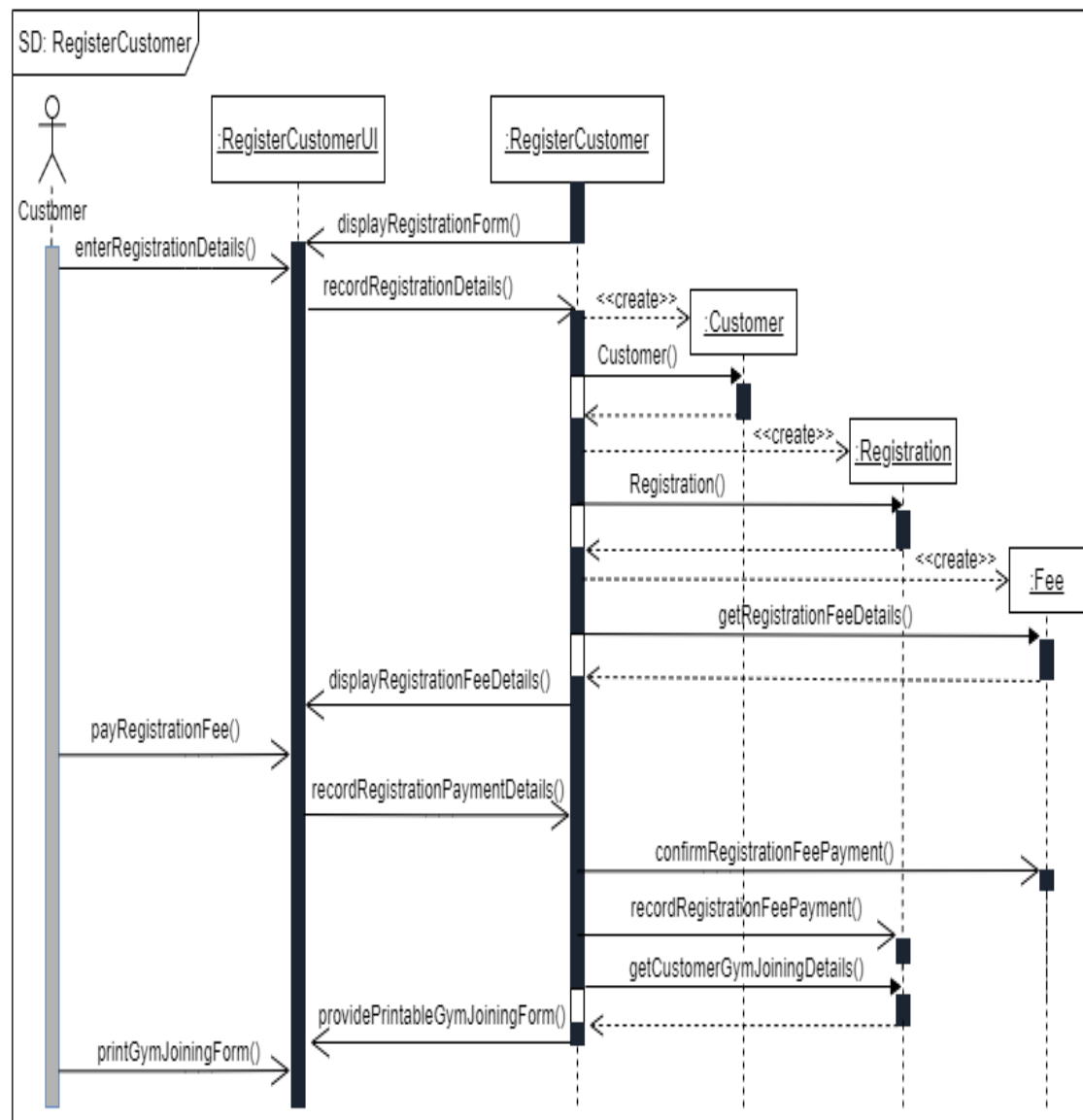


Figure 14: Drawing Domain class object lifeline

**g. Addition of frame**

*Figure 15: Perfect GYM Club's Sequence Diagram*

In the above figure, a frame is added for the whole sequence diagram, which is named as "SD:RegisterCustomer" as shown in the top left part of the figure. The final Sequence Diagram for Register Customer expanded use case description is shown in the above figure.

## 4. Class Diagram

Class Diagram is a structural Unified Modeling Language diagram that shows the system's classes, attributes, methods and relationships among different classes. A class diagram contains different classes, their attributes and functions, associations with different types, multiplicity, messages, aggregation, composition, inheritance classes and so on.

### 4.1 Steps involved in Class Diagram

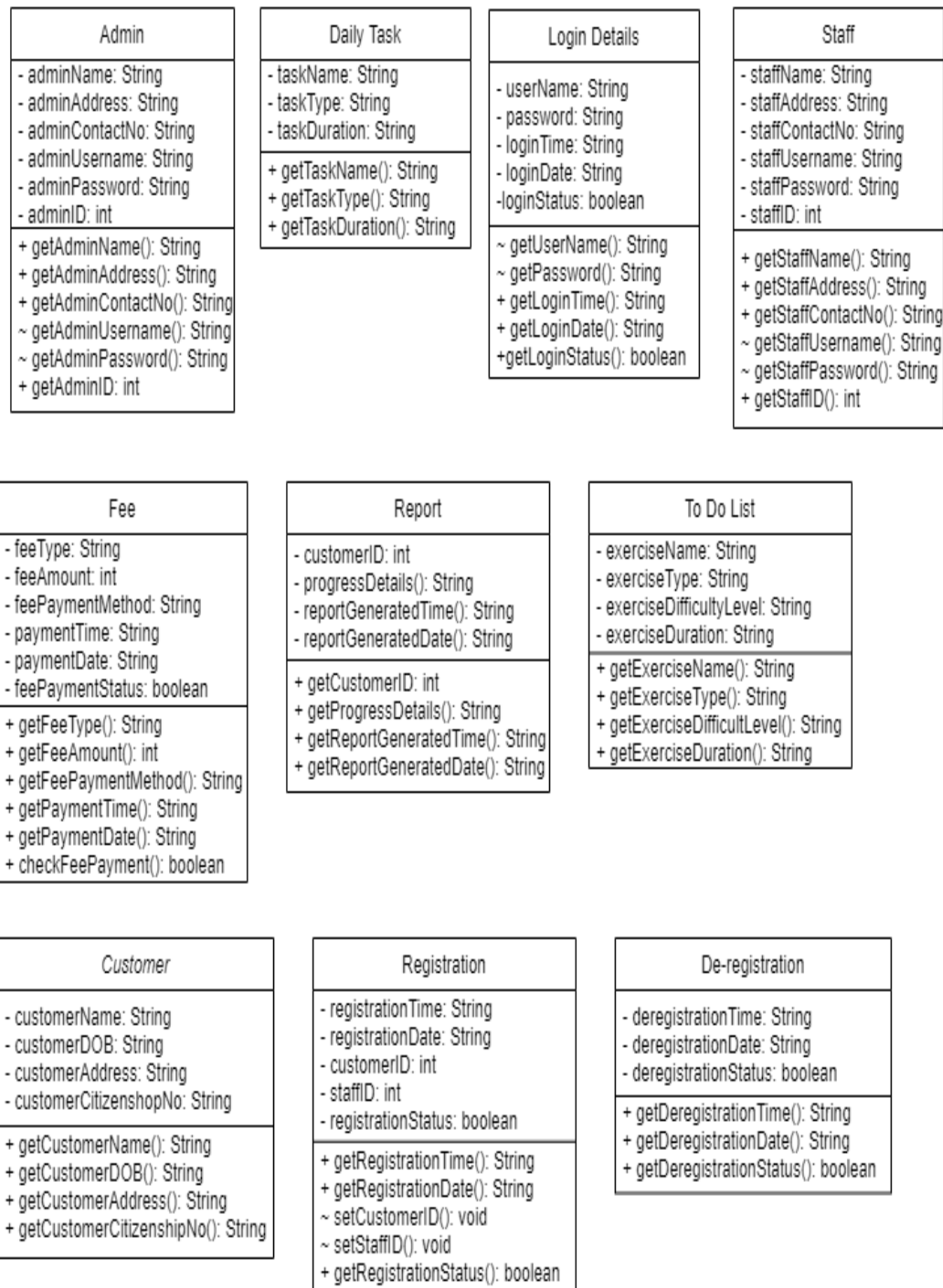
#### a. Finding the Unique Domain Classes

Use Case	Domain Classes
Register Customer	Customer, Fee, Registration
Login	Admin, Staff, Customer, Login Details
Create To Do List	Customer, Staff, To Do List
Pay fee	Customer, Fee
Check Payment	Customer, Fee
Register Staff	Admin, Staff, Registration
Define Daily Task	Admin, Daily Task
De-register Staff	Staff, De-registration, Admin
Generate Customer Report	Admin, Customer, Report
De-register Customer	Customer, De-registration, Admin

Table 13: Finding Unique Domain Classes

Therefore, the unique domain classes from all the use cases are:

- ✓ Customer
- ✓ Fee
- ✓ Registration
- ✓ Admin
- ✓ Staff
- ✓ Login Details
- ✓ To Do List
- ✓ Daily Task
- ✓ De-registration
- ✓ Report

**b. Drawing the Class Diagram for each domain classes***Figure 16: Domain Classes*

All the ten domain classes which were founded in the first step of Class Diagram are shown in the above figure.



### c. Adding Association Relationships

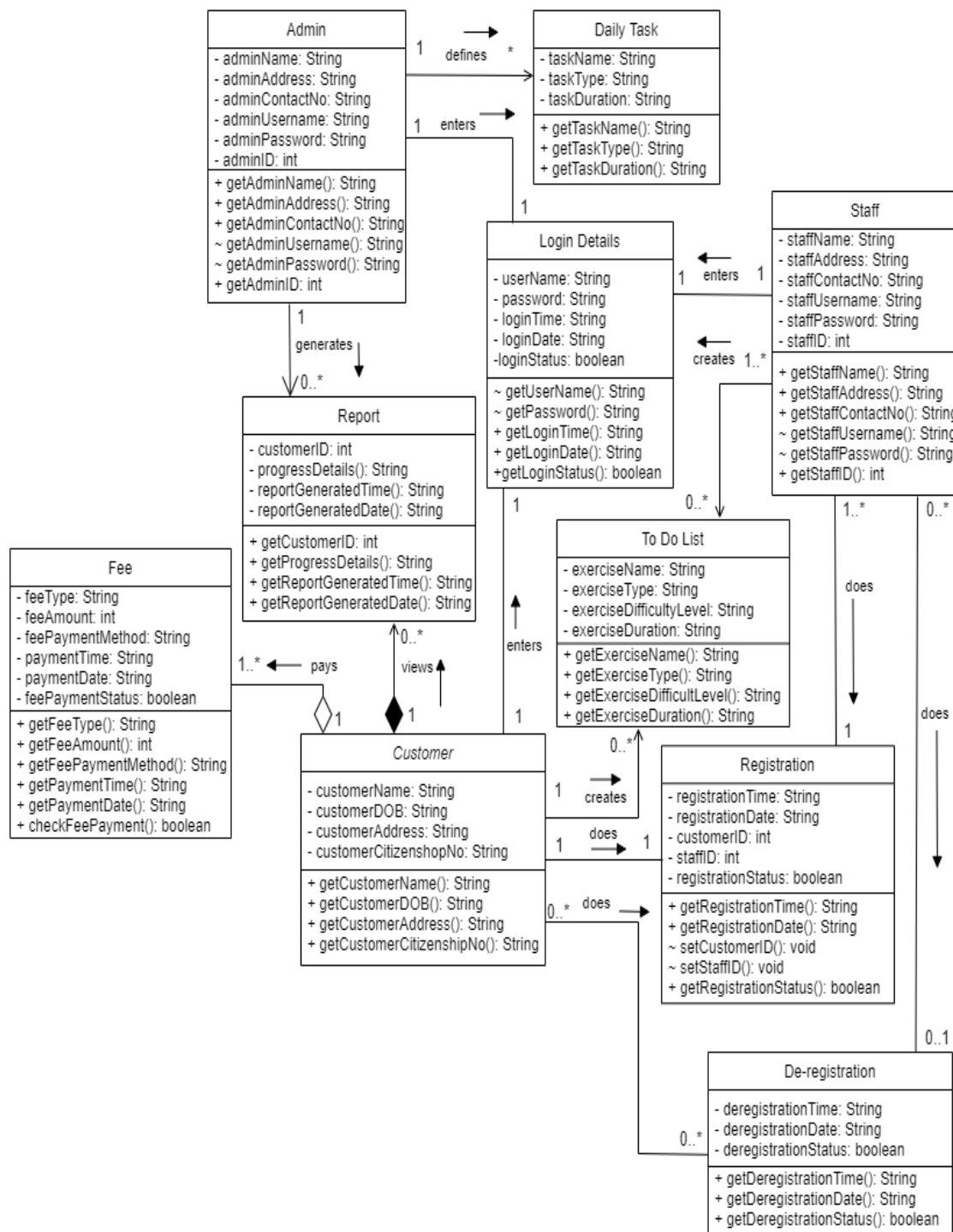


Figure 17: Addition of Associations

In this figure, associations are shown among the domain classes, along with the flow of messages and multiplicity. Here, Composition association relationship is shown between Customer and Report domain classes. Similarly, Aggregation association relationship is shown between Customer and Fee domain classes.

#### d. Adding Inheritance Relationship

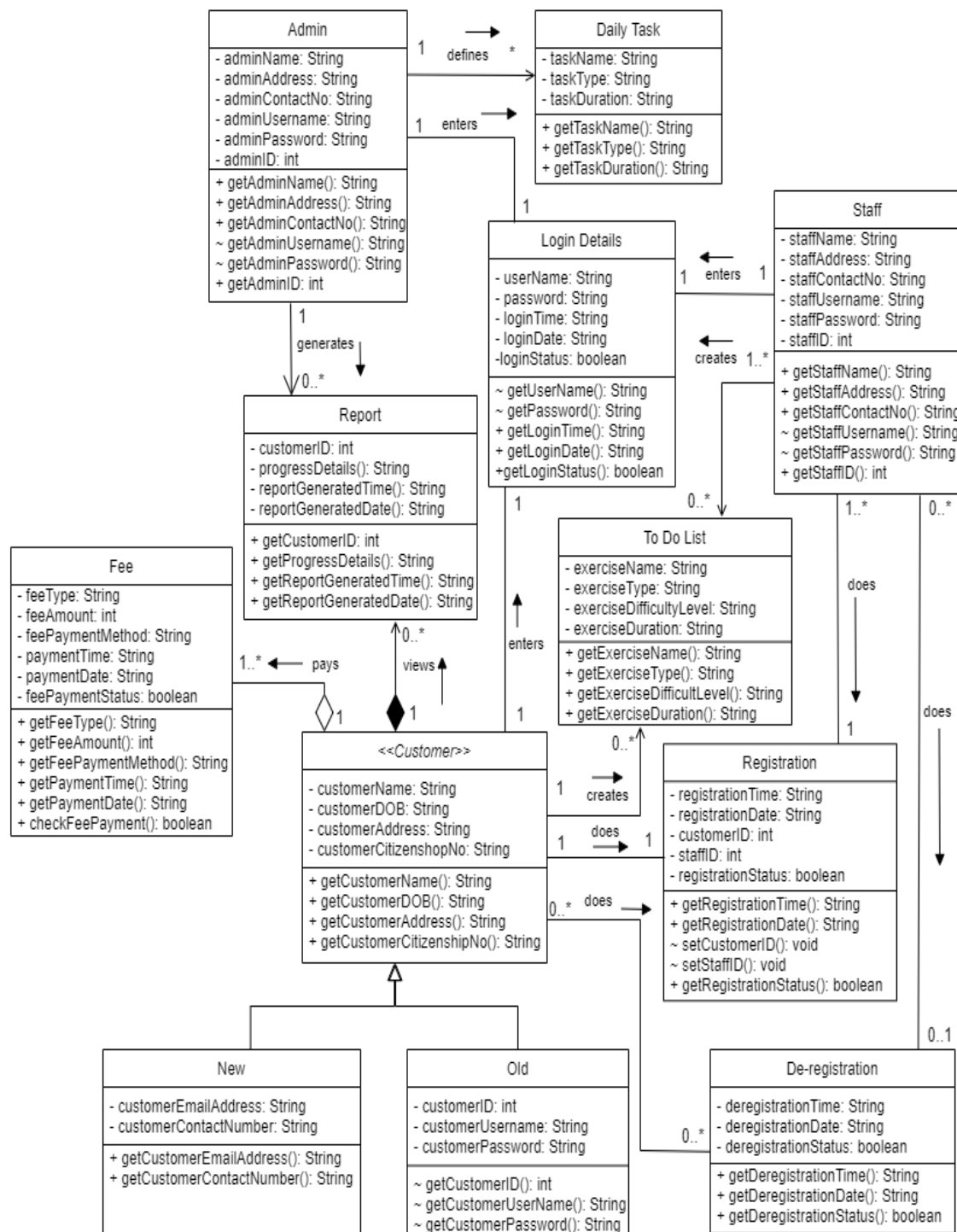


Figure 18: Perfect GYM Club's Class Diagram

In this figure, Customer's inheritance relationship is shown. Customer is Parent domain class, therefore it is named as "<<Customer>>". It's child classes are: "New" and "Old" which represents New Customer and Old Customer of the gym respectively. The Class Diagram of Perfect GYM Club is shown in the above figure.

## **5. Progressing forward from current development phase**

For this project, we followed the phases of Rational Unified Process (RUP). RUP has four phases, they are: Inception, Elaboration, Construction and Transition. Currently, we are at the beginning of the 3<sup>rd</sup> week, which means we are in the middle of the Elaboration phase as shown in the Gantt chart figure at the beginning of this report. Until now, we have already gathered the requirement analysis, scheduled resources, managed risk, time and cost estimation, and developed an initial prototype to show the system's model to the client. Similarly, we have already analysed the problem domain and designed the initial Unified Modeling Language (UML) diagrams such as Use Case diagram, Communication diagram, Sequence diagram and Class diagram of domain classes to represent the interactions and flow of information among different classes and objects of the system. These UML diagrams will be very much helpful to the developers, because it provides them an idea on how to code the program.

Although we have already drawn the UML diagrams, the designing stage for the project is still not completed, because we are following RUP methodology, which contains an Agile principle. In Agile principle, changes are welcomed at any stages during the development of the project, therefore requirements of the project can be changed frequently. When the requirement changes, the requirement for designing also changes, therefore the designers will have to design the UML diagrams in other RUP phases as well.

So now, our requirement analysis software engineers will work on verifying the project requirements and ask the client if any changes are to be made in the system. Here, analysts will also be involved to convert client's business language to the language that can be understood by designers and developers. Then, designers will modify the design of the system according to the changes as per request of the client. At the same time, developers will also start coding and along with it, testers will test the codes. Documentation of the system will also proceed along with these activities. So in this stage, developers, testers, analysts, software engineers and designers will all be involved.

At the same, the software engineers will start working on the project's planning, which are to be done in the Construction RUP phase. Since changes are accepted at any stages during the project development, designs are modified iteratively and new requirements are again coded by the developers. Then, testers will test the system with the help of different test cases. Since, agile principle is included in RUP methodology, clients are heavily involved during the project development. Therefore, we consider clients as part of our team. We will maintain a daily communication with our client in order to get their regular feedbacks. Regular feedbacks from clients are very much important in software development project, because it is easier to change the design and functionality of the system earlier during the initial stages, rather than late, when it is about time to release the system to the client. These processes will repeat continuously until the system is built and the client is satisfied. Once the system is built, the system manual and user guide will be created in the Construction phase.

Now, the system will finally be released and delivered to the client at the end of Transition phase on approximately 30<sup>th</sup> of April, 2020 as shown in the Gantt chart. In this phase, training will be provided to the client and their staff to give them information on how to run the system and its features. However, after we deliver the system to the client, some errors may still arise in the system. Hence, our team will keep in touch with the client and maintain the system by handling any errors that may arise after releasing the system. Since RUP is an iterative software development method, the system will be developed in different versions. Therefore, we will collect the user's reviews, feedbacks and additional functionalities requirements, and upgrade it in the system's next version. Therefore, there will be overlapping of all four RUP phases while developing the system.

The products or techniques that might be produced for the design are Entity Relation Diagram (ERD), Object-Oriented Design, and Data Dictionary. Firstly, ERD is known as the blue print of the system. It helps us in finding data requirements for developing a well-designed system. In our project, it might be used to obtain the information about the structure of our system with the help

of a diagram. It will also help to display the relationship between the entities and the attributes of our system (beginnersbook, 2020).

Similarly, we might use Object Oriented Design technique for constructing a detailed description by identifying how our system is going to be developed from the current stage. In this technique, system's context will be defined, system architecture will be designed, object of the system will be identified, design models will be designed, and object interfaces will be specified (tutorialspoint, 2020).

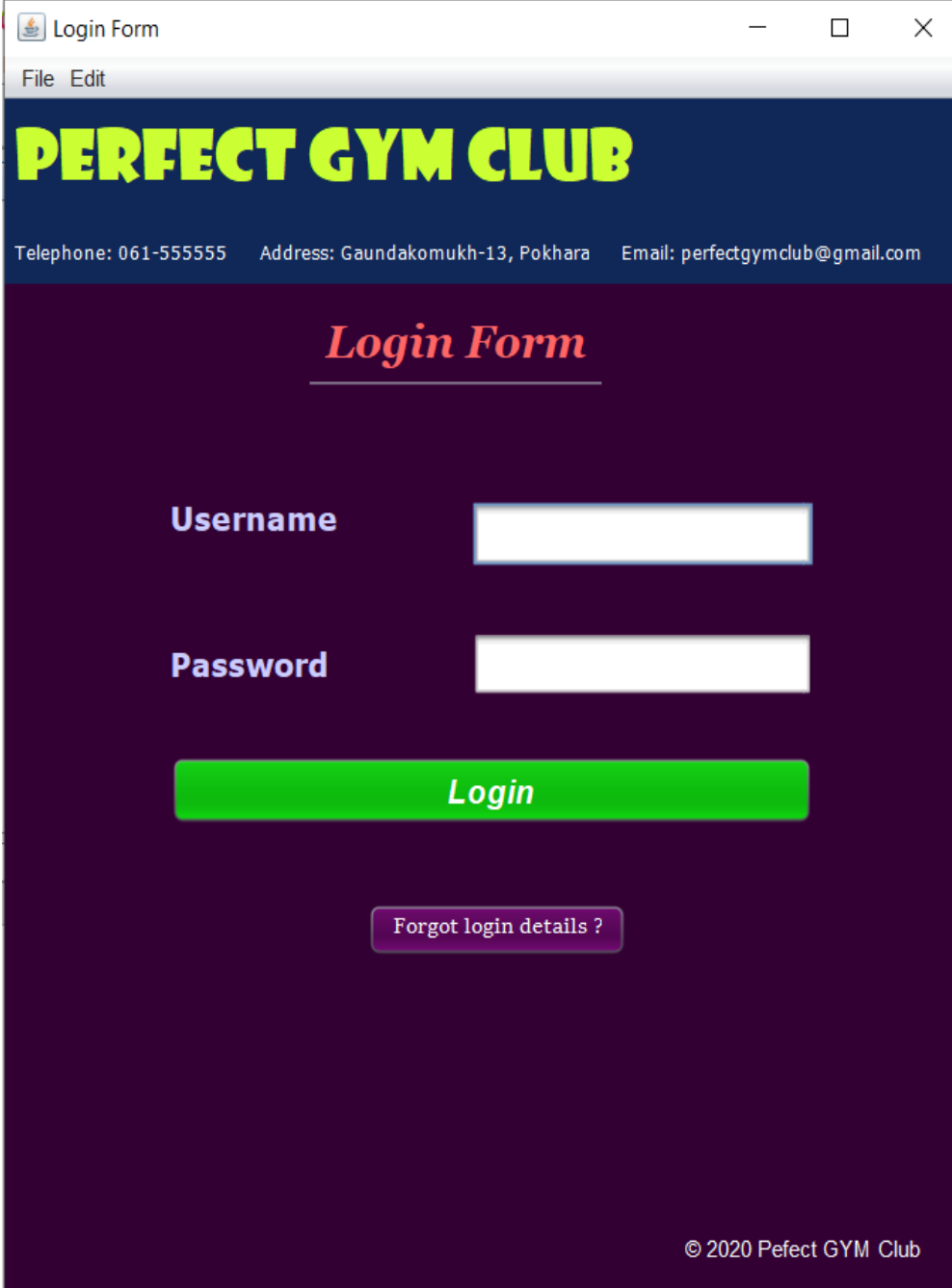
Correspondingly, we might also use Data Dictionary to avoid inconsistencies, improve data quality and reduce the data redundancy in our project. It also helps to provide clear and complete definition about the system requirements to the designers, which might simplify their work while designing the system (Janakiraman, 2020).

Hence, ERD, Object Oriented Design and Data Dictionary are the three products or techniques that we might produce for the system design.

## 6. Prototype

The prototype was developed with the help of Apache NetBeans tool. The prototype of Perfect GYM Club's application are shown in the figures below.

### a. Login form



The image shows a web browser window titled "Login Form" with a menu bar containing "File" and "Edit". The main content area has a dark blue header with the text "PERFECT GYM CLUB" in large, bold, yellow-green letters. Below the header, contact information is displayed: "Telephone: 061-555555", "Address: Gaundakomukh-13, Pokhara", and "Email: perfectgymclub@gmail.com". The central part of the form is on a dark purple background. It features the title "Login Form" in a red, italicized font, underlined. Below this, there are two input fields: "Username" and "Password", each with a white text box. A large green button labeled "Login" is positioned below the password field. At the bottom of the form area, there is a purple button labeled "Forgot login details ?". The footer of the page contains the copyright notice "© 2020 Pefect GYM Club".

Figure 19: Prototype of Login form

**b. Customer Registration Form**

Customer Registration form

File Edit

# PERFECT GYM CLUB

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

## Customer Registration Form

First name

Last name

Mobile number

Email address

Date of Birth

Gender ☐ Male ☐ Female ☐ Unspecified

Card type

Card number

New Username

New Password

Confirm Password

**Register**

[Cannot register ?](#) © 2020 Pefect GYM Club

Figure 20: Prototype of Customer Registration Form

**c. Customer Report Form**

Customer Report Form

File Edit

# PERFECT GYM CLUB

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

## Customer Report Form

Customer ID

Full name

Mobile number

Email address

Gender ☐ Male ☐ Female ☐ Unspecified

Age

Gym Joining Date

Month	Attendance (Days)	Weight (Kg)	Remarks
April	28	87	Overweight
May	29	82	Progressing well
June	27	75	Losing weight
July	20	70	Healthy weight

**Generate Customer Report**

[Problem ?](#) © 2020 Pefect GYM Club

*Figure 21: Prototype of Customer Report Form*



**d. Customer Deregistration Form**

The image shows a web browser window titled "Customer Deregistration Form". The browser's address bar shows "File Edit". The page has a dark blue header with the text "PERFECT GYM CLUB" in large, bold, yellow letters. Below the header, there is a dark blue bar with contact information: "Telephone: 061-555555", "Address: Gaundakomukh-13, Pokhara", and "Email: perfectgymclub@gmail.com". The main content area has a dark purple background. At the top of this area, the text "Customer Deregistration Form" is written in orange. Below this, there are several form fields: "Customer ID" with a white input box, "Full name" with a white input box, and "Reason" with a dropdown menu showing "Others (Please specify below)". Below the dropdown is a large white text area with a scrollbar. Further down, there are radio buttons for "GYM experience" with options "Poor", "Average", "Good" (selected), and "Excellent". Below this is the question "Do you plan to rejoin in the future ?" with radio buttons for "Yes", "No", and "May be" (selected). At the bottom, there is a white input box labeled "Enter Password". Below the password box is a large orange button with the text "Deregister" in white. At the very bottom, there is a small purple button with the text "Cannot deregister ?" and a copyright notice "© 2020 Pefect GYM Club".

Customer Deregistration Form

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

**Customer Deregistration Form**

Customer ID

Full name

Reason

GYM experience ☐ Poor ☐ Average ☒ Good ☐ Excellent

Do you plan to rejoin in the future ?

☐ Yes ☐ No ☒ May be

Enter Password

**Deregister**

[Cannot deregister ?](#)

© 2020 Pefect GYM Club

Figure 22: Prototype of Customer Deregistration Form

**e. Fee Payment Form**

Fee Payment Form

File Edit

# PERFECT GYM CLUB

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

## Fee Payment Form

Customer ID

Full name

Mobile number

Select a month for payment of your fee.

Choose a payment method.

Card number

Month	Year	Fee (Rs)	Fee status	Paid by (Card no)	Paid on
January	2020	1200	Paid	A087018000	2020-Feb-02

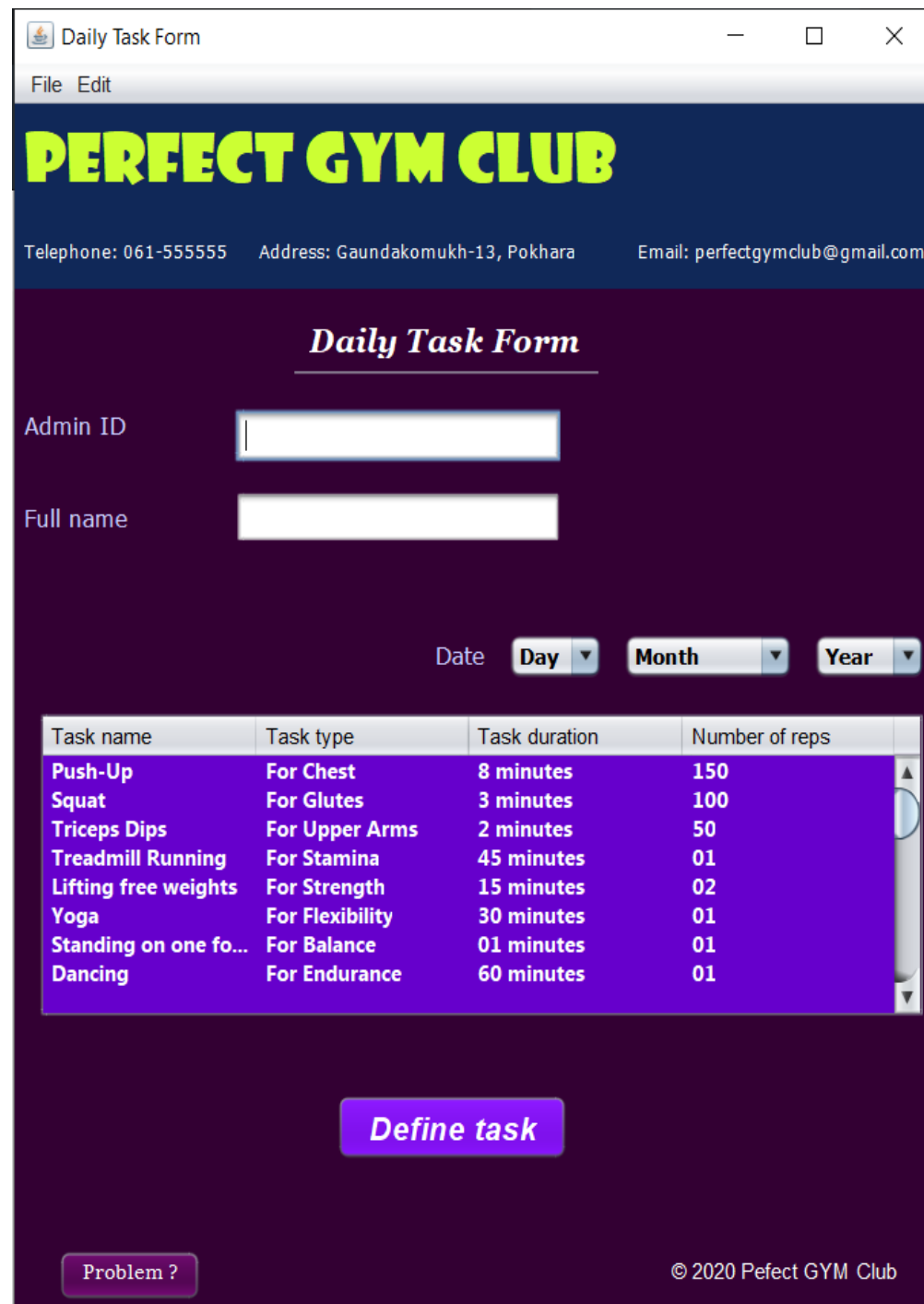
[Pay fee](#)

[Problem ?](#)

© 2020 Pefect GYM Club

*Figure 23: Prototype of Fee Payment Form*

## f. Daily Task Form



The image shows a web application prototype for a 'Daily Task Form'. The window title is 'Daily Task Form'. The header features the 'PERFECT GYM CLUB' logo in large, bold, yellow letters on a dark blue background. Below the logo, contact information is displayed: 'Telephone: 061-555555', 'Address: Gaundakomukh-13, Pokhara', and 'Email: perfectgymclub@gmail.com'. The main content area has a dark purple background. It features a title 'Daily Task Form' in a white, italicized font. Below the title are two input fields: 'Admin ID' and 'Full name'. A date selection section includes a 'Date' label and three dropdown menus for 'Day', 'Month', and 'Year'. A table with a light blue header and dark blue body lists various tasks. The table has four columns: 'Task name', 'Task type', 'Task duration', and 'Number of reps'. Below the table is a large orange button labeled 'Define task'. At the bottom left is a 'Problem ?' button, and at the bottom right is the copyright notice '© 2020 Pefect GYM Club'.

File Edit

# PERFECT GYM CLUB

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

## Daily Task Form

Admin ID

Full name

Date  Day  Month  Year

Task name	Task type	Task duration	Number of reps
Push-Up	For Chest	8 minutes	150
Squat	For Glutes	3 minutes	100
Triceps Dips	For Upper Arms	2 minutes	50
Treadmill Running	For Stamina	45 minutes	01
Lifting free weights	For Strength	15 minutes	02
Yoga	For Flexibility	30 minutes	01
Standing on one fo...	For Balance	01 minutes	01
Dancing	For Endurance	60 minutes	01

**Define task**

**Problem ?**

© 2020 Pefect GYM Club

Figure 24: Prototype of Daily Task Form

**g. To Do List Form**

**PERFECT GYM CLUB**

Telephone: 061-555555    Address: Gaundakomukh-13, Pokhara    Email: perfectgymclub@gmail.com

### *To Do List Form*

Customer ID

Full name

Mobile number

Email address

Exercise name	Exercise type	Exercise duration	Number of reps
Push-Up	For Chest	8 minutes	150
Squat	For Glutes	3 minutes	100
Triceps Dips	For Upper Arms	2 minutes	50
Treadmill Running	For Stamina	45 minutes	01

Reminder:

   © 2020 Pefect GYM Club

*Figure 25: Prototype of To Do List Form*

**h. Check Payment Form**

Check Payment Form

File Edit

**PERFECT GYM CLUB**

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

**Check Payment Form**

Customer ID

Full name

Mobile number

Email address

Select a month to check your fee payment status.

January ▼

Month	Year	Fee (Rs)	Fee status	Due date
January	2020	1200	Not paid	2020-Feb-07

**Check Payment**

[Problem ?](#)

© 2020 Pefect GYM Club

*Figure 26: Prototype of Check Payment Form*

**i. Staff Registration Form**

Staff Registration Form

File Edit

# PERFECT GYM CLUB

Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com

## Staff Registration Form

First name

Last name

Mobile number

Email address

Date of Birth

Gender ☐ Male ☐ Female ☐ Unspecified

Card type

Card number

New Username

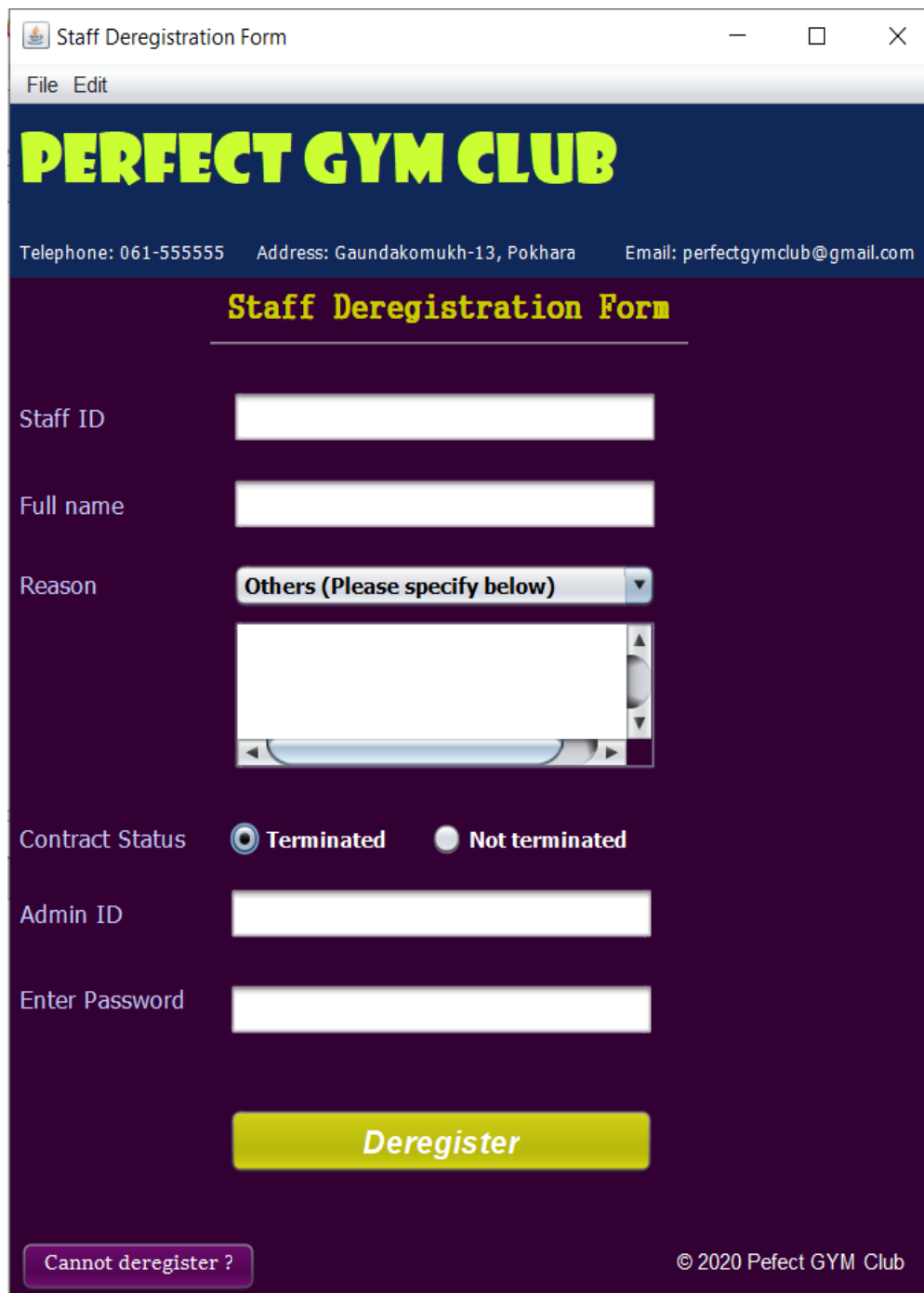
New Password

Admin Password

**Register**

[Cannot register ?](#) © 2020 Pefect GYM Club

*Figure 27: Prototype of Staff Registration Form*

**j. Staff Deregistration Form**

The image shows a web browser window titled "Staff Deregistration Form". The browser's address bar shows "File Edit". The website has a dark blue header with the text "PERFECT GYM CLUB" in large, bold, yellow letters. Below the header, there is a dark blue bar with white text: "Telephone: 061-555555 Address: Gaundakomukh-13, Pokhara Email: perfectgymclub@gmail.com". The main content area has a dark purple background. At the top of this area, the text "Staff Deregistration Form" is written in yellow. Below this, there are several form fields: "Staff ID" with a white text input field; "Full name" with a white text input field; "Reason" with a dropdown menu showing "Others (Please specify below)" and a white text area below it; "Contract Status" with two radio buttons, "Terminated" (selected) and "Not terminated"; "Admin ID" with a white text input field; and "Enter Password" with a white text input field. At the bottom of the form, there is a large yellow button with the text "Deregister" in black. In the bottom left corner, there is a small purple button with the text "Cannot deregister ?". In the bottom right corner, there is a small white text "© 2020 Pfect GYM Club".

Staff Deregistration Form

Staff ID

Full name

Reason Others (Please specify below)

☒ Terminated ☐ Not terminated

Admin ID

Enter Password

**Deregister**

[Cannot deregister ?](#)

© 2020 Pfect GYM Club

Figure 28: Prototype of Staff Deregistration Form

## Conclusion

The coursework was based on development of a computerized information system as a Software Engineer for Perfect GYM Club to record the gym's customers and their payment details. For this project, we followed Rational Unified Process (RUP). It is an iterative software development process which consists of four phases: Inception, Elaboration, Construction and Transition. Similarly, the system was designed with the help of Unified Modeling Language (UML). It is a standardized modelling language that consists of many structure and behavioral diagrams. In our project, Use Case diagram, Class diagram, Sequence diagram and Communication diagram of the system were designed.

In this coursework, the prototype of computerized system was developed with the help of Apache NetBeans tool. The prototype contains all the features as well as some additional features that were mentioned in the detailed specifications of the Perfect GYM Club. A Gantt chart reflecting the RUP method was also developed in the coursework to plan the project's development. The coursework was a tough one because of the nature of the module itself. Software Engineering is a tough module because while doing any project as a Software Engineer, a lot of theories, methods and principles needs to be followed and that is why, a lot of difficulties arose during this coursework as well. In my case, drawing Use Case diagram, Communication diagram, Sequence diagram and Class diagrams were the most difficult task. In UML diagrams, one diagram is related with another diagram, and therefore, if there is any error in one diagram, then all the remaining diagrams also needs to be changed. Therefore, a lot of discussions with friends, suggestions from module leaders, researches on the internet and reviews of lecture slides were required to overcome these difficulties.

This coursework was a very fruitful and productive project. I gained tons of knowledge on developing a system's prototype, RUP methodology and UML diagrams. Similarly, I got chance to work on this project as a Software Engineer, and therefore I got to know how challenging task it is for a Software Engineer to work on a project. Last but not the least, this coursework helped me to enhance my capability and endurance to work on a project under huge pressure, and for that, credit goes to this coursework.



## References

- beginnersbook, 2020. *beginnersbook*. [Online]  
Available at: <https://beginnersbook.com/2015/04/e-r-model-in-dbms/>  
[Accessed 28 April 2020].
- Britton, C. & Doake, J., 2005. *sciencedirect*. [Online]  
Available at: <https://www.sciencedirect.com/topics/computer-science/case-description>  
[Accessed 21 April 2020].
- Gantt, 2020. *Gantt*. [Online]  
Available at: <https://www.gantt.com>  
[Accessed 17 April 2020].
- Janakiraman, P., 2020. Data Dictionary and Normalization. *technowaveinc*.
- Powell-Morse, A., 2017. Rational Unified Process: What Is It And How Do You Use It?. *airbrake*, 14 March.
- tutorialspoint, 2020. *tutorialspoint*. [Online]  
Available at:  
[https://www.tutorialspoint.com/object\\_oriented\\_analysis\\_design/ooad\\_object\\_oriented\\_design.htm](https://www.tutorialspoint.com/object_oriented_analysis_design/ooad_object_oriented_design.htm)  
[Accessed 28 April 2020].
- uml-diagrams, 2020. *UML Use Case Diagrams*. [Online]  
Available at: <https://www.uml-diagrams.org/use-case-diagrams.html>  
[Accessed 19 April 2020].
- Wright, E., 2014. *slideshare*. [Online]  
Available at: <https://www.slideshare.net/ElizaWrightCarter/what-is-uml-unified-modeling-language>  
[Accessed 15 April 2020].