**Final Report  
7COM1025-0105-2021**

**COURSE MANAGEMENT SYSTEM**

**SHAYA SATHIYAN**

of semester B in the year 2022 in  
partial fulfillment of the requirements for the award of   
Master of Science   
in   
Advanced Computer Science

**Department of Computer Science   
University of Hertfordshire  
April 2022**

Contents

[Abstract 4](#_Toc100671517)

[1.0 Introduction 5](#_Toc100671518)

[1.1 Project Background 5](#_Toc100671519)

[1.2 Project aims / objectives 6](#_Toc100671520)

[1.2.1 Computerization 6](#_Toc100671521)

[1.2.2 No Redundant Data 6](#_Toc100671522)

[1.2.3 Automation 6](#_Toc100671523)

[1.2.4 Easy Interaction 6](#_Toc100671524)

[1.3 Project Description 7](#_Toc100671525)

[1.3.1 Functional Requirements 7](#_Toc100671526)

[1.3.2 Non-functional Requirements 7](#_Toc100671527)

[2.0 System Design and Implementation 8](#_Toc100671528)

**Table of Figures**

[**Figure 1: Entity Relationship (ER) Diagram 8**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690871)

[**Figure 2: Class Diagram 9**](#_Toc100690872)

[**Figure 3: Login Page 10**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690873)

[**Figure 4: Dashboard 11**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690874)

[**Figure 5: Booking Page 11**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690875)

[**Figure 6: When using a day to filter the timetable to book a lesson 11**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690876)

[**Figure 7: Student's Booking History 12**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690878)

[**Figure 8: Edit or Cancel the Upcoming Booking in Booking list 12**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690879)

[**Figure 9: Students rate and review the lessons they have attended. 12**](file:///E:\MSC\SemesterB\7COM1025%20PSE\Assign\course-management-system\Final%20Report.docx#_Toc100690877)

# Abstract

Implementing a system in Java using Swing can be a challenging task that necessitates knowledge of several major libraries as well as advanced Java concepts. A windowing toolkit is generally responsible for providing a framework that allows a graphical user interface (GUI) to display the right features on the screen at the right moment in a graphical system. In this report, I designed and developed a simple course management system project to manage student demands for group exercise courses. The goal of this project is to create a system that allows you to check the timetable and find an open slot for a given day, time, and lesson. Examining the cost of each lesson as well as the rating of each lesson. The Java programming language was used to build this project, which included components from the swing library. As a result of our project, students will be able to use FreeHand to book a class, rate the lesson, and perform any other activities in a simple manner.

**Keywords**: NetBeans IDE 8.2, Swing, GUI, Object-Oriented Programming, Java 8, Stream, SOLID Principles, Data Transfer Object (DTO), Data Access Object (DAO), ArrayList, Map, Set, Collection Framework.

# 1.0 Introduction

This chapter gives an overview about the aim, objectives and background of the system. It also mentioned the assumptions that I made when developing this system.

# 1.1 Project Background

Online systems form the backbone of every nation, and hence it is important to provide a strong technology foundation to younger generations to ensure the development of open-minded global citizens securing the future of everyone. Advanced technology available today can play a crucial role in streamlining all processes to promote solidarity among people. This application is designed to reduce the inconveniences & disadvantages of the manual based booking in course management.

Online Course Management System is a system which University Sports Centre (USC) needs software for managing the bookings of group exercise lessons made by the students. It must maintain track of the price of each lesson, the rating of each lesson, the timetable, and the available slot for a specific day, time, and lesson. This is very difficult to manage manually. Maintenance of all this information manually is a very complex task. Owing to the advancement of technology, organization of an Online Course Management becomes much simple. The Online Course Management has been designed to computerize and automate the operations performed over the information about the student’s booking issues and rating and all other operations. This computerization of course helps in many instances of its maintenances. It minimizes management's workload by reducing the amount of manual work required.

Course management system consists of task such as login to the student account, checking the timetable, booking a lesson, keeping controlling the booking, producing the monthly reports, and providing rating and reviews.

# 1.2 Project aims / objectives

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

1. Check the course timetable.
2. Book a course.
3. Change a booking.
4. Cancel the future booking.
5. Provide a numerical rating.
6. Write a review.
7. Search availability of course and time.
8. Facility to generate the monthly reports. (Monthly lesson report & champion exercise report)

## 1.2.1 Computerization

1. To save the stationary wastage.
2. To provide Prior information about lesson’s availability.
3. To provide a computerized management of reviews, rating and attendance.

## 1.2.2 No Redundant Data

1. To provide Reliable update on student’s attendance and monthly reports.
2. To provide the review and rating for the service.

## 1.2.3 Automation

1. To reduce the time that spends to store/ update / retrieve the records.
2. To provide an automated attendance, monthly lesson report and champion exercise report.

## Easy Interaction

1. To designed for better interaction between student and management.
2. To provide easy platform to store day today lesson activities.

# Assumptions

In order to construct this project, I made some assumptions.

1. When the user wants to filter the timetable by date, he or she cannot combine day, time, or course with date. They can only filter by date.
2. When the user provides the numerical rating only, his/her status changed to “Attended”.
3. I maintained the records for booking availability until June 5, 2022

# 2.0 System Analysis

# 2.1 Software Requirement Specification

This section explains what the system will do and how it will be expected to perform. It also defines the features that the product must have to meet the needs of end-users.

## 2.1.1 Functional Requirements

1. Course Management System shall be Online based.
2. Student shall login by his/her account.
3. Student shall check the timetable.
4. Student shall book new lesson.
5. Student shall change the lesson and time according to the availability.
6. Student shall cancel the future booking.
7. Student shall view future booking of their lessons.
8. Student shall view history of their lessons.
9. Student shall view their attendance.
10. Student shall generate the monthly reports.
11. Student shall provide the review and rating for their lessons.

## 2.1.2 Non-functional Requirements

1. The System should be easy to use.
2. The System should be available 24 hour.
3. The System should response on time.
4. The System should provide specific information to specific user.
5. The System should not fail.
6. Right information is available to right student at right time.

# 2.2 Tools

1. NetBeans 8.2
2. GitHub

# 2.3 Technologies

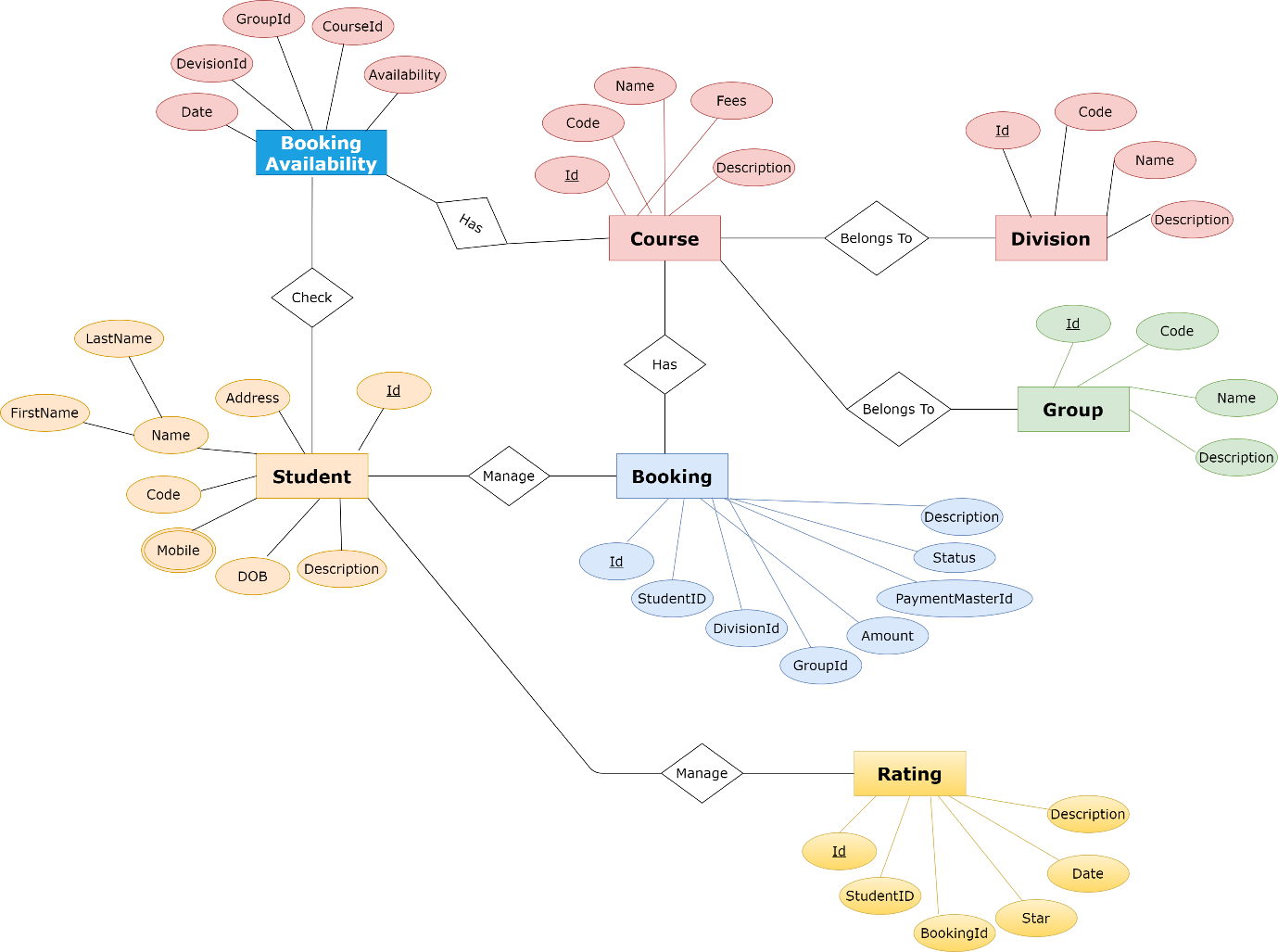
1. Java 8
2. Collection Framework (ArrayList, Map and Set)
3. Swing
4. Stream API

# 3.0 System Design and Implementation

The entity relationships, UI designs, and how the system works all are discussed in this section. I further highlighted the best practices I followed to implement this system in place.

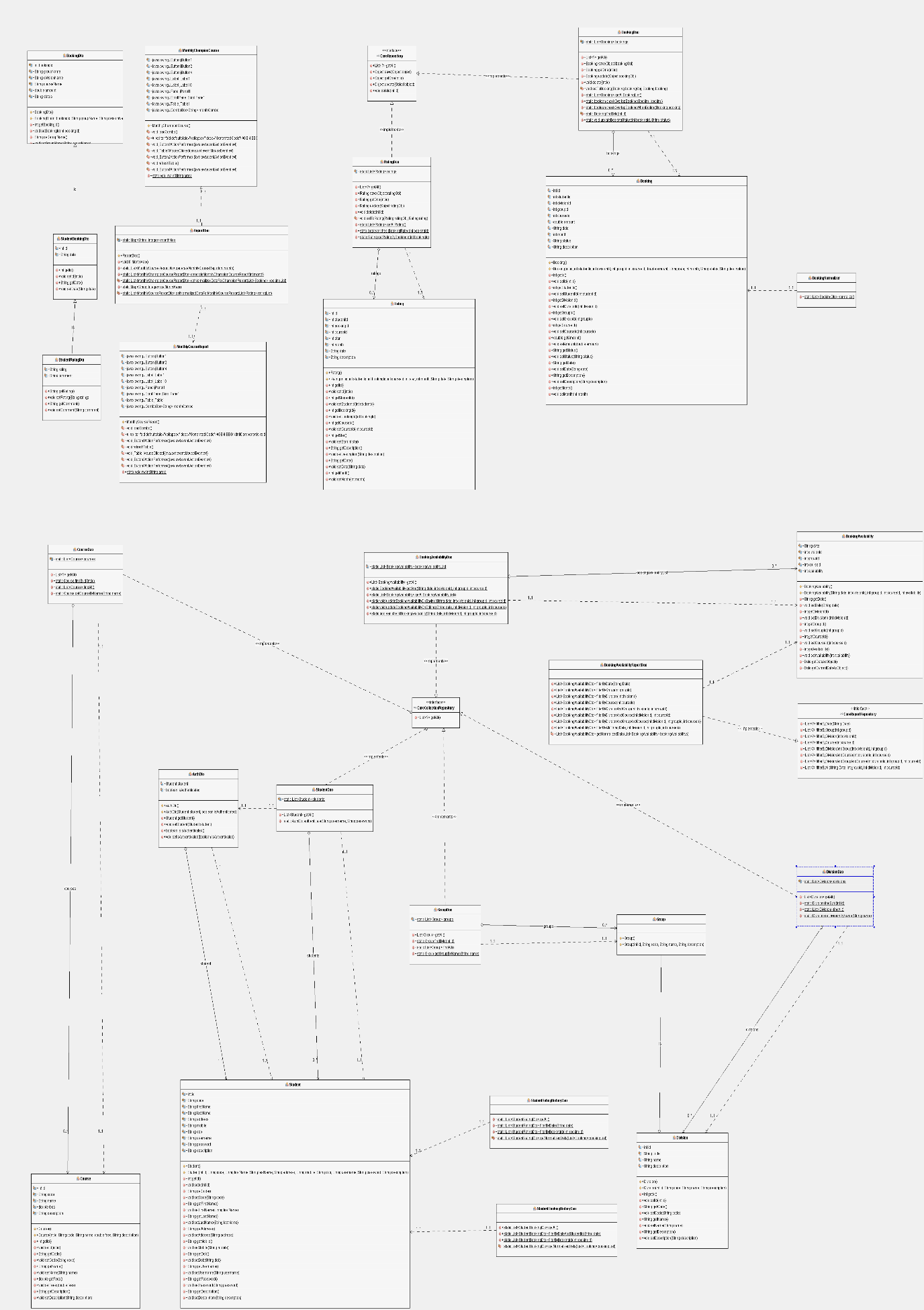
# 3.1 UML Diagram

## 3.1.1 ER Diagram



**Figure 1: Entity Relationship (ER) Diagram**

## 3.1.2 Class Diagram

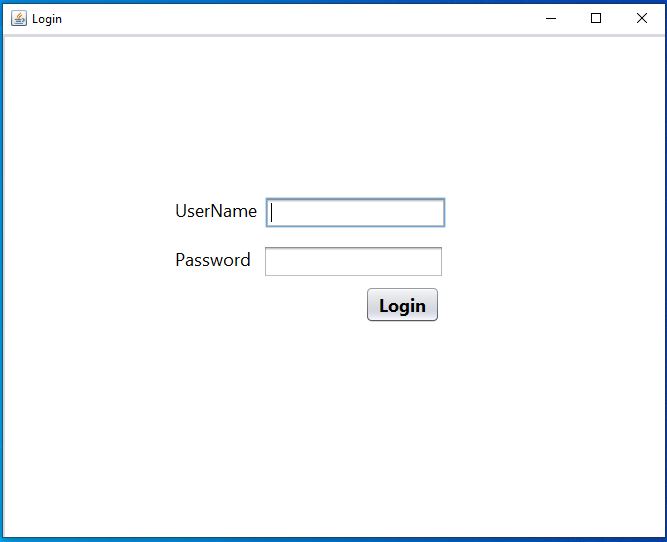


**Figure 2: Class Diagram**

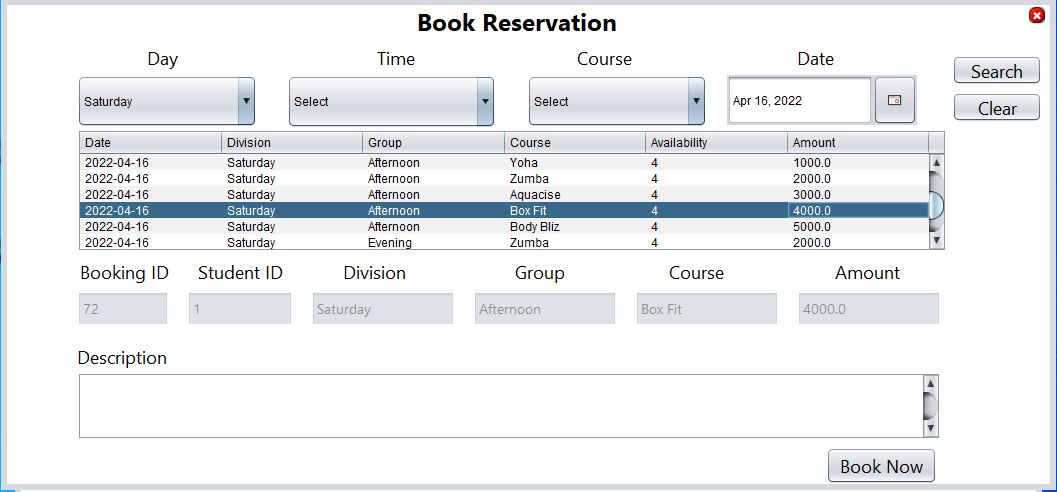
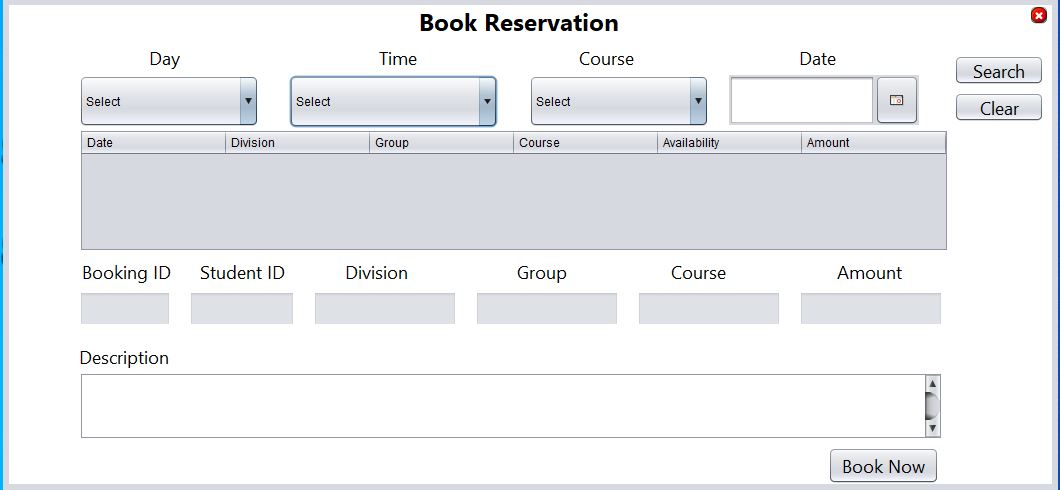
# 3.2 Basic GUI

A GUI program provides a much more advanced user interface, in which the user interacts with GUI components such as windows, buttons, text input boxes, and so on, using a mouse and keyboard. In this case, the GUI was implemented using Swing. In Swing,

1. JFrame and JPanel: Java has a built-in class to represent windows. Although there are various distinct types of windows, the JFrame class represents the most prevalent (which is included in the package javax.swing).A JFrame is an independent window that can be the main window of an application. JPanel is a content, display panel, which is used as a drawing surface and holds other components.
2. Events and Listeners: The physical appearance of a GUI is determined by the structure of containers and components, but it reveals nothing about how the GUI behaves. The majority of graphical user interfaces are event-driven, which means that the software waits for actions that are triggered by the user's activities. The application responds to an event by calling an event-handling method that developers build the methods to respond to the events that the user is involved in. The most common technique for handling events in Java is to use event listeners. A listener is an object that includes one or more event-handling methods.

Here, I have included some screenshots of my system's user interface.

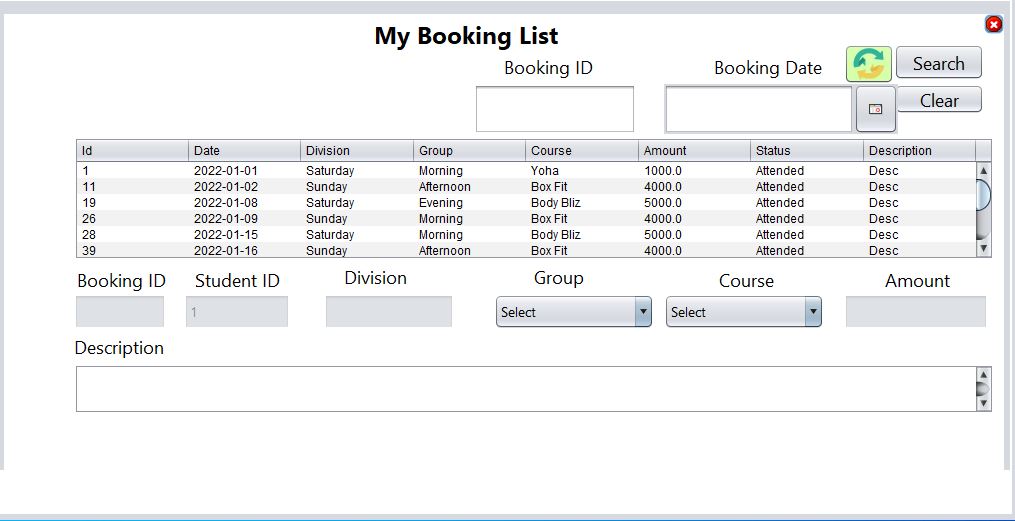
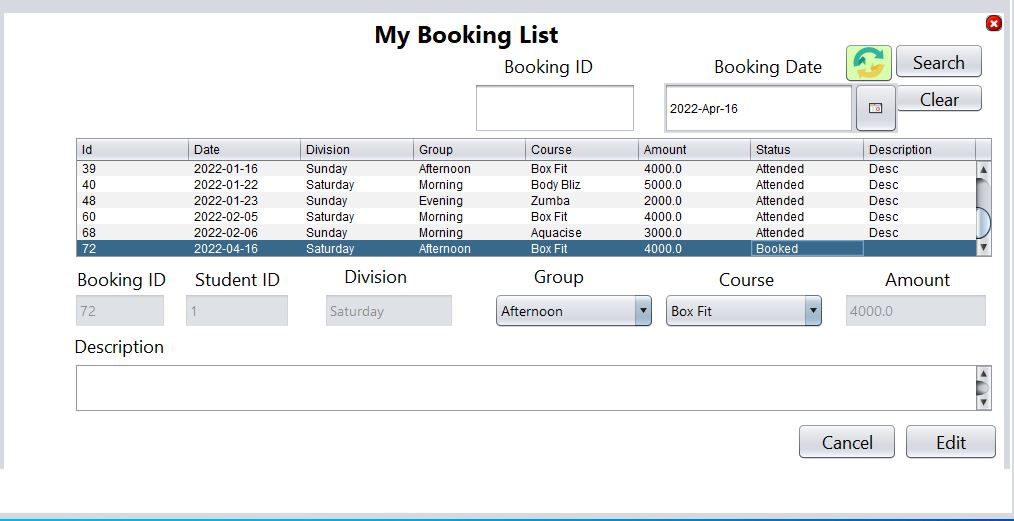
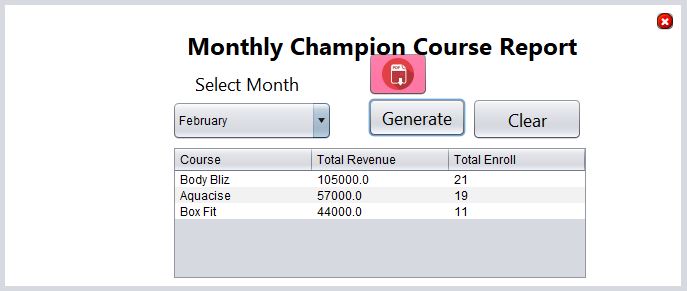
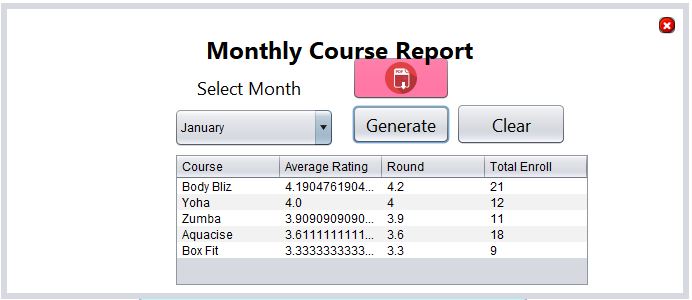
**Figure 3: Login Page**



**Figure 4: Dashboard**

**Figure 5:** **Booking Page**

**Figure 6**: **When using a day to filter the timetable to book a lesson**

****

**Figure 9: Students rate and review the lessons they have attended.**

**Figure 8: Edit or Cancel the Upcoming Booking in Booking list**

**Figure 7: Student's Booking History**

**Figure 10:** **Generate the Monthly Course Report**

**Figure 11: Generate the Monthly Champion Course Report**

# 3.3 Overall Structure

This is a stand-alone application for booking a group fitness lesson. I used Java 8 to do this project. This project does not use any external databases. To develop this system, I used a variety of best practices, including design patterns, SOLID principles, Collection framework, Stream API, and version control.

The system's key feature is the ability to schedule a lesson. The student must first check the lesson's availability, price, and timing. After successful login, the user is redirected to a dashboard (Figure 4). There are five features in the Dashboard, including book my course, student booking list, student rating, and two types of monthly reports. If a student needs to register a lesson, he or she should select "Book My Course" from the dashboard menu. It takes you to the page where you may make a reservation (Figure 5). There, students can check the timetable in a variety of ways:

1. one is by specifying the day (Saturday or Sunday).
2. the other one is by specifying the course name (Yoga, Zumba…).
3. the other one is by specifying the time (Morning, Afternoon or Evening).
4. the other one is by specifying the date.
5. the other one is by specifying the time and day.
6. the other one is by specifying the time, course name and day.
7. the other one is by specifying the Course name and day.
8. the other one is by specifying the time, date, course name and day

After screening the timetable, the student chooses a course and books the lesson by clicking the Book button. The page is navigated to the Booking List after a successful booking. Students can see their past and future lessons that they have already registered on that list. Students can verify their attended, cancelled, changed and booked lesson from that list. Also, if he or she needs to change or cancel the future booking, pick that row and click the edit button to alter or click the cancel button to cancel the booking according to availability and preference. User cannot edit or cancel the past events. After their action, the status of the lesson is updated as changed or cancelled, depending on what they did.

If a student takes a class, he or she is required to rate that class. A button labelled "My Rating" can be found on the Dashboard. When you click that button, you'll be taken to My Rating List (Figure 9). In such situation, Student will be unable to leave any feedback on future bookings. Only students have the ability to rate previous lessons that they attended. If the student gives a rating only, the status changes to attended; otherwise, the status does not change to attended. Providing a numerical rating to a course is a must-do task for students. But if they want to give review then they can send review as well.

In every month, the system generates and print the two monthly reports. Those are,

1. Monthly Course Report (Figure 10) – It containing the number of students per group exercise lesson on each day, along with the average rating.
2. Monthly Champion Course Report (Figure 11) - It containing the group exercise which has generated the highest income, counting all the same exercise lessons together.

When the student has completed his or her tasks, he or she can log out of the system at any moment. I assumed the students were already registered in the system based on the coursework guidelines.

For implementing the project, Design Patterns, SOLID Principles, Collection Framework, and Version Control were some of the best practices I followed.

## 3.3.1 Object Oriented programming principles

Java supports object-oriented programming techniques that are based on a hierarchy of classes and well-defined and cooperating objects. One object-oriented concept that helps objects work together is inheritance. The relationship is one of parent to child, with the child or extending class inheriting all of the parent class's attributes. All classes in Java derived from **java. lang.Object** and shared its methods. The **java.lang.Object** methods are also shown because they are inherited and implemented by all of its subclasses, which are all of the Java API libraries' classes. Encapsulation was also used to achieve loose coupling and protects an object from unwanted access by users. I used private variables in classes to accomplish this. Not only these two concepts used, but also abstraction and polymorphism as well. In Java, I used interface to achieve abstraction. So, in order to develop this system, I used all four object-oriented concepts.

## 3.3.2 Design Patterns

I can make my code more flexible, reusable, and maintained by adopting design patterns. Because java follows design patterns internally, this is the most significant aspect. Here I used Data Transfer Object (DTO) and Data Access Object (DAO).