|  |
| --- |
| 7COM1025 |
| PROGRAMMING FOR SOFTWARE ENGINEERS |
| [Document subtitle] |

|  |
| --- |
| ADMIN  [Date] |

# Table of content

1. Introduction………………………………………………………….2
2. Class diagram………………………………………………………..2
3. Assumptions…………………………………………………………5
4. Design patterns………………………………………………………5
5. Refactoring…………………………………………………………..7
6. Software artefact…………………………………………………….8
7. Testing and evaluation……………………………………………...11
8. References………………………………………………………......16

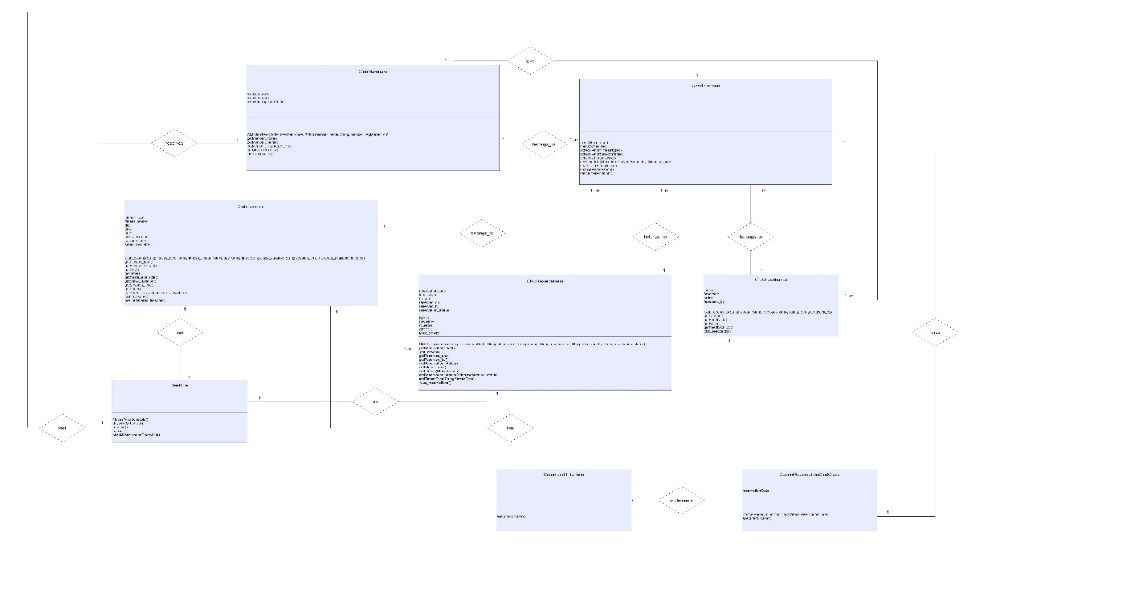
# 1.Introduction

In this fast paced era, businesses are keen on leveraging advanced and high-tech solutions for improvising business processes so that service quality can be improved. Companies seek to achieve this goal by cutting down customer waiting times and simplifying core business processes through deployment of automated business solutions. One such business initiative is undertaken by a company by the name of “Weekend Fitness Club” that has been operational in fitness industry from past many years (Batool et al., 2015). The board members of WFC are now planning to deploy a software solution with which they can automate their order booking process. For this purpose, a JAVA based software solution is developed that can satisfy all key requirements that are laid out by WFC staff with respect to software system. The report below highlights UML class diagram, assumptions, design principles, JUnit testing results and artefact which has been developed.

# 2.Class Diagram

A UML class diagram has been prepared in conjunction with the WFC system, to show classes, their corresponding attributes, key methods and associations/links to other classes in the table provided. The class diagram is attached below-

**Class Diagram: -**



In the above presented class diagram, it can be observed that there are a total of 8 classes which are created, and each class has its own attributes, key methods and links to other classes (Mulge et al., 2018). Different classes and their corresponding attributes, key methods and their associations are showed below-

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 1 “Club Members” | Member\_code  Member\_name  Member\_registered\_on | clubMembers()  getMember\_code()  getMember\_name()  getMember\_registered\_on()  getClub\_members()  club\_members() | 1-1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 2 “ClubLessons” | Fitness\_type  Fitness\_lesson  Day  Time  Price  Clas\_duration  Session\_time  Seats\_available | clubLessons()  getFitnes\_type()  getFitness\_lesson()  getDay()  getTime()  getSeats\_available()  getClass\_duration()  getSession\_time()  getPrice()  setSeats\_available()  club\_lessons()  pre\_registered\_lessons() | 1-1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 3 “TestFile” | - | fitnessTypeSchedule()  daywiseSchedule()  reerve()  cancel()  checkReservationCodeValid() | 1-1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 4 “ClubReservations” | reservationCode  fitnessType  lesson  reserved\_on  reserved\_by  reservation\_status  BOOK  CHANGE  CANCEL  ATTEND  MAX\_SEATS | ClubReservations()  getReservtionCode()  getLesson()  getReservered\_on()  getReserved\_by()  getReservation\_status()  getFitnessType()  setLesson()  setReservation\_status()  SetFitnessType()  Club\_reservations() | 1-M |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 5 “UserInterface” | - | Main()  checkSchedule()  SCheduleForFitnesType()  ScheduleForSessionTime()  ScheduleForDaywise()  getSchedule()  seatsLeft()  changeReservation()  cancelReservation() | 1-M |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 6 “ClubFeedbacks” | Lesson  Feedback  Rating  Feedback\_by | ClubFeedbacks()  getLesson()  getFeedback()  getRating()  getFeedback\_by()  club\_feedbacks() | 1-M |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 7 “Command: Interface” | - | executeFunction() | 1-1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Attributes** | **Key Methods** | **Associations** |
| Class 8 “CancelReservationCmdClass” | reservationCode | CancelReservationCmdClass() | 1-1 |

# 3.Assumptions

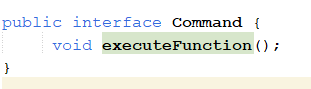
When writing code for the proposed software system, there were a few assumptions which were considered. These assumptions are identified below-

* Customers can check timetables by entering session time – either by selecting morning or evening batch
* System will generate repots for the lessons according to booking time of the lessons and status of booking will be set as “attended”
* Before attending class, the booking status should either be “booked” or “changed”. In case, it is “attended” or “cancelled”, then user will not be allowed to add the feedback for the lesson
* Members will be added to system if they book at least one lesson
* Members will be able to check number of seats that are available for lessons while checking schedule (Pinto et al., 2015).

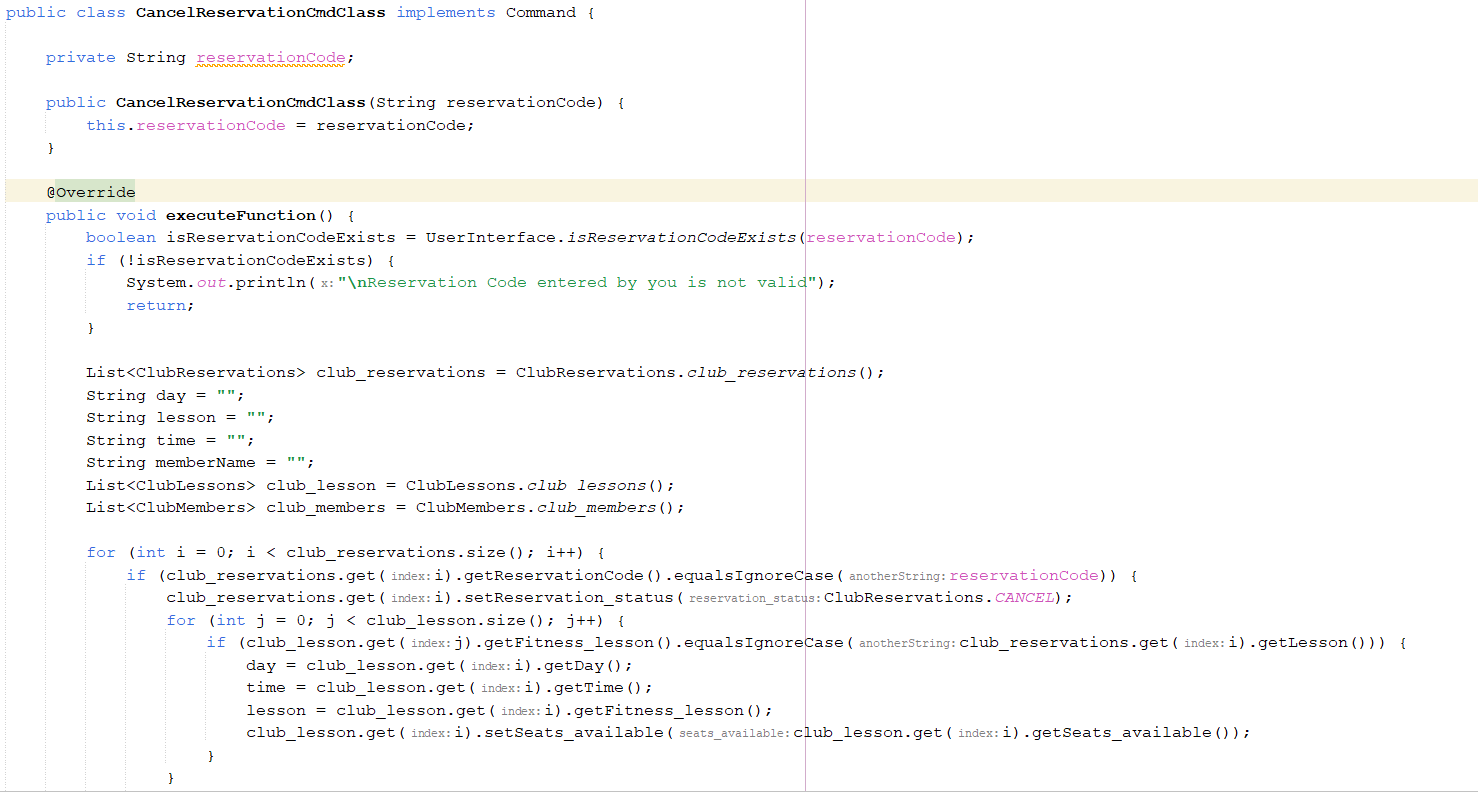
# 4.Design Patterns

As far as design patterns are concerned, it is really important to understand that patterns of “Command pattern” design approach will be taken into consideration. It is worth noting that command pattern approach works by encapsulating a request as an object which is called in classes.

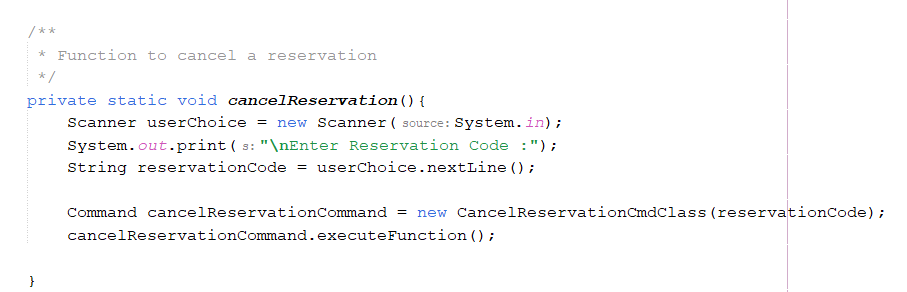
In context of the WFC system project, command pattern is applied through creation of an interface namely “Command” in which a method has been created – executeFunction(). It can be seen as follows-



After an interface is created, the next step involves creation of a class - “CancelReservationCmdClass” which works through implementation of above method. In this way, it will encapsulate the cancel reservation functionality.



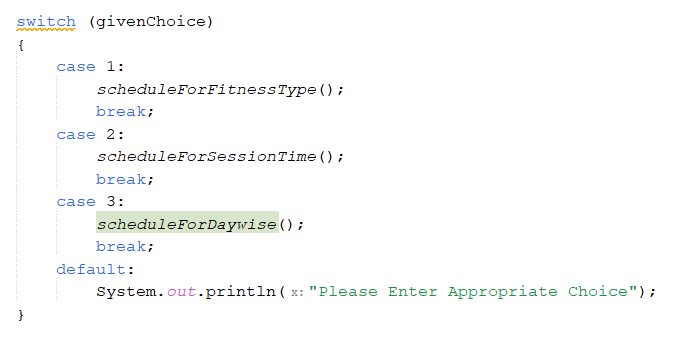
Now, an instance has been created of “CancelReservationCmdClass” so that it can perform cancel functionality.



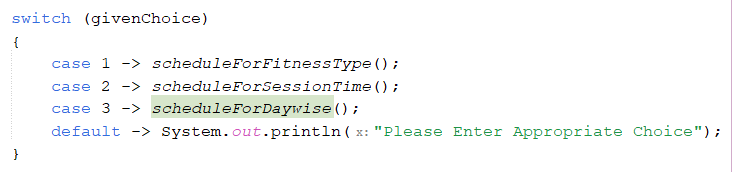
In this way, cancel reservation facility has been separated, as a new instance and class is created which is done to make code more flexible and maintainable such that it can handle future complexities in an easy and efficient manner.

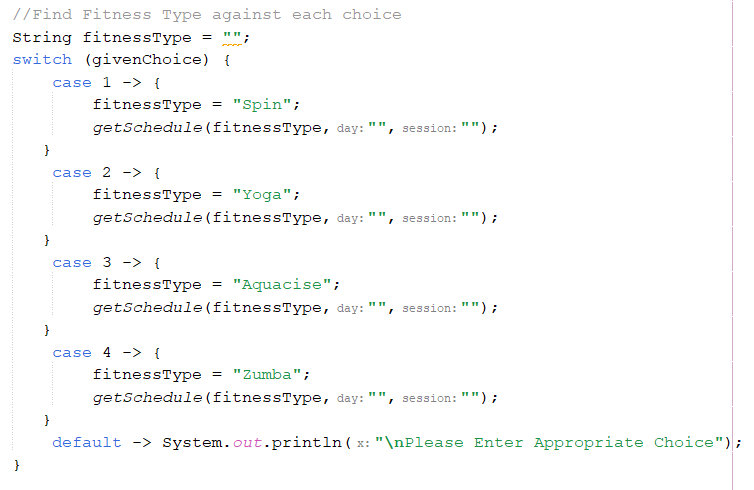
# 5.Refactoring

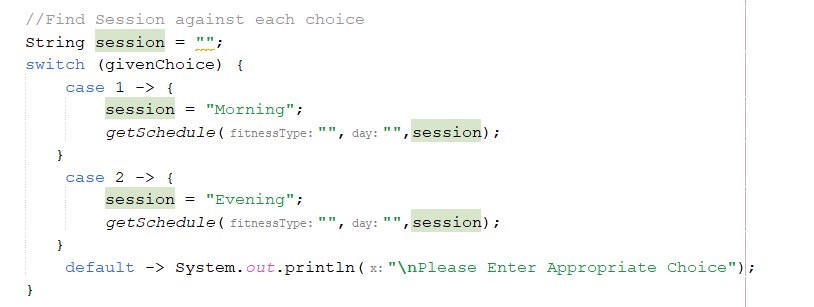
Refactoring has been done in the original code to make code more flexible and maintainable. Previously, ‘switch’ case was used, which made the code really complex to manage as only limited number of cases could be added. The use of simple ‘switch’ case can be seen in screenshot below-



The code has been refactored, in which switch case is improved, this improved version of switch case can be seen in the code snip which is appended below-





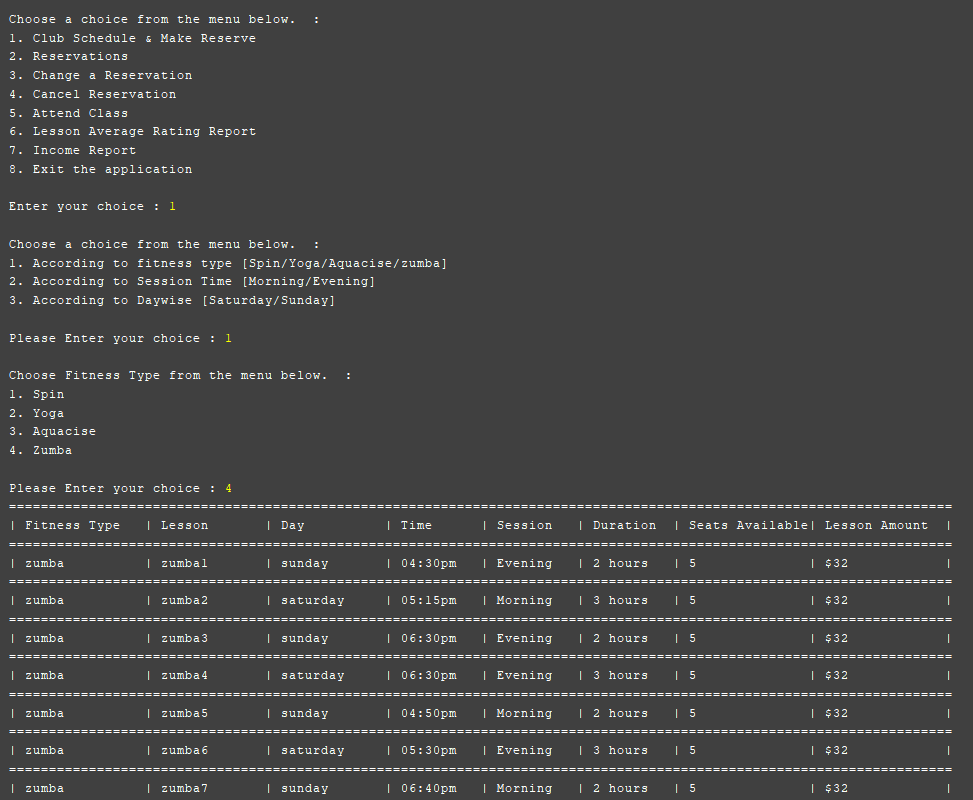


# 6.Software Artefact

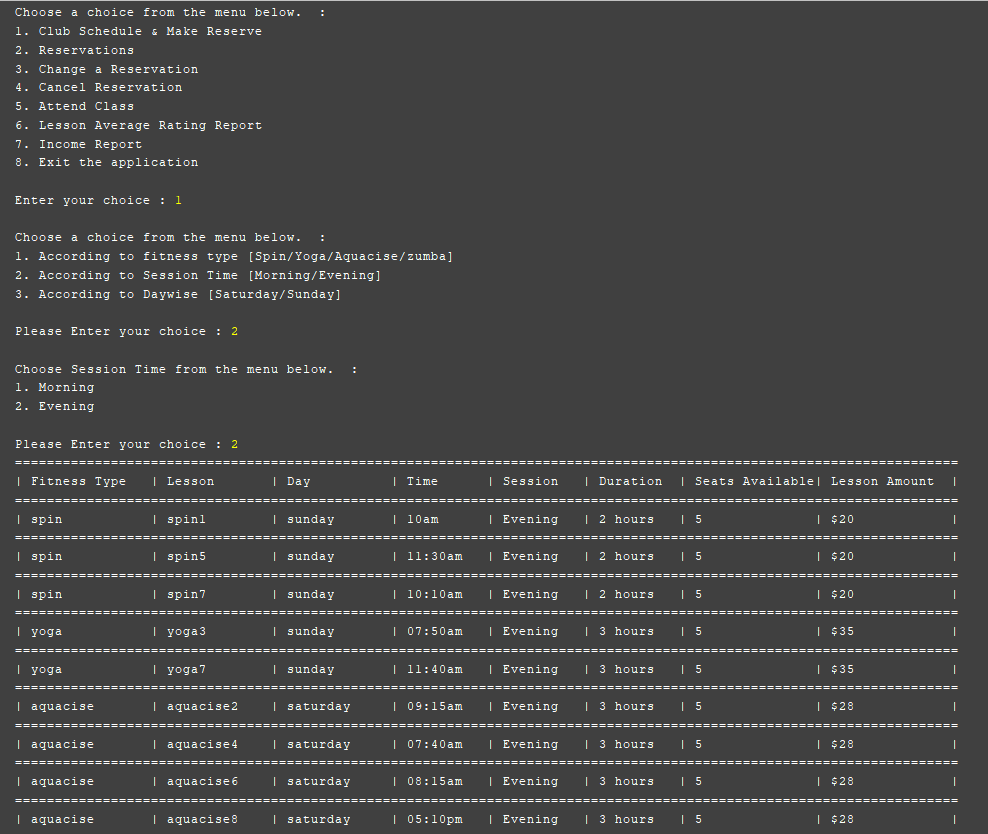
In this part, software artefact which has been designed and developed using JAVA scripting language will be displayed. The artefact is shown below as-

1. Club schedule

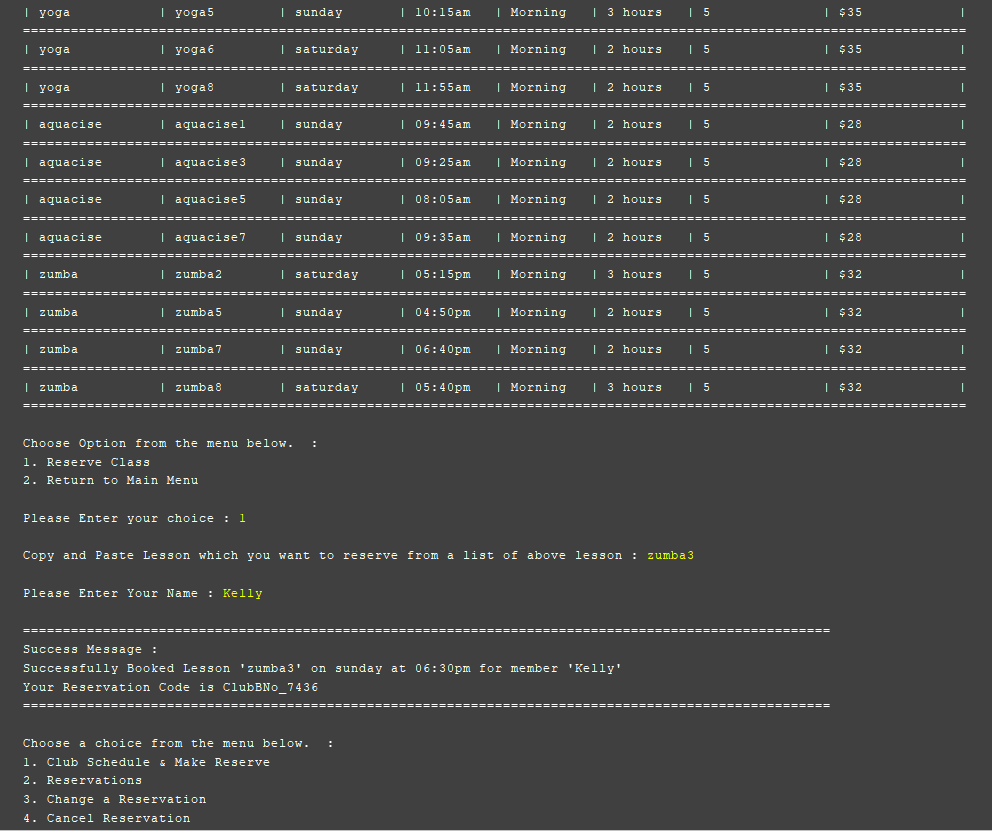
To check whether system is operating in intended manner, CMD prompt has been used in which different inputs were provided. If user wants to check club schedule according to fitness type, then user can do so by entering relevant input as shown below-



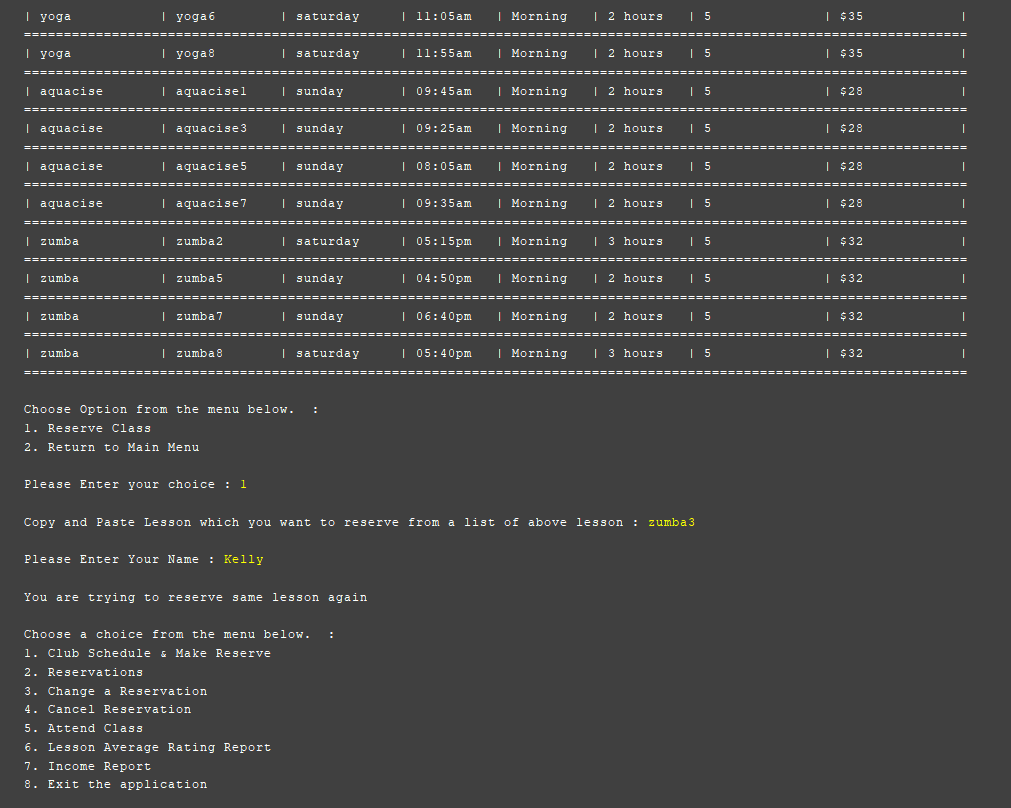
If user wants to check club schedule according to session time, then user can do so by selecting appropriate option that is displayed below-



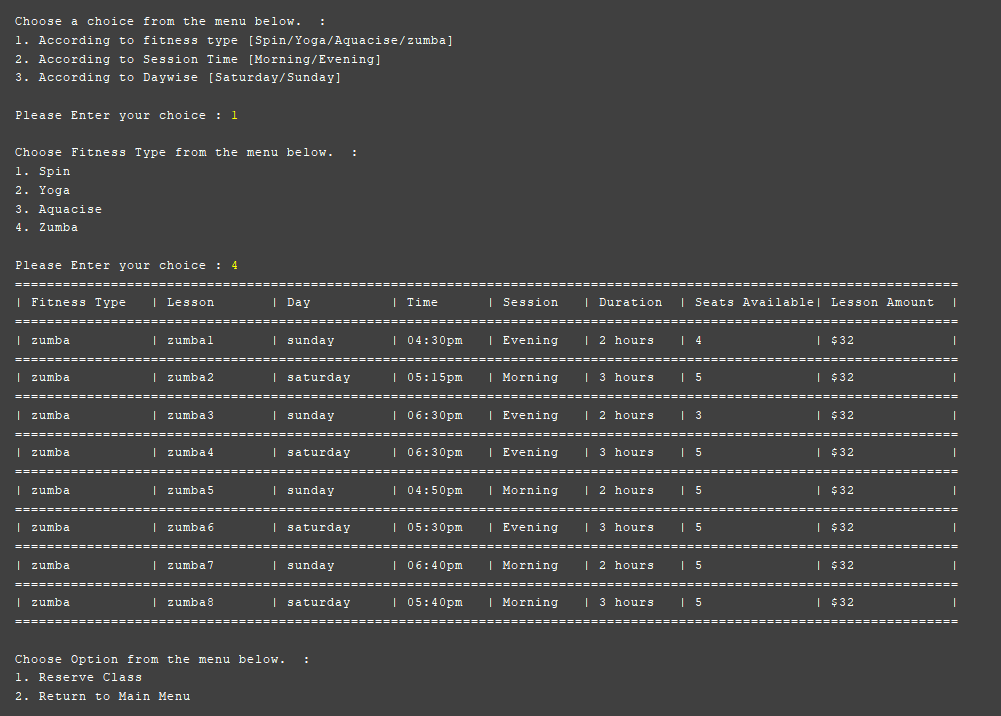
1. User can also book a class by specifying relevant option and then selecting time slot for which class is to be booked.



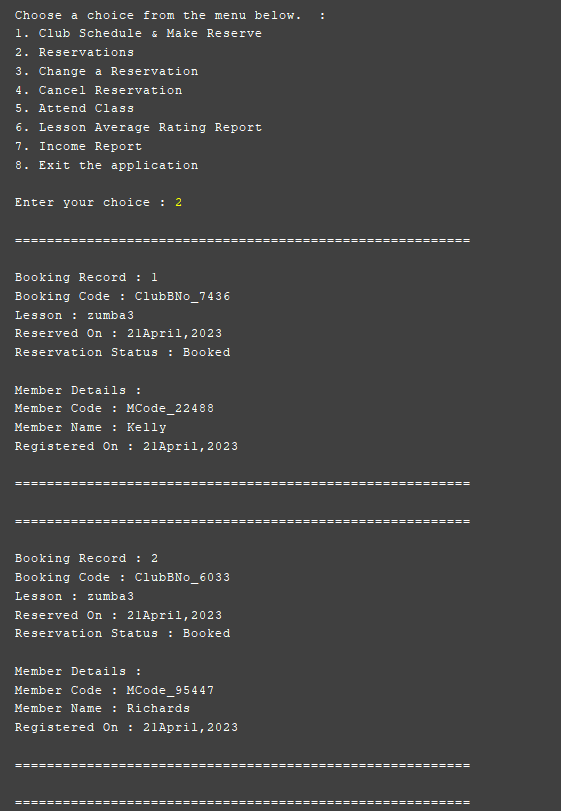
1. If user tries to book an already booked class for the second time, then system will generate an error notification, which is displayed below-

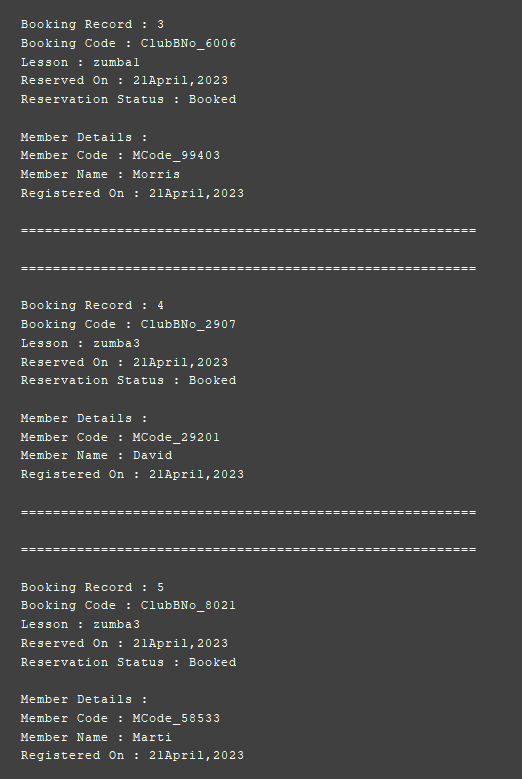


1. If user wants to check total number of seats available in the schedule, then user can do so by performing below actions on command prompt-

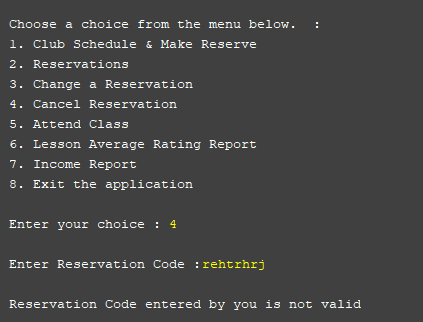


1. If admin user wants to check all the bookings made by customers on the portal, then admin user can do so by selecting appropriate option as bellows-





1. If user wants to cancel any particular lesson for booking, then user can do so by providing appropriate booking code; however, if code does not exist, then system will generate an error message-



# 7.Testing and Evaluation

The testing of different modules of the software code will be done using JUnit testing technique. Test cases are identified below-

**Test Case 1 🡪 To check whether the schedule is displayed according to selected fitness type or not**

|  |  |
| --- | --- |
| * **Test data used** | Fitness Type |
| * **Expected output** | The schedule should be displayed according to given fitness type |
| * **Actual output** | The schedule is displayed according to given fitness type |
| * **Screenshot** |  |
| * **Status** | Pass |

**Test Case 2 🡪 To check whether the schedule is displayed according to selected day or not**

|  |  |
| --- | --- |
| * **Test data used** | Day |
| * **Expected output** | The schedule should be displayed according to given day |
| * **Actual output** | The schedule is displayed according to given day |
| * **Screenshot** |  |
| * **Status** | Pass |

**Test Case 3 🡪 To book a class**

|  |  |
| --- | --- |
| * **Test data used** | Lesson, membername |
| * **Expected output** | Display success message with lesson detail |
| * **Actual output** | Displays success message with lesson detail |
| * **Screenshot** |  |
| * **Status** | Pass |

**Test Case 4 🡪 To check cancel reservation (to see whether booking is cancelled or not)**

|  |  |
| --- | --- |
| * **Test data used** | Reservation Code |
| * **Expected output** | Reservation should be cancelled with success message |
| * **Actual output** | Reservation is cancelled with success message |
| * **Screenshot** |  |
| * **Status** | Pass |

**Test Case 5 🡪 To check whether booking code inputted is valid or not (To check whether system is able to detect whether inputted booking id is valid or not)**

|  |  |
| --- | --- |
| * **Test data used** | Reservation Code |
| * **Expected output** | Display message – “valid” if it is valid and display message - “Not valid”, if it is not valid |
| * **Actual output** | Displays message – “valid” if it is valid and displays message - “Not valid”, if it is not valid |
| * **Screenshot** |  |
| * **Status** | Pass |

**Test Case 6 🡪 To check whether system allows user to book same lesson again**

|  |  |
| --- | --- |
| * **Test data used** | lesson, membercode |
| * **Expected output** | Display message “You are trying to reserve same lesson again” if the member tries to reserve same lesson twice. |
| * **Actual output** | Displays message – “You are trying to reserve same lesson again” if the member tries to reserve same lesson twice. |
| * **Screenshot** | |
| * **Status** | Pass |

# 8. References

Batool, A. *et al.* (2015) “Impact and comparison of programming constructs on Java and C# source code readability,” *International Journal of Software Engineering and Its Applications*, 9(11), pp. 79–90. Available at: https://doi.org/10.14257/ijseia.2015.9.11.07.

Mulge, M.Y. *et al.* (2018) “Impact and consideration of programming constructs on Java and C# source code readability,” *International Journal of Emerging Research in Management and Technology*, 6(11), p. 106. Available at: https://doi.org/10.23956/ijermt.v6i11.53.

Pinto, G. *et al.* (2015) “A large-scale study on the usage of Java’s concurrent programming constructs,” *Journal of Systems and Software*, 106, pp. 59–81. Available at: https://doi.org/10.1016/j.jss.2015.04.064.