INDIVIDUAL ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

AAPP004-4-2-JP

JAVA PROGRAMMING

UCDF2005(1) ICT(DI)

RESORT ROOM BOOKING SYSTEM DOCUMENTATION

Name: Chan Ming Li

TP Number: TP060774

Hand out Date: 12 FEBUARY 2022

Hand in Date: 10 APRIL 2022

Weightage: 60%

Lecturer Name: Dr. Kadhar Batcha Nowshath

Table of Contents

1.0 Introduction

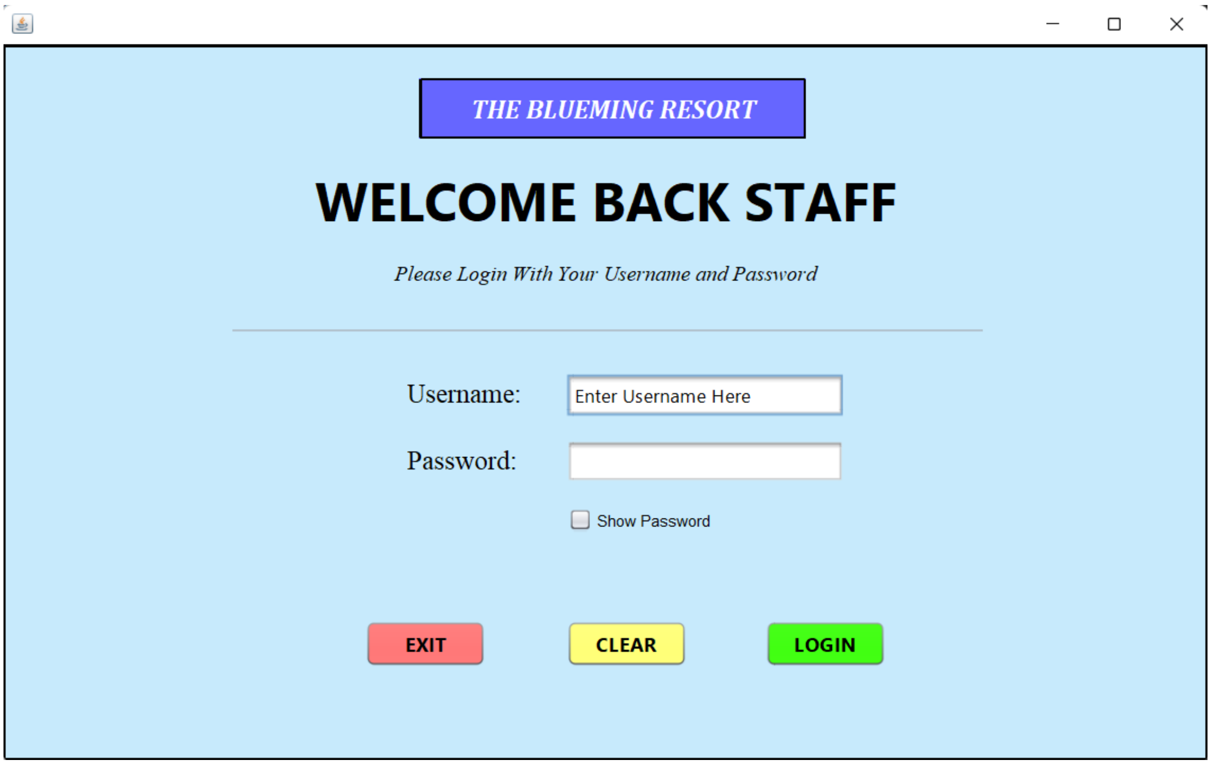
The Blueming Resort is providing rooms with different views which is Jungle and Sea view for booking. The Blueming Resort has decided to utilize an online room booking system for their staffs in order to enhance their performance in room bookings. As the advancement of technology, online room booking system helps staffs to avoid the laborious and time-consuming room booking procedures as it supports automatic saving booking and customer information, view available rooms and more functions. In order to improve customer satisfaction and experiences, a new online room booking system is designed and developed by Chan Ming Li by implementing the object-oriented programming concepts in Java. The object-oriented programming concepts are implemented as they allow us to maintain the codes easier, avoid redundant code blocks, enhance the security measures of data and more. Therefore, the newly developed room booking system will boost the growth of the resort by speeding up the room booking process in order to provide customers the best room booking experiences.

1.1 Assumptions

During the planning phase of The Blueming Resort Room Booking System, there are a few assumptions are created by the developer in order to clarify the functionalities of the system and develop the program efficiently. Firstly, the staffs who are using the system will have different username and password which is stored in a text file as a staff credentials database. Staffs are not able to register or change their password in the system as their account are oversee or prepared by the admins before they have the access to the system. Besides, the customer could only book at most 7 days as the resort have a high booking demand. Moreover, the customer could only book the rooms within 7 days from the current date in order to ensure fairness between customers and prevent too-early bookings from customers. It would be more practical and comfortable for the room booking system of The Blueming Resort by following the assumptions.

2.0 Functionalities / JFrame of The Room Booking System

2.1 Login Page



THE BLUEMING RESORT

WELCOME BACK STAFF

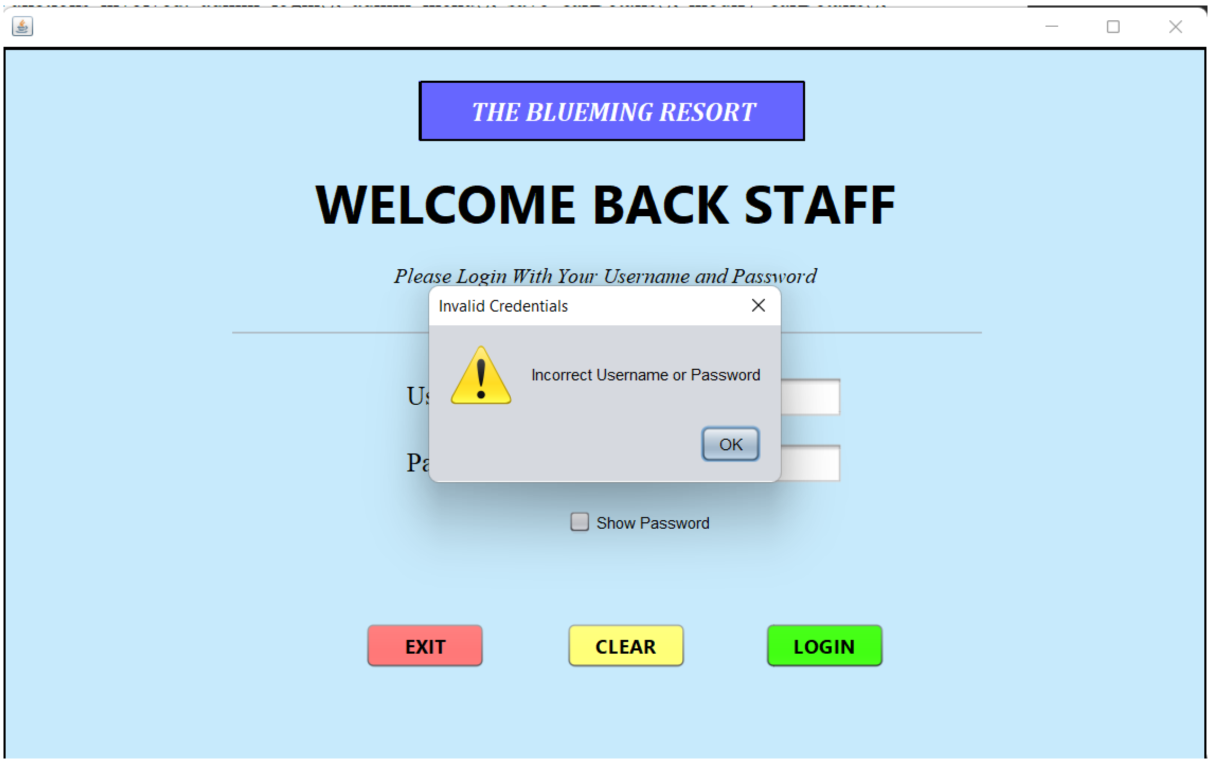
Please Login With Your Username and Password

Username: Enter Username Here

Password:

Show Password

Figure 2.1(a): Login Page Output



THE BLUEMING RESORT

WELCOME BACK STAFF

Please Login With Your Username and Password

Invalid Credentials

Incorrect Username or Password

Figure 2.1(b): Warning Message When Incorrect Credentials is Entered

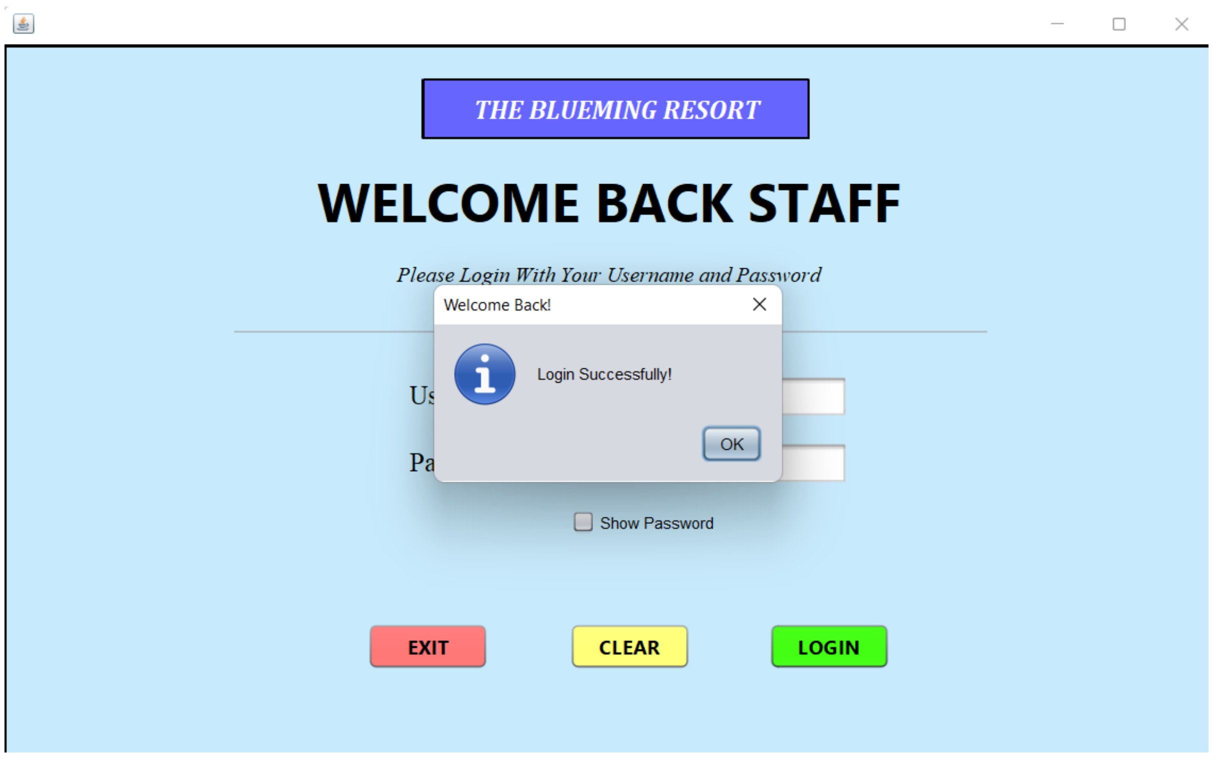


Figure 2.1(c): Welcome Message When Correct Credentials are Entered

The login page of the room booking system as illustrated in *Figure 2.1(a)* will be displayed when the system is launched. The staffs are required to enter their username and password correctly to login into the system successfully. If wrong credentials are detected by the system, a warning message will pop out as shown in *Figure 2.1(b)* to indicate that wrong credentials are entered and request the staff to enter his/her credentials again. If the staff entered the correct credentials, the system would pop out a login successfully message as illustrated in Figure 2.1(c) and navigate him/her to the staff home page for further features. Besides, there is a “clear” function provided to staffs in order clear all the entered text to save time if the staff wanted to re-enter his/her credentials. Additionally, an “exit” button is provided for staffs to exit the system.

2.2 Staff Home Page

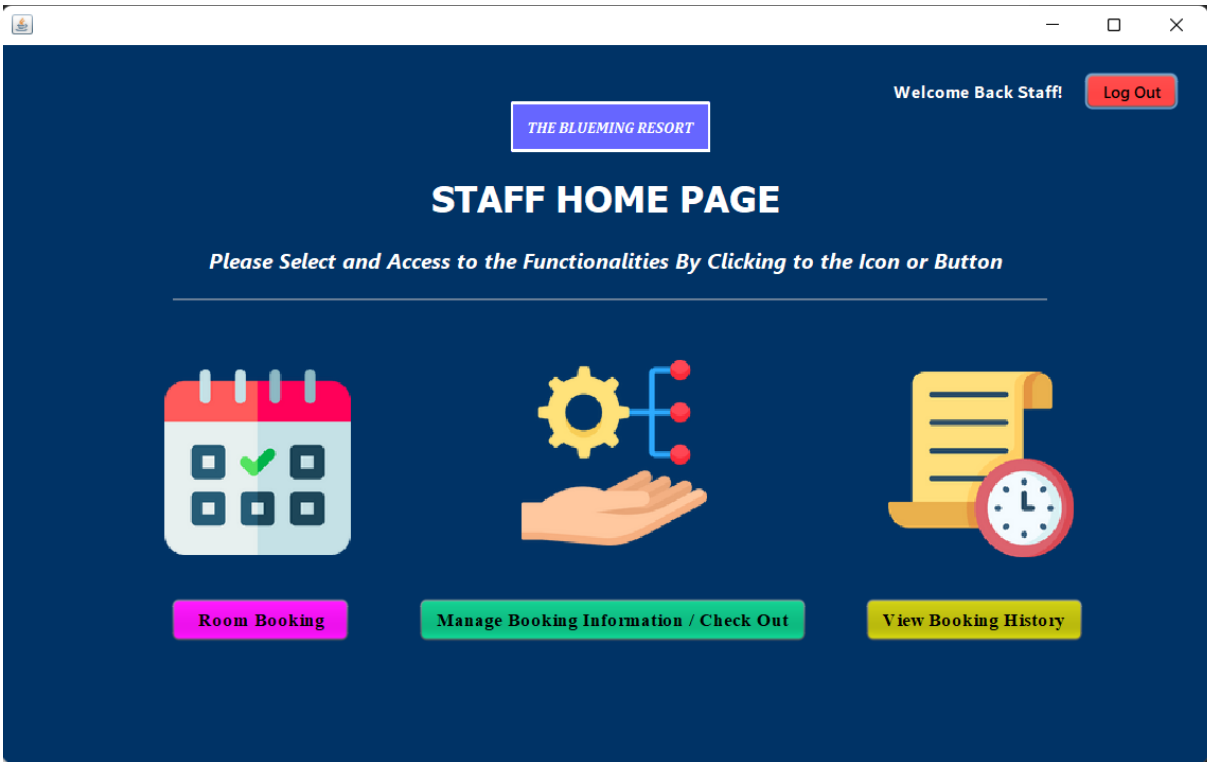
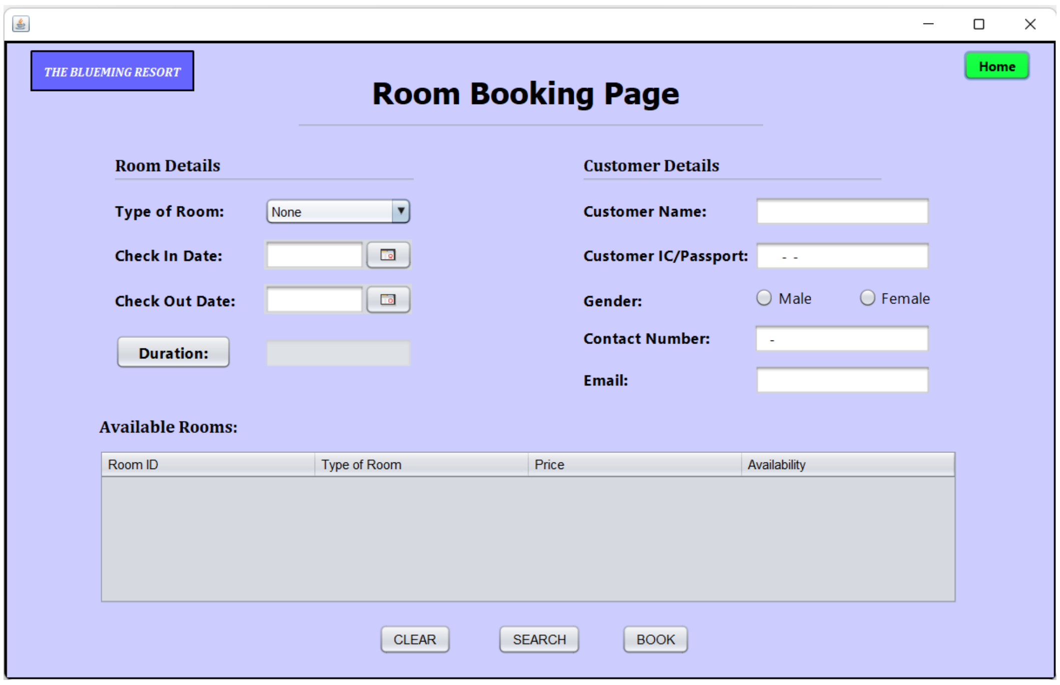

Welcome Back Staff!
Log Out
THE BLUEMING RESORT
STAFF HOME PAGE
Please Select and Access to the Functionalities By Clicking to the Icon or Button
Room Booking
Manage Booking Information / Check Out
View Booking History

Figure 2.2(a): Staff Home Page Output

After login successfully from the login page, staffs will be navigated to staff home page as shown in Figure 2.2(a) to access to different functionalities provided in the system. There are three main functions provided in this page which is room booking, manage booking information or check out and view booking history. Besides, there is a log out button located in the top right corner of the page which allows staffs to log out from their account and navigate back to the login page.

2.3 Room Booking Page


THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room:	None
Check In Date:	
Check Out Date:	
Duration:	

Customer Details

Customer Name:	
Customer IC/Passport:	- -
Gender:	○ Male ○ Female
Contact Number:	-
Email:	

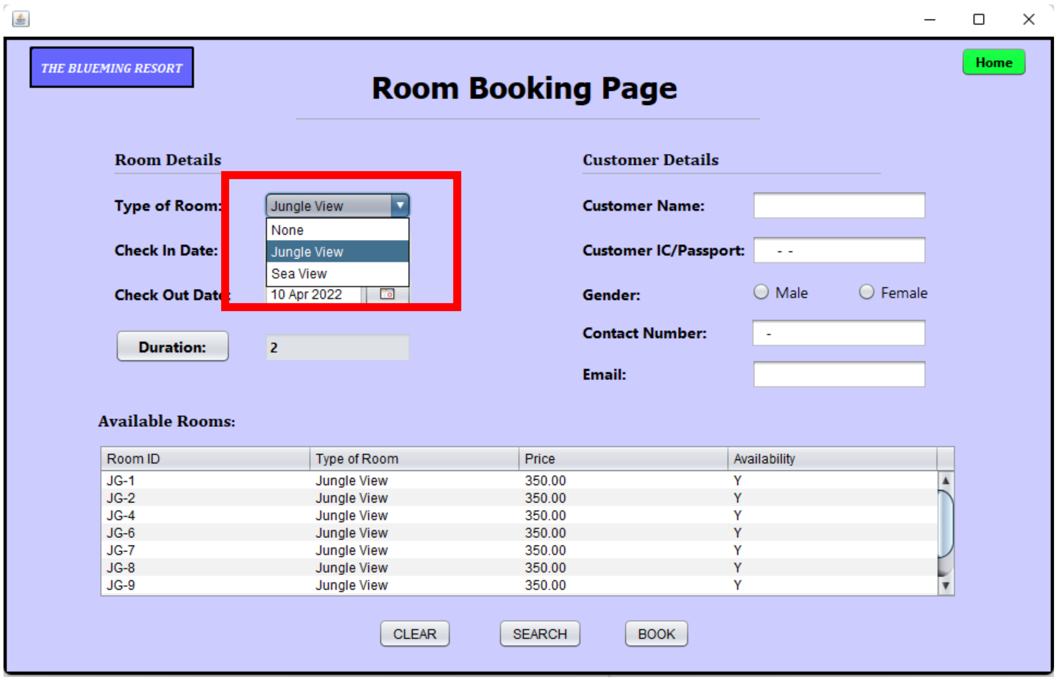
Available Rooms:

Room ID	Type of Room	Price	Availability

CLEAR SEARCH BOOK

Figure 2.3(a): Login Page Output

The room booking page of the resort consists of three main sections which is room details, customer details and available rooms as illustrated *Figure 2.3(a)*. The main purpose of the room booking page is to staffs to enter different details and provide information of the customers in order to book the room successfully in the system. The JTable located at the bottom of the page is utilized to show room details of available rooms such as room id, room type, price per day and availability.



Room Booking Page

Room Details

Type of Room:	Jungle View (None / Jungle View / Sea View)
Check In Date:	
Check Out Date:	10 Apr 2022
Duration:	2

Customer Details

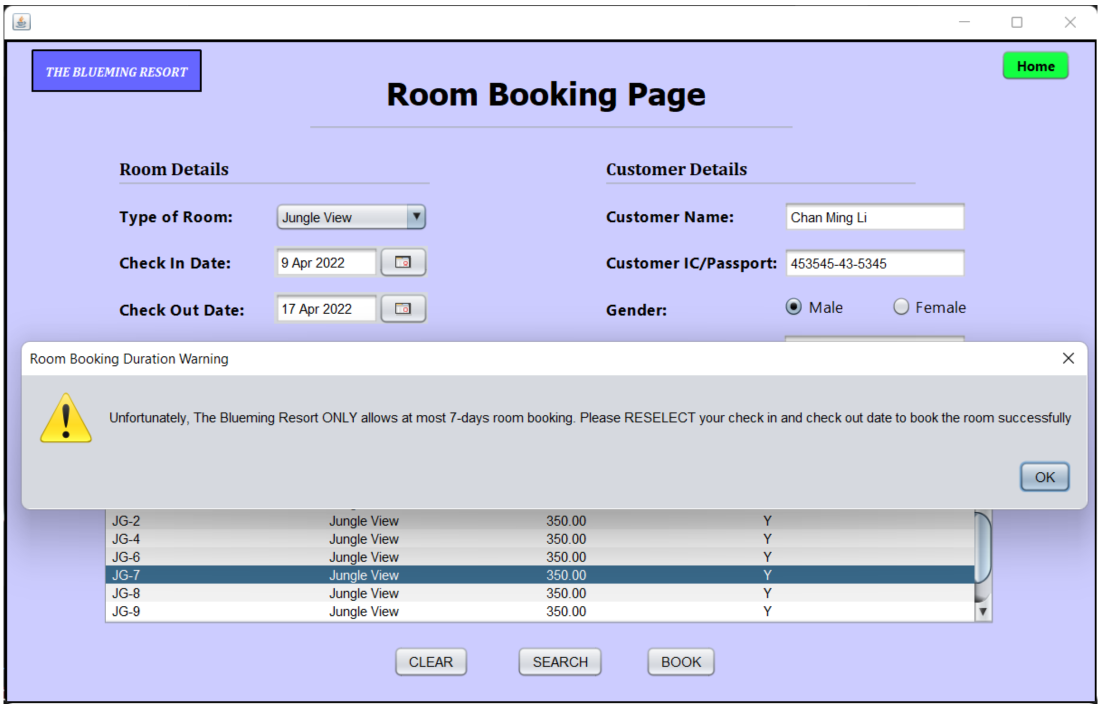
Customer Name:	
Customer IC/Passport:	- -
Gender:	Male / Female
Contact Number:	-
Email:	

Available Rooms:

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y
JG-9	Jungle View	350.00	Y

Figure 2.3(b): Search for Available Rooms Function

In order to display the available rooms, the staff is required to select either jungle view or sea view as shown in *Figure 2.3(b)*. The table below will show all the available rooms for the specific type of room after staff has chosen the room type from the combo box.



Room Booking Page

Room Details

Type of Room:	Jungle View
Check In Date:	9 Apr 2022
Check Out Date:	17 Apr 2022

Customer Details

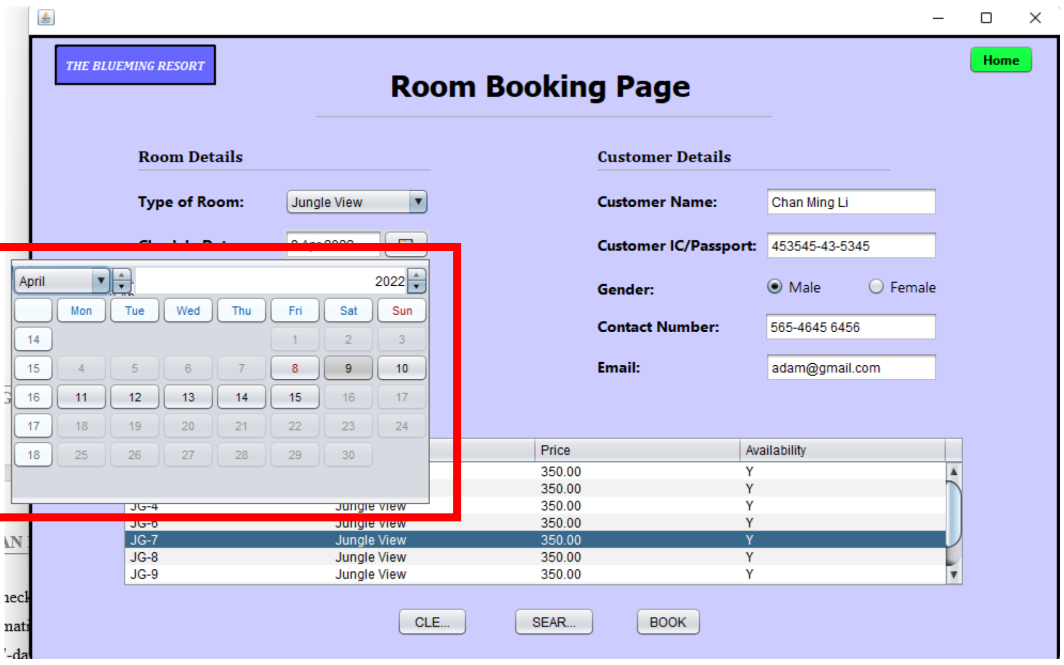
Customer Name:	Chan Ming Li
Customer IC/Passport:	453545-43-5345
Gender:	[✓ Male] Female

Room Booking Duration Warning

Unfortunately, The Bluemimg Resort ONLY allows at most 7-days room booking. Please RESELECT your check in and check out date to book the room successfully

JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y
JG-9	Jungle View	350.00	Y

Figure 2.3(c): Room Booking Duration Warning



THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room: Jungle View

Customer Details

Customer Name: Chan Ming Li

Customer IC/Passport: 453545-43-5345

Gender: (● Male) ○ Female

Contact Number: 565-4645 6456

Email: adam@gmail.com

April 2022

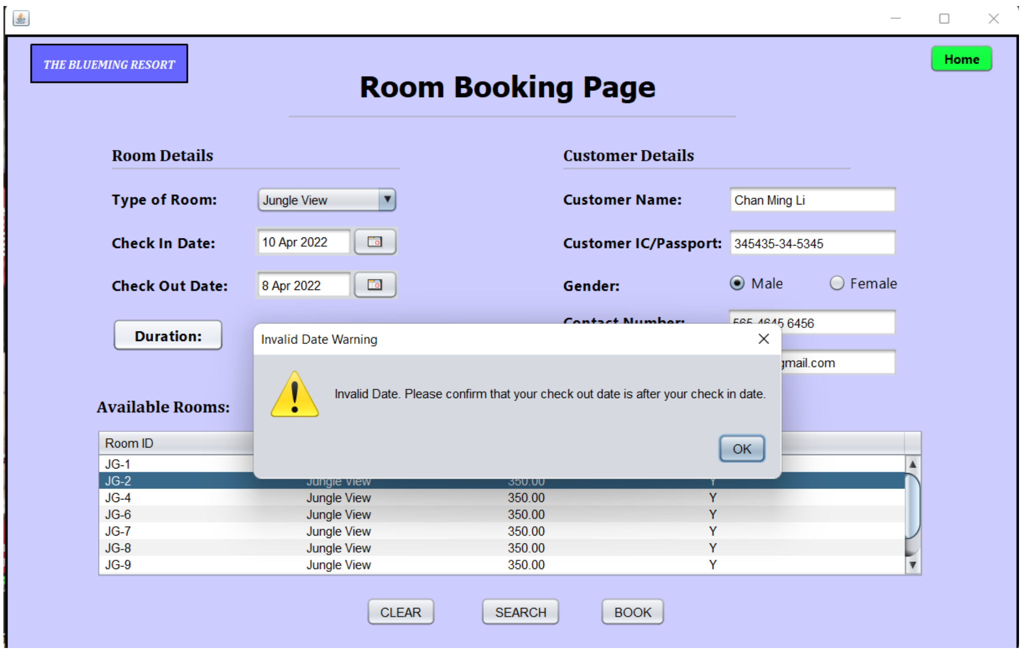
	Mon	Tue	Wed	Thu	Fri	Sat	Sun
14					1	2	3
15	4	5	6	7	8	9	10
16	11	12	13	14	15	16	17
17	18	19	20	21	22	23	24
18	25	26	27	28	29	30	

Room ID	Type	Price	Availability
		350.00	Y
		350.00	Y
JG-4	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y
JG-9	Jungle View	350.00	Y

CLE... SEAR... BOOK

Figure 2.3(d): Limit Check in Date Within 7 Days in JCalendar Chooser

When the staff is selecting the check in and check out date, the system will calculate the duration between both dates automatically. As the resort has a high demand, it only allows customers to book rooms at most 7-days as shown in *Figure 2.3(c)* in order to ensure fairness between customers and avoid long-term booking duration. Besides, the customer could only book the rooms within one week from the current date to avoid too-early bookings from the customers. Therefore, the room booking system has developed to allow staffs to only select check in dates within a week from the current date as illustrated in *Figure 2.3(d)*.



THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room: Jungle View

Check In Date: 10 Apr 2022

Check Out Date: 8 Apr 2022

Duration:

Available Rooms:

Customer Details

Customer Name: Chan Ming Li

Customer IC/Passport: 345435-34-5345

Gender: (● Male) ○ Female

Contact Number: 565-4645 6456

Email: ...mail.com

Invalid Date Warning

Invalid Date. Please confirm that your check out date is after your check in date.

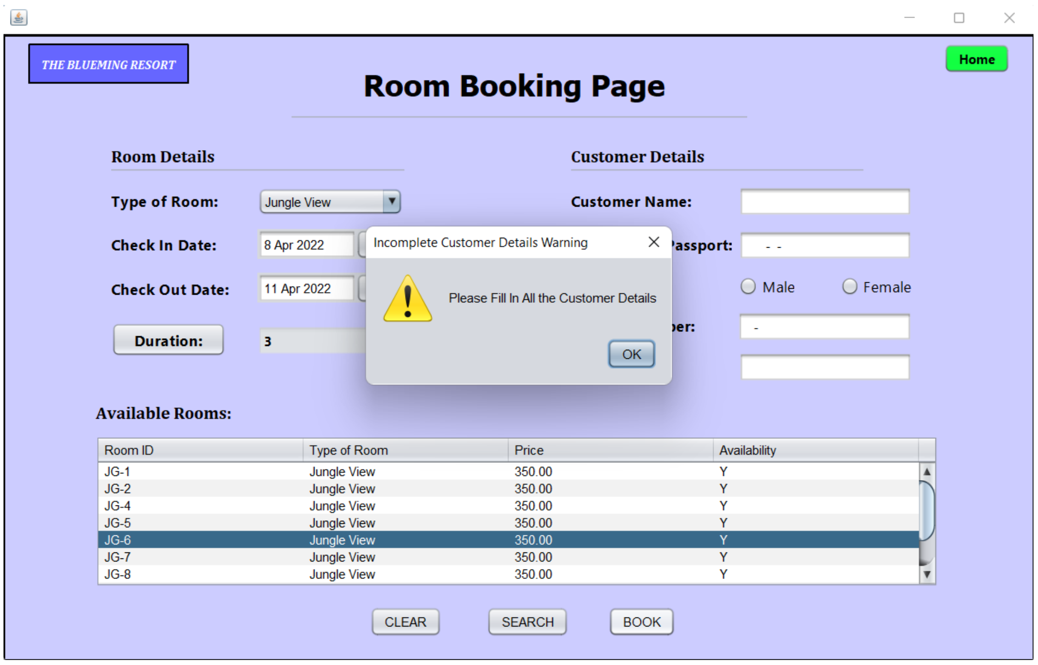
OK

Room ID	Type	Price	Availability
JG-1			
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y
JG-9	Jungle View	350.00	Y

CLEAR SEARCH BOOK

Figure 2.3(e): Invalid Date Warning Message

Based on Figure 2.3(e), there will be an invalid date warning message if the checkout date selected is earlier than check in date selected which causes the invalidity of date and duration.


THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room:	Jungle View
Check In Date:	8 Apr 2022
Check Out Date:	11 Apr 2022
Duration:	3

Customer Details

Customer Name:

Passport: - -

Male Female

-

Incomplete Customer Details Warning

Please Fill In All the Customer Details

OK

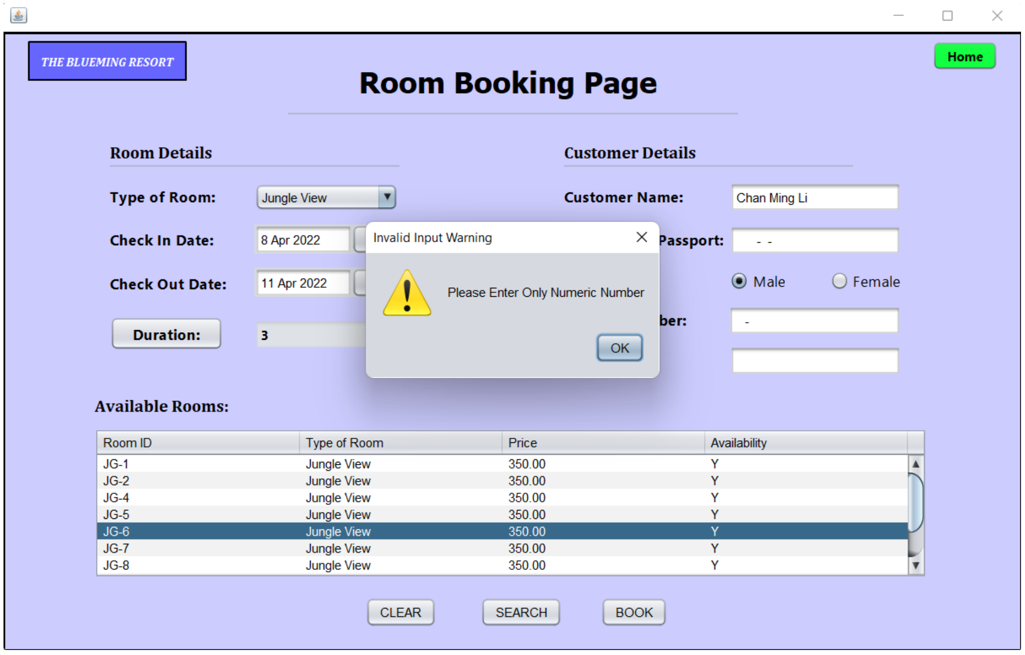
Available Rooms:

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-5	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y

CLEAR SEARCH BOOK

Figure 2.3(f): Missing Customer Details Warning Message

Customer details such as name, IC, gender, contact number and email are required to book the room successfully in the system. If there are any missing customer details, an incomplete customer details warning message will pop out as shown in *Figure 2.3(f)* so that the staff is aware that there are missing customer details.


THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room:	Jungle View
Check In Date:	8 Apr 2022
Check Out Date:	11 Apr 2022
Duration:	3

Customer Details

Customer Name: Chan Ming Li

Passport: - -

[✓ Male] Female

-

Invalid Input Warning

Please Enter Only Numeric Number

OK

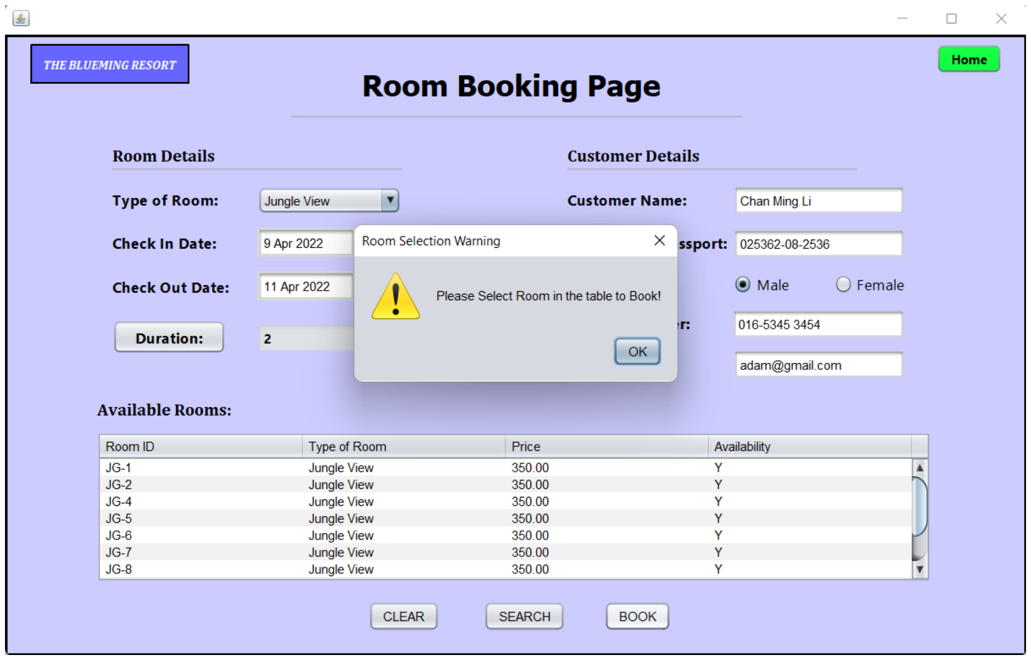
Available Rooms:

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-5	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y

CLEAR SEARCH BOOK

Figure 2.3(g): Invalid Input in Customer IC & Contact Number Field

Furthermore, there are some input validations pre-programmed in the customer IC and contact number input field in order to avoid invalid inputs. As shown in *Figure 2.3(g)*, staffs are only able to enter numeric values for customer IC and contact number input field.



THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room: Jungle View

Check In Date: 9 Apr 2022

Check Out Date: 11 Apr 2022

Duration: 2

Customer Details

Customer Name: Chan Ming Li

ssport: 025362-08-2536

● Male ○ Female

r: 016-5345 3454

adam@gmail.com

Room Selection Warning

Please Select Room in the table to Book!

OK

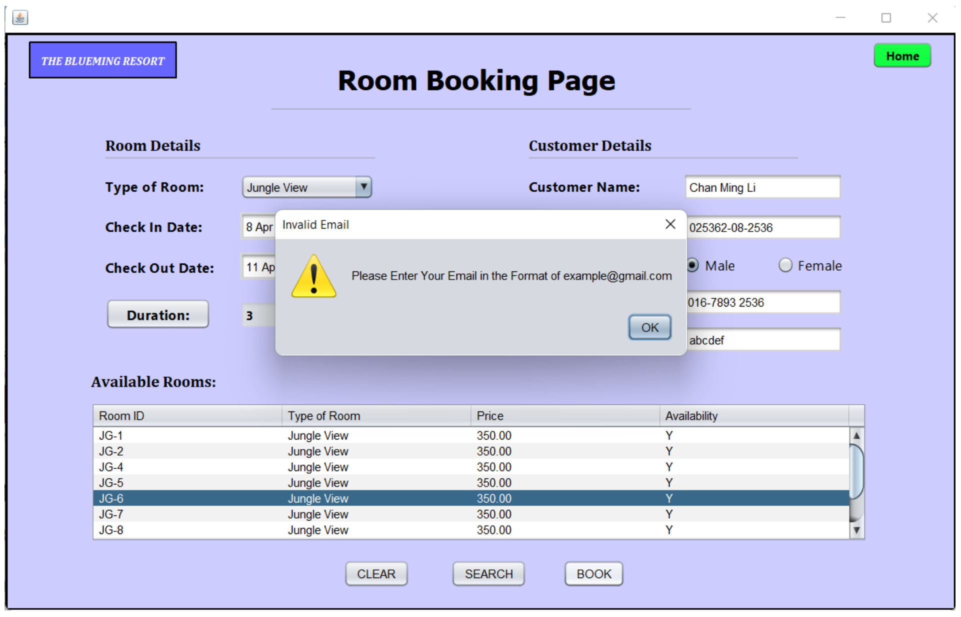
Available Rooms:

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-5	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y

CLEAR SEARCH BOOK

Figure 2.3(h): Missing Room Selection Before Booking

Moreover, staffs must select one of the available rooms displayed in the table which indicates that the selected room will be booked for the customer if the booking is successfully done. If none of the available rooms are selected, the system will pop out a room selection warning message as a reminder to the staff shown in *Figure 2.3(h)*.



THE BLUEMING RESORT

Home

Room Booking Page

Room Details

Type of Room: Jungle View

Check In Date: 8 Apr

Check Out Date: 11 Ap

Duration: 3

Customer Details

Customer Name: Chan Ming Li

025362-08-2536

● Male ○ Female

016-7893 2536

abcdef

Invalid Email

Please Enter Your Email in the Format of example@gmail.com

OK

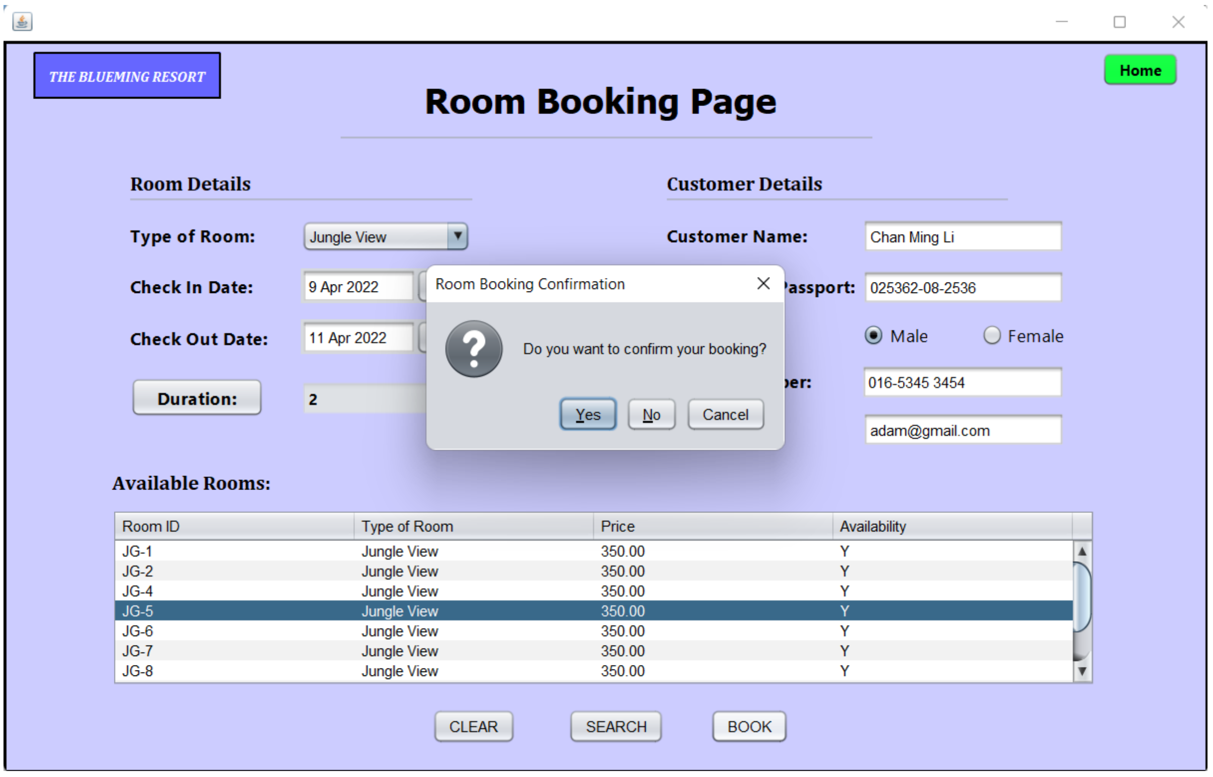
Available Rooms:

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-5	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y

CLEAR SEARCH BOOK

Figure 2.3(i): Invalid Email Format

Besides, an email format validation also exists in the system which only allows staff to input the correct and valid email to avoid invalid data entry and ensure the integrity of the data entered. If the staff entered email in an invalid format such as “mingli@gaiill.com” , an invalid email warning message will be displayed in the screen as illustrated in *Figure 2.3(i)*.



Type of Room: Jungle View

Check In Date: 9 Apr 2022

Check Out Date: 11 Apr 2022

Duration: 2

Customer Name: Chan Ming Li

Passport: 025362-08-2536

● Male ○ Female

016-5345 3454

adam@gmail.com

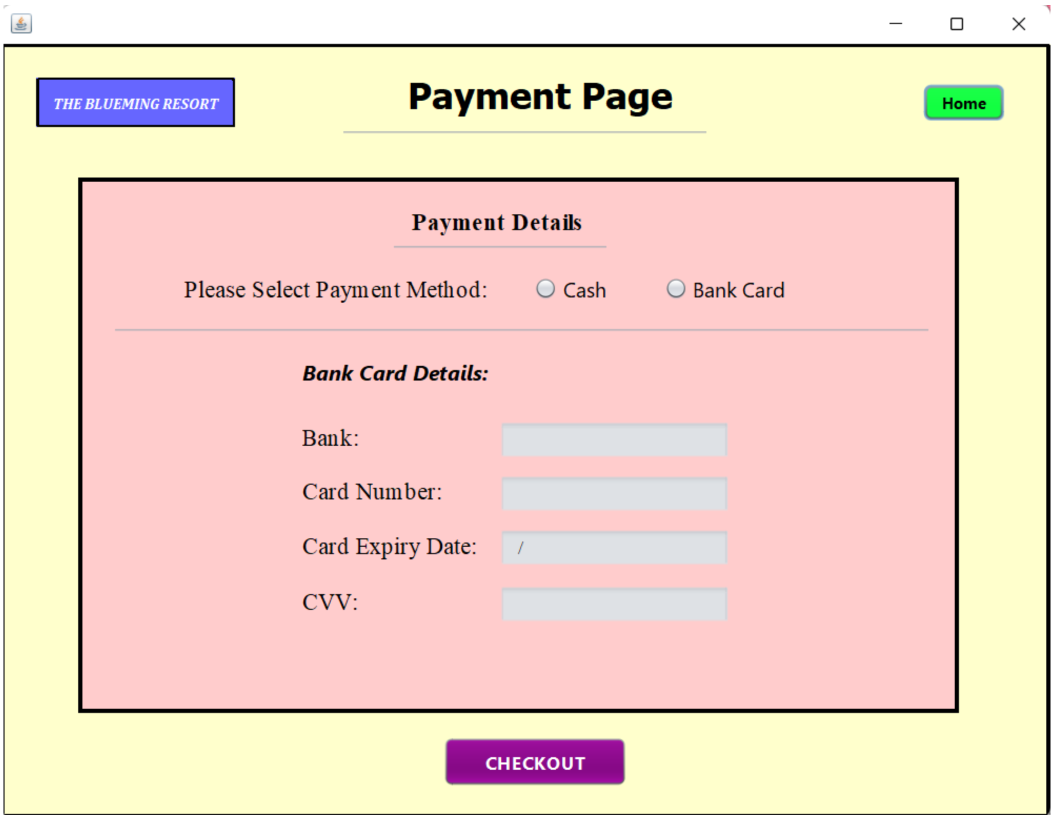
Do you want to confirm your booking?

Room ID	Type of Room	Price	Availability
JG-1	Jungle View	350.00	Y
JG-2	Jungle View	350.00	Y
JG-4	Jungle View	350.00	Y
JG-5	Jungle View	350.00	Y
JG-6	Jungle View	350.00	Y
JG-7	Jungle View	350.00	Y
JG-8	Jungle View	350.00	Y

Figure 2.3(j): Booking Confirmation Message

After getting all the room information and customer details, staff will click the “book” button and navigate to the payment page to continue the room booking process. Before navigating to the payment page, there is a booking confirmation message displayed as shown in *Figure 2.3(j)* to avoid mistakenly clicking on the “book” button. If the “Yes” is clicked, it will navigate to the payment page and all the booking information will be passed to the payment page.

2.4 Payment Page



THE BLUEMING RESORT

Payment Page

Home

Payment Details

Please Select Payment Method: ○ Cash ○ Bank Card

Bank Card Details:

Bank:

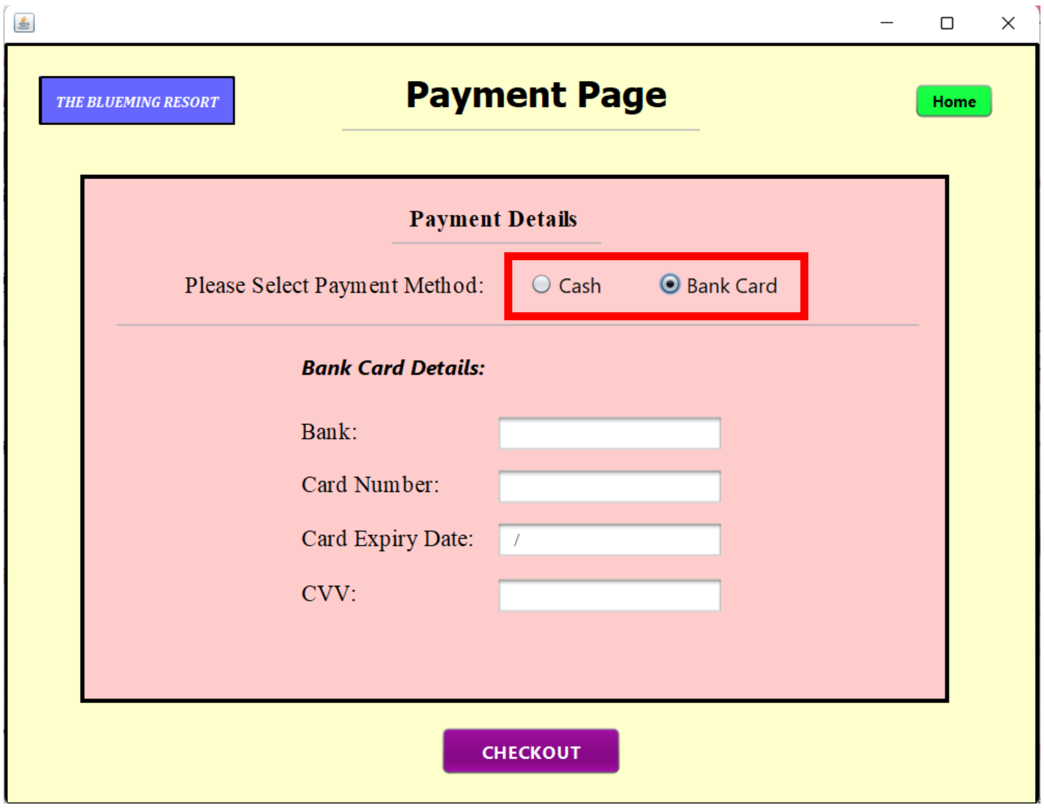
Card Number:

Card Expiry Date: /

CVV:

CHECKOUT

Figure 2.4(a): Payment Page Output



THE BLUEMING RESORT

Payment Page

Home

Payment Details

Please Select Payment Method: Cash [✓ Bank Card]

Bank Card Details:

Bank:

Card Number:

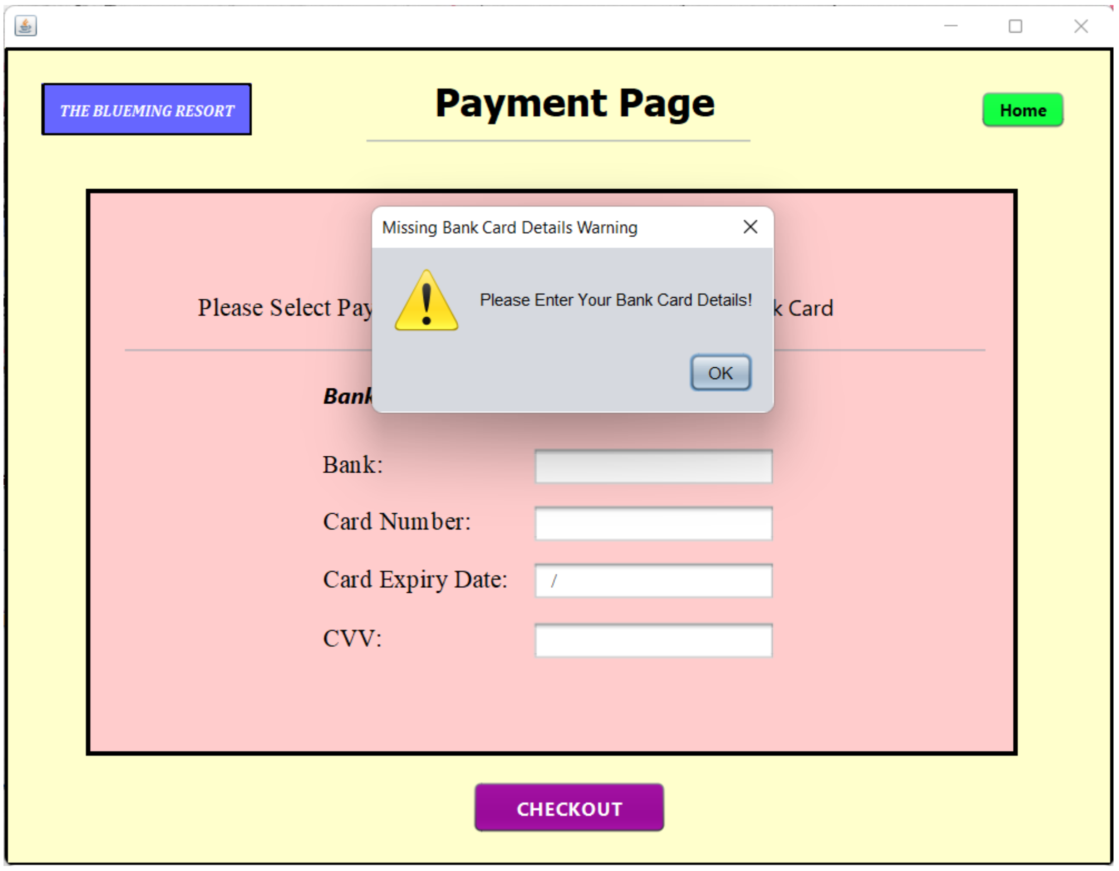
Card Expiry Date: /

CVV:

CHECKOUT

Figure 2.4(b): Bank Card Details' Input Field Enabled if Bank Card is Selected

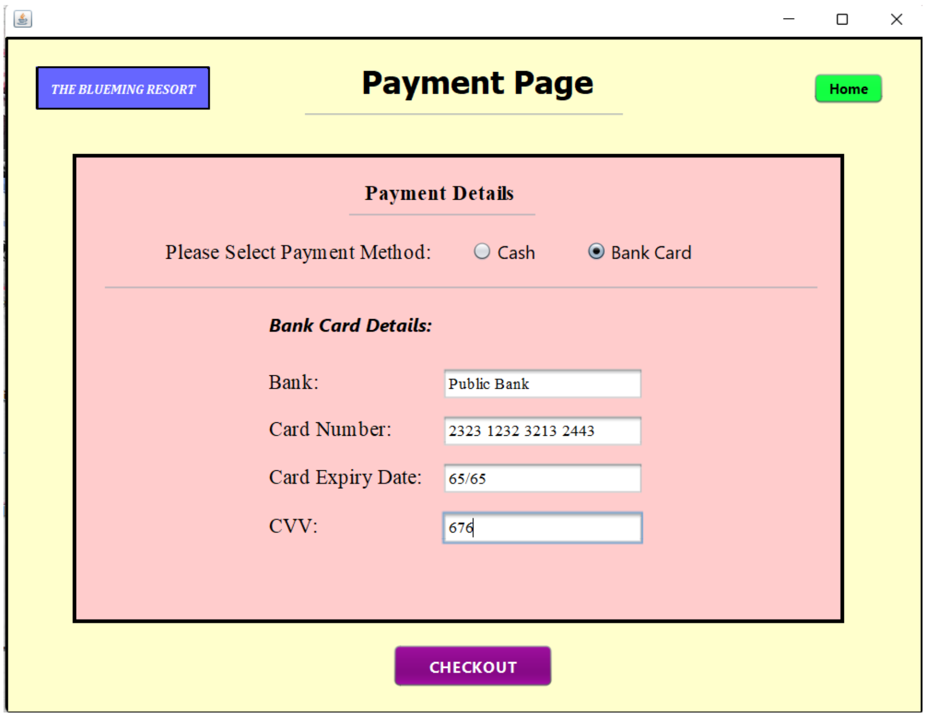
After entering all the booking information in the booking page, payment page will be displayed to allow staff to select for the customer preffered payment method. All input fields available under the bank card details are disabled in default to avoid any irrelvant data entry from the staffs as illustrated in *Figure 2.4(a)*. The input fields will be enabled only if bank card is selected shown in *Figure 2.4(b)* as only bank card has the details of it to enter.


Missing Bank Card Details Warning

Please Enter Your Bank Card Details!

Figure 2.4(c): Missing Bank Card Information if Bank Card is Selected

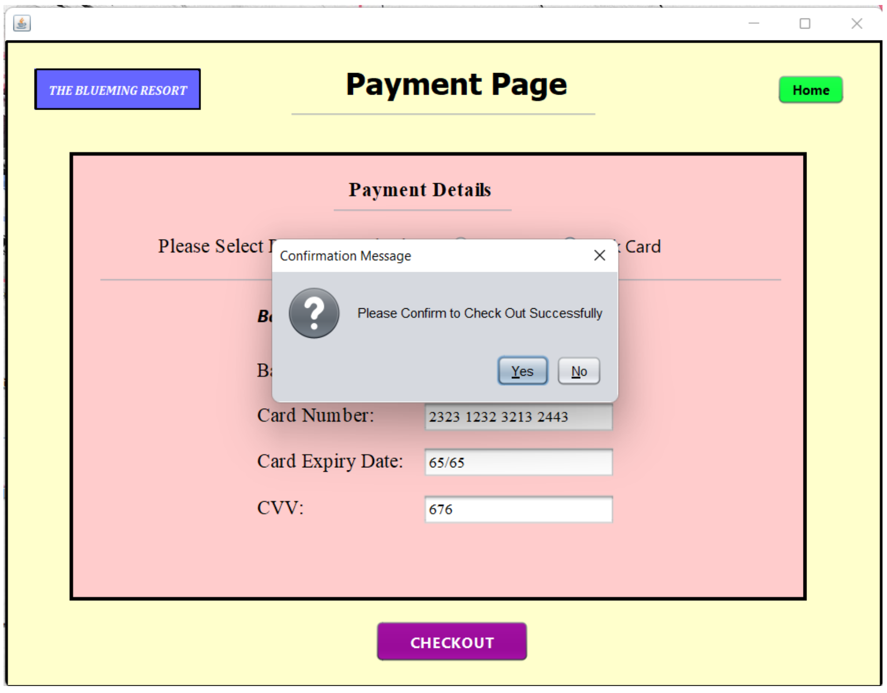
If bank card is selected, staff is required to enter the bank card details such as bank name, card number, expiry date and card verification value (CVV). Hence, if the bank card details are missing, the system will be displaying a missing bank card details warning message and request staff the enter those details as illustrated in *Figure 2.4(c)*.



Bank: Public Bank
Card Number: 2323 1232 3213 2443
Card Expiry Date: 65/65
CVV: 676

Figure 2.4(d): Formatting in Input Field of Bank Card Details

According to *Figure 2.4(d)*, the input fields for bank card details are formatted automatically when staff is entering the values. For instance, the card expiry date is formatted to "##/##" which only allows staffs to enter 4 numeric values with a "/" in the middle of the values. Besides, staffs can only enter exact 3 numeric values in the "CVV" input field as all CVV for bank card is formatted in 3 numerical values.



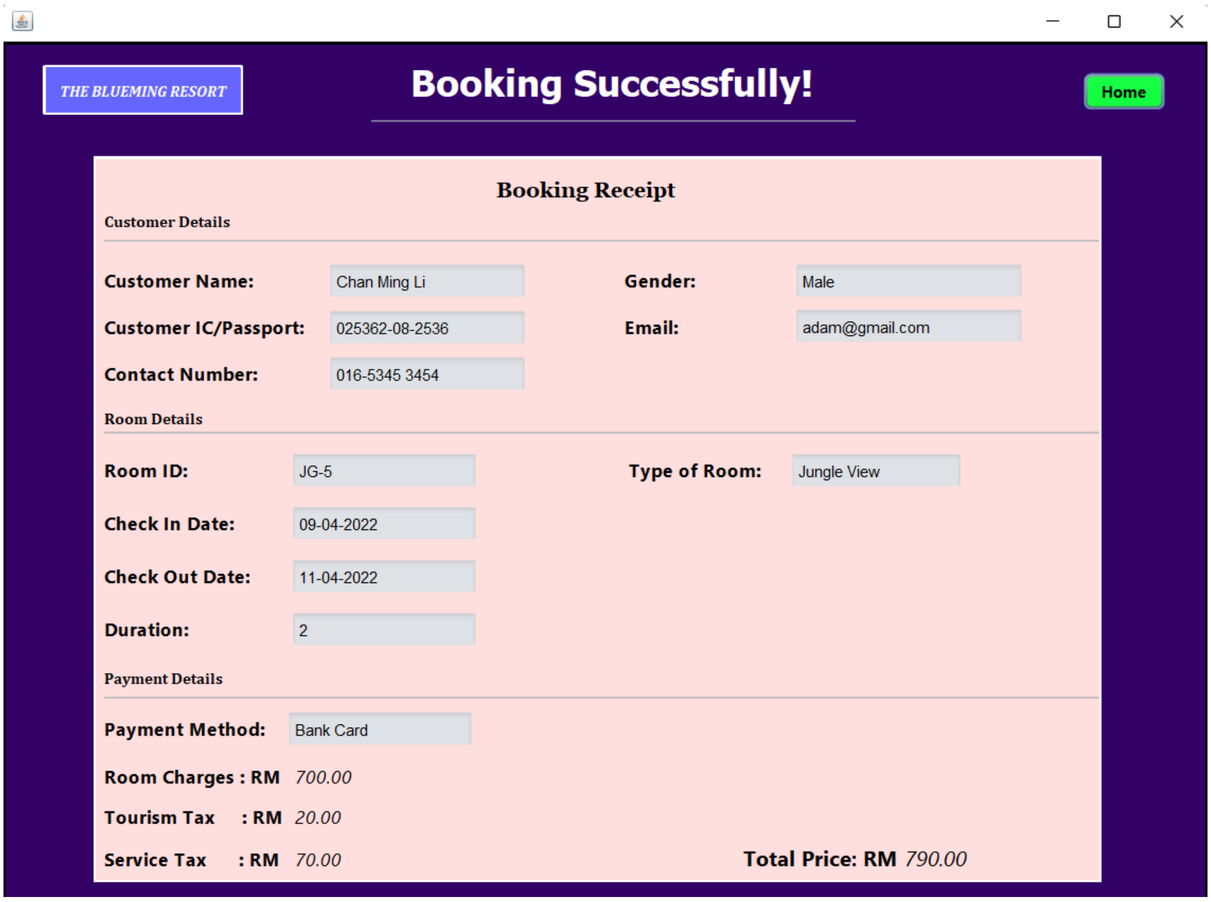
Please Confirm to Check Out Successfully

Card Number: 2323 1232 3213 2443
Card Expiry Date: 65/65
CVV: 676

Figure 2.4(e): Check Out Confirmation

After all values are entered correctly, the system will display a confirmation message as shown in *Figure 2.4(e)* to ensure that the staff wanted to checkout for the payment. Staff will be navigated to a payment receipt page to proof that the booking and payment is done successfully.

2.5 Receipt Page



Booking Successfully!

Booking Receipt

Customer Details

Customer Name:	Chan Ming Li	Gender:	Male
Customer IC/Passport:	025362-08-2536	Email:	adam@gmail.com
Contact Number:	016-5345 3454		

Room Details

Room ID:	JG-5	Type of Room:	Jungle View
Check In Date:	09-04-2022		
Check Out Date:	11-04-2022		
Duration:	2		

Payment Details

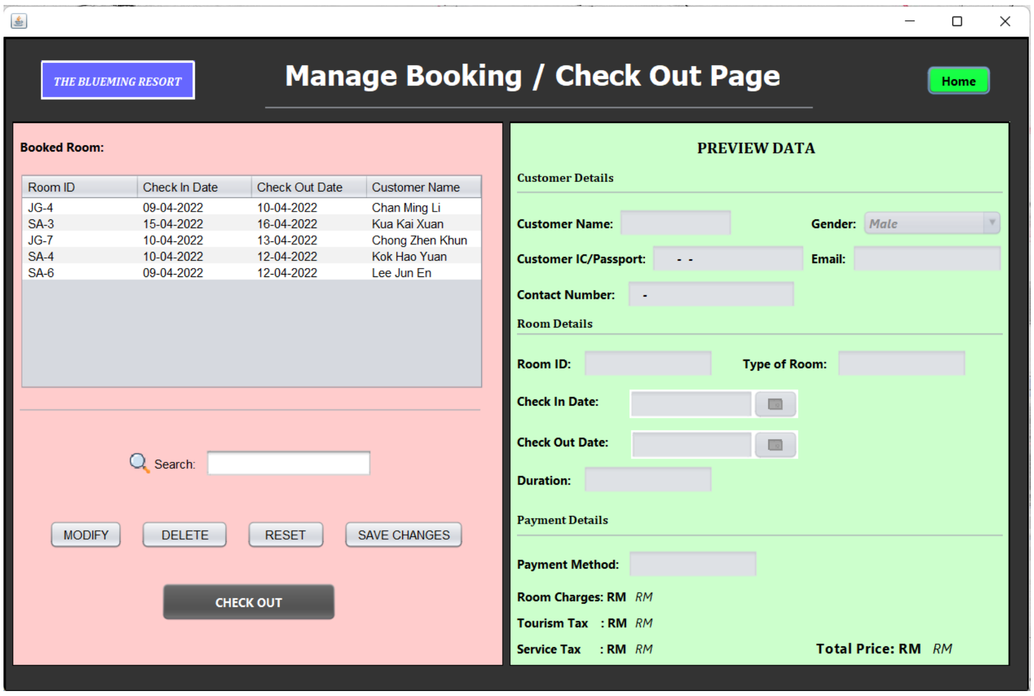
Payment Method:	Bank Card
Room Charges : RM	*700.00*
Tourism Tax : RM	*20.00*
Service Tax : RM	*70.00*

Total Price: RM *790.00*

Figure 2.5(a): Receipt Page Output

The booking receipt page as illustrated in *Figure 2.5(a)* will be displayed after the booking and payment information is confirmed. The receipt consists of information such as customer information, room details, payment method, charges and more. All the information in the information will be displayed in the receipt automatically once the staff checkout from the payment page. After viewing the receipt, staffs can return back to home page by clicking the “home” button located at the top right corner of the page.

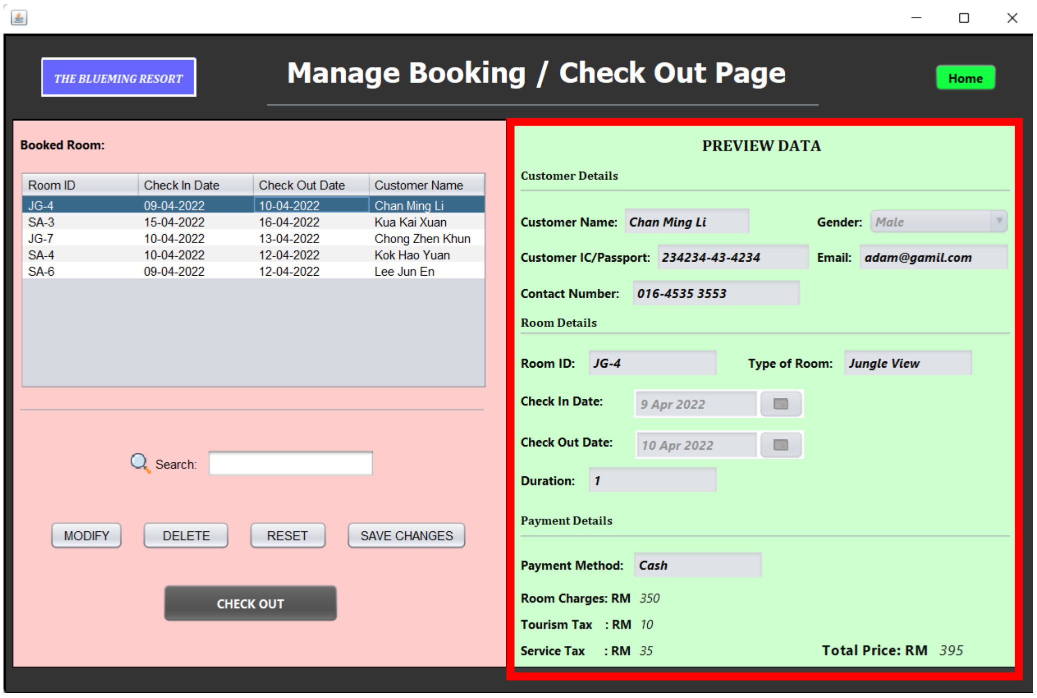
2.6 Manage Booking / Room Check Out Page



Room ID	Check In Date	Check Out Date	Customer Name
JG-4	09-04-2022	10-04-2022	Chan Ming Li
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En

Figure 2.6(a): Manage Booking / Room Check Out Page

The purpose of having a manage booking / room check out page as shown in *Figure 2.6(a)* is to have an overview of the booked rooms' information and allows staff to execute some functionalities which is provided in the page. The functionalities available in the page including preview, modify, delete, search booked rooms' information and check out rooms.

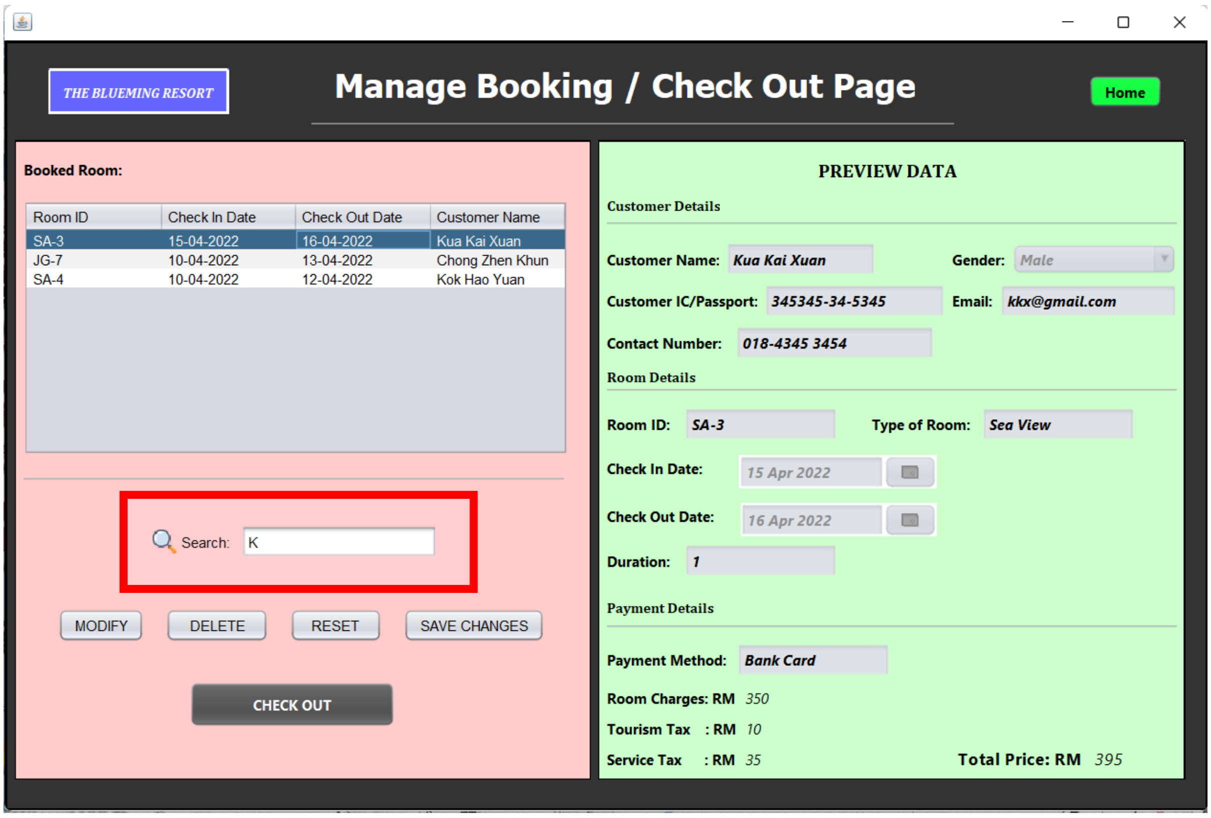


Room ID	Check In Date	Check Out Date	Customer Name
JG-4	09-04-2022	10-04-2022	Chan Ming Li
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En

Customer Name: *Chan Ming Li*
Gender: *Male*
Customer IC/Passport: *234234-43-4234*
Email: *adam@gamil.com*
Contact Number: *016-4535 3553*
Room ID: *JG-4*
Type of Room: *Jungle View*
Check In Date: *9 Apr 2022*
Check Out Date: *10 Apr 2022*
Duration: *1*
Payment Method: *Cash*
Room Charges: RM *350*
Tourism Tax : RM *10*
Service Tax : RM *35*
Total Price: RM *395*

Figure 2.6(b): Preview Data by Clicking the Rows in Booked Rooms Table

Staffs are able to preview the information of every booked room by clicking the rows in the booked rooms table located in the left panel of the page as illustrated in *Figure 2.6(b)*. Every row in the table indicates different booked rooms in the resort. When a particular row is selected, the information of the booked room including customer details, room details and charges will be displayed in the right panel.



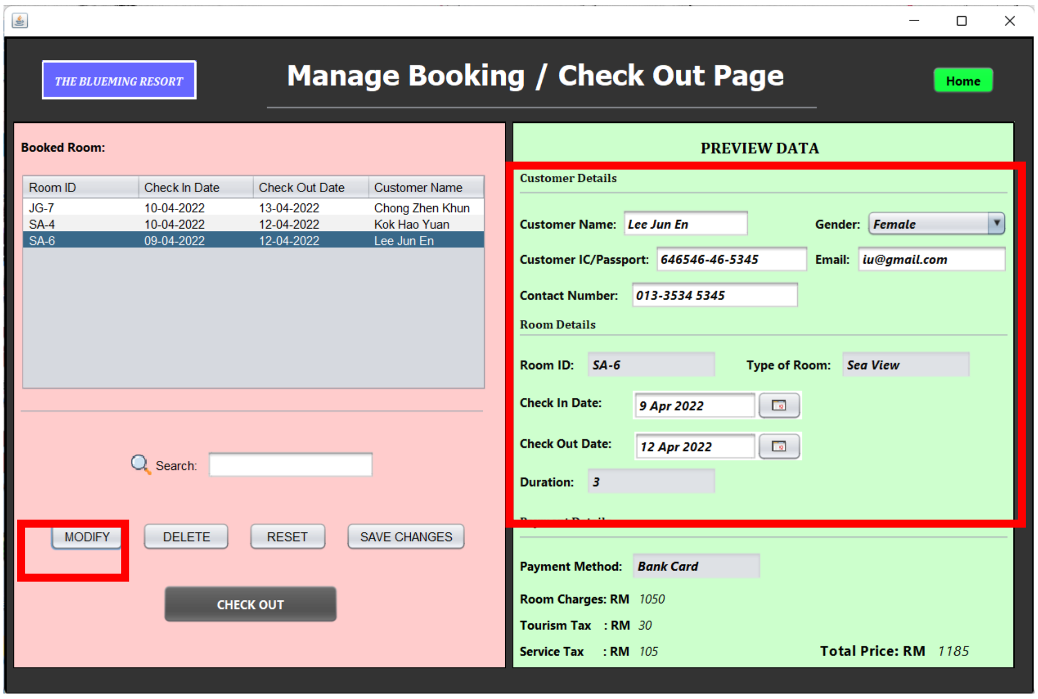
Room ID	Check In Date	Check Out Date	Customer Name
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan

Search: K

Customer Name: Kua Kai Xuan
Gender: Male
Customer IC/Passport: 345345-34-5345
Email: kkx@gmail.com
Contact Number: 018-4345 3454
Room ID: SA-3
Type of Room: Sea View
Check In Date: 15 Apr 2022
Check Out Date: 16 Apr 2022
Duration: 1
Payment Method: Bank Card
Room Charges: RM 350
Tourism Tax : RM 10
Service Tax : RM 35
Total Price: RM 395

Figure 2.6(c): Search Function

Furthermore, there is a search function available in the page which increase work efficiency of staffs when they are looking for a specific room's information. As the search bar is case sensitive, the search results will only include the information that contains the exact same searched input. For instance, when the staff entered a "K" in the search bar as shown in *Figure 2.6(c)*, the search results will return information that contains "K" only.



Booked Room:

Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En

Search:

Customer Details

Customer Name: Lee Jun En
Gender: Female
Customer IC/Passport: 646546-46-5345
Email: iu@gmail.com
Contact Number: 013-3534 5345

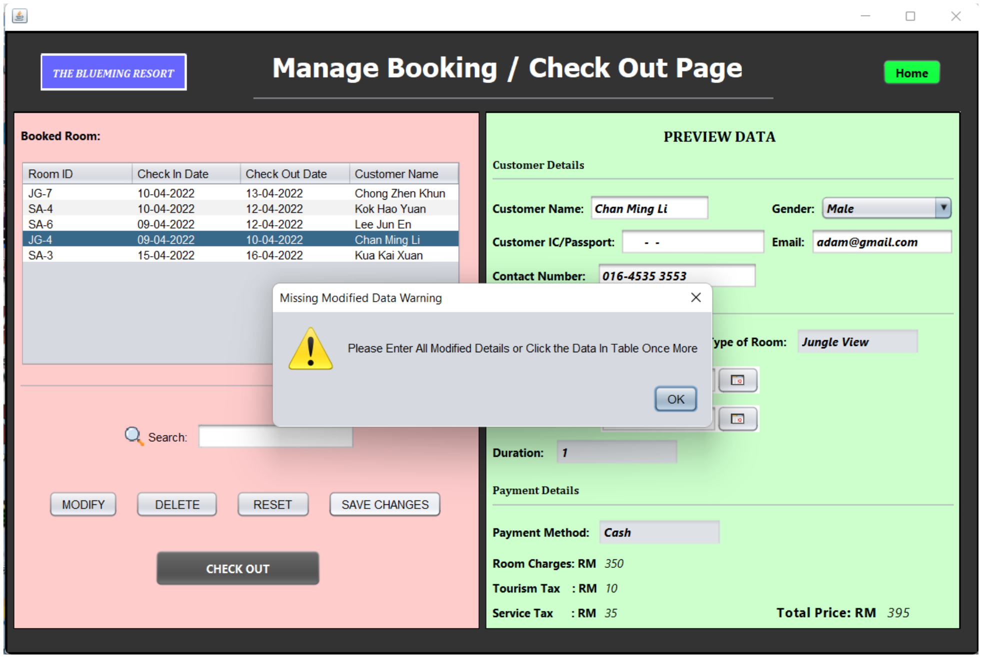
Room Details

Room ID: SA-6
Type of Room: Sea View
Check In Date: 9 Apr 2022
Check Out Date: 12 Apr 2022
Duration: 3

Payment Method: Bank Card
Room Charges: RM 1050
Tourism Tax : RM 30
Service Tax : RM 105
Total Price: RM 1185

Figure 2.6(d): Input Fields Enabled When Modify Button is Clicked

When the modify button is clicked, the input fields in the preview data section will be enabled as shown in *Figure 2.6(d)* and allow staffs to make changes for some information. There is some information that is not allowed to modify such as room id, room type, payment method and charges. If there is a change in the check in or out date which will leads to the change of duration, the charges will automatically update when the modified information is confirmed and save changes button is clicked.



Booked Room:

Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En
JG-4	09-04-2022	10-04-2022	Chan Ming Li
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan

Search:

Customer Details

Customer Name: Chan Ming Li
Gender: Male
Customer IC/Passport: - -
Email: adam@gmail.com
Contact Number: 016-4535 3553

Missing Modified Data Warning

Please Enter All Modified Details or Click the Data In Table Once More

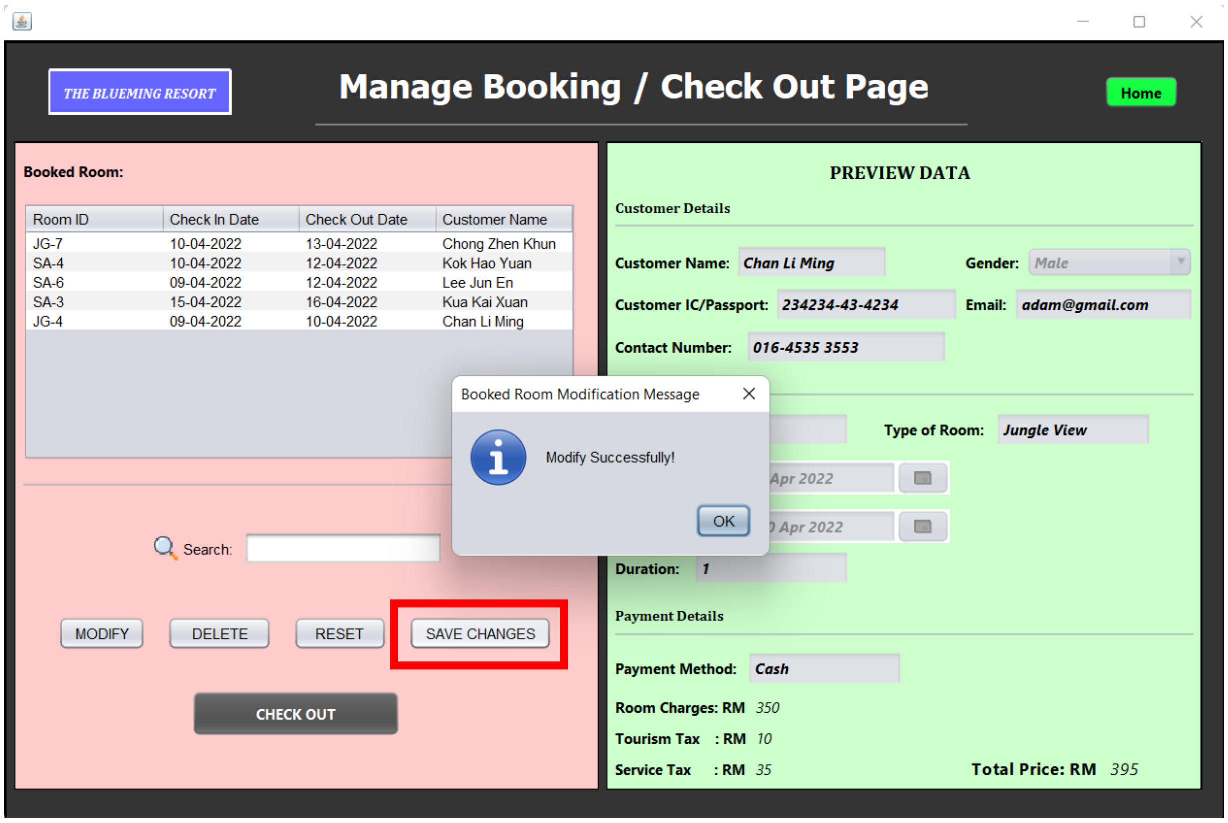
ype of Room: Jungle View
Duration: 1

Payment Details

Payment Method: Cash
Room Charges: RM 350
Tourism Tax : RM 10
Service Tax : RM 35
Total Price: RM 395

Figure 2.6(e): Missing Modified Information in Input Field

If the staff leave the input field blank when modifying the information, the system will display a missing modified data warning message. The missing data can be restored by clicking the row in the booked room table once.


THE BLUEMING RESORT

Manage Booking / Check Out Page

Home

Booked Room:

Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan
JG-4	09-04-2022	10-04-2022	Chan Li Ming

Search:

MODIFY DELETE RESET SAVE CHANGES

CHECK OUT

PREVIEW DATA

Customer Details

Customer Name: Chan Li Ming Gender: Male

Customer IC/Passport: 234234-43-4234 Email: adam@gmail.com

Contact Number: 016-4535 3553

Type of Room: Jungle View

Apr 2022

Apr 2022

Duration: 1

Payment Details

Payment Method: Cash

Room Charges: RM 350

Tourism Tax : RM 10

Service Tax : RM 35

Total Price: RM 395

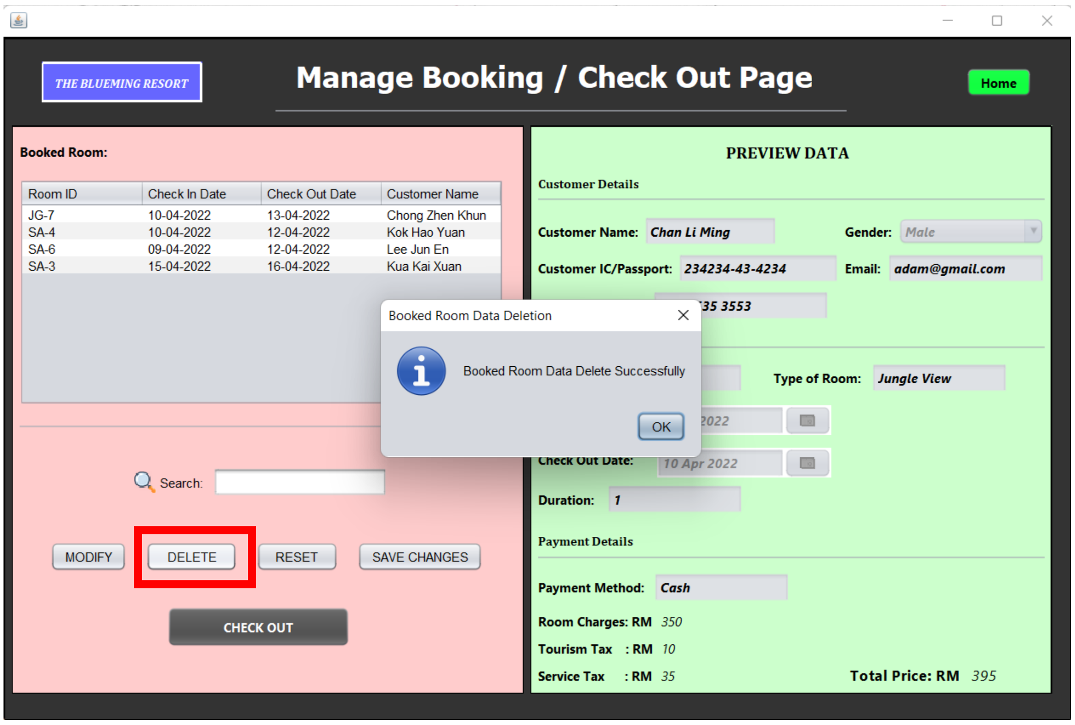
Booked Room Modification Message

Modify Successfully!

OK

Figure 2.6(f): Modify Successfully Message

After modifying all the information, staff is required to click the save changes button in order to save all the modified information. Then, the system will display a modification successfully message as illustrated in *Figure 2.6(f)*.



Manage Booking / Check Out Page

Booked Room:

Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En
SA-3	15-04-2022	16-04-2022	Kua Kai Xuan

Search:

Booked Room Data Deletion

Booked Room Data Delete Successfully

PREVIEW DATA

Customer Details

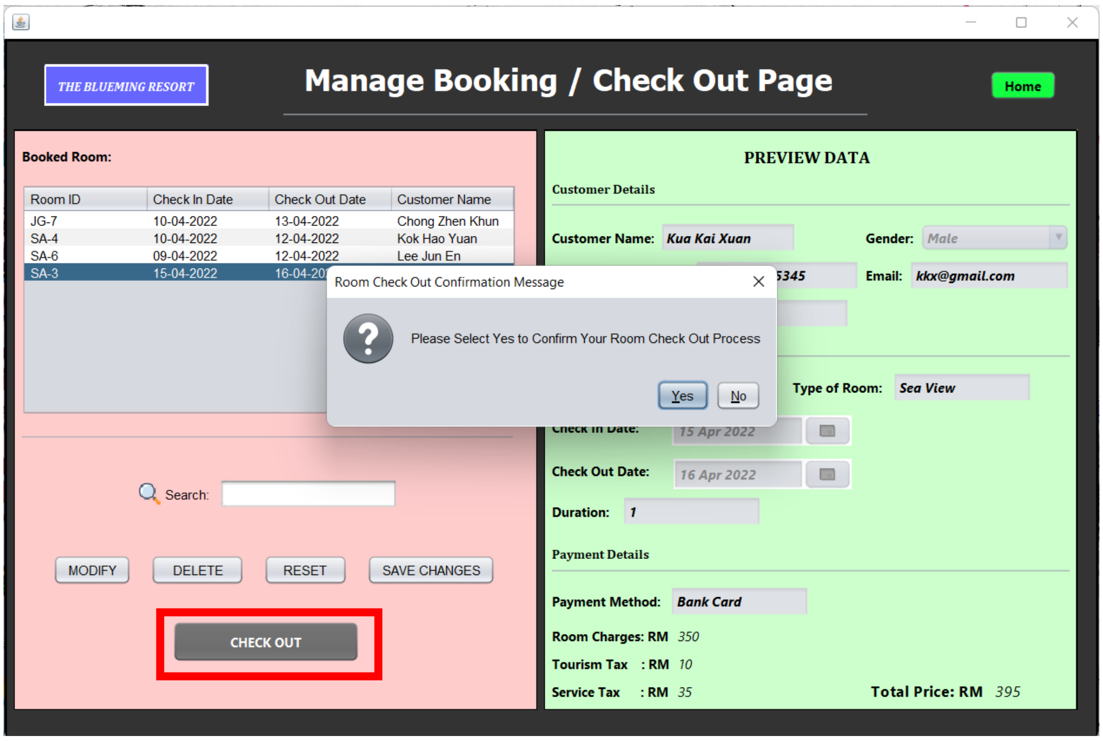
Customer Name: Chan Li Ming Gender: Male
Customer IC/Passport: 234234-43-4234 Email: adam@gmail.com
35 3553
Type of Room: Jungle View
2022
Check Out Date: 10 Apr 2022
Duration: 1

Payment Details

Payment Method: Cash
Room Charges: RM 350
Tourism Tax : RM 10
Service Tax : RM 35 Total Price: RM 395

Figure 2.6(g): Delete Booking Function

Delete booking function is available in the system where staff is able to select a specific booking data to delete and the booked room will automatically become available. When a customer wanted to cancel his booking, this function can be used by the staff to delete the specific booked room's information.



Manage Booking / Check Out Page

Booked Room:

Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En
SA-3	15-04-2022	16-04-20	

Search:

Room Check Out Confirmation Message

Please Select Yes to Confirm Your Room Check Out Process

PREVIEW DATA

Customer Details

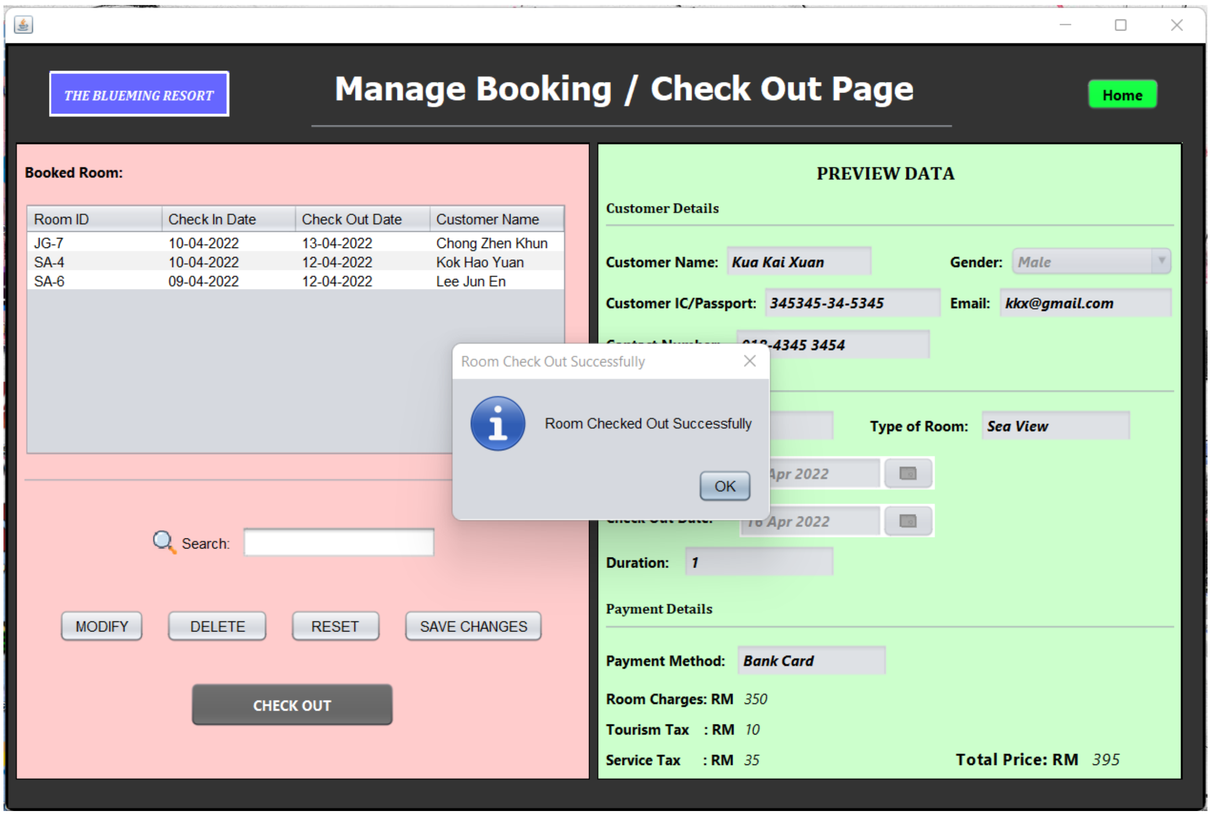
Customer Name: Kua Kai Xuan Gender: Male
5345 Email: kkx@gmail.com
Type of Room: Sea View
Check In Date: 15 Apr 2022
Check Out Date: 16 Apr 2022
Duration: 1

Payment Details

Payment Method: Bank Card
Room Charges: RM 350
Tourism Tax : RM 10
Service Tax : RM 35 Total Price: RM 395

Figure 2.6(h): Room Check Out Confirmation Message

By clicking the room checkout button, staffs can check out the room when the room reaches its check out date. After that, the system will display a room checkout confirmation message as shown in *Figure 2.6(h)* to avoid any incident happens.



Room ID	Check In Date	Check Out Date	Customer Name
JG-7	10-04-2022	13-04-2022	Chong Zhen Khun
SA-4	10-04-2022	12-04-2022	Kok Hao Yuan
SA-6	09-04-2022	12-04-2022	Lee Jun En

Room Check Out Successfully

Room Checked Out Successfully

Customer Name: Kua Kai Xuan
Gender: Male
Customer IC/Passport: 345345-34-5345
Email: kkx@gmail.com
Contact Number: 012-4345 3454
Type of Room: Sea View
Apr 2022
Check Out Date: 16 Apr 2022
Duration: 1
Payment Method: Bank Card
Room Charges: RM 350
Tourism Tax : RM 10
Service Tax : RM 35
Total Price: RM 395

Figure 2.6(j): Room Check Out Successfully Message

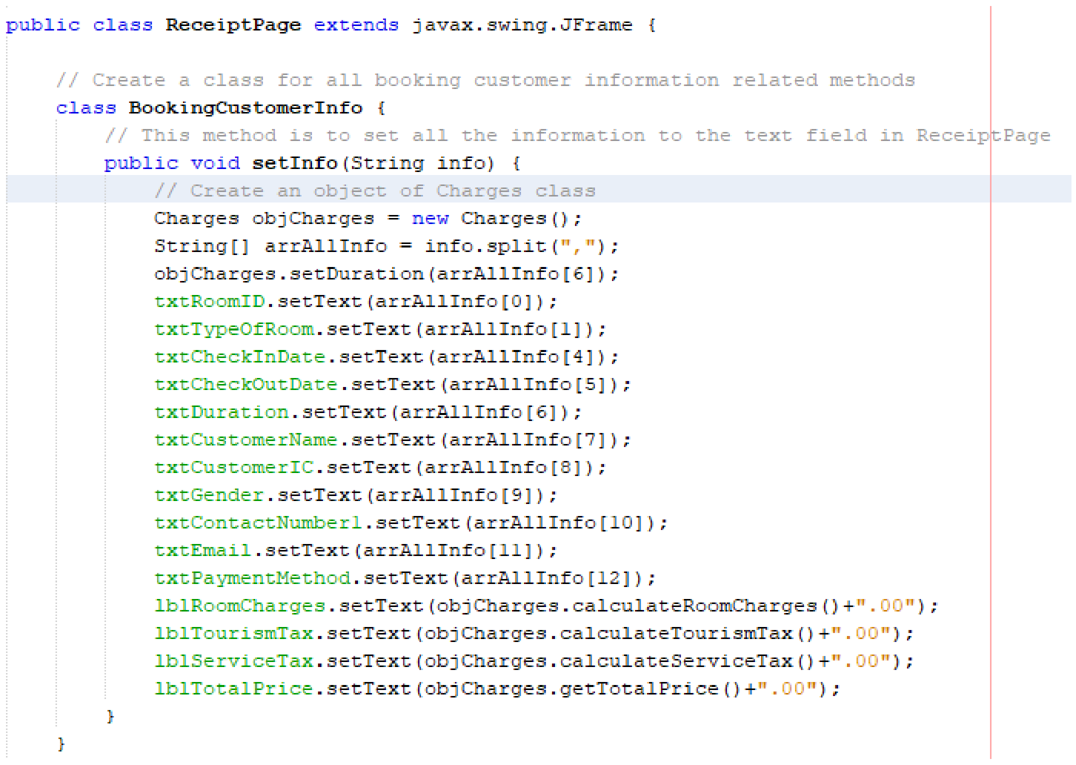
After the staff confirmed to check out the room, the room is successfully checked out if there is a message displayed as illustrated in *Figure 2.6(j)*. The information of the checked-out rooms will be added into a text file as a booking history database that stores all the information of booked rooms. Therefore, the booking history of the resort is still available if the staffs wanted to utilize it to generate sales report or extract any insights from the past data.

3.0 OOP Concepts & Source Codes Explanation

3.1 Object Oriented Programming (OOP) Concepts Implemented

3.1.1 Class & Object – Receipt Page

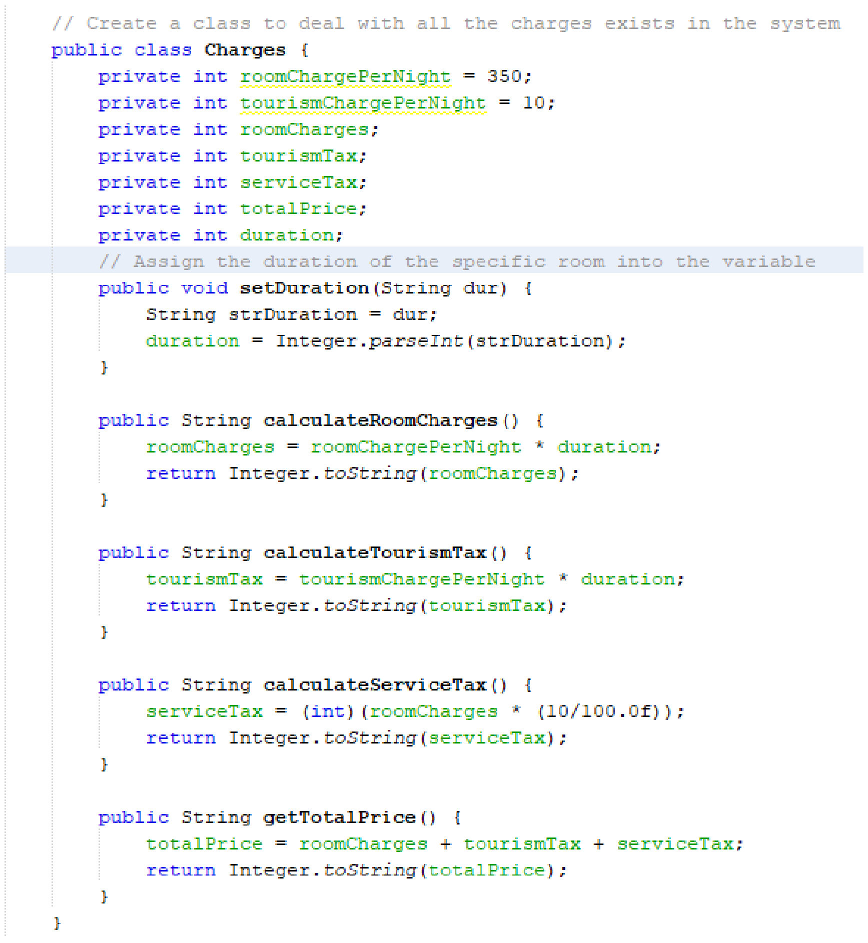
The main elements in object-oriented programming concepts are class and object. Class can be considered as a template for objects while objects is an instance of a class. Class contains the same characteristics or attributes of all objects (Hartman, 2022). By implementing class and object, redundant codes can be avoided and provides a clear structure of codes for the developr. Hence, class and object are also implemented in the room booking system.



```
public class ReceiptPage extends javax.swing.JFrame {

    // Create a class for all booking customer information related methods
    class BookingCustomerInfo {
        // This method is to set all the information to the text field in ReceiptPage
        public void setInfo(String info) {
            // Create an object of Charges class
            Charges objCharges = new Charges();
            String[] arrAllInfo = info.split(",");
            objCharges.setDuration(arrAllInfo[6]);
            txtRoomID.setText(arrAllInfo[0]);
            txtTypeOfRoom.setText(arrAllInfo[1]);
            txtCheckInDate.setText(arrAllInfo[4]);
            txtCheckOutDate.setText(arrAllInfo[5]);
            txtDuration.setText(arrAllInfo[6]);
            txtCustomerName.setText(arrAllInfo[7]);
            txtCustomerIC.setText(arrAllInfo[8]);
            txtGender.setText(arrAllInfo[9]);
            txtContactNumber1.setText(arrAllInfo[10]);
            txtEmail.setText(arrAllInfo[11]);
            txtPaymentMethod.setText(arrAllInfo[12]);
            lblRoomCharges.setText(objCharges.calculateRoomCharges()+".00");
            lblTourismTax.setText(objCharges.calculateTourismTax()+".00");
            lblServiceTax.setText(objCharges.calculateServiceTax()+".00");
            lblTotalPrice.setText(objCharges.getTotalPrice()+".00");
        }
    }
```

Figure 3.1.1(a): BookingCustomerInfo Class in Receipt Page

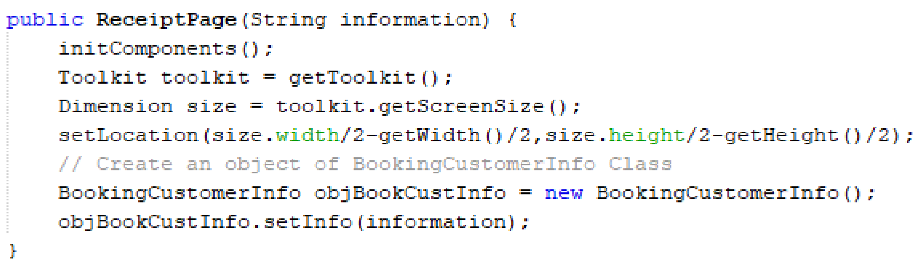
One of the example codes that implemented class and object is in the Receipt Page. There is a class known as “BookingCustomerInfo” that is purposely created for booking and customer information related objects and methods as illustrated in *Figure 3.1.1(a)*. The class consists of a superclass of “ReceiptPage” and a method called “setInfo()” which receives all the booking and customer information as a argument and set the relevant information to the text fields in the Receipt Page. In the “setInfo()” method, an object of the “Charges” class known as “objCharges” is created in order to get the return values from the “Charges” class which is the different charges of the particular room. Besides, the “setDuration” method is called from “objCharges” to pass the duration value to the “Charges” class for calculation purposes.



```
// Create a class to deal with all the charges exists in the system
public class Charges {
    private int roomChargePerNight = 350;
    private int tourismChargePerNight = 10;
    private int roomCharges;
    private int tourismTax;
    private int serviceTax;
    private int totalPrice;
    private int duration;
    // Assign the duration of the specific room into the variable
    public void setDuration(String dur) {
        String strDuration = dur;
        duration = Integer.parseInt(strDuration);
    }

    public String calculateRoomCharges() {
        roomCharges = roomChargePerNight * duration;
        return Integer.toString(roomCharges);
    }

    public String calculateTourismTax() {
        tourismTax = tourismChargePerNight * duration;
        return Integer.toString(tourismTax);
    }

    public String calculateServiceTax() {
        serviceTax = (int)(roomCharges * (10/100.0f));
        return Integer.toString(serviceTax);
    }

    public String getTotalPrice() {
        totalPrice = roomCharges + tourismTax + serviceTax;
        return Integer.toString(totalPrice);
    }
}
```

Figure 3.1.1(b): Charges Class in Receipt Page

In the Receipt Page, there is another class known as “Charges” to store all objects and classes related to charges as illustrated in *Figure 3.1.1(b)*. In the class, there are five methods available including “setDuration()”, “calculateRoomCharges()”, “calculateTourismTax()”, “calculateServiceTax()” and “getTotalPrice()”. For the “setDuration” method, it receives the duration value from the “setInfo()” method in “BookingCustomerInfo” class and assign the value to a string variable known as “duration”. For the other methods, the method’s name clearly describes the purpose of the method.



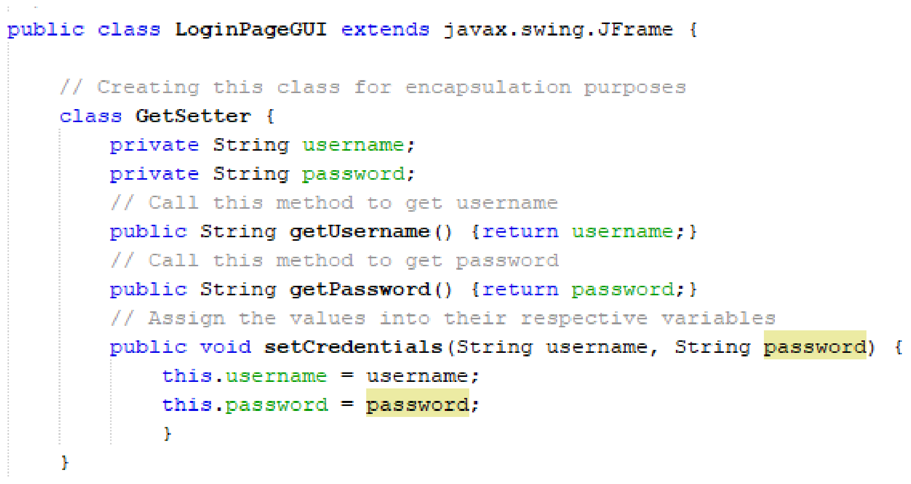
```
public ReceiptPage(String information) {
    initComponents();
    Toolkit toolkit = getToolkit();
    Dimension size = toolkit.getScreenSize();
    setLocation(size.width/2-getWidth()/2,size.height/2-getHeight()/2);
    // Create an object of BookingCustomerInfo Class
    BookingCustomerInfo objBookCustInfo = new BookingCustomerInfo();
    objBookCustInfo.setInfo(information);
}
```

Figure 3.1.1(c): Calling setInfo() Method

According to *Figure 3.1.1(c)*, an object of "BookingCustomerInfo" class, "objBookCustInfo" is created in order to call the "setInfo()" method and pass the all the information received by the Receipt Page from the Payment Page to the method. Hence, the receipt is able to display all the information automatically in the input field when staffs are navigated to the Receipt Page.

3.1.2 Encapsulation – Login Page

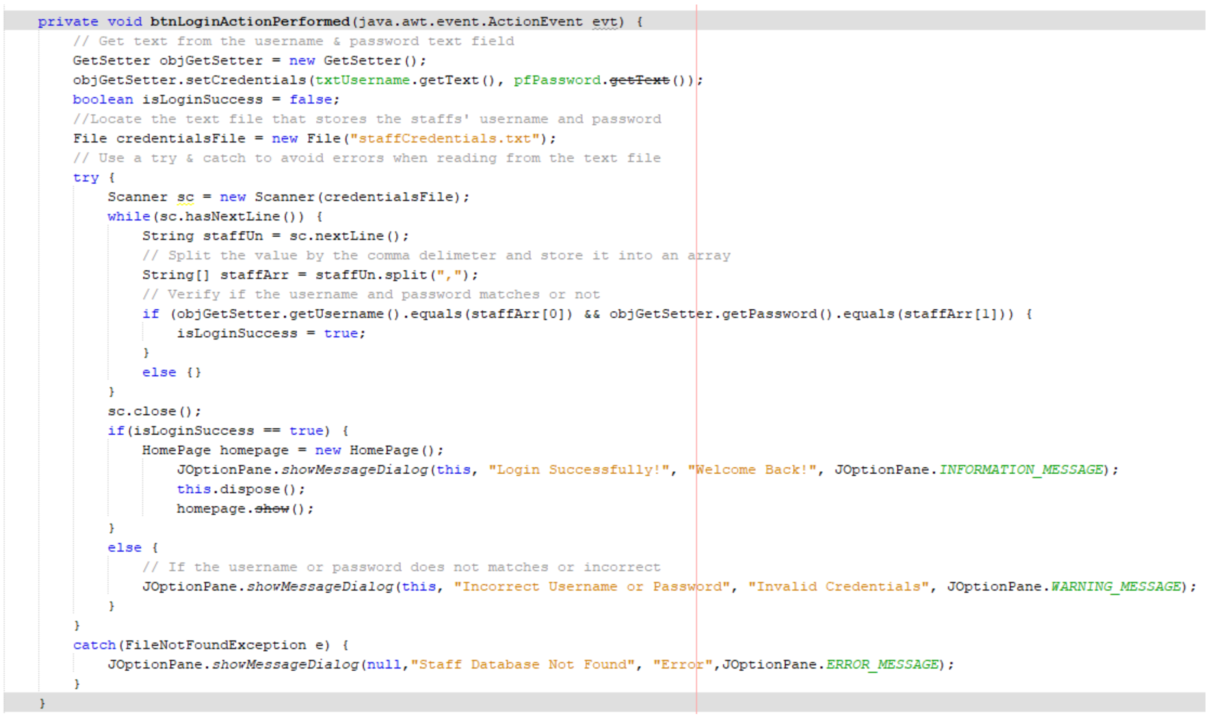
Encapsulation is often used in object-oriented programming to wrap or bind the data and code which provides a protective shield to prevent the data accessed by the code outside from the shield. The data can only be accessed by the methods which is in the same class in order to ensure the data security and protection (Great Learning Team, 2022). In the room booking system, encapsulation is also implemented for the login username and password as the credentials is extremely important to access the system.



```
public class LoginPageGUI extends javax.swing.JFrame {

    // Creating this class for encapsulation purposes
    class GetSetter {
        private String username;
        private String password;
        // Call this method to get username
        public String getUsername() {return username;}
        // Call this method to get password
        public String getPassword() {return password;}
        // Assign the values into their respective variables
        public void setCredentials(String username, String password) {
            this.username = username;
            this.password = password;
            }
    }
```

Figure 3.1.2(a): GetSetter Class for Encapsulation in Login Page

Based on *Figure 3.1.2(a)*, there is a class named as "GetSetter" which contains methods to set and get the credentials of staffs when they are logging into the system. The main purpose of creating this class is to ensure the data security and provide protection for the data by encapsulating the credentials of the staffs. There are two private variables declared to prevent access from the codes which is not in the class. The method within the class called "setCredentials" will receive the data when it is called by an object and assign the data to the respective variable which is username and password. Besides, there are two other methods known as "getUsername" and "getPassword" is created to return the two values which is username and password if they are called.



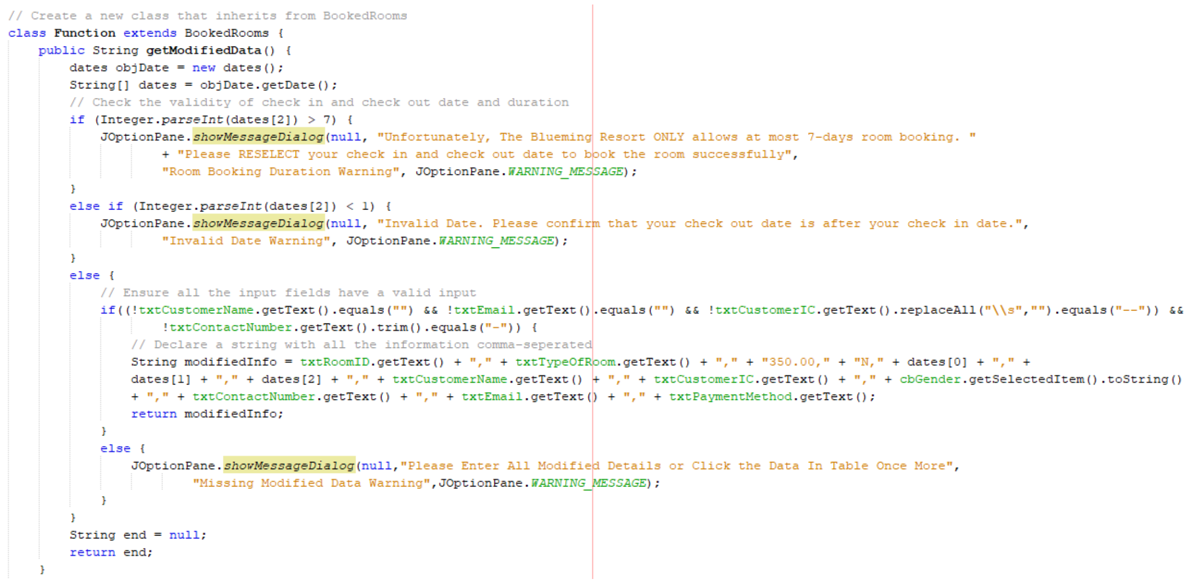
```java
private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
    // Get text from the username & password text field
    GetSetter objGetSetter = new GetSetter();
    objGetSetter.setCredentials(txtUsername.getText(), pfPassword.~~getText~~());
    boolean isLoginSuccess = false;
    //Locate the text file that stores the staffs' username and password
    File credentialsFile = new File("staffCredentials.txt");
    // Use a try & catch to avoid errors when reading from the text file
    try {
        Scanner sc = new Scanner(credentialsFile);
        while(sc.hasNextLine()) {
            String staffUn = sc.nextLine();
            // Split the value by the comma delimeter and store it into an array
            String[] staffArr = staffUn.split(",");
            // Verify if the username and password matches or not
            if (objGetSetter.getUsername().equals(staffArr[0]) && objGetSetter.getPassword().equals(staffArr[1])) {
                isLoginSuccess = true;
            }
            else {}
        }
        sc.close();
        if(isLoginSuccess == true) {
            HomePage homepage = new HomePage();
                JOptionPane.showMessageDialog(this, "Login Successfully!", "Welcome Back!", JOptionPane.INFORMATION_MESSAGE);
                this.dispose();
                homepage.~~show~~();
        }
        else {
            // If the username or password does not matches or incorrect
            JOptionPane.showMessageDialog(this, "Incorrect Username or Password", "Invalid Credentials", JOptionPane.WARNING_MESSAGE);
        }
    }
    catch(FileNotFoundException e) {
        JOptionPane.showMessageDialog(null,"Staff Database Not Found", "Error",JOptionPane.ERROR_MESSAGE);
    }
}
```

Figure 3.1.2(b): Verifying the Credentials of Staffs by Calling the GetSetter Class

Firstly, an object of “GetSetter” class will be created as “objGetSetter” in order to call “setCredentials” method which is located in the class. When calling the method, the username and password entered by the staff will be passed into the method for encapsulation. After that, the system verifies if the credentials entered by staff is correct and valid or not by checking from the text file, “staffCredentials.txt” which stores all the staffs’ credentials. In the verifying process, “getUsername” and “getPassword” is called by the object created earlier in order to retrieve the credentials entered so that the credentials cannot be accessed by the other code blocks. If the credentials are verified successfully, the staff will be navigated to the staff home page with a welcome message. Hence, encapsulation concept is useful when the system wanted to hide the data from other classes to ensure the data security.

3.1.3 Inheritance – Manage Booking / Room Check Out Page (Modify)

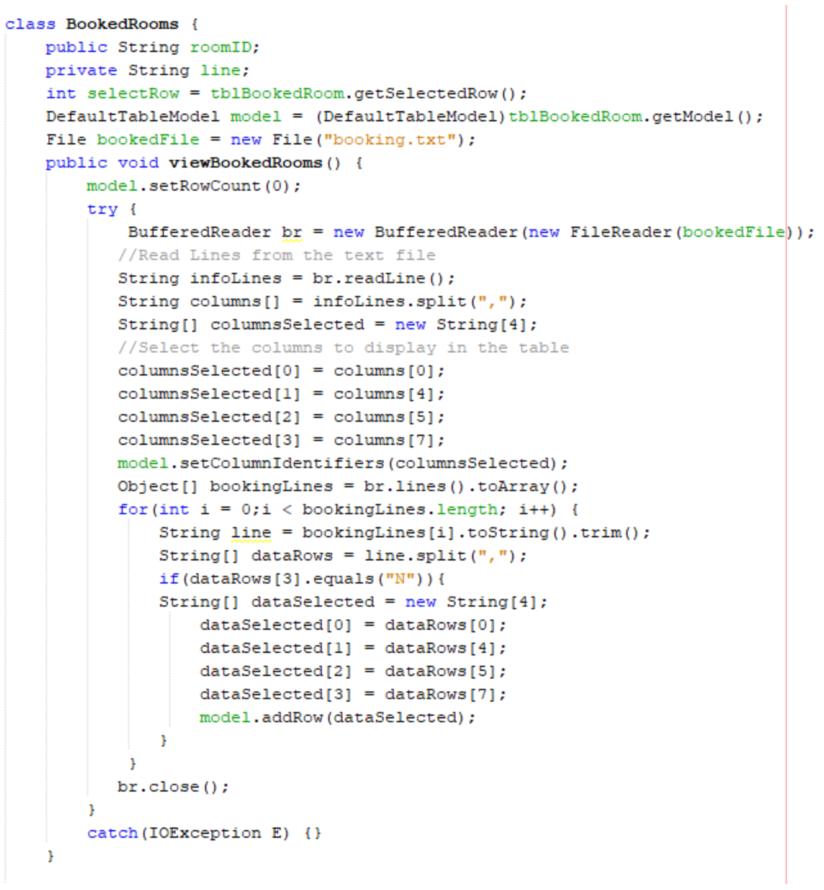
Inheritance is the main concept that supports the reusability of code blocks as it will inherits the features, methods, and properties of another class. We could always derive a new class from an existing class when there are some code blocks that is needed for the new class (Biswal, 2021). Hence, the room booking system have utilized inheritance concept in the source code in order to structure the codes clearly and emphasize on the reusability of the codes.



```java
// Create a new class that inherits from BookedRooms
class Function extends BookedRooms {
    public String getModifiedData() {
        dates objDate = new dates();
        String[] dates = objDate.getDate();
        // Check the validity of check in and check out date and duration
        if (Integer.parseInt(dates[2]) > 7) {
            JOptionPane.showMessageDialog(null, "Unfortunately, The Blueming Resort ONLY allows at most 7-days room booking. "
                    + "Please RESELECT your check in and check out date to book the room successfully",
                    "Room Booking Duration Warning", JOptionPane.WARNING_MESSAGE);
        }
        else if (Integer.parseInt(dates[2]) < 1) {
            JOptionPane.showMessageDialog(null, "Invalid Date. Please confirm that your check out date is after your check in date.",
                    "Invalid Date Warning", JOptionPane.WARNING_MESSAGE);
        }
        else {
            // Ensure all the input fields have a valid input
            if((!txtCustomerName.getText().equals("") && !txtEmail.getText().equals("") && !txtCustomerIC.getText().replaceAll("\\s","").equals("--")) &&
                    !txtContactNumber.getText().trim().equals("-")) {
                // Declare a string with all the information comma-seperated
                String modifiedInfo = txtRoomID.getText() + "," + txtTypeOfRoom.getText() + "," + "350.00," + "N," + dates[0] + "," +
                dates[1] + "," + dates[2] + "," + txtCustomerName.getText() + "," + txtCustomerIC.getText() + "," + cbGender.getSelectedItem().toString()
                + "," + txtContactNumber.getText() + "," + txtEmail.getText() + "," + txtPaymentMethod.getText();
                return modifiedInfo;
            }
            else {
                JOptionPane.showMessageDialog(null,"Please Enter All Modified Details or Click the Data In Table Once More",
                        "Missing Modified Data Warning",JOptionPane.WARNING_MESSAGE);
            }
        }
        String end = null;
        return end;
    }
```

Figure 3.1.3(a): Function Class that Inherits BookedRooms Class

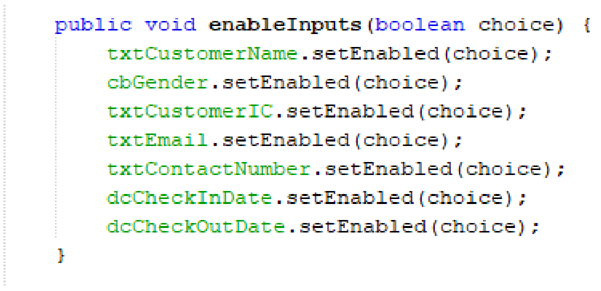
In the Manage Booking / Room Checkout Page, there is a class called “Function” that inherits the “BookedRooms” class as “BookedRooms” class contains some code blocks that could be reuse in the new class. There are a few methods within the “Function” class such as “getModifiedData()”, “search()”, “enabledInputs()”, “checkOut()” and more. When declaring the new class, the “extends” keyword is used to indicate that it inherits the “BookedRooms” class as illustrated in *Figure 3.1.3(a)*. In the “getModifiedData()” method, an object of the “dates” class is created in order to call the “getDate” method that will return the modified check in and check out date. After getting the check in and check out date, there is an if else statement which is used to validate the modified check in and check out date before compiling all information into a string variable.



```
class BookedRooms {
    public String roomID;
    private String line;
    int selectRow = tblBookedRoom.getSelectedRow();
    DefaultTableModel model = (DefaultTableModel)tblBookedRoom.getModel();
    File bookedFile = new File("booking.txt");
    public void viewBookedRooms() {
        model.setRowCount(0);
        try {
            BufferedReader br = new BufferedReader(new FileReader(bookedFile));
           //Read Lines from the text file
           String infoLines = br.readLine();
           String columns[] = infoLines.split(",");
           String[] columnsSelected = new String[4];
           //Select the columns to display in the table
           columnsSelected[0] = columns[0];
           columnsSelected[1] = columns[4];
           columnsSelected[2] = columns[5];
           columnsSelected[3] = columns[7];
           model.setColumnIdentifiers(columnsSelected);
           Object[] bookingLines = br.lines().toArray();
           for(int i = 0;i < bookingLines.length; i++) {
               String line = bookingLines[i].toString().trim();
               String[] dataRows = line.split(",");
               if(dataRows[3].equals("N")){
               String[] dataSelected = new String[4];
                   dataSelected[0] = dataRows[0];
                   dataSelected[1] = dataRows[4];
                   dataSelected[2] = dataRows[5];
                   dataSelected[3] = dataRows[7];
                   model.addRow(dataSelected);
               }
            }
           br.close();
        }
        catch(IOException E) {}
    }
```

Figure 3.1.3(b): BookedRooms Class in Manage Booking / Check Out Page

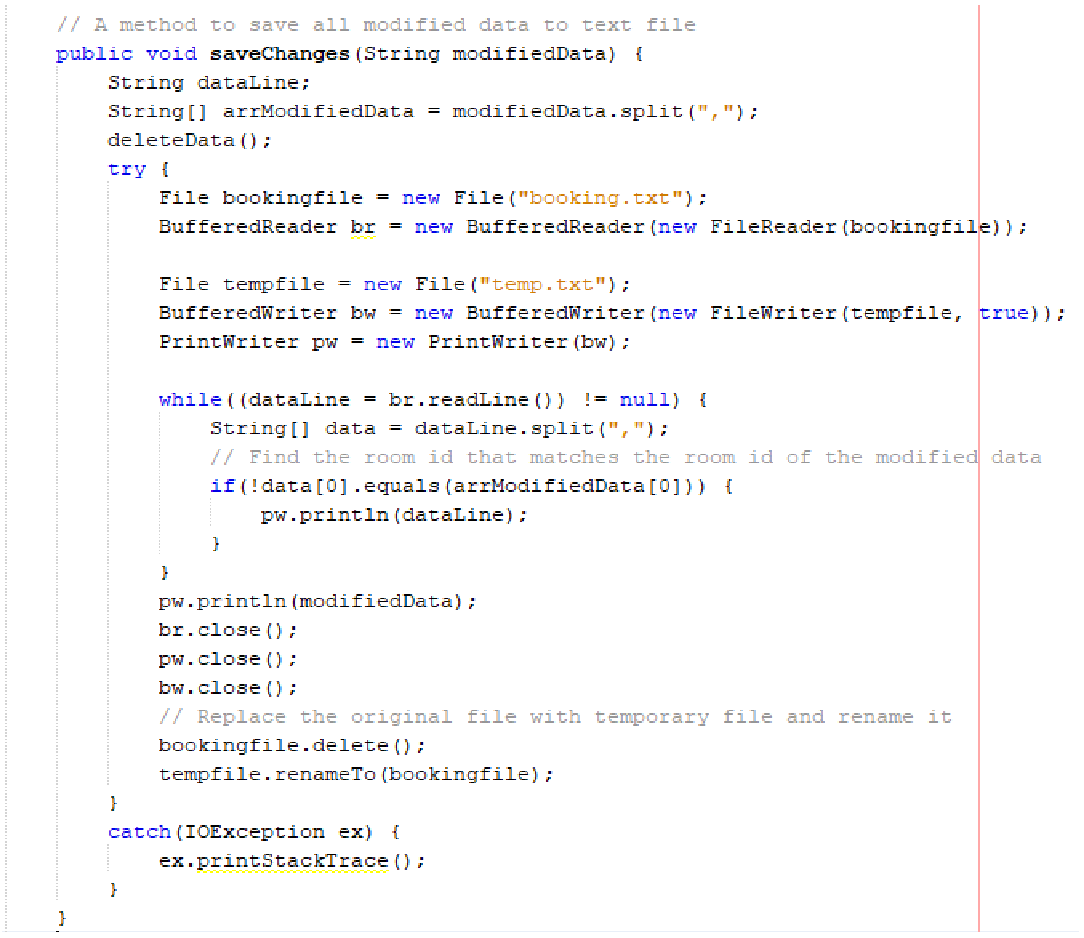
"BookedRooms" class is inherited by the "Function" class, therefore the existing properties and methods could be use by the "Function" class. The current selected table row is stored in a variable as it will be widely used in both classes. Besides, the file path of the booking text file is also stored in a variable so that it could be accessed easily by the class that inherited this class. Furthermore, the existing methods can be accessed without creating an object by the inherited classes. For instance, the "viewBookedRooms()" method is created to import all the information with "N" availability within the booking text file into the table that shows all the booked rooms information as shown in *Figure 3.1.3(b)*.



```
public void enableInputs(boolean choice) {
    txtCustomerName.setEnabled(choice);
    cbGender.setEnabled(choice);
    txtCustomerIC.setEnabled(choice);
    txtEmail.setEnabled(choice);
    txtContactNumber.setEnabled(choice);
    dcCheckInDate.setEnabled(choice);
    dcCheckOutDate.setEnabled(choice);
}
```

Figure 3.1.3(c): enableInputs() Method in Function class

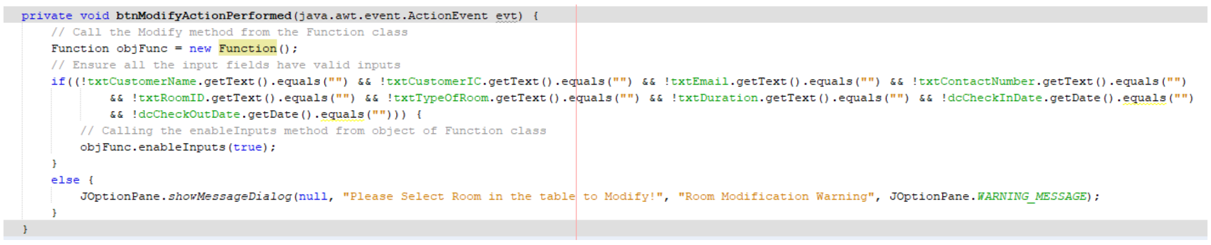
The “enableInputs()” method is located within the “Function” class that enables or disables some of the input fields when it is called as enabled input fields are required for modification purposes. The method will receive a boolean value to identify if the input field is needed to be enabled or disabled as the input fields are disabled in default.



```java
// A method to save all modified data to text file
public void saveChanges(String modifiedData) {
    String dataLine;
    String[] arrModifiedData = modifiedData.split(",");
    deleteData();
    try {
        File bookingfile = new File("booking.txt");
        BufferedReader br = new BufferedReader(new FileReader(bookingfile));

        File tempfile = new File("temp.txt");
        BufferedWriter bw = new BufferedWriter(new FileWriter(tempfile, true));
        PrintWriter pw = new PrintWriter(bw);

        while((dataLine = br.readLine()) != null) {
            String[] data = dataLine.split(",");
            // Find the room id that matches the room id of the modified data
            if(!data[0].equals(arrModifiedData[0])) {
                pw.println(dataLine);
            }
        }
        pw.println(modifiedData);
        br.close();
        pw.close();
        bw.close();
        // Replace the original file with temporary file and rename it
        bookingfile.delete();
        tempfile.renameTo(bookingfile);
    }
    catch(IOException ex) {
        ex.printStackTrace();
    }
}
```

Figure 3.1.3(d): saveChanges() Method in Function class

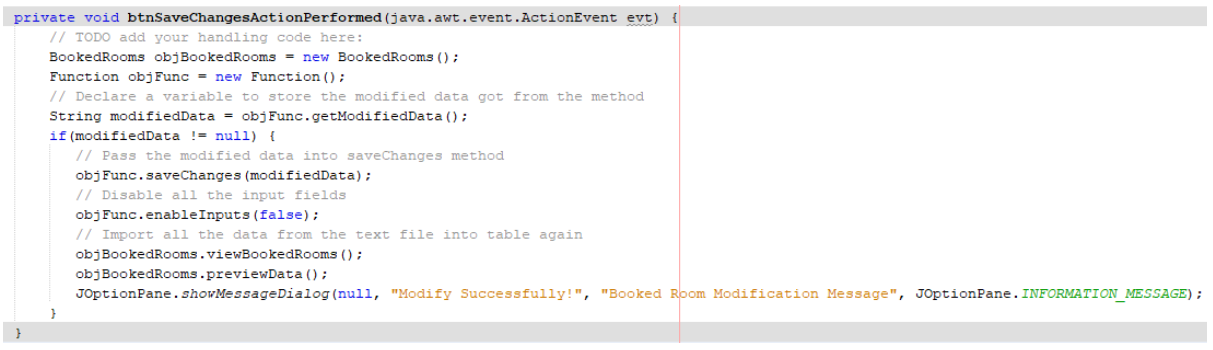
After modifying all the information, staffs are required to click the save changes button in order to save the modified information into the booking text file. According to the method shown in *Figure 3.1.3(d)*, the “saveChanges()” method will receive the modified data as an argument and rewrite the data including the modified data into the booking text file.



```java
private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {
    // Call the Modify method from the Function class
    Function objFunc = new Function();
    // Ensure all the input fields have valid inputs
    if((!txtCustomerName.getText().equals("") && !txtCustomerIC.getText().equals("") && !txtEmail.getText().equals("") && !txtContactNumber.getText().equals("")
            && !txtRoomID.getText().equals("") && !txtTypeOfRoom.getText().equals("") && !txtDuration.getText().equals("") && !dcCheckInDate.getDate().equals("")
            && !dcCheckOutDate.getDate().equals(""))) {
        // Calling the enableInputs method from object of Function class
        objFunc.enableInputs(true);
    }
    else {
        JOptionPane.showMessageDialog(null, "Please Select Room in the table to Modify!", "Room Modification Warning", JOptionPane.WARNING_MESSAGE);
    }
}
```

Figure 3.1.3(e): Calling enableInputs() when Modify Button is Clicked

When the staffs clicked the modify button, the “enabledInputs()” method is called by the object created from the “Function” class. Therefore, the staffs are able to enter the modified information into the text fields.



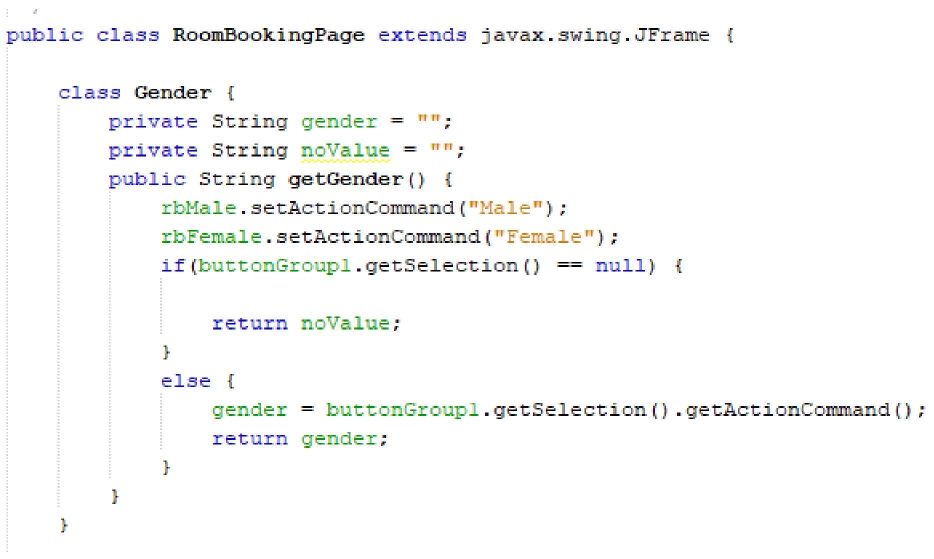
```java
private void btnSaveChangesActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    BookedRooms objBookedRooms = new BookedRooms();
    Function objFunc = new Function();
    // Declare a variable to store the modified data got from the method
    String modifiedData = objFunc.getModifiedData();
    if(modifiedData != null) {
        // Pass the modified data into saveChanges method
        objFunc.saveChanges(modifiedData);
        // Disable all the input fields
        objFunc.enableInputs(false);
        // Import all the data from the text file into table again
        objBookedRooms.viewBookedRooms();
        objBookedRooms.previewData();
        JOptionPane.showMessageDialog(null, "Modify Successfully!", "Booked Room Modification Message", JOptionPane.INFORMATION_MESSAGE);
    }
}
```

Figure 3.1.3(f): Getting the Modified Data & Saving into the Text File

When save changes button is clicked, a variable is declared to store the modified data from the “getModifiedData()” and pass into the “saveChanges()” method if the modified data is not null. After that, the input fields will be disabled, and the booked rooms table will be refreshed by importing the data from the booking text file again.

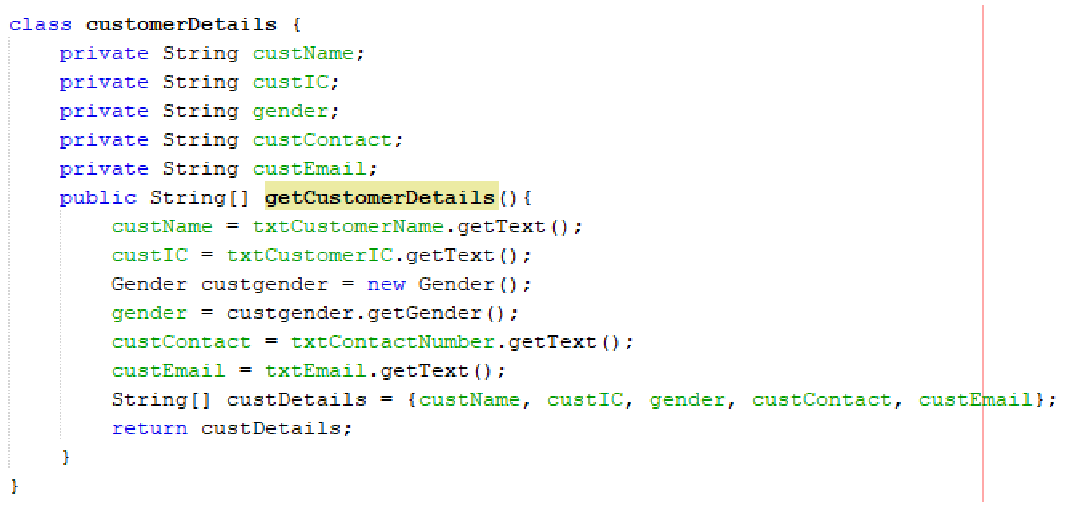
3.1.4 Composition – Room Booking Page (Get Customer Details & Gender)

Composition is considered as a strong association in object-oriented programming. When the parent object is deleted, all of the child objects will not exist automatically too. Therefore, this indicates that there will be no independent existence in a composition relationship (Hartman, 2022). The composition concept is also implemented in the room booking system.



```java
public class RoomBookingPage extends javax.swing.JFrame {

    class Gender {
        private String gender = "";
        private String noValue = "";
        public String getGender() {
            rbMale.setActionCommand("Male");
            rbFemale.setActionCommand("Female");
            if(buttonGroup1.getSelection() == null) {

                return noValue;
            }
            else {
                gender = buttonGroup1.getSelection().getActionCommand();
                return gender;
            }
        }
    }
```

Figure 3.1.4(a): getGender() Method in Gender Class



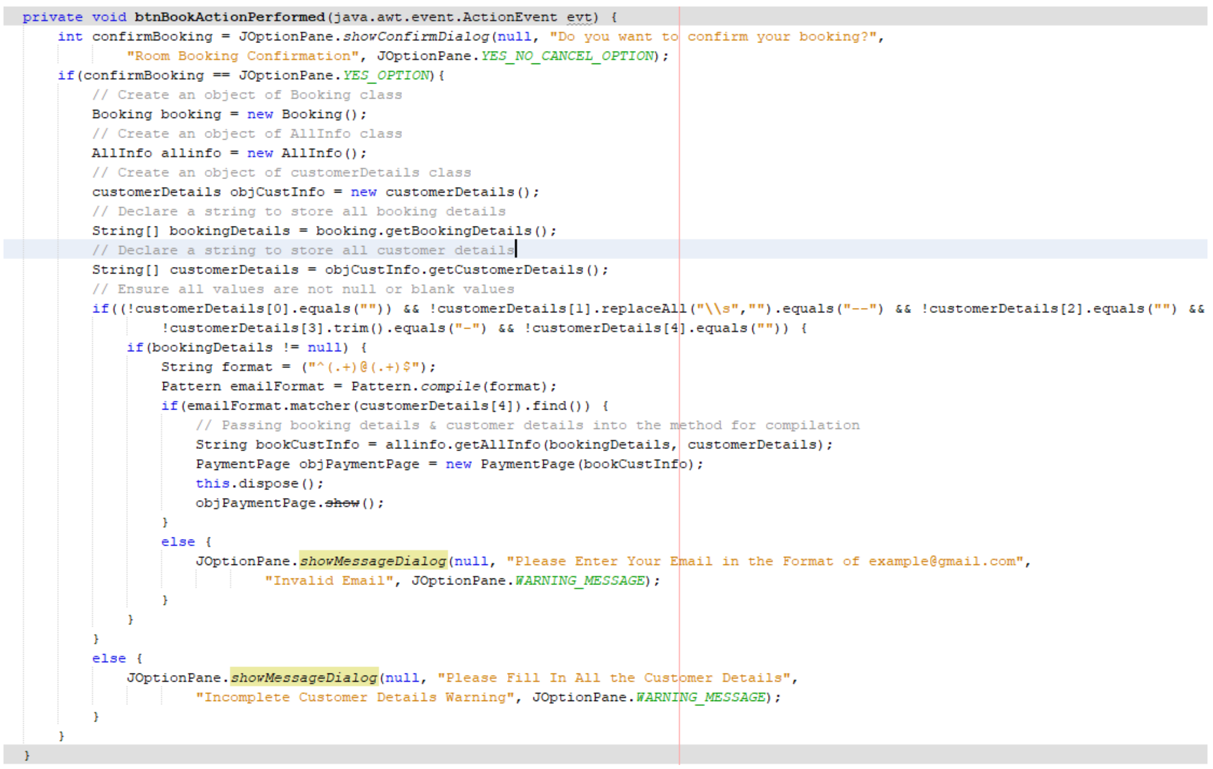
```
class customerDetails {
    private String custName;
    private String custIC;
    private String gender;
    private String custContact;
    private String custEmail;
    public String[] getCustomerDetails(){
        custName = txtCustomerName.getText();
        custIC = txtCustomerIC.getText();
        Gender custgender = new Gender();
        gender = custgender.getGender();
        custContact = txtContactNumber.getText();
        custEmail = txtEmail.getText();
        String[] custDetails = {custName, custIC, gender, custContact, custEmail};
        return custDetails;
    }
}
```

Figure 3.1.4(b): Composition between Gender and customerDetails Class

For example, one of the details that is required by the customer is their gender which can acquired by using the "getGender()" method in the "Gender" class. The object of "Gender" class is created in the "getCustomerDetails()" method in order to call the "getGender()" method as shown in *Figure 3.1.4(b)*. The "Gender" class is only accessed by the "customerDetails" class in the whole system. Therefore, it is considered as composition because if we can't get the customer details, the customer gender will not exist too.

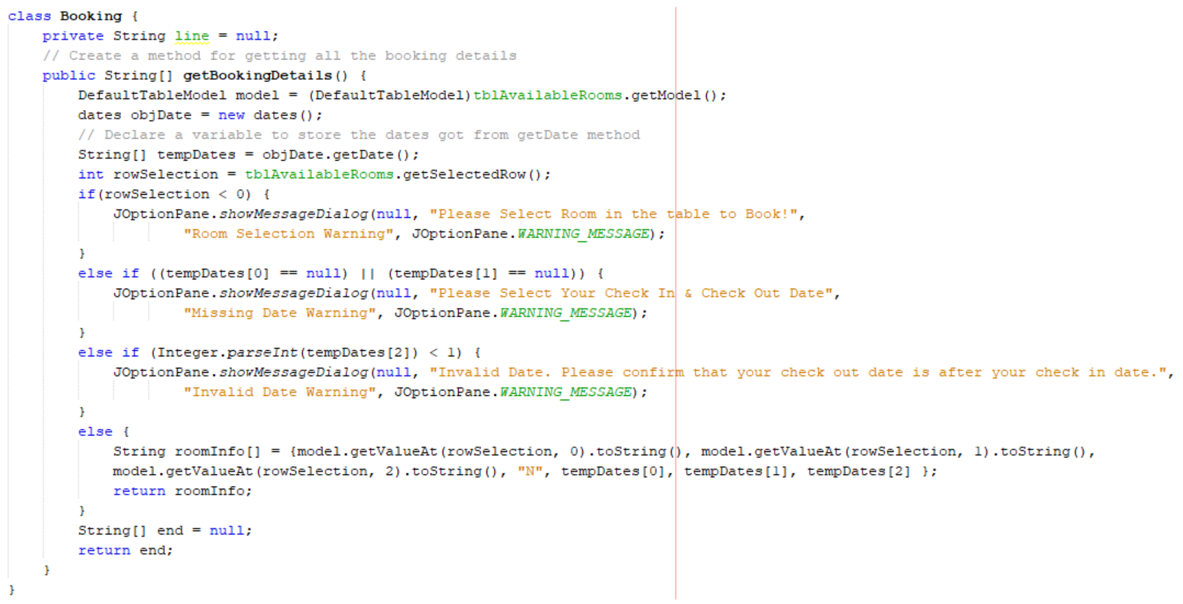
3.1.5 Aggregation – Room Booking Page (Booking)

Aggregation is known as a weak relationship between objects. Although the object is a part of another object, but the object will still exist if the other object is deleted (Bhattacharjee, 2020). The aggregation association is implemented in the room booking system too.



```java
private void btnBookActionPerformed(java.awt.event.ActionEvent evt) {
    int confirmBooking = JOptionPane.showConfirmDialog(null, "Do you want to confirm your booking?",
            "Room Booking Confirmation", JOptionPane.YES_NO_CANCEL_OPTION);
    if(confirmBooking == JOptionPane.YES_OPTION){
        // Create an object of Booking class
        Booking booking = new Booking();
        // Create an object of AllInfo class
        AllInfo allinfo = new AllInfo();
        // Create an object of customerDetails class
        customerDetails objCustInfo = new customerDetails();
        // Declare a string to store all booking details
        String[] bookingDetails = booking.getBookingDetails();
        // Declare a string to store all customer details
        String[] customerDetails = objCustInfo.getCustomerDetails();
        // Ensure all values are not null or blank values
        if((!customerDetails[0].equals("")) && !customerDetails[1].replaceAll("\\s","").equals("--") && !customerDetails[2].equals("") &&
                !customerDetails[3].trim().equals("-") && !customerDetails[4].equals("")) {
            if(bookingDetails != null) {
                String format = ("^(.+)@(.+)$");
                Pattern emailFormat = Pattern.compile(format);
                if(emailFormat.matcher(customerDetails[4]).find()) {
                    // Passing booking details & customer details into the method for compilation
                    String bookCustInfo = allinfo.getAllInfo(bookingDetails, customerDetails);
                    PaymentPage objPaymentPage = new PaymentPage(bookCustInfo);
                    this.dispose();
                    objPaymentPage.show();
                }
                else {
                    JOptionPane.showMessageDialog(null, "Please Enter Your Email in the Format of example@gmail.com",
                            "Invalid Email", JOptionPane.WARNING_MESSAGE);
                }
            }
        }
        else {
            JOptionPane.showMessageDialog(null, "Please Fill In All the Customer Details",
                    "Incomplete Customer Details Warning", JOptionPane.WARNING_MESSAGE);
        }
    }
}
```

Figure 3.1.5(a): Aggregation in Getting Booking Details and Customer Details



```java
class Booking {
    private String line = null;
    // Create a method for getting all the booking details
    public String[] getBookingDetails() {
        DefaultTableModel model = (DefaultTableModel)tblAvailableRooms.getModel();
        dates objDate = new dates();
        // Declare a variable to store the dates got from getDate method
        String[] tempDates = objDate.getDate();
        int rowSelection = tblAvailableRooms.getSelectedRow();
        if(rowSelection < 0) {
            JOptionPane.showMessageDialog(null, "Please Select Room in the table to Book!",
                    "Room Selection Warning", JOptionPane.WARNING_MESSAGE);
        }
        else if ((tempDates[0] == null) || (tempDates[1] == null)) {
            JOptionPane.showMessageDialog(null, "Please Select Your Check In & Check Out Date",
                    "Missing Date Warning", JOptionPane.WARNING_MESSAGE);
        }
        else if (Integer.parseInt(tempDates[2]) < 1) {
            JOptionPane.showMessageDialog(null, "Invalid Date. Please confirm that your check out date is after your check in date.",
                    "Invalid Date Warning", JOptionPane.WARNING_MESSAGE);
        }
        else {
            String roomInfo[] = {model.getValueAt(rowSelection, 0).toString(), model.getValueAt(rowSelection, 1).toString(),
            model.getValueAt(rowSelection, 2).toString(), "N", tempDates[0], tempDates[1], tempDates[2] };
            return roomInfo;
        }
        String[] end = null;
        return end;
    }
}
```

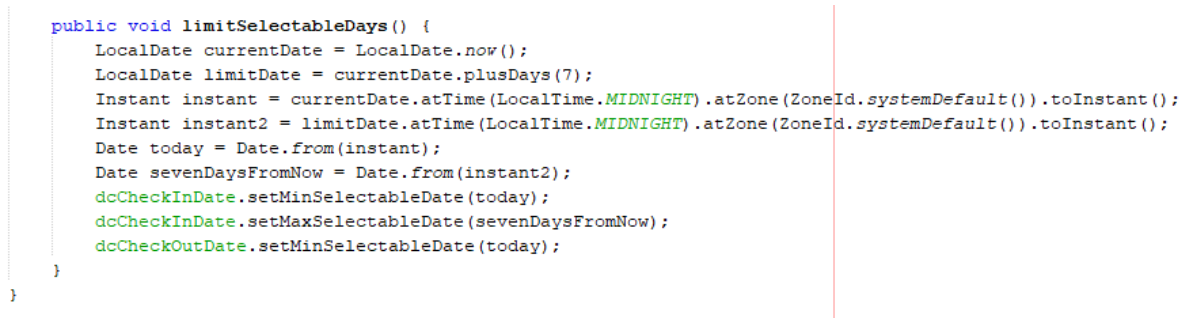
Figure 3.1.5(b): getBookingDetails() Method in Booking class

From the source code shown in *Figure 3.1.5(a)*, there are three objects being created for “Booking” class, “AllInfo” class and “customerDetails” class. The program will retrieve the booking details by calling the “getBookingDetails()” method and customer details by calling the “getCustomerDetails()” method. After that, both information will be passed into a method called “getAllInfo()” to compile all the information together and pass it to the payment page.

In this case, the booking details will still exist if the customer details are deleted for some reasons as they are having aggregation association.

3.2 Source Code Explanation

3.2.1 Limit Selectable Date on JDateChooser

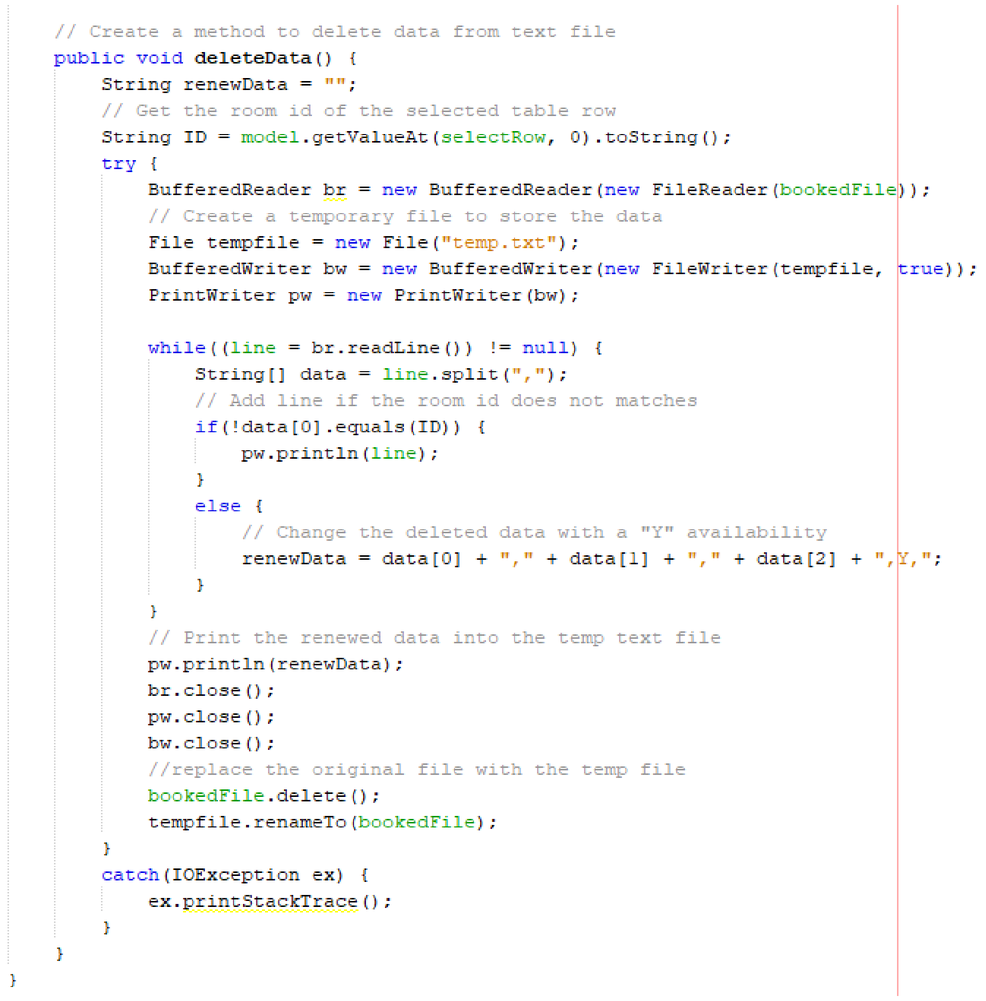


```java
    public void limitSelectableDays() {
        LocalDate currentDate = LocalDate.now();
        LocalDate limitDate = currentDate.plusDays(7);
        Instant instant = currentDate.atTime(LocalTime.MIDNIGHT).atZone(ZoneId.systemDefault()).toInstant();
        Instant instant2 = limitDate.atTime(LocalTime.MIDNIGHT).atZone(ZoneId.systemDefault()).toInstant();
        Date today = Date.from(instant);
        Date sevenDaysFromNow = Date.from(instant2);
        dcCheckInDate.setMinSelectableDate(today);
        dcCheckInDate.setMaxSelectableDate(sevenDaysFromNow);
        dcCheckOutDate.setMinSelectableDate(today);
    }
}
```

Figure 3.2.1(a): limitSelectableDays() Day Method in Booking Page

As mentioned in assumption section, the customer is able to book within 7 days from the current date. Therefore, the limitSelectableDays() method is created to block all the dates in JDateChooser which is not within 7 days from the current date. From the code block shown in *Figure 3.2.1(a)*, the system has called several methods in order to acquire the current date and the date after 7 days from now. After that, the check in and check out JDateChooser is configured with the minimum and maximum selectable date to limit staffs on choosing the check in and check out dates.

3.2.2 Delete in Manage Booking / Room Check Out Page

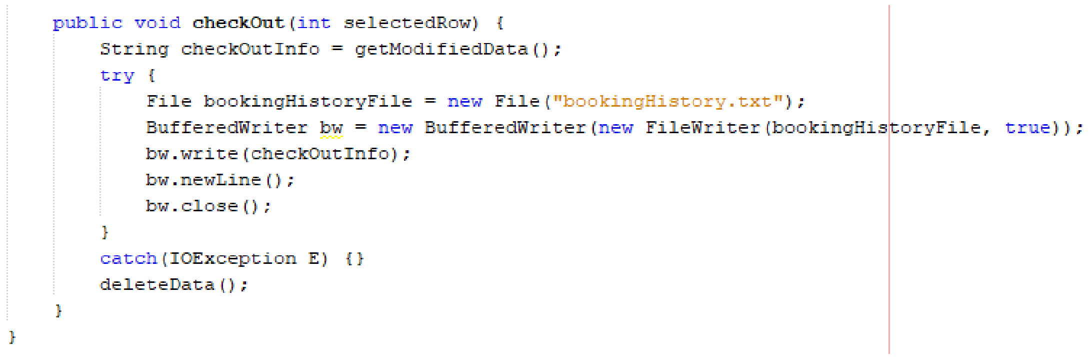


```
    // Create a method to delete data from text file
    public void deleteData() {
        String renewData = "";
        // Get the room id of the selected table row
        String ID = model.getValueAt(selectRow, 0).toString();
        try {
            BufferedReader br = new BufferedReader(new FileReader(bookedFile));
            // Create a temporary file to store the data
            File tempfile = new File("temp.txt");
            BufferedWriter bw = new BufferedWriter(new FileWriter(tempfile, true));
            PrintWriter pw = new PrintWriter(bw);

            while((line = br.readLine()) != null) {
                String[] data = line.split(",");
                // Add line if the room id does not matches
                if(!data[0].equals(ID)) {
                    pw.println(line);
                }
                else {
                    // Change the deleted data with a "Y" availability
                    renewData = data[0] + "," + data[1] + "," + data[2] + ",Y,";
                }
            }
            // Print the renewed data into the temp text file
            pw.println(renewData);
            br.close();
            pw.close();
            bw.close();
            //replace the original file with the temp file
            bookedFile.delete();
            tempfile.renameTo(bookedFile);
        }
        catch(IOException ex) {
            ex.printStackTrace();
        }
    }
}
```

Figure 3.2.2(a): deleteData() Method in Manage Booking / Room Check Out Page

For the delete function in the room booking system, it is mainly used for staff to delete the booking and customer information when the customer requested to cancel his/her booking. Firstly, the room id of the selected table row is stored in a string variable as it will be used for finding the specific data in the booking text file. The existing data in the booking text file will be copied to a temporary file except for the selected row data. Then, the selected row data will be renewed by changing its' availability to "Y" and wrote into the temporary text file. Lastly, the original booking text file will be deleted, and the temporary text file will be renamed as booking text file. Hence, the data deletion is completed by utilizing the file handling technique.

3.2.3 Check Out in Manage Booking / Room Check Out Page



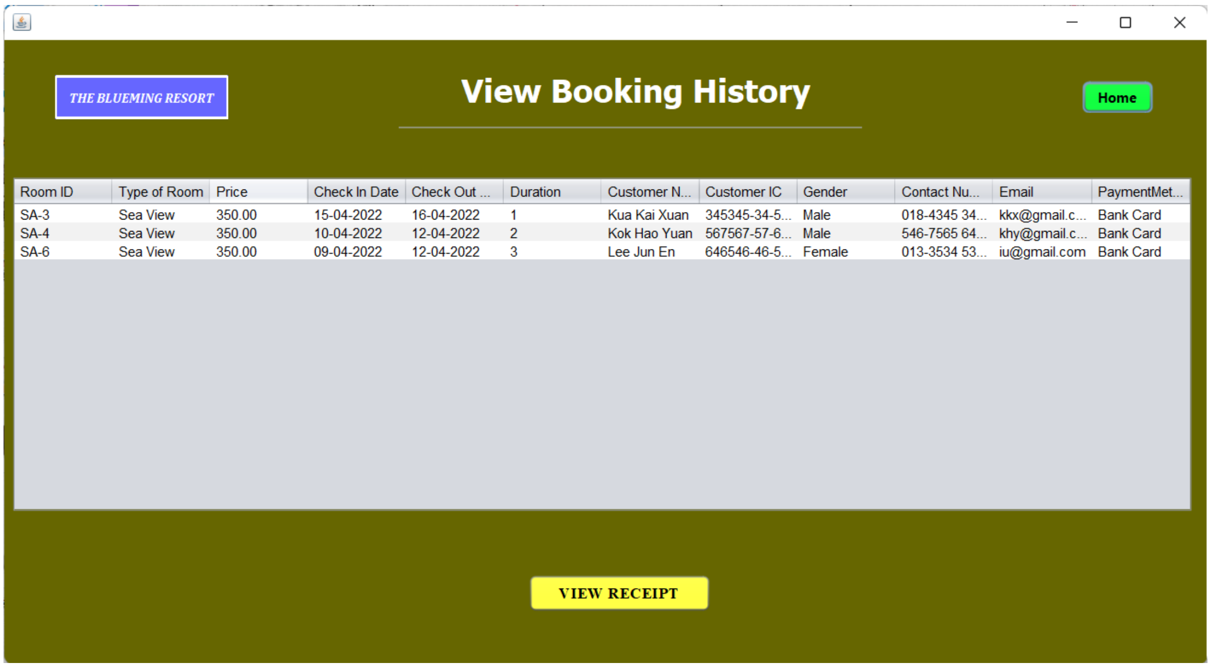
```java
    public void checkOut(int selectedRow) {
        String checkOutInfo = getModifiedData();
        try {
            File bookingHistoryFile = new File("bookingHistory.txt");
            BufferedWriter bw = new BufferedWriter(new FileWriter(bookingHistoryFile, true));
            bw.write(checkOutInfo);
            bw.newLine();
            bw.close();
        }
        catch(IOException E) {}
        deleteData();
    }
}
```

Figure 3.2.3(a): checkOut() method in Manage Booking / Room Check Out Page

When the staff wanted to check out the room, the system will acquire the room information by calling the “getModifiedData()” method in order to save the booking data into a text file act as a booking history database. After saving all the information of the room that is checking out, the specific room data will be deleted from the booking text file and the booked room table by calling the “deleteData” method as shown in *Figure 3.2.2(a)* and the room will become available for booking.

4.0 Additional Features of The Room Booking System

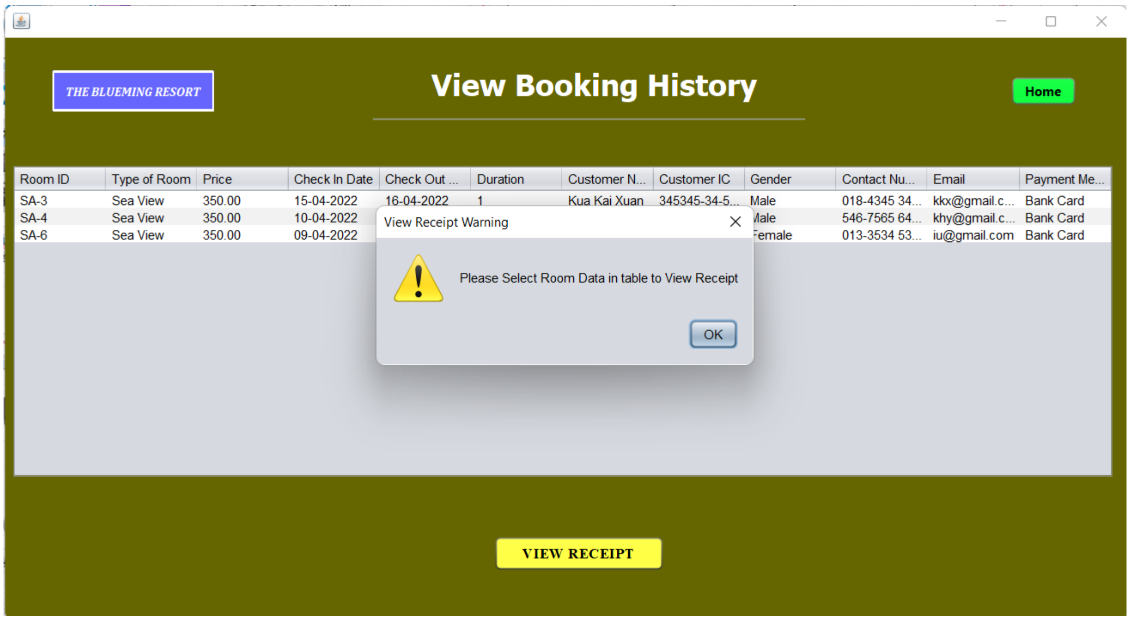
4.1 View Booking History



Room ID	Type of Room	Price	Check In Date	Check Out ...	Duration	Customer N...	Customer IC	Gender	Contact Nu...	Email	PaymentMet...
SA-3	Sea View	350.00	15-04-2022	16-04-2022	1	Kua Kai Xuan	345345-34-5...	Male	018-4345 34...	kkx@gmail.c...	Bank Card
SA-4	Sea View	350.00	10-04-2022	12-04-2022	2	Kok Hao Yuan	567567-57-6...	Male	546-7565 64...	khy@gmail.c...	Bank Card
SA-6	Sea View	350.00	09-04-2022	12-04-2022	3	Lee Jun En	646546-46-5...	Female	013-3534 53...	iu@gmail.com	Bank Card

Figure 4.1(a): View Booking History Page Output

Staffs are able to access to the view booking history page from home page. The view booking history page contains a table that consists of information such as room id, room type, check in and out date, customer name, payment method and more as illustrated in *Figure 4.1(a)*.



Room ID	Type of Room	Price	Check In Date	Check Out ...	Duration	Customer N...	Customer IC	Gender	Contact Nu...	Email	Payment Me...
SA-3	Sea View	350.00	15-04-2022	16-04-2022	1	Kua Kai Xuan	345345-34-5...	Male	018-4345 34...	kkx@gmail.c...	Bank Card
SA-4	Sea View	350.00	10-04-2022					Male	546-7565 64...	khy@gmail.c...	Bank Card
SA-6	Sea View	350.00	09-04-2022					Female	013-3534 53...	iu@gmail.com	Bank Card

View Receipt Warning

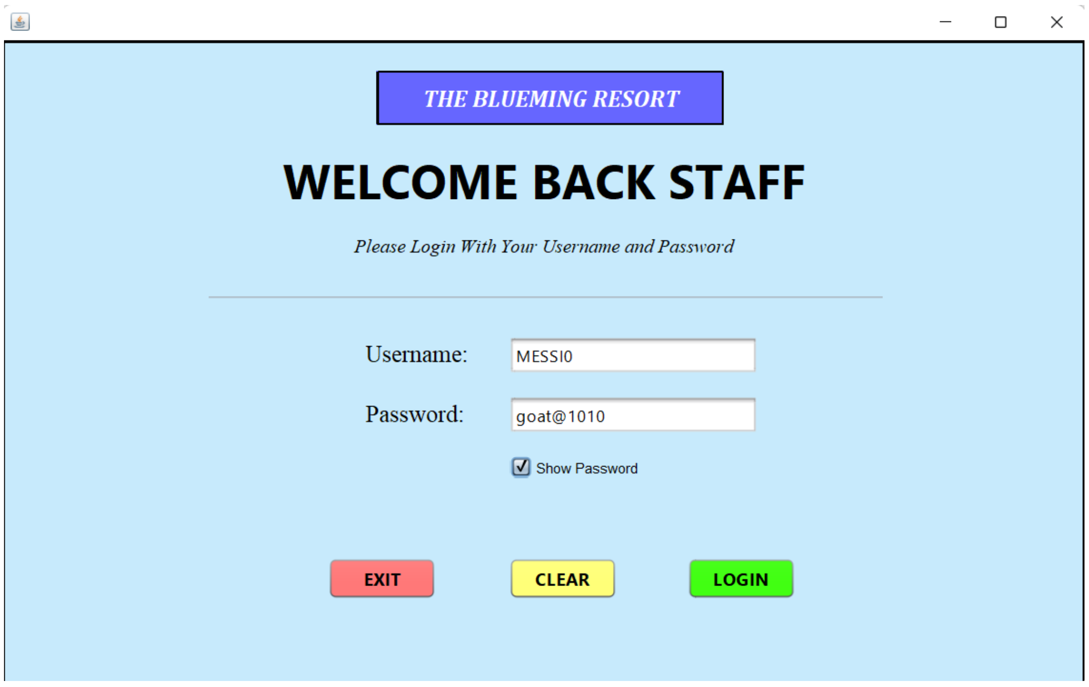
Please Select Room Data in table to View Receipt

OK

Figure 4.1(b): Missing Selected Data to View Receipt

If there is no data row selected from the table, the system will display a view receipt warning shown in *Figure 4.1(b)* as the system could not identify the booking history that the staff wanted to view. The staff should always select a specific data row to view the receipt. After clicking the view receipt button, the staff will be navigated to the receipt page with the relevant information same as *Figure 2.5(a)*.

4.2 Show Password in Login Page



Username: MESSI0

Password: goat@1010

☑ Show Password

Figure 4.2(a): Show Password Function in Login Page

The show password function is an additional feature provided for the staff as sometimes the staff might want to double-check his/her password before logging into the system. The password field originally is encrypted, and it will decrypt when the show password is checked as illustrated in *Figure 4.2(a)*.

5.0 Conclusion

To put it succinctly, the room booking system is developed in Java as it is one of the common object-oriented programming languages which helps in code reusability, easier maintenance, and development and more. By utilizing different functions available in the system, staffs could book and manage all the room bookings in a more efficient and effective way which will increase the satisfaction and retention rate of customers. Besides, all the booking history will be stored and recorded by using the system as it might be used to generate insights or reports from it. In short, the newly developed room booking system consists of advantages that would boost and increase the performance of the resort’s business.

6.0 References

Bhattacharjee, D. (2020, December 31). *Object-Oriented-Programming Concepts in Java*. Retrieved from Baeldung: https://www.baeldung.com/java-oop

Biswal, S. (2021, June 10). *OOPs Concept in Java*. Retrieved from SOAIS: https://www.soais.com/oops-concept-in-java/

Great Learning Team. (2022, February 5). *OOPs concepts in Java | What is OOPs in Java?* Retrieved from Great Learning: https://www.mygreatlearning.com/blog/oops-concepts-in-java/#Encapsulation

Hartman, J. (2022, February 19). *OOPs Concepts in Java | What is, Basics with Examples*. Retrieved from Guru99: https://www.guru99.com/java-oops-concept.html