**Booking System**

# Introduction

The Hatfield Junior Swimming School offers swimming lessons to children between the ages 4-11 wherein learners are segmented into 6 groups (i.e. grade 0 - 5). The swimming school utilizes a spreadsheet and manual accounting software for organizing and storing lesson booking which is time-consuming and prone to errors. Thus the swimming school intends to develop a new system for managing lesson bookings, using java script the company aims to develop a new booking management system whose development, development and testing procedure is described in this report.

# Assumptions

1. Instructor data: The instructors conducting the swimming lessons will have to register over the GUI software to store instructor data within the CSV files.
2. Store user and platform data over CSV files: All the user and platform data collected from students and instructors will be stored within a CSV document.

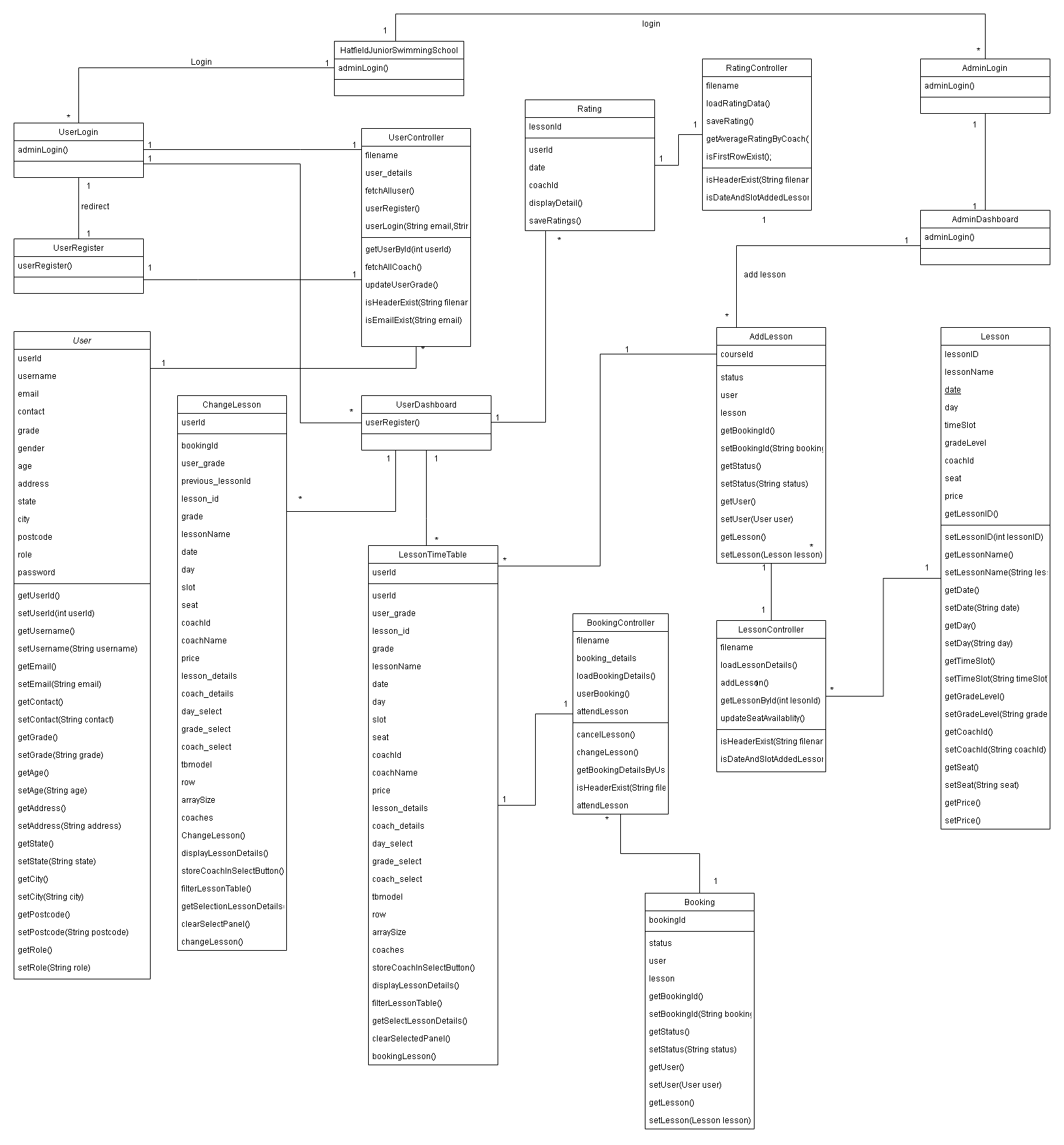
# Structure

The software utilizes a class structure to call functions, additionally, the code utilizes an inheritance structure which creates a hierarchy between classes by inheriting from other classes. This structure enhances the efficiency of the code while reducing the complexity of understanding the code by removing code duplications. A brief description of the different classes, associated attributes and methods is presented below

| Classes | Methods |
| --- | --- |
| User | UserRegister() |
| UserDeregister() |
| UserLogin() |
| Lesson Time table | ViewLessons() |
| ViewBookedLessons() |
| CreateLessons() |
| Booking Lesson | BookLessons() |
| CancelBooking() |
| Rating or Review | AddReview() |
| ViewReview() |
| CalculateAvergaeRating() |
| Admin Login | AdminLogin() |

# Class diagram

The class diagram helps visualize a static view of the application in which the major classes, their attributes and the associated methods are described. Furthermore, the relationship between these classes is also represented in this diagram.



# Design Pattern

Design patterns serve as a blueprint or template for developers to solve recurring design problems within the product source code (Koffman and Wolfgang, 2021). The GUI software of Hatfield Junior Swimming School utilizes the MVC (Model View Controller) design pattern wherein it separates the software blueprint into three components (Controller, View, Model), This feature helped the developers team to manage and maintain the code base which enabled the reusability of components. Furthermore, the MVC design pattern provided developers with get and set methods, to create user lessons and fetch booking details. Additionally, user inputs such as User name, Passwords, etc. were fetched using the get method to store the user data within a CSV file.

# Junit Testing

**Test 01:** View Booked Lessons

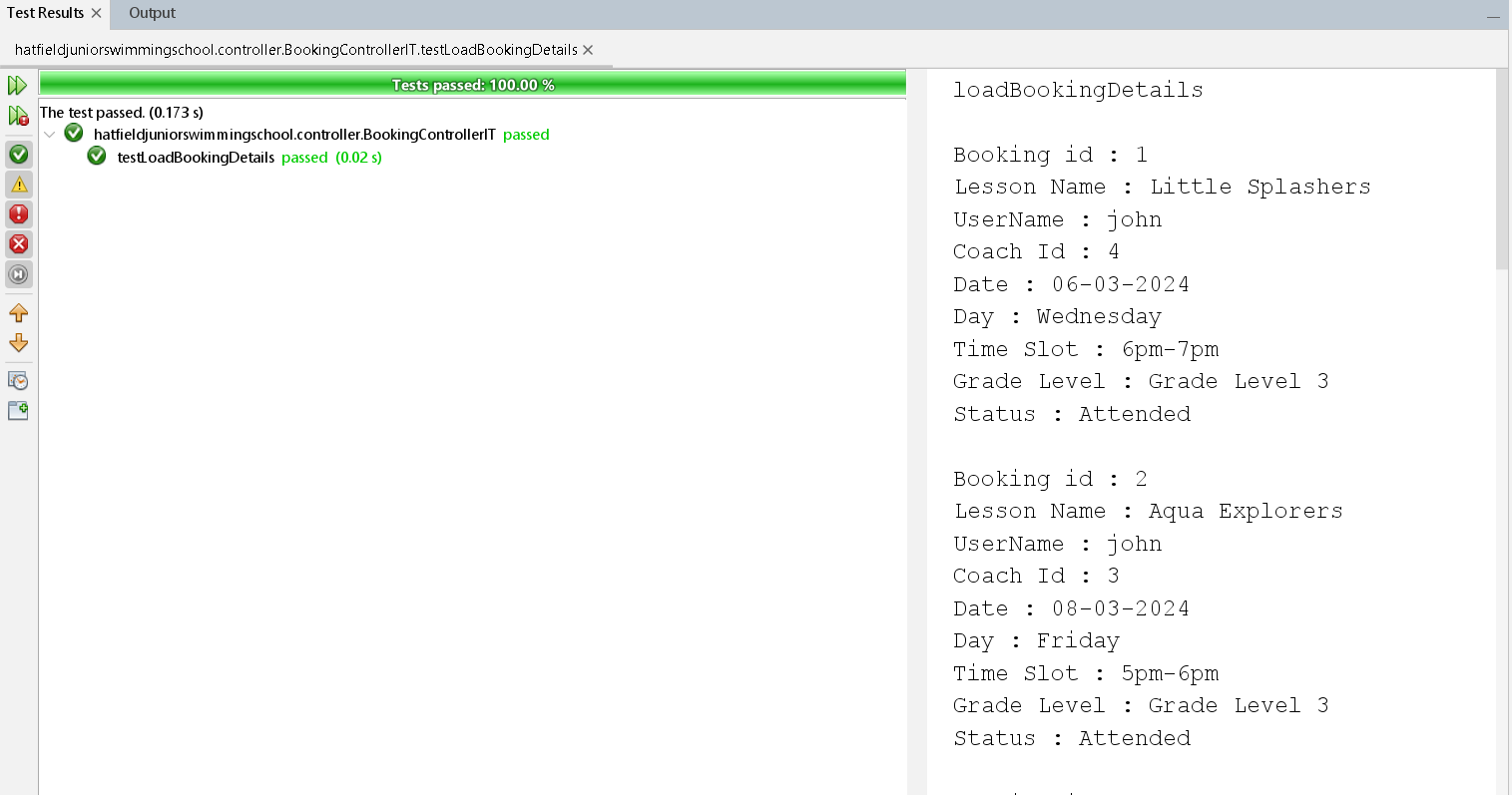
Input: Booking date or user ID or Coach Name

Expected Output: List the learners who booked swimming lessons.

Actual Output: List the learner's booked swimming lessons.

****

**Output :**

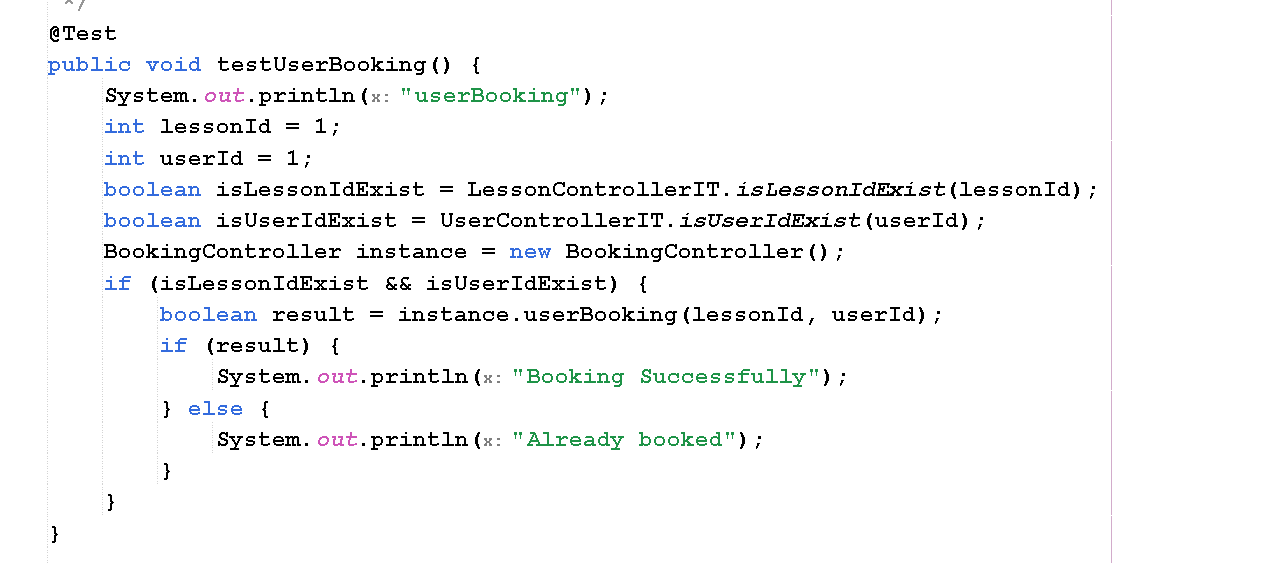
****

**Test 02:** Book Swimming lessons over the software

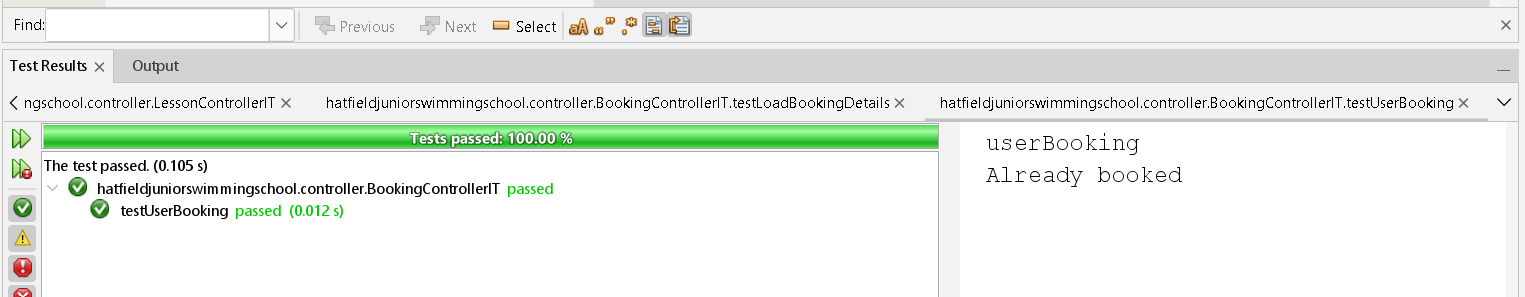
Input: The learner's data and the day for which the learner is booking

Expected Output: Reflect the successful booking for the learner in the view booking tab

Actual Output: The View booking tab reflects upon the successful bookings performed by the learner.

****

**Output:**

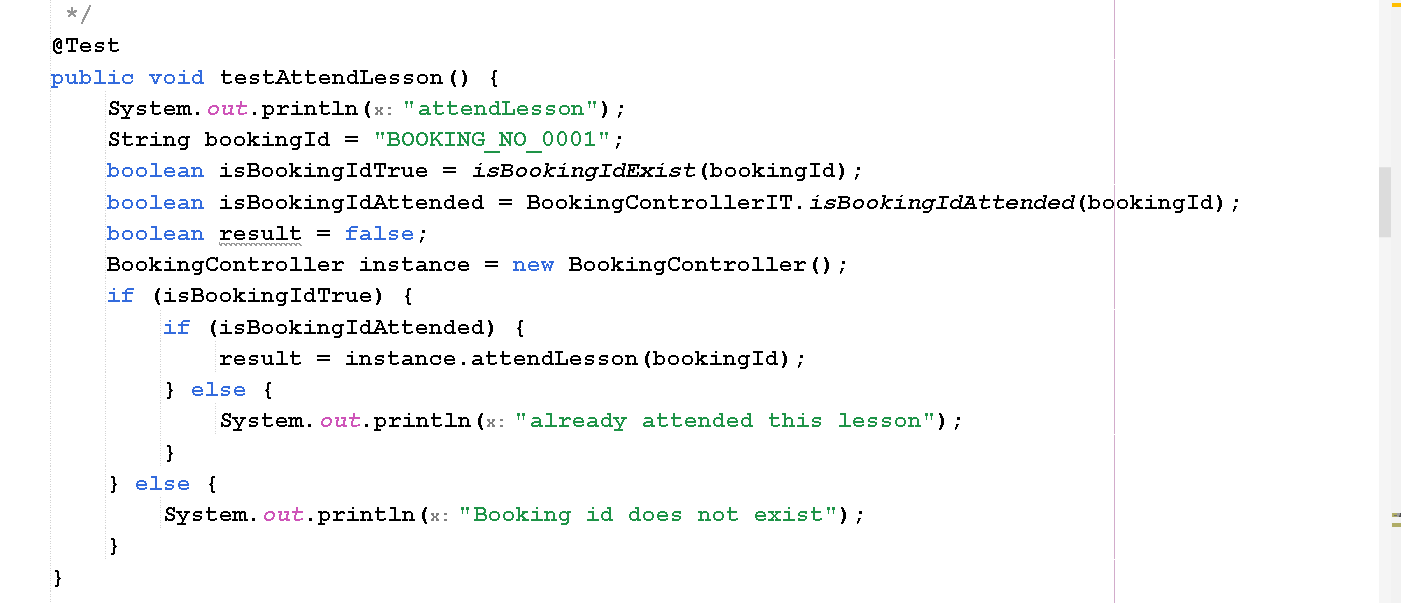
****

**Test 03:** Reflect on the lesson attendance over the software

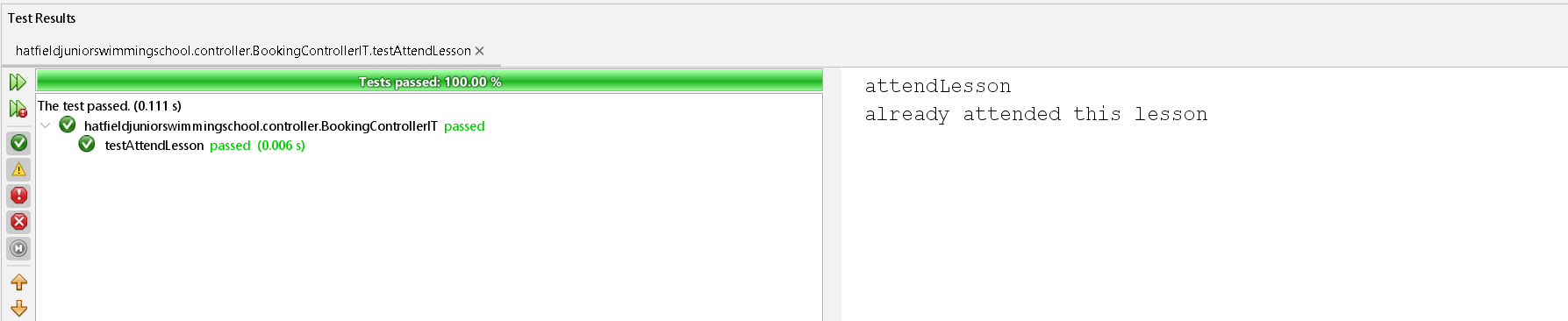
Input: Booking date and learner’s id

Expected Output: Reflect “present” over the lessons attended by the learner.

Actual Output: Reflects “present” over the lessons attended by the learner.

****

**Output :**

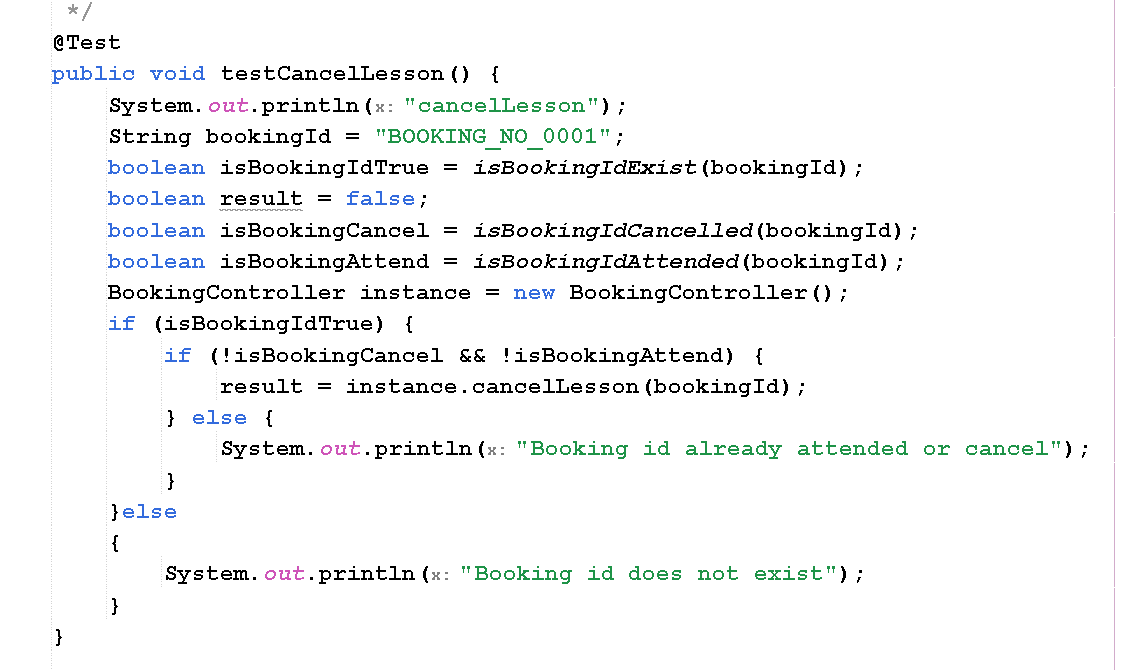
****

**Test o4:** Cancel lessons/ booking over the software

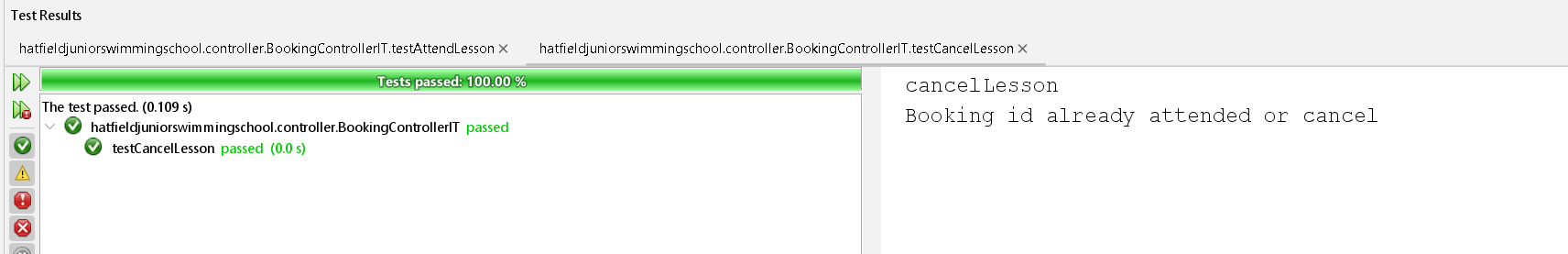
Input: Booking details and user details.

Expected Output: Reflect the cancellation over the view bookings tab.

Actual Output: Changes the booking status to cancel.

****

**Output :**

****

**Test 05:** Update lessons over the bookings

Input: lesson\_ID and booking\_ID

Expected Output: Update the lesson details of the bookings

Actual Output: Updates the lesson details of these bookings

# References :

Koffman, E.B. and Wolfgang, P.A.T. (2021). Data Structures: Abstraction and Design Using Java. [online] Google Books. John Wiley & Sons. Available at: https://books.google.co.in/books?hl=en&lr=&id=8nwnEAAAQBAJ&oi=fnd&pg=PA1&dq=java+design+pattern+&ots=alTu13st2m&sig=nUOwBBg39tJC7TY\_NUOR5Ysqa7w&redir\_esc=y#v=onepage&q=java%20design%20pattern&f=false [Accessed 20 Mar. 2024].