

**College of Engineering**

**Programming 2(CSC202)**

**Project Report**

Spring 2024-2025 | Section 77 AA

Submitted by: **Group 8**

|  |  |  |
| --- | --- | --- |
| No. | Student Name | Student ID |
| 1 | Malak Salah | 1098490 |

1. **Thought process:**

It started by deciding between doing windows or normal straightforward output. To integrate GUI into the project and since interacting with windows provides a better user experience, I chose the former.

The Facilities, Staff, and Student classes were made first because their content is clear and because they act as a foundation to the BookingSystem class (the core class).

I decided to have the main method that runs the program in the BookingSystem class because it is the core of the program. Writing the BookingSystem class began by writing code for log in and registration. Log in is one method, and registration is another method. They are separated so they open different windows and maintain correct logic.

The User class was made after that to declare and implement the verifyLogin method invoked in the Login method in the BookinSystem class.

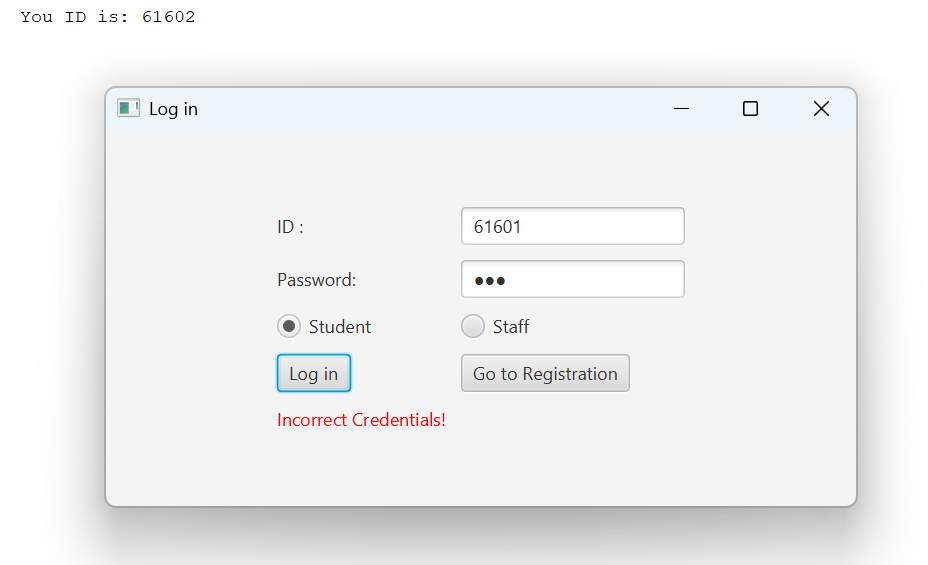


Fig 1.1 Log in does not proceed if credentials entered are incorrect as seen above.

I decided to have fixed Facility objects as it makes the system simpler and easier to work on than allowing adding and removing them. Six Facility objects were created as data fields of the BookingSystem class. This way, they can be used by all the methods in the class.

First window to open after log in is the FacilitiesAndSearch window which displays the six facilities and has a search field and search button for filtration. The search part of the window creates a FacilitySearchEngine object from the FacilitySeacrhEngine. The FacilitySearchEngine class implements the methods defined in the interface it is implementing (the IFacilitySearch interface). Those methods are responsible for returning an ArrayList of the facilities that match user search. If the user chooses to search by type, the method that searches by type is invoked for example. I had the radio buttons combined into a toggle group to know which radiobox is selected and conduct the search based on that as shown is figure 1.2 below.

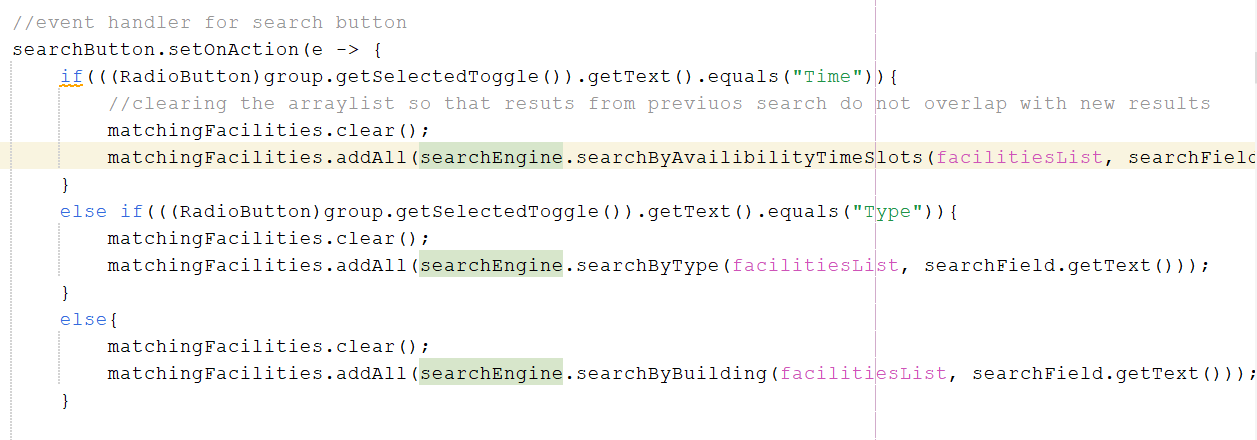


Fig 1.2 code that checks which radio button is selected and conducts the search based on that

After the ‘Search’ button is hit, another window appears and displays the results of the search. If the logged in user is a student, the Student Dashboard window is opened through calling the StudentDashboard method, otherwise, the StaffDashboard method is invoked. The facilities that appear in the search result will have checkboxes next to them.

In the Student Dashboard, the checkbox of some facilities might not be clickable if the facility’s occupancy limit is exceeded, or if the facility is a premium one and the student logged in is not eligible. Eligibility is checked through a method defined in the User class, which is of course implemented in the Student and the Staff classes. For students, only engineering students are eligible for premium facilities. This is where the FacilityLimitExceededException and the AccessDeniedException2 might be thrown. More clearly, if one of these exceptions are thrown after clicking a checkbox, it means that the facility associated with the checkbox cannot be booked for either of the two reasons, and the checkbox is automatically deselected. If a checkbox is selected successfully, however, timeslot radio buttons appear for the facility – and when unchecked the timeslot radio buttons for the facility are removed. If the selected time slot is not available for the facility, a TimeSlotUnavailableException is thrown, otherwise, a ‘Book’ button appears. When clicked, the Payment and Booking Processing window is opened through calling the paymentWindow method. There, the student should transfer the correct amount of money for the booking to be registered. When a booking is registered, the facility’s currentOccupancy property is incremented, and the facility is added to the bookedServices array list of the Student object, and to the students.txt file.

A screenshot of a computer

AI-generated content may be incorrect.

Fig 1.3 message appears on console if unavailable time slot for facility is selected.

In the Staff Dashboard, all checkboxes are always clickable because staff do not add to the occupancy of the facilities (they don’t book them, they manage them) and they are all eligible for premium services. The same logic of selecting and deselecting the checkboxes applies for the Staff Dashboard, as well as the one related to the TimeSlotUnavailableException, except a ‘Manage’ button appears instead of a ‘Book’ button if the exception is not thrown. When the ‘Manage’ button is clicked, the facility is added to the managedFacilities array list of the Staff object, and to the staff.txt file.

From the Log in window that appeared in the beginning, the ‘Go to Registration’ button opens the Registration window and registers new Users within the same run of the program.

**More Buttons:**

In the Student Dashboard window, there is a ‘Check my booked services’ button that invokes a method that opens new small-sized window that displays the services books by the logged in student.

A screenshot of a computer

AI-generated content may be incorrect.

Fig 1.4 the Your Booked Services window display example

In the Staff Dashboard, there are three button as shown below.

A screenshot of a computer

AI-generated content may be incorrect.

Fig 1.5 notice the three buttons at the bottom of the window

Check Fig 1.5. The first button on the left displays the facilities managed by the staff member. The one beside it, reads from the students.txt file, and the last one reads from the staff.txt file. I decided to keep the reading of files possible to staff only, which is why the last two buttons are only in the Staff Dashboard not the Student Dashboard.

1. **Challenges Faced**

One of the challenges faced was when a new log in happened in the same run, the new Facilities and Search window duplicates the facilities list and shows 2 or more of them. Then I figured that the array list facilitiesListed adds the facilities everytime the Facilies and Search window opens, so I used an if statement with the condition if facilitiesList.isEmpty(), and had the adding of facilities within it. This way, only the first time the window opens the facilities are added to the array list of facilities used throughout the method.

Another challenge was having the three buttons at the bottom of the Staff Dashboard be positioned at the bottom as shown in Fig 1.5. The window uses a GridPane, and the content of the window is dynamic, it could change, so there is no fixed number of rows. To fix this, I put the three buttons inside an HBox as I wanted them to be aligned horizontally, then created a BorderPane object and had the GridPane set in the center of it, and the HBox set at the bottom of it. Then the BorderPane was passed to the Scene.

1. **References**

*W3schools.com*. W3Schools Online Web Tutorials. (n.d.-a). https://www.w3schools.com/java/default.asp

Comment, info, M., kartik Follow, kartik Follow, Kartik, & Follow. (2025, May 19). *Java tutorial - learn Java Programming*. GeeksforGeeks. https://www.geeksforgeeks.org/java/