OCR GCE A

COMPUTER SCIENCE PROJECT

H446-03

# Name : <MABROOR AHMED >

Candidate Number : <190064>

<Institution Name> : <MABROOR AHMED>

Title of Project : <Booking system >

H446-03 – Project CONTENTS

Table of Contents

[Name : <MABROOR AHMED > 1](#_Toc67671730)

[A. Analysis 5](#_Toc67671731)

[Features that make my problem solvable by computational methods 5](#_Toc67671732)

[Why should it be solved using computational methods? 5](#_Toc67671733)

[Suitable Stakeholders 5](#_Toc67671734)

[Existing Solutions 6](#_Toc67671735)

[Researching the problem in depth 8](#_Toc67671736)

[Interview time 8](#_Toc67671737)

[Interview questions 8](#_Toc67671738)

[Interview answers 9](#_Toc67671739)

[Observations/what I will implement. 10](#_Toc67671740)

[Success criteria 10](#_Toc67671741)

[Limitations 11](#_Toc67671742)

[B. Design 11](#_Toc67671743)

[Problem decomposition 11](#_Toc67671744)

[Variable identification 13](#_Toc67671745)

[Usabilitiy and Features 14](#_Toc67671746)

[Login page 14](#_Toc67671747)

[Register page 15](#_Toc67671748)

[Main page 16](#_Toc67671749)

[Class DIagram 17](#_Toc67671750)

[Post-development Testing 18](#_Toc67671751)

[C. Developing the coded solution (“The development story”) 22](#_Toc67671752)

[Prototype 1 23](#_Toc67671753)

[Prototype 1 Overview 23](#_Toc67671754)

[Test plan 24](#_Toc67671755)

[setting up the database 25](#_Toc67671756)

[Prototype 2 26](#_Toc67671757)

[Problem 26](#_Toc67671758)

[Solution: 27](#_Toc67671759)

[Prototype 2 overview 30](#_Toc67671760)

[Test plan 30](#_Toc67671761)

[Prototype 3 33](#_Toc67671762)

[Main page planning 33](#_Toc67671763)

[Main page 34](#_Toc67671764)

[Overview 35](#_Toc67671765)

[test plan 36](#_Toc67671766)

[overview 40](#_Toc67671767)

[Post-Development Testing 42](#_Toc67671768)

[Test for functionalities and robustness 42](#_Toc67671769)

[Functionality - Stakeholder 43](#_Toc67671770)

[Usability Testing 45](#_Toc67671771)

[D. Evaluation 49](#_Toc67671772)

[Success criteria 49](#_Toc67671773)

[usability Features 52](#_Toc67671774)

[Maintenance 53](#_Toc67671775)

[Limitations and Future Development 54](#_Toc67671776)

[Project Appendixes 55](#_Toc67671777)

[Design 55](#_Toc67671778)

[Login button(on the login page) 56](#_Toc67671779)

[Reset button(on the login page) 57](#_Toc67671780)

[Register button. (on the login page) 58](#_Toc67671781)

[Register button(on the register page) 58](#_Toc67671782)

[Back button(on the register page) 59](#_Toc67671783)

[Booking button(on main page) 59](#_Toc67671784)

[Manage rooms button(on main page) 60](#_Toc67671785)

[Add room button(on Room management page) 60](#_Toc67671786)

[**edit button(on Room management page)** 62](#_Toc67671787)

[Delete button(on Room management page) 63](#_Toc67671788)

[Back button(on Room management page) 64](#_Toc67671789)

[Code for filling the Jtable 65](#_Toc67671790)

[Book button (on the booking page) 66](#_Toc67671791)

[edit button 70](#_Toc67671792)

[add new room button 71](#_Toc67671793)

# A. Analysis

## Features that make my problem solvable by computational methods

Hotels have been using record books to store information about customers for a long time however with the increase of customers and the addition of new rooms it is hard to keep track of data. The process of finding out which rooms are booked and the ones which are not booked is a time-consuming process. Data also had to be recorded by hand with multiple customers in line there is a build-up of customer traffic. This problem can be solved by computational methods with the use of a data base which can store information and with the use of SQL data can be manipulated easily and efficiently. As algorithms can also be used to filter rooms and bookings. This would allow a fast and easy way to check which bookings have been made or which rooms are currently occupied. Bookings can simply be made with a click of a button to tick rooms in use.

## Why should it be solved using computational methods?

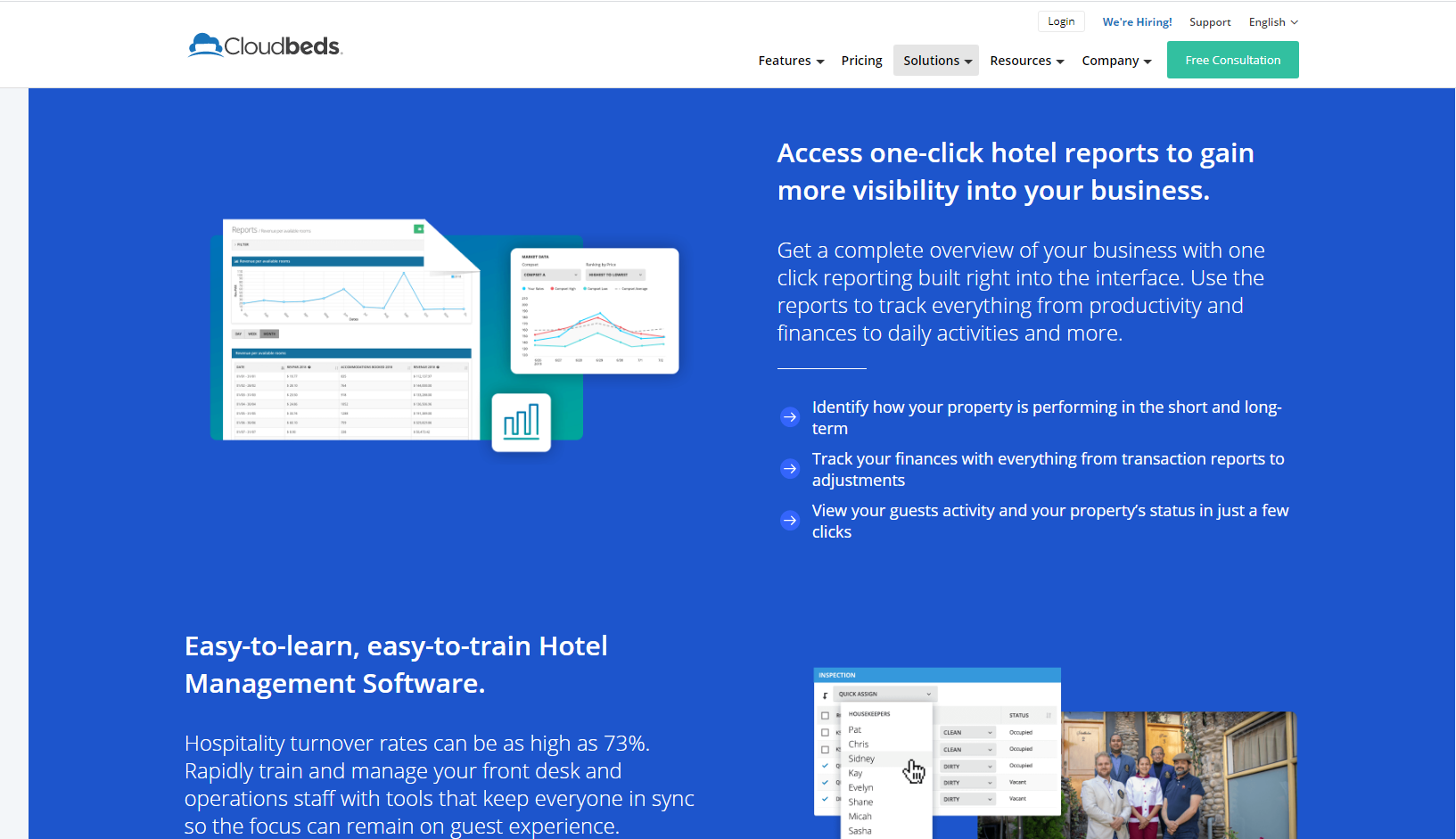
This problem should be solved using computational methods as databases with SQL statements are well suited for a hotel booking system. A database could be used to hold the booking information, the program will be made for employees where they will be able to make bookings for customers, this removes the problem of employees constantly marking rooms as occupied or not. A login system could also be created as customers information will also be stored in the database ;this will protect the customers private details .The rooms would be filtered which will help in the navigation of rooms saves the time of looking through the rooms one by one. The rooms would be assigned to a maximum number of customers, this will remove the error of hotel company employees accidently assigning multiple customers the same room.

## Suitable Stakeholders

The stakeholders for this solution would be hotel companies and their employees. The program will aid the workers with making bookings easier and with less errors, the employees can quickly make bookings with the help of queries filtering of bookings will allow faster booking times. For the customers makings bookings will also be fast and easy as it can be done from anywhere with a phone call. This booking system will decrease the number of errors in bookings and the loss of data as the bookings will not be written on paper which can be destroyed or lost easily.

## Existing Solutions

Solution 1



Overview:

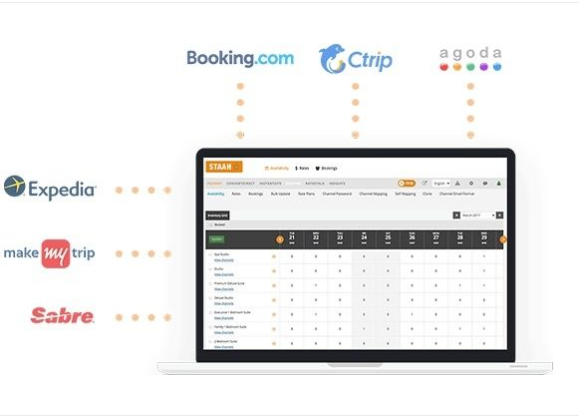
Cloudbeds allows hotels to you to manage the customers stay with the use of online bookings. One of the features of Cloudbeds is you are presented with an overview of your business with one click reporting. This is used to track productivity of the customers and it is used to also track the company’s finances. Another feature of Cloudbeds is the Over the air (OTA) channels are kept up to date. This use of third party allows different rooms prices to be compared preventing overbooking. Cloudbeds allows fast and easy bookings with the company keeping track of what rooms have been booked and by who the room is booked

what I will use for my solution

I will have an overview of the rooms to keep track of the rooms which are booked and the rooms which are free, as Cloudbeds have an overview of the business. I will separate the sections into different rooms which are on different floors and different sizes of.

Solution 2





Overview

STAAH is another existing solution which uses integration of OTA channels to reach a wider audience making it simple for customers as they are directly sent to the bookings page. STAAH uses cloud ensuring that data will always be backed up and it is ideal for properties of all sizes. Cloud storage is a great solution for large companies as storage is not a problem. The hotel booking engine is an easy to use and simple booking system with a variety of features system, for example there are many ways payments can be made including vouchers. This attracts customers as bookings are fast and they do not have to deal with any staff. The solution can also be used on multiple devices . The use of customer feedback helps incoming customers to filter out hotels they would like to stay at and make bookings for.

Parts I could use on my solution

I could use filtering which would make it easy for staff to see which rooms are occupied and which aren’t .

## Researching the problem in depth

Researching the problem in depth I need to get more information on how bookings were made previously and how the booking system worked before. I will carry out an interview with a manager of a hotel with questions such as “how are bookings currently made” etc.

## Interview time

An interview will be taking place on the 26st of January 2020 with the manager of the hotel. The interview will be conducted online with the manager and the aim is to conduct the interview that has been planned to be able to understand the old booking system and the problems related to it. This interview should allow me to learn what parts of the current method need improving and will give me ideas for my program.

## Interview questions

|  |  |
| --- | --- |
| **Question** | **Reason for asking** |
|  |
| How does the system create bookings? | This question is to understand what the hotel currently does to create bookings and how the time can be decreased. |
| What room types exist? | To be able to create a booking system we need to know what type of rooms exist so those can be added to the database. |
| Are customers information kept after they have left ? | This is to find out what kind of information needs to be stored and the period it needs to be stored for |
| What type of information is held in the booking system? | This will tell me what information I need to store in my solution. |
| How do employees keep track of the rooms which are booked and the ones which are not booked? | This will help me find a better way of the method which is currently used |
| Is there a way that the booking system is protected so only staff can access it? | This is to check the security of the system right now  If there is no way to ensure security, a login can be added to the system to ensure security |

## Interview answers

* Currently we keep logbooks of each room and write down customer names under the correct time to create a booking.
* There are single double and king size rooms.
* No, the books are not kept after the customers have left.
* The most booked rooms, how much the rooms cost, and which rooms are under maintenance.
* Checking which rooms are booked and which are not is noted with a column titled “booked” if there a date written next to the room that means it is booked until then.
* The booking system is all in books and they are kept in a safe place where only workers can access them, however there is no other way it can be kept safe.

## Observations/what I will implement.

The current hotel booking system is mostly on paper, meaning saving customers information is made difficult as with time files will pile up, however I can use a data base which is more organized, and the files are more safe. whereas the current system there is only one way to keep the files from being accessed, which is to keep the books safe. The customers information would be safe in a data base as I will make login system only allowing employees to access the information keeping it safe from unauthorized access. With a data base system, it would also be easier to access information and will be easier therefore to check which rooms are booked and which are not booked. When customers are booking rooms, they will also know the different booking prices which are allocated to different types of rooms.

## Success criteria

* Create a login system where staff and customers can login to
* Create a Register system if the customers or staff do not already have a login.
* Login system has a menu making it easier for people to navigate through the page.
* Create a page where staff can add new Rooms.
* staff can change Edit and delete the rooms.
* a system to filter our different type of rooms .
* button to book a room which is currently free.
* A receipt made for the customers and staff printing out what room is booked and the type of room it is.

## Limitations

The limitation I will mainly face is the time I have available to complete this project, as I will need to finish planning around January, so to adapt to this situation I will keep my design very minimal. Though booking online would be faster and more efficient some people for example the old generation might find it difficult to navigate through the online booking. Another limitation is if internet is not available the customer would not be able to use the online booking system.

# B. Design

<See H446-03 Project Advice Booklet for help and guidance of what must go here.>

## Problem decomposition

* The main page will have two option with the options being customer login and staff login, with the information being stored into customer and staff database.
* The staff will have access to the customer data bases as well as the staff information, this is so staff can also access information about customers to for example see which rooms have been books and for how long the rooms they have booked for.
* There is a bookings page as the customer logs in which then the rooms are available to be booked.
* The rooms will be different prices as there are different sizes.

## Variable identification

|  |  |  |
| --- | --- | --- |
| **Input** | **Process** | **Output** |
| Login detail  Room information  ‎‏‏‎ ‎‏‏‎ ‎‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎  ‎‏‏‎ ‎‏‏‎ ‎‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ | Login details will be stored in a database  Booking a room  Editing a room  Deleting a room | Confirmation if the details are properly stored or not  Room information will be used when booking are made and login detail will be assigned to a room |

User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **variable** | **type** | **identifier** | **description** | **validation** |
| Username  password  ‎‏‏‎ ‎‏‏‎ ‎‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ | String  String | username  Password | Stores the name of user.  Stores the password of the staff. | Should only be string.  Should only be string. |

Room

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **variable** | **type** | **identifier** | **description** | **validation** |
| type  number  Reserved  ‎‏‏  ‎‏‏‎ ‎‏‏‎ ‎‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎  ‎‏‏‎ ‎‏‏‎ ‎‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏‏‎ ‎ ‏ | String  integer  Boolean | Roomtype  Roomnumber  Reserved | Stores the different types of rooms.  Stores the number of the rooms which are different for every room.  Stores if a room is booked or not | Should only be string.  Should only be numbers.  Should only be yes or no. |

## Usabilitiy and Features

### Login page

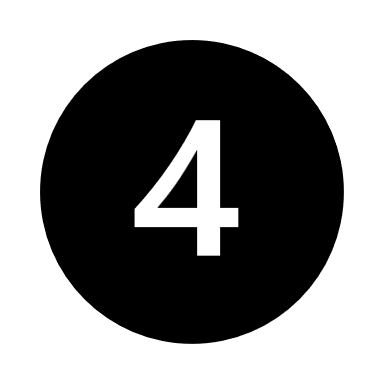
Login



Username



Password



reset

Login

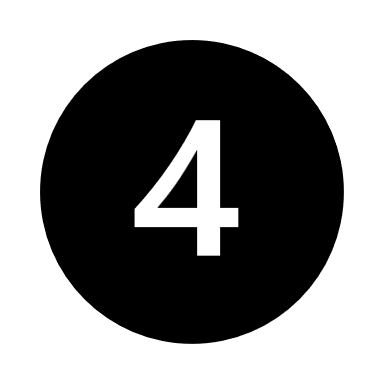


The username field to get the username of the worker logging in the system.



The password field to get the password of the user logging in the system.



The login button which will check if the username and password match any which are in the system ,depending on that the system will respond with directing the user to the main menu or sending a error message.

This button will clear the field so it is made easy for users to renter their details.

### Register page

Register



Username



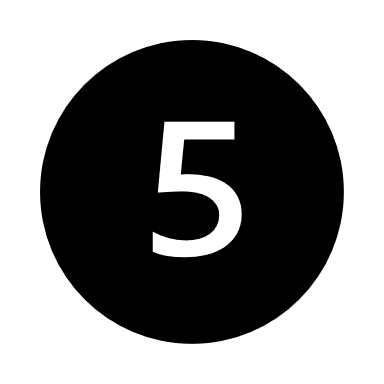
Password

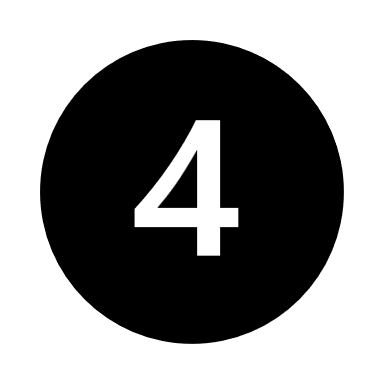


reset

Back

register







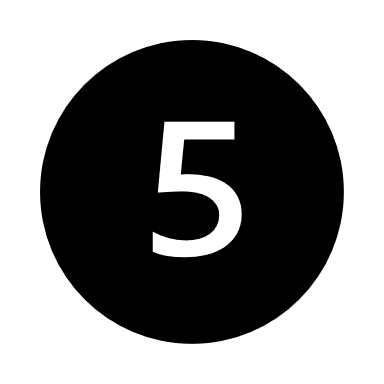
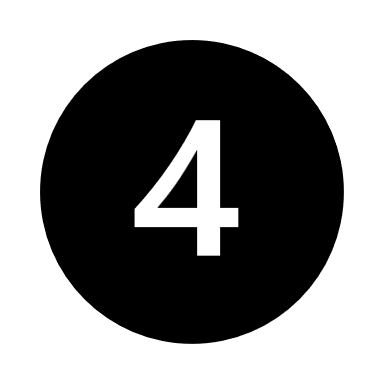
The username field to get the username of the user to store in the database .



The password field to get the password of the user to store in the database.



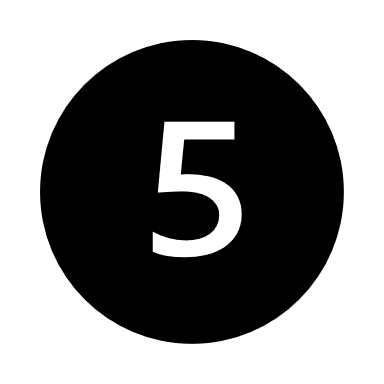
The login button which will check if the username and password match any which are in the system ,depending on that the system will respond with directing the user to the main menu or sending a error message.



This button will lead the user back to the login page so they can login once their login details have been registered.

This button will clear the fields, so it is made easy for users to renter their details.

### Main page





|  |  |  |
| --- | --- | --- |
| **Room**  **number** | **type** | **Reserved** |
|  |  |  |
|  |  |  |
|  |  |  |

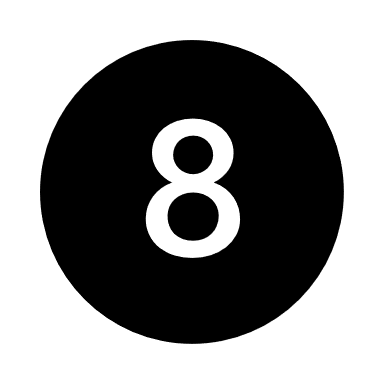
Room Type

Room number

Username





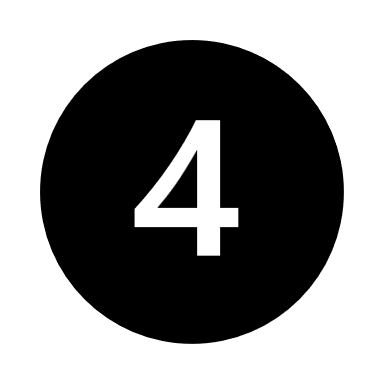


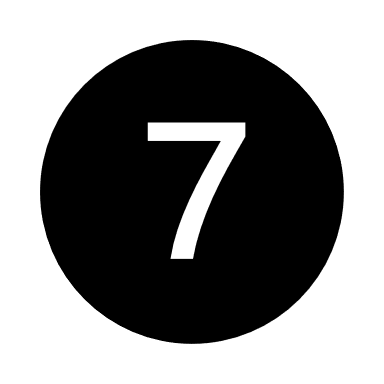
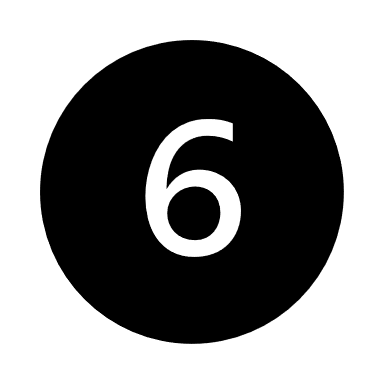
reset

delete

Edit

Book room





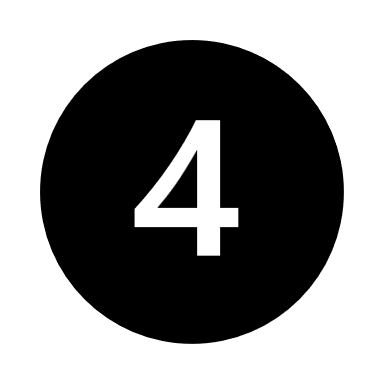


This will store the username which will be later used for the receipt when the book room button is clicked.

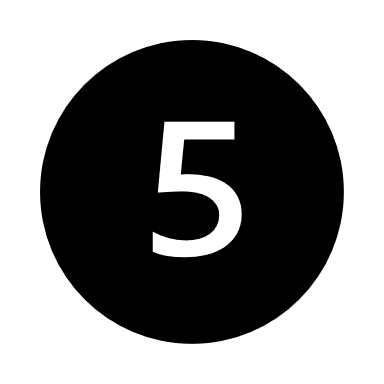
This is the room number which is unique and assigned to every room.

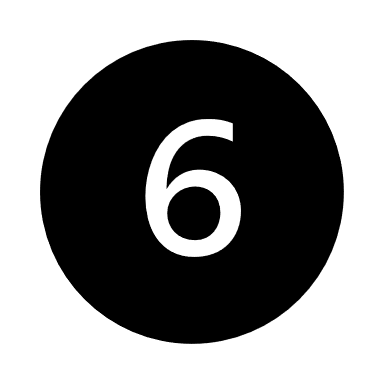


This will tell the system which room type the user wants to book.

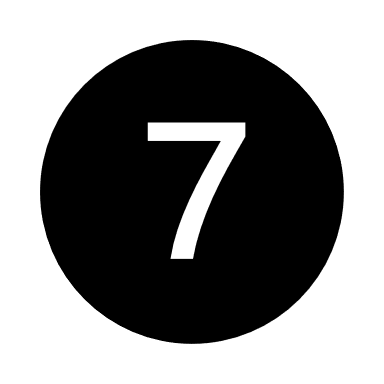


The book room button will take in the information from the user ,their username, the room the want to book, the room type and will send a message back (which is the receipt) informing them which room they have booked.

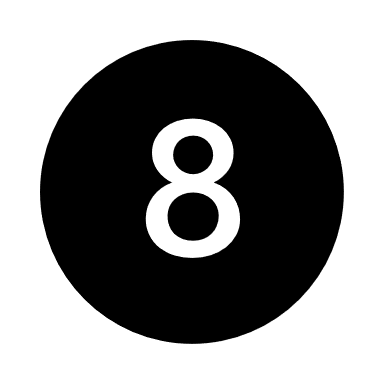
This is a table which will be connected to a database which will have the fields Room number , type, and reserved. This database table will be to show which rooms are booked and which are not .This will make it easy and simple for the workers and navigating to check which rooms are occupied will be right in front of them while making the booking.



This button will clear the fields, so it is made easy for users to renter their details.



This edit button would allow the worker to edit rooms, for example the if they just booked a room, they can change the room to reserved updating the table ,this will help other workers when they are making new bookings .



This delete will be used to delete rooms ,a use for this could be to not show rooms which are currently under maintenance by removing them.

## Class DIagram

Initially I wanted to make 3 classes with the parent class as login and child as register and rooms however, I thought it would be much better to join the login and register class and make the programme simpler.

|  |
| --- |
| Rooms |
| Room number: integer  Room type : String  Reserved: Boolean |
| editroom(Roomnumber,Roomtype,Reserved);  Deleteroom(Roomnumber);  getLogin(username);  Bookroom(username,Roomnumber,Roomtype);  Filltable(Roomnumber,Roomtype,Reserved); |

|  |
| --- |
| Login |
| Username : String  Password : String |
| new(username,password):constructor  confirmLogindetails();function  register(username,password); function |

Both classes are vital for my project , login is used at the start to allow access into the main programme , with the help of a database SQL will be used to manipulate the data. The link between the two databases will be made through the foreign key username will be used. The rooms class will be used when the user interacts with the main page to make booking or alter anything to with the rooms , it will also used at the end when a booking is made.

## Post-development Testing

**Login page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | Testing data | Expected  Output | Actual |
| 1 | Validation that the login system is bypassed when login data is correct. | Username=”mabroor”  Password=”ahmed” | Directs the user to the login page |  |
| 2 | Validation to see if the login system is bypassed even when the login is not equal to the correct login but contains the details of the login | Username=”mabroor11”  Password=”ahmed123” | This should not direct the user to the login page it should reply with “Wrong login” |  |
|  | Validation to see what the response of the programme would be if the login button is clicked without any login details | Username=””  Password=”” | Wrong login |  |
|  | Validation to see if the reset button works and clears the data in the username and password text fields. | Username=”asdasd”  Password=”1234” | Empties the data in the fields. | The data from the fields is removed |

**Register page.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test number** | **Description** | **Testing data** | **Expected**  **Output** | **Actual** |
| **1** | Validation that the Register button directs the user to the register page | The Register button | Stores the data inputted by the user into the database |  |
| **2** | Validation to see if entering new details in the Register page stores the details in the database | Username=”Testing”  Password=”Testing123” | This should add the Username “Testing“and the Password “Testing123” also to my database. |  |
| **3** | Validation to see the back button on the Register page works | The back button on the register page | Directs the user to the login page |  |
| **4** | Validation to see if you can login after you have registered | Username=”Testing”  Password=”Testing123” | Should respond with “Correct login” and should Direct the user to the main page |  |

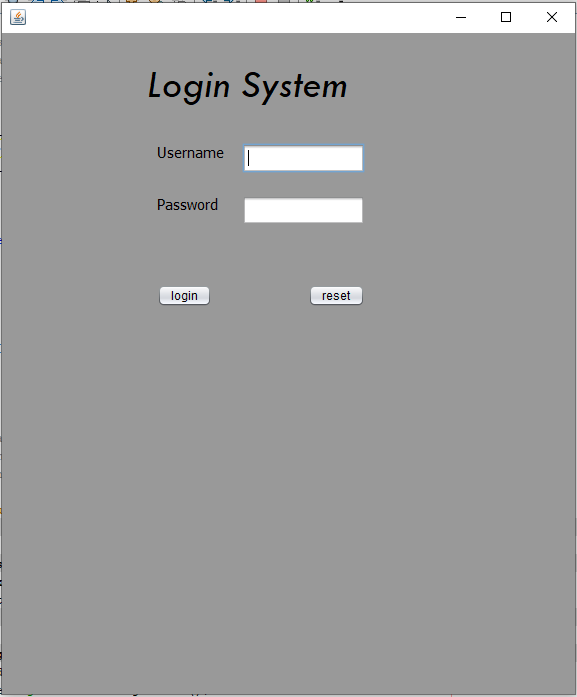
**Main page.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test number** | Description | Testing data | Expected  Output | Actual |
| **1** | Validation to see if the add new button on the room management page works. | Room number=”13”,Type=”King”,  Reserved=”no” | The room “13” is stored in  the database, the Roomtype  is “King” and it should be stored as “no” for the Reserved . |  |
| **2** | Validation to see if the remove button in the room management works | Room number=”12”,  Type=”Single”,Reserved=”no”. | This should remove the room with the roomnumber”12” |  |
| **3** | Validation to see if the edit button on the Room management page works. | Room number=”1”,  Type=”Single”,  Reserved=”yes”. | This should edit the room with the Roomnumber” 1”, the Reserved field which should be “no” as the start,  this should now change to “yes”. |  |
| **4** | Validation to see if the back button on the room management | The back button is clicked |  |  |

# C. Developing the coded solution (“The development story”)

<See H446-03 Project Advice Booklet for help and guidance of what must go here.>

## 2Prototype 1



Login verification.

A procedure is used for verifying the user’s input; however, the condition is wrong as the username currently is only set to contain for example the string “mabroor” to pass the login phase. The same condition applies to the password however with a different value. This needs to be changed and further changes will be made in prototype 2.

### Prototype 1 Overview

The login verification needs to be changed so the user is only sent to the main page when the login details exactly match the required ones, the user should not be sent to the main page when the login details contain some of the login details. An improvement that needs to be made is there needs to be a database where login details are stored. Another improvement could be that register system should be made so you don’t manually have to code the user’s login information.

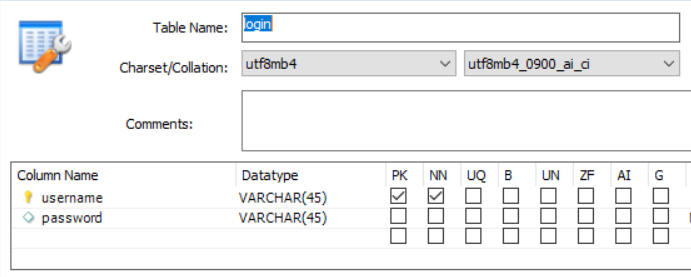
### Test plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | Testing data | Expected  Output | Actual |
| 1 | Validation that the login system is bypassed when login data is correct. | Username=”mabroor”  Password=”ahmed” | Directs the user to the login page |  |
| 2 | Validation to see if the login system is bypassed even when the login is not equal to the correct login but contains the details of the login | Username=”mabroor11”  Password=”ahmed123” | This should not direct the user to the login page it should reply with “Wrong login” |  |
|  | Validation to see what the response of the programme would be if the login button is clicked without any login details | Username=””  Password=”” | Wrong login |  |
|  | Validation to see if the reset button works and clears the data in the username and password text fields. | Username=”asdasd”  Password=”1234” | Empties the data in the fields. | The data from the fields is removed |

**The errors I found.**

The validation for the function which is made for the login system currently is incorrect, this needs to be improved and corrected to make sure only the users with the exact login details can enter the hotel booking system.

### setting up the database



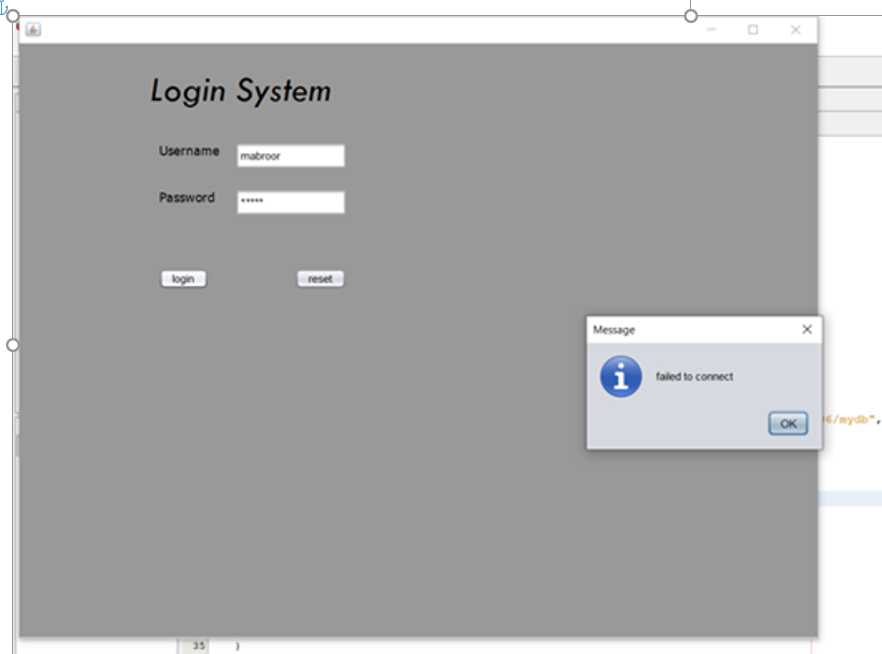
***At first a table was made in MySQL with the name login, it included 2 columns with the username’s column being the primary key, the password was not chosen to be the primary key as usernames are more suitable for identifying a user’s login details.***

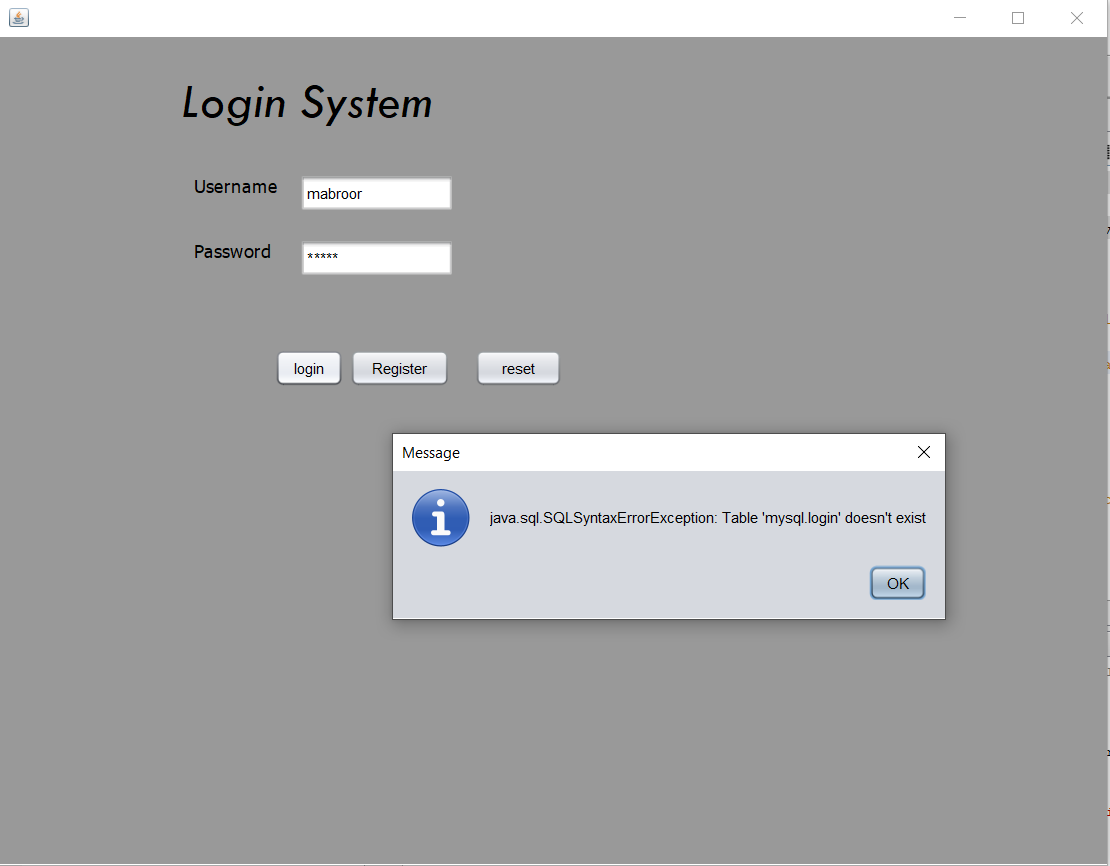
**This is a try catch statement which is used to test if a block of code will work if it does not work the catch part sends the error back I used this as the pop up “login failed” message made it very clear if something was wrong and that I was not going in the right lines .This part of code was created to initially connect to the database and then after the user has inputted their information. I used SQL to check if their information existed in my database and if it was the same, the confirmation of these two things then means the user is directed to the Main page.**

## Prototype 2

### Problem

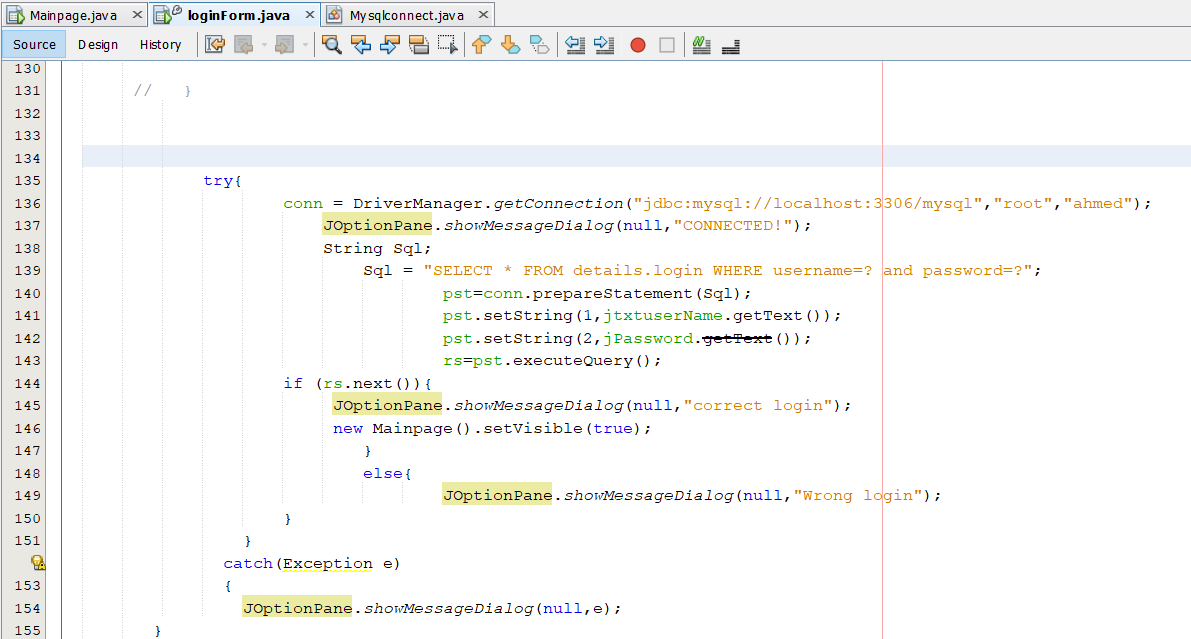
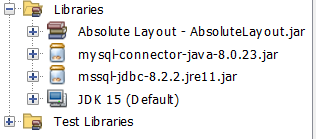
This function was used at the start to test my connection from NetBeans to a database in MySQL. With the help of a try catch statement which identifies a mistake if an error has been made However, it did not give the response which i wanted it to give as it displayed (“failed to connect”) on the screen.





### Solution:

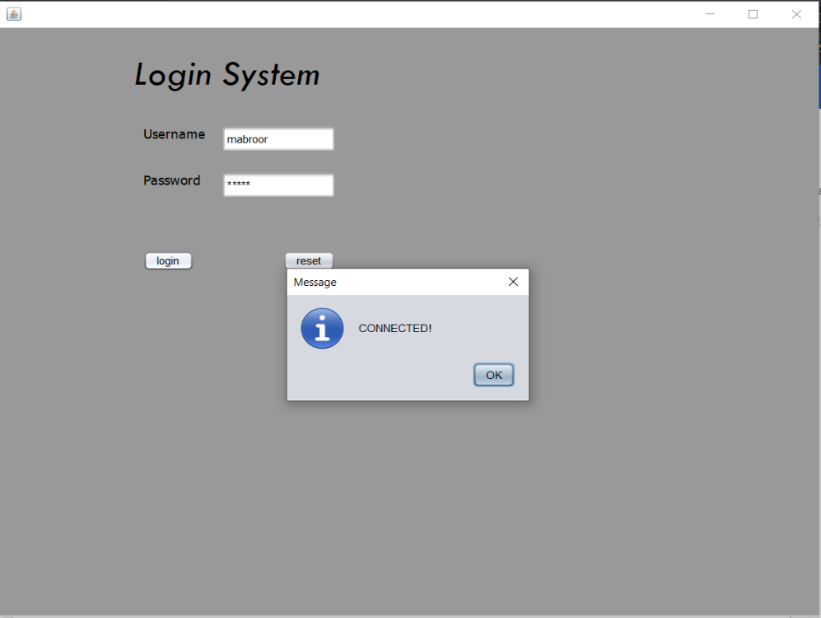
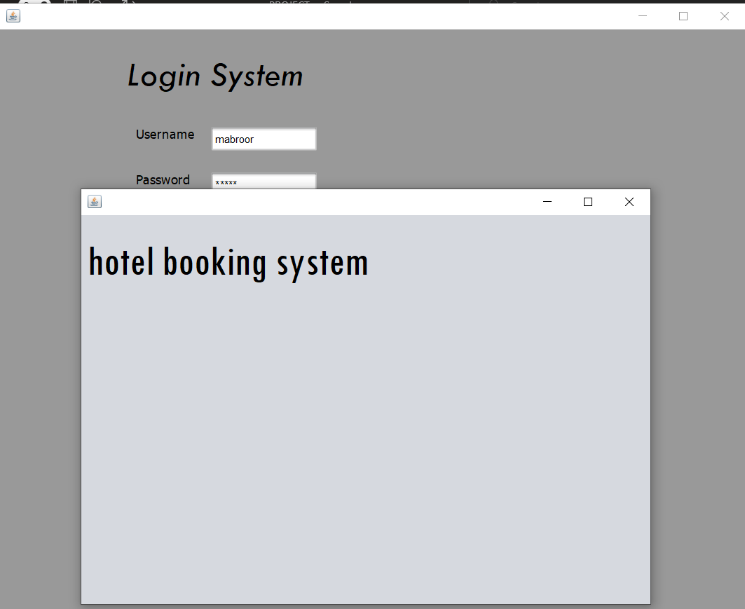
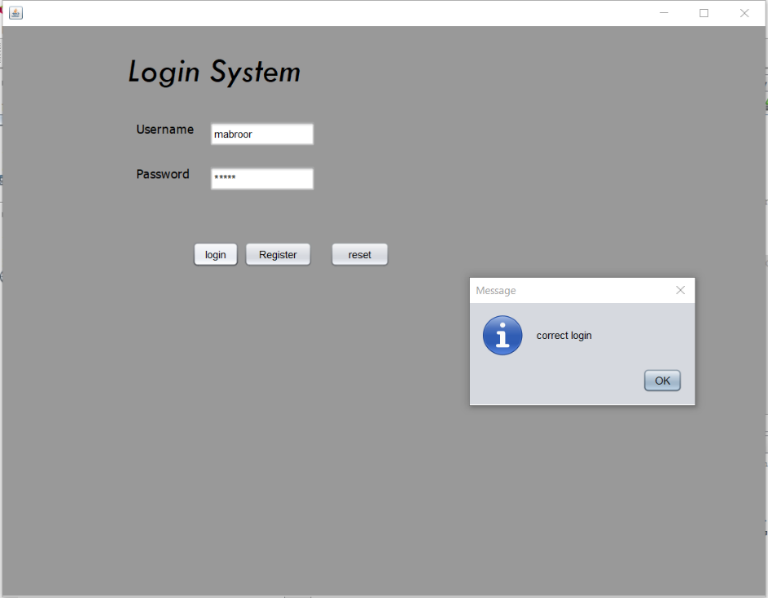
I fixed my problem of the database not being able to connect by using the right parameters and syntax when connecting to my database to NetBeans and my syntax error was solved by changing the “select from logins” to “select from details.login”.

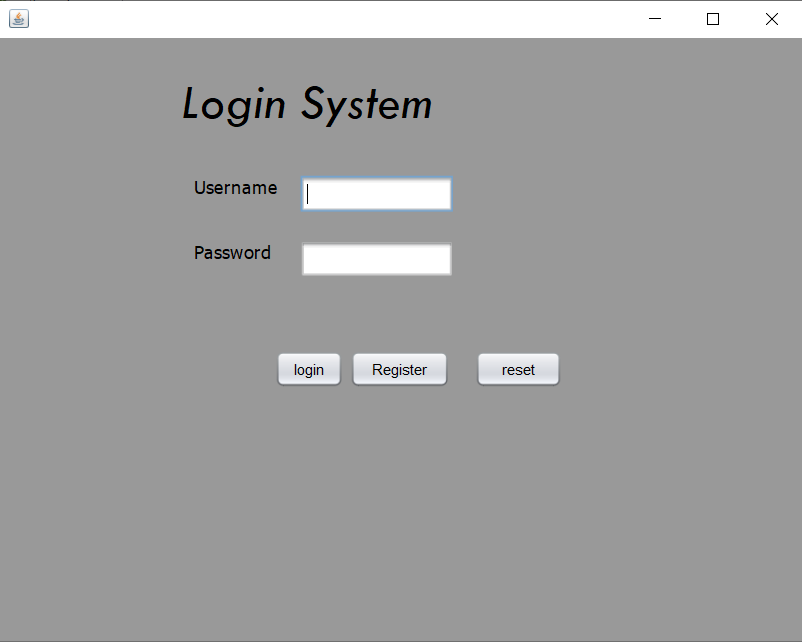
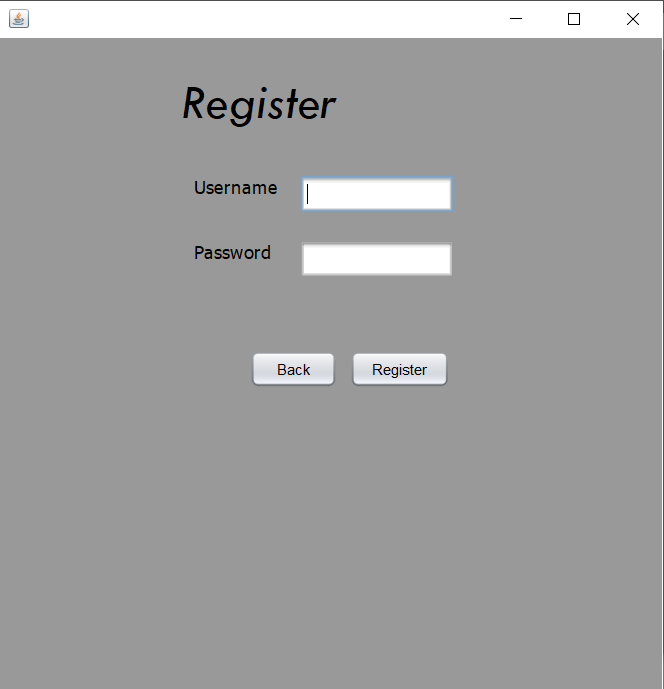
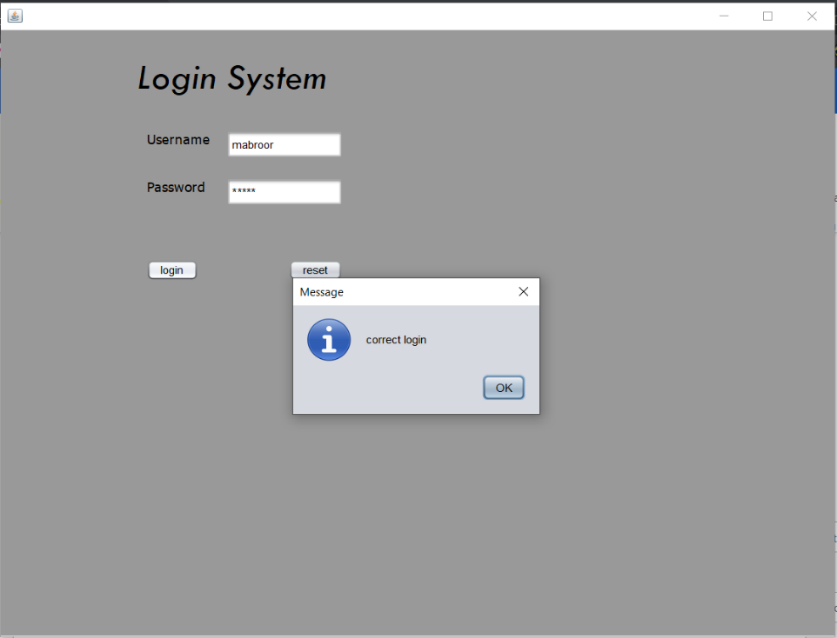


The connection error was fixed by adding a” my sql connector” library, with that an SQL statement was also used to check if login details are already presented in the database would that log a user in. The result for this was that it did log you in if the current login detail was held in the database.









Register system.

***I created a register system where the users could input their own usernames and passwords without the need of manually putting in details, a function was used to insert data into my database. This means that when you go back to the login page after registering with your details, you can login as your details are stored in the database.***

### Prototype 2 overview

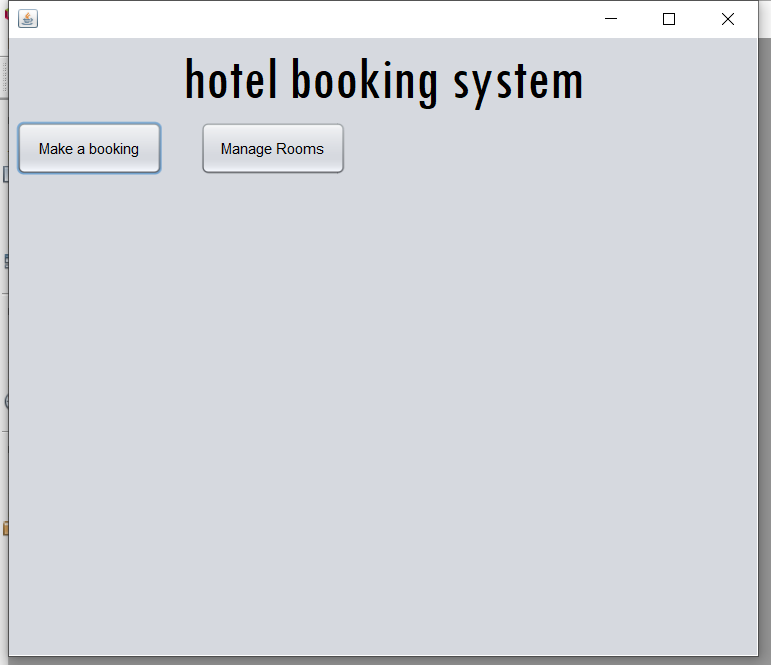
I created the register page and login page with also connecting my login system to a database, however I have not yet created my main page which will allow the booking to happen and the user receiving a receipt at the end .I also want to make it easy for the workers when it comes to checking which rooms are available and which are not ,so I would like to make something that will allow for rooms to be managed easily.

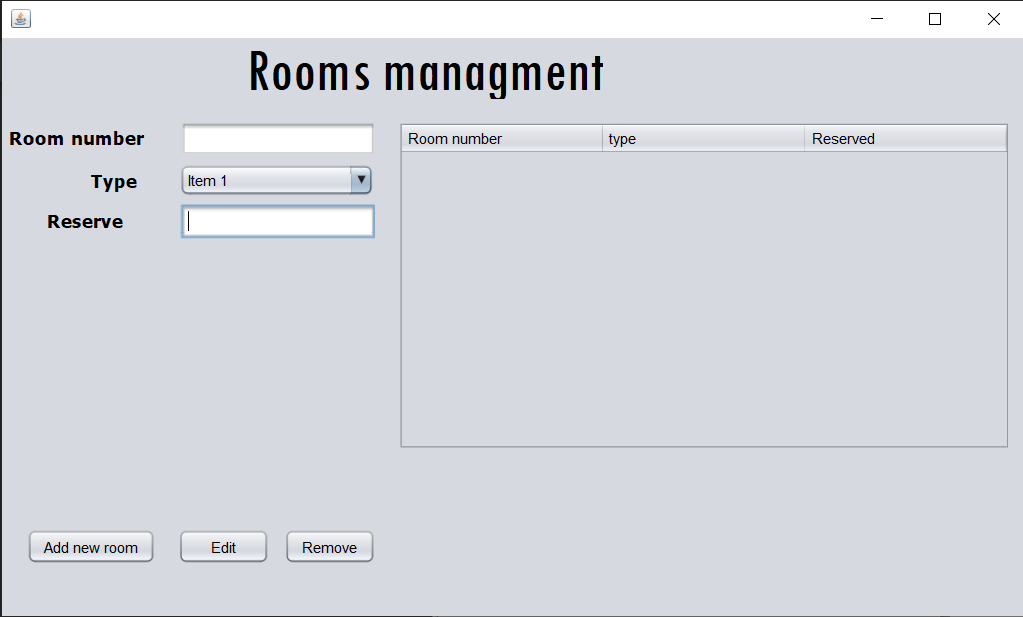
### Test plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test number** | **Description** | **Testing data** | **Expected**  **Output** | **Actual** |
| **1** | Validation that the Register button directs the user to the register page | The Register button | Directs the user to the register page |  |
| **2** | Validation to see if entering new details in the Register page stores the details in the database | Username=”Testing”  Password=”Testing123” | This should add the Username “Testing“ to my database and the Password “Testing123” also to my database in MYSQL and the output should be “Registered”  **Database Before input** | **Database** **after** |
| **3** | Validation to see the back button on the Register page works | The back button on the register page | Directs the user to the login page |  |
| **4** | Validation to see if you can login after you have registered | Username=”Testing”  Password=”Testing123” | Should respond with “Correct login” and should Direct the user to the main page | Badge with solid fillBadge 1 with solid fill |

## Prototype 3

### Main page planning

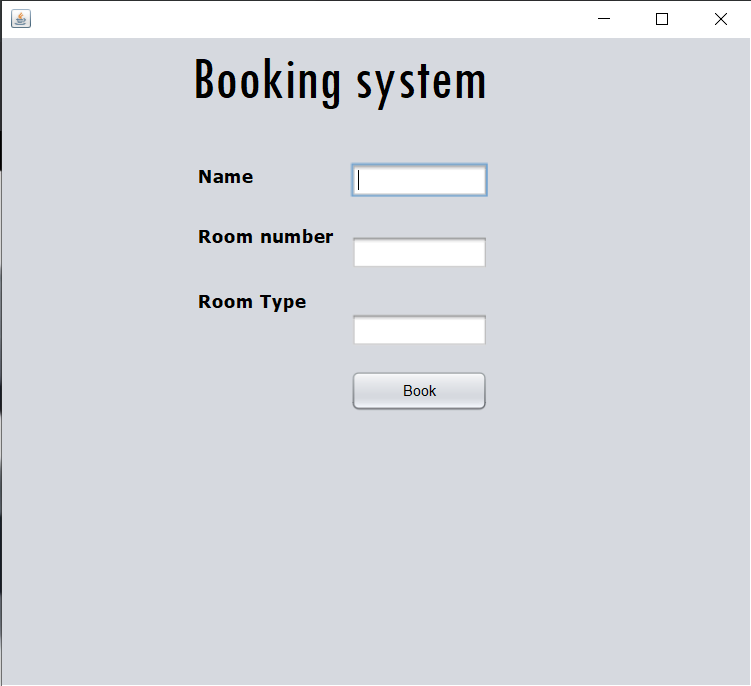




**The main page will include the features of making a booking and managing rooms. In Room management I plan on showing all the rooms in the from the database which will be created on the MYSQL just as the login table was created for the login system.**

### Main page

**Another table called room was created to store the information about rooms, the room number which is the primary key with the other fields including the room type and if it is reserved or not. The room number was the primary key as it is the unique value for each room.**



**For the booking system the two tables “room” and “login” were linked to each other using the foreign key “username” then an SQL update statement is used to make a booking, the worker then receives a receipt with the room that has been booked, the room type and**

### Overview

The system contains two main pages after the user has logged in, one is room management where the workers can add rooms and see which rooms are reserved or not ,this makes it easy and fast for workers to navigate through the available rooms also making bookings easy. The other page is where the bookings will be made the bookings page has 3 text fields which are used to see which room is booked username ,room number and room type .Once the worker has inputted the details the database is updated with the room information and that room is reserved along with the person it is reserved for.

### test plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | Testing data | Expected  Output | Actual |
| **1** | Validation to see if the add new button on the room management page works. | Room number=”13”,Type=”King”,Reserved=”no” | “Room added” |  |
| **2** | Validation to see if the remove button in the room management works | Room number=”12”,Type=”Single”,Reserved=”no”. | This should remove the row with the room number 12 |  |
| **3** | Validation to see if the edit button on the Room management page works. | Room number=”1”,Type=”Single”,Reserved=”yes”. | This should change the room one which is a single room to reserved and respond with “Room Edited”. |  |
| **4** | Validation to see if the back button on the room management | The back button is clicked | Should send the user back to the main page |  |

**The errors I found.**

The remove button is currently not working and needs to be fixed.

### overview

The system contains two main pages after the user has logged in, one is room management where the workers can add rooms and see which rooms are reserved or not ,this makes it easy and fast for workers to navigate through the available rooms also making bookings easy.

## Post-Development Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **description** | **justification** | **Test data** | **Expected output** | **Actual output** | **Status** |
| Each room should have a unique Room number. | This makes sure there is no duplicates rooms | Adding a room which already | Room cannot be added | As expected. | Checkmark with solid fill |
| What happens when a room which is not in the database is edited | This is to make sure no room is edited when it does not exist, and that the system does not add the room instead of editing an existing room | Editing a room which is not in the database | Room cannot be edited | As expected. | Checkmark with solid fill |
| All the button works | Making sure the system is functional and has no simple faults. | Clicking all the buttons and checking they do their purpose] | All the buttons work properly. | As expected. | Checkmark with solid fill |
| The program user cannot change the room type | This is to check if a secondary key in a MYSQL database table can be changed or not | Changing the type of a room while the “room type” is a secondary key in the database table | Should still be able to change the datatype | An error is given about “SQL statement integrity”. | Close with solid fill |
| The program can register a new user and then the user can use the back button and login? | This is to just check for functionality | New details are entered into the register ,which are later used to login. | The user should be able login | As expected | Checkmark with solid fill |

### Test for functionalities and robustness (done by me)

### Functionality - Stakeholder

|  |  |  |
| --- | --- | --- |
| **Action to test** | **Working** | **Comments from the stakeholder** |
| All the buttons work? | Checkmark with solid fill | All the buttons work in their manner. |
| Does the login system work? | Checkmark with solid fill | I was able to login easily |
| Does the register page work? | Checkmark with solid fill | I was able to register with new logins |
| Are you able to login once you have registered? | Checkmark with solid fill | Yes, I can login once I have registered. |
| Does the main page work to its functionality? | Checkmark with solid fill | The two buttons lead me to the right pages. |
| Does the edit button work in the room management page? | Checkmark with solid fill | Yes, I can change room information which is then updated on the table. |
| Does the delete button work in the room management page? | Checkmark with solid fill | Yes, I was able to remove a room which also updated into |
| Does the back button work on the room management page? | Checkmark with solid fill | Yes, it worked fine |
| Are you able to make a booking and get a receipt back ? | Checkmark with solid fill | I was able to get a receipt back for the room I booked. |

From the stake holder meeting , it was clear that my programme worked to its function .

### Usability Testing

A small survey also took place with the stakeholders to get their responses of the usability of the programme.

YES =100%

NO= 0%

YES =100%

NO= 0%

No =67%

Yes= 33%

Easy =67%

Okay =16%

Hard =16%

Easy =83%

Okay =17%

Hard =0%

Very well = 67%

Okay= 33%

Not at all= 0%

**From the survey I can conclude that**

-The user found it very easy to use the programme

-The users were able to use the login and register system correctly to reach the main page.

-The user did not like the design and colouring of the programme , however 33% did like the design.

-There were some mixed views on the understanding of the error messages 67% found it easy to understand,16% found it okay and another 16% found it hard.

-The user mainly found it easy to navigate through the program with 83% responded with easy out of the option easy, okay, and hard. The last 17% found it okay , there was no one who found it hard to navigate through the programme.

-The last question was aimed at the functionality of the programme and though 67% said that the programme did what it was supposed to do very well, 33% said it was okay and no one said it did not serve its purpose at all .

# D. Evaluation

<See H446-03 Project Advice Booklet for help and guidance of what must go here.>

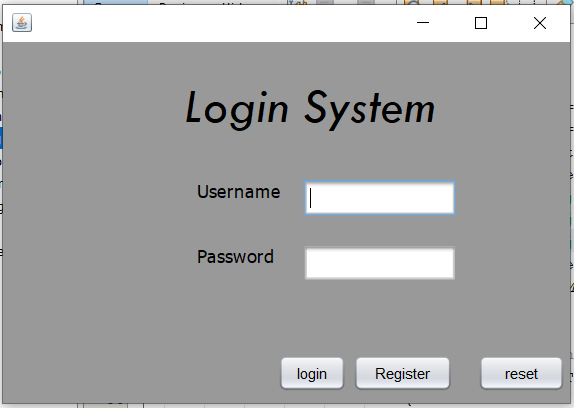
## Success criteria

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria number** | **Description** | **Has it been met** | **Evidence/Further Development** |
| **1** | Create a login system where staff and customers can login to . | Checkmark with solid fill | In the usability survey, all the stakeholders said that they were able to use the login system to reach the main page. |
| **2** | Create a Register system if the customers or staff do not already have a login. | Checkmark with solid fill | In the usability survey , the stakeholders also said that they were able to use the register system too. |
| **3** | Login system has a menu making it easier for people to navigate through the page. | Close with solid fillCheckmark with solid fill | This was partially done as there was not a specific menu in the login system ,however there was a register button which had a back button .This made navigation easy. The stakeholders also said that the navigation was easy as in the usability survey 83% said it was “easy”. |
| **4** | Create a page where staff can add new Rooms. | Checkmark with solid fill | This feature was added as on the room management page there was a edit button which is fully functional and was tested in the test plan for prototype 3. |
| **5** | staff can change Edit and delete the rooms. | Checkmark with solid fill | These features were also added on the room management page and were fully tested with a test plan as evidence . |
| **6** | a system to filter our different type of rooms . | Close with solid fill | This feature was unfortunately not added due to time and I was not able to finish this algorithm. |
| **7** | button to book a room which is currently free. | Checkmark with solid fill | This button was added on the Booking page and the evidence for this is in the test plan for prototype 3 where it was tested. |
| **8** | A receipt made for the customers and staff printing out what room is booked and the type of room it is. | Checkmark with solid fill | This feature was added on the Booking page and there is evidence as there are screen shots in prototype 3. |

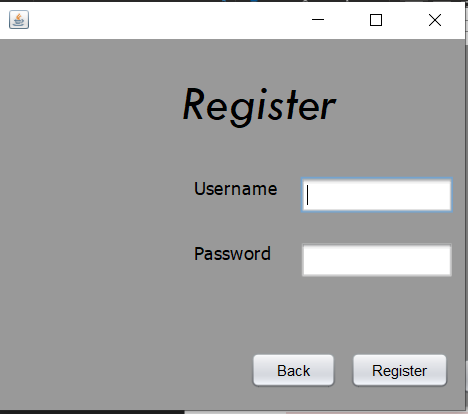
**Conclusion/what could be done(further development) .**

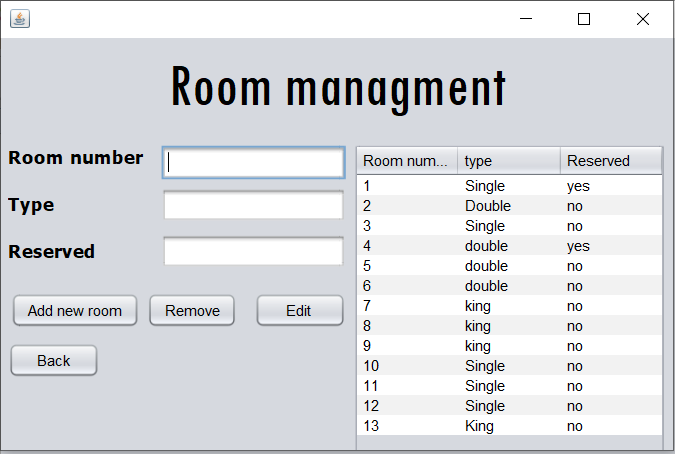
* An option to add a filter button which filters rooms .This could be done with an algorithm mainly with the use of if statements filtering of different rooms could be made with the user’s entries. Then an SQL query could be used to search through the database and pick out the rooms which meet the criteria .
* A full menu on the login page. Looking back, it doesn’t look like I needed this as the stakeholders are already satisfied with the navigation of the system ,however I could have implemented a bar at the top of the page leading to the Register page and for the main page I could do the same thing but with the room management page and booking page at the top.

## usability Features



A feature on the first page was to clear the username and password field , this makes it easy for user to enter login details again if they accidently wrote the wrong ones .I found this very useful when I had to do a lot of testing as it made the process faster. I also made the design and colour scheme very simple and not annoying with bright colours, though my design had bad reviews by the stake holders in the usability survey as they had other opinions. The login page also did not show the password as it was being typed by the user ,this would give the user a better sense of security and prevents other near you from looking at your password.

On the second page there was a back button which helped the user navigate back to the login page after they have register , though I did not have a menu for navigation as said in my success criteria my stakeholders were happy with the navigation system as 83% said it was “easy” to navigate through the system. I would say this small feature was very helpful as it made it very easy for the users to navigate through the programme.

On the Room management page, I created a table which was connected to the room table database in MYSQL and displayed the room and if they were booked or not , I think this made it very easy and simple for the users to make bookings according to which rooms were booked and which were not , and when the user makes a booking they could come back to this page and edit the room to reserved which is then updated in the database and the table. I also used a back button here to make navigation easy especially to the booking page.

## Maintenance

The databases are well connected and are normalised to a degree, the function I coded for the connection and the querying of the data in the database are very simple to understand and should be understood by other programmer if there is a need to alter anything in the system or for future development. The database can function on its own if the application MYSQL workbench is downloaded however the right version 8.0 would be needed as it is the latest version and is the version, which was used to create the MYSQL server too, therefore the database most likely will not need to be maintained. The program requires NetBeans latest version with the MySQL connector package to run, the code is split up nicely into functions making the coding simple and efficient ,the code is annotated making it easier for other programmers to understand when It comes to others adding a button or a filter system which I was not able to do due to time. This splits up of the program making it simple and easy for the other programmer who wants to add a filter system or alter any code. The program requires inputs from the users this does not need to be maintained as there is no need of changing code as there are no problems currently , however my programming skills have a limit of its own , I might not realize something is wrong or inefficient until a professional programmer checks the code itself.

## Limitations and Future Development

One of the current limitations of the program is that there is not a feature which is able to filter rooms , adding this feature would make it even easier for the workers to make booking as for example the rooms which are booked can be filtered out on the table or for example the rooms which are only of type double could be filtered out . This can be done by coding an algorithm which takes in the user’s input , then with a SQL query using an if statement the keywords could be searched for in the database with the table “room” . This would then follow up with code to display the selected data which was searched through using the users keyword input.

For further development, I could create a change interface section, this was due to the bad reviews I had about my design from the stakeholders in the usability survey. Most of the stakeholders said that the design was not good as 67% of the stakeholders did not like it. I would also like to make this program online into a webpage which is backed up by a could network. This would secure the data as it will be backed up by a cloud service which will not get rid of the data when a system shuts down. this would also be good as it would help hotels which are not located next to each other to use the system, the online system could also have some new features such as displaying the room prices to customers as they enter the website. With the connection of the databases the system could also have a feature which shows all the room price rates throughout the companies different hotels across a country for example.

# Project Appendixes

Insert as many project appendixes as you need for your project.

These might include, but are not limited to:

* Complete Code Listing (ESSENTIAL)
* Interview Transcripts
* Meeting notes
* Observation notes or questionnaires

## Design

I made the design from the start ; however it was made with the help of NETBEANS.

## Login button(on the login page)

private void jbtnlogiinActionPerformed(java.awt.event.ActionEvent evt) {

try{

conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");//connecting to the database created in MYSQL

String Sql;

Sql = "SELECT \* FROM details.login WHERE username=? and password=?"; //SQL Query used to munipulate

pst=conn.prepareStatement(Sql);

pst.setString(1,jtxtuserName.getText()); //data from the textfeilds are reterived and then sent to the database

pst.setString(2,jPassword.getText());

rs=pst.executeQuery();

if (rs.next()){

new Mainpage().setVisible(true);

}

else{

JOptionPane.showMessageDialog(null,"Wrong login");

}

}

catch(Exception e) //for mistakes

{

JOptionPane.showMessageDialog(null,"login failed");

}

}

## Reset button(on the login page)

private void jbtnresetActionPerformed(java.awt.event.ActionEvent evt) {

jPassword.setText(null);

jtxtuserName.setText(null);

}

## Register button. (on the login page)

private void RegisterbtnActionPerformed(java.awt.event.ActionEvent evt) {

new Register().setVisible(true); //making the register page visible

}

## Register button(on the register page)

private void RegisterbtnActionPerformed(java.awt.event.ActionEvent evt) {

try{

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");

String Sql;

Sql = "insert into details.login(username,password) values(?,?)"; //sql statement to putin the users details in the database

PreparedStatement pst = conn.prepareStatement(Sql);

pst.setString(1,jtxtuserName.getText());

pst.setString(2,jPassword.getText());

pst.executeUpdate();

JOptionPane.showMessageDialog(null,"Registered!");

}

catch(Exception e)

{

JOptionPane.showMessageDialog(null,e);

}

}

## Back button(on the register page)

private void jbtnresetActionPerformed(java.awt.event.ActionEvent evt) {

new loginForm().setVisible(true); // leads back to the login page

}

## Booking button(on main page)

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

new Booking().setVisible(true); // leads to the booking page

}

## Manage rooms button(on main page)

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

new Managerooms().setVisible(true);// leads to the Room management page

}

## Add room button(on Room management page)

public void addingroom(){

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");

String addQuery = "insert into details.room(R\_number,Type,Reserved) values(?,?,?)";

PreparedStatement pst;

pst= conn.prepareStatement(addQuery);

pst.setString(1,jTextRoomNUM.getText()); // getting text from the fields

pst.setString(2,jTextRoomTYPE.getText());

pst.setString(3,jTextRESERVE.getText());

pst.executeUpdate();

JOptionPane.showMessageDialog(null,"Room added");

} catch (SQLException ex) {

//to respond if no information was inputted

if (jTextRESERVE.getText().equals("") || jTextRoomNUM.getText().equals("") || jTextRoomTYPE.getText().equals(""))

{

JOptionPane.showMessageDialog(null,"Fill in all the details");

}

else

{

JOptionPane.showMessageDialog(null,"Room cannot be added");

}

Logger.getLogger(Rooms.class.getName()).log(Level.SEVERE, null, ex);

}

}

## edit button(on Room management page)

public void Editingroom(){

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");

String editQuery;

editQuery = "UPDATE details.room SET Type=?, Reserved=? WHERE R\_number=? ";

PreparedStatement pst;

pst= conn.prepareStatement(editQuery);

pst.setString(1,jTextRoomTYPE.getText()); //getting data from the fields

pst.setString(2,jTextRESERVE.getText());

pst.setString(3,jTextRoomNUM.getText());

pst.executeUpdate();

JOptionPane.showMessageDialog(null,"Room Edited");

} catch (SQLException ex) {

// response to if the fields are empty

if (jTextRESERVE.getText().equals("") || jTextRoomNUM.getText().equals("") || jTextRoomTYPE.getText().equals(""))

{

JOptionPane.showMessageDialog(null,"Fill in all the details to Edit a Room");

}

else

{

JOptionPane.showMessageDialog(null,"Room cannot be edited");

}

Logger.getLogger(Rooms.class.getName()).log(Level.SEVERE, null, ex);

}

}

## Delete button(on Room management page)

public void removeroom(){

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");

String deleteQuery;

deleteQuery = "DELETE FROM details.room WHERE (R\_number=?,Type=?,Reserved=?)";

PreparedStatement pst;

pst= conn.prepareStatement(deleteQuery);

pst.setString(1,jTextRoomNUM.getText()); // getting data from the fields

pst.setString(2,jTextRoomTYPE.getText());

pst.setString(3,jTextRESERVE.getText());

pst.executeUpdate();

} catch (SQLException ex) {

if (jTextRESERVE.getText().equals("") || jTextRoomNUM.getText().equals("") || jTextRoomTYPE.getText().equals(""))

{

JOptionPane.showMessageDialog(null,"Fill in all the details to DELETE a Room");

}

else

{

JOptionPane.showMessageDialog(null,"Room cannot be DELETED");

}

Logger.getLogger(Rooms.class.getName()).log(Level.SEVERE, null, ex);

}

}

## Back button(on Room management page)

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

new Mainpage().setVisible(true);

}

## Code for filling the Jtable

public void fill\_in\_thetable(JTable table){

ResultSet rs;

PreparedStatement pst;

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed"); //connecting the database

String SelectQuery = "SELECT \* FROM details.room"; //data from the database is selected to be put on the jtable

pst = conn.prepareStatement(SelectQuery);

rs = pst.executeQuery();

DefaultTableModel tableModel;

tableModel = (DefaultTableModel)table.getModel();

Object[] row;

while(rs.next()) // while loop to increment until all rows are filled

{

row = new Object [3];

row[0] = rs.getInt(1);

row[1] = rs.getString(2);

row[2] = rs.getString(3);

tableModel.addRow(row);

}

} catch (SQLException ex) {

Logger.getLogger(Managerooms.class.getName()).log(Level.SEVERE, null, ex); // checks for error

}

}

## Book button (on the booking page)

public void MAKbooking(){

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql","root","ahmed");

String editQuery;

editQuery = "UPDATE details.room SET username=?, Type=? WHERE R\_number=? ";

PreparedStatement pst;

pst= conn.prepareStatement(editQuery);

pst.setString(1,jtxtuserName.getText());

pst.setString(2,jTextRoomTYPE.getText());

pst.setString(3,jTextRoomNUM.getText());

pst.executeUpdate();

JOptionPane.showMessageDialog(null,"Room booked");

} catch (SQLException ex) {

if (jtxtuserName.getText().equals("") || jTextRoomNUM.getText().equals("") || jTextRoomTYPE.getText().equals(""))

{

JOptionPane.showMessageDialog(null,"Fill in all the details to book a Room");

}

else

{

JOptionPane.showMessageDialog(null,"Room cannot be booked");

}

Logger.getLogger(Rooms.class.getName()).log(Level.SEVERE, null, ex);

}

}