# Design

# Section 1: Inputs, Outputs, and User Interface.

**Inputs outputs and user interface design**

In this section of the project, the inputs, and outputs related will be depicted, Furthermore, the first draft of the plan of the design of the interface will also be depicted. I will be explaining the functions and reasoning behind each element in the design of the interface, The choices made will not include specific elements but will also be mentioned, to show a precise idea of what my system aspires to be. I will use tools on Microsoft Word to draw and label all the elements about each form, relating to each objective. Some objectives may not include any inputs or outputs, and so this will be disclosed for the relevant objectives. This section will be laid out with subheadings for each objective.

# Customers should be able to sign up.

**Success criteria:** There should be a form, asking customers to enter their details. Then ask them to create a username and password which, should be saved on the database. Then redirecting them to the login page.

Signing in is the first step for any customer to proceed further on the program Therefore, it must be effective and efficient. Once the user opens up the form “frmCust” which is where this all takes place. The user should see textboxes date time picker and buttons each for a different purpose. Here are all the prefixes used in frmCust and what they indicate:

textboxes = “txt”, buttons = “btn”, datetimepicker = “dat”. Below, there is the design for formCust;

A screenshot of a computer screen

Description automatically generated

Here the user can press the button “btnBack” to redirect to the “frmLogIn” where users may input their login and proceed further on. Furthermore, the button “btnNew” ,if the data is validated, will complete the sign-in process and show the following message box (“You have successfully made an account”) in addition, All of the data entered will be stored in the database and the user will be redirect them to the “frmLogIn”. However, if all the textboxes are empty or data is not validated it should show the following message box(“Sorry, Please double check that your information is right”). Furthermore, If the user is under 18 it should show the following message box (“Sorry, You must be 18 or over to use the service”).

# Staff members should be able to sign up.

**Success criteria:** Another staff member will record the other staff members' details. Then ask them to create a username and password, which should be saved on the database.

Here is a similar objective as the customer one but it is slightly different for staff members for security and monitoring purposes. For example, to prevent customers from finding out about any secret methods or staff passwords as some systems have check validations following a pattern. Therefore, I want to make another staff member be required to make the account for another staff member for the purposes mentioned above. This will all take place in the “frmStaff” where staff members can add, delete, update and search for other staff members' details. Furthermore, the form will consists of pannels, buttons, text boxes, a list box and a date time picker. Here are all the prefixes used in frmCust and what they indicate:

textboxes = “txt”, buttons = “btn”, datetimepicker = “dat” and listboxes = “lst”. Below, there is the design for formCust;

A screenshot of a computer

Description automatically generated

Here, another staff member will need to log in to add the new staff member. Firstly, you will need to input the new staff member details in the text boxes and date time picker. Once the details have been inputted the staff members whose account belongs to will need to press the “btnNew” button. Once it's pressed and the data is validated it shows the following message (“You have successfully added the new staff member”). Otherwise, it will show the following message box (“Sorry please double check the details”). Furthermore, if the staff member wants to clear the details, that can be done with the “btnClear” button as this will clear each textbox and reset the date time picker.

# The customer should be able to log in.

**Success criteria:** There should be a form, asking customers to input their username and password. If inputted correctly it should be able to let them log in, otherwise it will display an error message.

The Login Form, which will be the first thing the client will see once the program starts, consists of labels that will be used to indicate what a certain function does, textboxes that will allow the client to input information that may or may not affect their experience, and buttons. Buttons will redirect to another form.

Here are all the prefixes used in frmLogIn and what they indicate:

labels = “lbl”, textboxes = “txt”, buttons = “btn”etc… Furthermore, within this system panels will be used a lot because it’s a simple way to separate things. Below, there is the design for formLogIn;

A screenshot of a login form

Description automatically generated

Here ‘btnLogin’ will open up the form for the main page for customers if the user inputs the right login details. However, if the user inputs the wrong login details a message box should show up (“you have entered wrong login details, please try again”) and will **not** direct the user to another page ‘btnClear’ will eliminate any characters entered in ‘txtusername’ and ‘txtpassword’. ‘btnFormCust’ will open up the form for creating a customer login ‘btnFormStaffMain’ will open the main page for staff members.

# Staff members should be able to log in.

**Success criteria:** Like the customer, in a separate form, staff members should receive a form, asking them to input their name and password. If inputted correctly it should be able to let them log in, otherwise it will display an error message.

The Staff Login Form, which will be the first thing the Staff member will see once the program starts, consists of labels that will be used to indicate what a certain function does, textboxes that will allow the staff member to input information that may or may not affect their experience, and buttons. Buttons will redirect to another form.

Here are all the prefixes used in frmStaffLogIn and what they indicate:

labels = “lbl”, textboxes = “txt”, buttons = “btn” etc… Furthermore, within this system panels will be used a lot because it’s a simple way to separate things. Below, there is the design for formLogIn;

A screenshot of a computer screen

Description automatically generated

Here, the staff member will need to input their own details on the textboxes. Afterwards, the staff member would need to press the “btnLogin” button to log in. Once its pressed , if data is validated,it should redirect them to “frmStaffMain”. Otherwise, it should show the following message box (“Sorry, Please double check your details”). Furthermore, if the staff member wants to clear the details, that can be done with the “btnClear” button as this will clear each textbox. In addition, there is also a “btnBack” if the staff member wishes to go back to “frmLogIn”.

# Should authorize staff to add, edit, delete, and find customers.

**Success criteria:** There should be a search function where you can search for customers with either customer name or customer ID, once found the customer they should be able to edit, add, and delete that customer's information.

Staff need to manage and help other customers as staff are trained to do so. This will also help in particular foreign customer as they may not understand English. Therefore, Staff members should be able to add, edit, delete and search for customers data to create the best possible brand image and customer experience.

Here are all the prefixes used in “frmCustomers”;

textboxes = “txt”, buttons = “btn”, datetimepicker = “dat”, datagridview = “dgv” and listboxes = “lst”. Below, there is the design for formCustomerst;

A screenshot of a computer

Description automatically generated

Here, staff members will, if creating a new account, need to input the customer's data in the text boxes and date time picker and press “btnNew” button. Once pressed, if data is validated, it should show the following message box (“You have successfully added the customer”). Otherwise, it will show the following message (“Sorry, Please double check the details”). Furthermore, if the staff member wants to clear the details, that can be done with the “btnClear” button as this will clear each textbox and reset the date time picker.

If the client wishes to update his details then the staff member may need to search for the customer's details before they can update the customer's details. To search, the staff member will need to type in either the customer's “ID” or “Surname” on their own respective text boxes. I chose these 2 values as if the customer forgets his “CustomerID” they could use his “Surname” as the customer will definitely know this to proceed with the update. Once typed, the staff member will need to press the “btnSearch” button to search for that customer's details. If the customer are validated, it should show their details on the textboxes and date time picker on the left. Otherwise, it should show the following message box(“Sorry, Unable to find this customer's details”). Furthermore, the staff member can also clear the textboxes used for searching with the “btnClear” button on the right.

Once the customer's details are shown. The staff member can edit all his details except the customer “ID” for searching and security purposes. Once the changes are done, the staff member should press the “btnNew” button to update the details. Next, if validated, the following message box(“ You have successfully updated the customer's details”). Otherwise, it should show the following message box(“Sorry, Please double-check the details”).

To delete a customer the staff member must search or have already added a customer or else the “btnDelete” button will be enabled. Once you have the customer's details the staff member should press the “btnDelete” button. Once pressed it should show the following message box(“Are you sure you want to delete this customer”) with a “yes” button and “no” button. If the staff member presses the “yes” button the customer's data will be deleted. If the staff member presses the “no" button the customer's data will not be deleted. This is for verification and security purposes.

# There should be pictures of the room in the system.

**Success criteria:** Staff members and customers should be able to see a picture of the room, with accessories, and decoration. This is to get a brief idea of what you are paying for. Staff members should be able to add and edit the picture/s.

Pictures play a huge role when purchasing. This is before it gives the customer a sensation of what they could expect. Rooms in this instance are the main selling point. Therefore, I must make sure I incorporate this in the project.

Here are the prefixes used in “frmAddRoom”;

Textboxes = “txt”, button = “btn”, combox = “cbm” , checkbox = “rich textbox = “rtb” and data grid view = “dgv”.

A screenshot of a computer

Description automatically generated

In this form , staff members can add pictures but before that the staff members should have added rooms in the system to work. I will be explaining 3 different scenarios for adding rooms.

The first situation is when creating a new room. Firstly, for this to happen, the staff members must input the data into the com boxes, textboxes, and rich text boxes. Afterward, the staff members would need to press the “btnAddRoom” button to add the room to the database. Once the room has been added, the staff member should press on the “pictureboxroom” picture box to add the image using the base64 method. Once that’s done, if the data is validated, the following message box should show (“You have successfully added the picture to the database”). Otherwise, this message box should show up (“Sorry, there was an issue when adding this picture”).

On the other hand, in the second situation, the staff member should have already added a room. The staff member should select the “index” on the left of each room, for the room the staff member wants to add the image to. After clicking on the “index” the details of the room selected should show up. Once the details show up, the staff member should click on the “picutureboxroom” picture box and select the image for the room using base64. Once that’s done, if the data is validated, the following message box should show (“You have successfully added the picture to the database”). Otherwise, this message box should show up (“Sorry, there was an issue when adding this picture”).

Finally, if the staff member decides he wants to update the room picture because of some adjustments to the room, he may do that by first the staff member should have already added a room. The staff member should select the “index” on the left of each room, for the room the staff member wants to add the image to. After clicking on the “index” the details of the room selected should show up. Once the details show up, the staff member should click on the “picutureboxroom” picture box and select the image for the room using base64. Once that’s done, if the data is validated, the following message box should show (“You have successfully added the picture to the database”). Otherwise, this message box should show up (“Sorry, there was an issue when adding this picture”). Lastly the staff member should press the “btnUpdate” button to update the details.

Once the picture has been added to the database. Customers may see the picture on the form where all the rooms are shown.

Here are the prefixes used in “frmRoom1”;

Label = “lbl”, rich textbox = “rtb”, list box = “lst”

A screenshot of a computer

Description automatically generated

Here, the customer may see the picture of the room in the “pictureboxroom” picture box but before that the customer should select the room he wants to see the picture of, out of the rooms shown. This is shown in the “lstroom” listbox. As soon as the room is selected, the customer should see the details of the room the customer has selected along with its details.

# The system should display a brief description of the room with what the room comes with, for example, a coffee machine.

**Success criteria:** Staff members and customers should be able to see a brief description of the room and what it comes with. This also gives a brief idea of what you are paying for. Staff members should be able to add and edit the description.

.

# Customers should be able to edit saved information.

**Success criteria:** Customers should be able to edit saved information, for example, if a customer wants to change their name, they could do so in the form.

Customers should be able to change/ update their details. This is because it reduces the work for staff members so that they can deal with other matters with customers. Furthermore, it will enhance the customer experience. Therefore, I will be adding this to the program. Here are all the prefixes used in frmEditInfo and what they indicate:

textboxes = “txt”, buttons = “btn”, datetimepicker = “dat”. Below, there is the design for frmEditInfot;

A screenshot of a computer screen

Description automatically generated

The only value the customer may not change is the “customerID” for security and monitoring purposes. Here the user can press the button “btnBack” to redirect to the “frmLogIn” where users may input their login and proceed further on. Furthermore, the button “btnNew”, if the data is validated, will complete the sign-in process, and show the following message box (“You have successfully made an account”) in addition, all of the data entered will be stored in the database and the user will be redirect them to the “frmLogIn”. However, if all the textboxes are empty or data is not validated it should show the following message box (“Sorry, please double check that your information is right”).

# Customers should be able to see previous bookings.

**Success criteria:**  Customers should be able to select any booking that they customer made and display all the information about that booking.

Having a feature that can make you remember previous bookings is important as this will or will not make a customer come back to the hotel again. This is because it makes the client relax and know that they are going to experience the same experience that the customer went through previously. Therefore, I will be adding this feature to the program.

Here are the prefixes used and what they indicate in “frmRoom”.

Label = “lbl”, combo box = “cbm”, date time picker = “dat”, list box = “lst” button = “btn” and check box = “check”.

A screenshot of a computer

Description automatically generated

For this to work a customer must make a booking, otherwise, it will not work as the “listBooking” list will be empty.

To get started the customer should click on one of the previous bookings listed in the “ListBooking” list. After selecting a booking, the details of that booking will show in the same format as the “frmBooking” form (filling in the combo boxes, labels, check boxes and the date time pickers). I decided to do it in this manner because it is something that the customers will already see. Therefore, making my program as consistent as possible.

# Should be able to authorize staff to change the availability of the room.

**Success criteria:** Staff members should be able to change the availability of the room, for example, if the room is “occupied” and that person is no longer staying in staff members can change the “occupied” into “available”.

Changing the availability of the room is important because if there is a booking made through a call staff members can untick the availability box. This will change the booking status of that room as it will not be shown to customers when booking in “frmBooking” form.

Here are the prefixes used and what they indicate in “frmAddRoom”.

Combo box = “cmb”, textbox = “txt”, rich textbox = “rtb” button = “btn”, check box = "check” and data grid view = “dgv”.

A screenshot of a computer

Description automatically generated

There are 2 different scenarios to change to availability of the room.

The first scenario would be when creating a room. Staff members will fill in all the combo boxes, text boxes, and checkboxes. If the check box is ticked it means that the room is available. Otherwise, it is not available. Lastly, the staff member should press the “btnAddRoom”. This will add the room to the database, ready to book/ not book depending on the availability of the room.

On the other hand, the second scenario would be when updating the availability of an existing room. This can be done by using the “dgvrooms” data grid view. The staff member would have to click on the “index” on the left of each room to display all the data of that room. After getting all the details up, depending on the situation, the staff member can tick/ untick the “checkAvailibility” check box. Lastly, the staff member should press the “btnUpdate” button to update the details.

# Customers should be able to log out.

**Success criteria:** Once the customer has finished what they wanted to do, the customers should be able to log out or exit the system.

Logging out is a fundamental role of any system as it allows the user to protect their data and not allow another user to use that user's account. Therefore, I will be including it in my program.

Here are the prefixes used and what they indicate in “frmCustMain” form;

Label = “lbl”, button = “btn”,.

A screenshot of a computer

Description automatically generated

Here the customer should simply just press on the “btnLogOut” button shown in the “frmCustMain” only. After pressing the button, the customer will be redirected to the “frmLogIn” form where the customer will need to re-enter the details to log in again. Preventing other users from accessing that customer's account.

# Staff members should be able to log out.

**Success criteria:** Like the customer, once the staff member has finished what they wanted to do, the staff members should be able to log out or exit the system.

Logging out is a fundamental role of any system as it allows the user to protect their data and not allow another user to use that user's account. However, for staff members have a different form to log out as the staff members are more important to not let anyone use their accounts as it will empower that user and could lead to business failure. Therefore, I will be including it in my program.

Here are the prefixes used and what they indicate in “frmStaffMain” form;

Label = “lbl”, button = “btn”,.

A screenshot of a computer

Description automatically generated

Here the staff member should simply just press on the “btnLogOut” button shown in the “frmStaffMain” only. After pressing the button, the staff member will be redirected to the “frmLogIn” form where the staff member will need to re-enter the details to log in again. Preventing other users from accessing that staff member's account.

# Should authorize staff members to make changes to rooms.

**Success criteria:** Enable Staff members to make changes to the room eg bed type.

Updating room information is important as if there are any adjustments to make, it should be updated. For example, let's say a customer broke a king size bed and the hotel does not have enough money for another king-size bed, the hotel would buy a queen-size bed or below and update the room. Otherwise, the hotel would receive fines for false marketing, which could lead to business failure. Therefore, I will be adding this to the program.

Here are the prefixes used and what they are intended for in “frmAddRoom”;

Textboxes = “txt”, button = “btn”, combox = “cbm” , checkbox = “rich textbox = “rtb” and data grid view = “dgv”.

A screenshot of a computer

Description automatically generated

To update the room content, the staff member should first select the room that needs to be updated. then staff members should select the “index” on the left of each room, after clicking on the “index” the details of the room selected should show up. Once the details show up, the staff member should update the values and click on the “btnUpdate” button to update the values which will be later seen when looking at the information of the room in “frmRoom1”.

# Should display data in tables (data grid view).

**Success criteria:** The Right Data Should be displayed in data grid views.

Displaying data in tables is a quite simple but effective way to represent data. This is because of how easy to understand it is for both staff members and customers. Therefore, I will be adding this to the program.

Here are the prefixes used and what they are intended to do in “frmBooking”.

Date time picker = “dat”, combo box = “cmb”, data grid view = “dgv” button = “btn”.

A screenshot of a computer screen

Description automatically generated

Here, the use of “dgvroom” data grid view, effectively shows the rooms available, the price of room, the type of bed, etc…. This will enhance the customer experience as they can see every detail of the room in one place.

A grid with a number of squares

Description automatically generated with medium confidence

Here for the staff side of things, they can see all the details of the room booked and who booked them with the extra choices as well. Again it is a simple and effective way to communicate.

A diagram of a company

Description automatically generated

# Section 2: data structures and methods of access

I will be using 4 tables in the database. The four are Customer, Staff, Room, and booking. Each table will be in the 3rd normalized form. I will also be using multiple variables and arrays throughout the project

**ERD**

A grid with white rectangles and black text

Description automatically generated

All entities in ERD will be turned into tables:

**Customer Table:**

A grid with black and white squares

Description automatically generated

This table will hold all the customer data. Whenever new customers sign in, their data will be registered in this table. As a result, the number of customers will continue to increase. Furthermore, I will be accessing the SQL queries to the database such as INSERT INTO when a new customer signs up or UPDATE SET when updating customer data or even DELETE FROM when deleting a customers data.

**Staff Table:**

A black and white grid with black squares

Description automatically generated

This table will hold all the staff data. Whenever a new staff member sign in, their data will be registered in this table. As a result, the number of staff will continue to increase but not as fast as the customers. Furthermore, I will be accessing SQL queries to the database such as INSERT INTO when a new staff member signs in or UPDATE SET when updating the staff members' data or even DELETE FROM when deleting a staff members data.

**Room Table:**

A screenshot of a computer screen

Description automatically generated

This table will hold all the room data. I am planning on having 10 rooms in total which explains why I set RoomNumberID as Integer instead of AutoNumber. The strings do not have limits as most of these are going to be composed of combo boxes. Furthermore, I will be accessing SQL queries to the database such as INSERT INTO whenever creating a room or UPDATE SET when updating a room, or even DELETE FROM when deleting a room.

**Booking Table:**

A screenshot of a computer screen

Description automatically generated

This table will hold all the booking data. Whenever bookings are placed all the data will be stored here. As a result, the content will carry out expanding and growing per booking. Furthermore, I will also be accessing SQL queries such as INSERT INTO when a booking is placed or a SELECT FROM WHERE to see all the bookings a customer may have made.

# Section 3: Processing stages

Here I am going to show you how I break down problems and objectives into sub-problems to help build the project. I will also show Dataflow diagrams for the objectives that I have not mentioned in my “problem” below.

PROBLEM- I will show you how a staff member will create a new staff member and also update the price per night of room 2 to £32 and log out.

Top Down diagram

A diagram of a solution

Description automatically generated

Objective 3: log in staff

A diagram of a log in

Description automatically generated

**LOGING IN PSEUDOCODE**

SQL (Select \*

From staffTbl

Where StaffUsername = txtUsername and StaffPassword = txtPassword

If correct then

Show FrmMain

Else

“Incorrect Log in details”

Objective 5: Sign up for staff

A diagram of a company

Description automatically generated

**Sign Up Staff Pseudocode**

if txtFirstname.Lenght > 50 or txtSurname.Lenght >50 or txtEmail > 50 or txtTelNo > 10 then

“Sorry, please double check your details as they exceed out limit”

Else

SQL( Insert Into Staff Table

Values (Firstname, surname, email, dateofbirth, telephonenumber, username, password)

CurrentstaffID = currentstaffID + 1

Objective 18: Staff members should be able to make changes to rooms

A diagram of a software update

Description automatically generated

**Updating Price/ night Pseudocode**

SQL ( Update RoomTbl

Set PricePerNight = £32)

Else

“Sorry there was a problem when updating”

Objective 17: Staff members should be able to log out

A diagram of a log in form

Description automatically generated

**Loging Out Pseudocode**

If Button LogOut is pressed then

Show frmLogin

Objective 2: log in customer

A diagram of a log in

Description automatically generated

**Log in pseudocode**

SQL (Select CUsername, Cpassword

From CustomerTbl

Where CUsername = @Username & Cpassword = @Password)

If @Username = txtUsername & @Password = txtPassword then

Show frmLogin

Else

“Sorry, please double check your log in details”

End procedure

For the rest of the objective I will be showing a dataflow diagram.

Objective 4: Sign up for customer

A diagram of a account

Description automatically generated

Objective 8: staff should be authorized to add, edit, update, delete and search for customers.

A diagram of a diagram

Description automatically generated

Objective 10: Adding and displaying pictures

A diagram of a diagram

Description automatically generated

Objective 11: adding and displaying room description

A diagram of a data flow

Description automatically generated

Objective 12: Update customer information

A diagram of a software update

Description automatically generated

Objective 15: Change availability of room.

A diagram of a customer

Description automatically generated

Objective 16: Customer should be able to log out

A diagram of a customer pressing out but redirected to log in form

Description automatically generated

# Section 4: Validations

Validations are measures used to restrict data being entered by the user. This is done to minimize the chances of errors as they check certain values. For example, a minimum character/presence check to ensure the user has made a username, entered their first name, etc… This will ensure that we know at least the bare minimum about the user to track them in case there is any emergency. As a result, validations are crucial for the program, Therefore, I will be adding them ion the program.

frmLogIn:

A screenshot of a login form

Description automatically generated

Validations:

A green and white box with black text

Description automatically generated

frmCust:

A screenshot of a computer screen

Description automatically generated

Validations:

A table with text on it

Description automatically generated

frmStaffLogIn:

A screenshot of a computer screen

Description automatically generated

Validations:

A green and white box with black text

Description automatically generated

frmStaffMain:

A screenshot of a computer

Description automatically generated

Validations:

No validations required as there is no input.

frmCustMain:

A screenshot of a computer

Description automatically generated

Validations:

no validations required as there is no input.

frmStaff:

A screenshot of a computer

Description automatically generated

Validations:

A screenshot of a computer

Description automatically generated

frmCustomer:

A screenshot of a computer

Description automatically generated

Validations:

A screenshot of a computer

Description automatically generated

frmAddRoom:

A screenshot of a computer

Description automatically generated

Validations:

A screenshot of a data analysis

Description automatically generated

frmReport:

A grid with a number of squares

Description automatically generated with medium confidence

Validations:

no validations required as there is no input.

frmViewRooms:

A diagram of a diagram

Description automatically generated

Validation:

no validations required as there is no input.

frmRoom1:

A screenshot of a computer

Description automatically generated

Validations:

A green and black text

Description automatically generated

frmBooking

A screenshot of a computer screen

Description automatically generated

Validations:

A screenshot of a data analysis

Description automatically generated

frmRoom:

A screenshot of a computer

Description automatically generated

Validations:

A screenshot of a data analysis

Description automatically generated

frmEditInfo

A screenshot of a computer screen

Description automatically generated

Validations:

A table with text on it

Description automatically generated

Data Type Check- Ensures the correct data type has been entered.

Presence Check- Ensures data has been entered.

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a login form

Description automatically generated

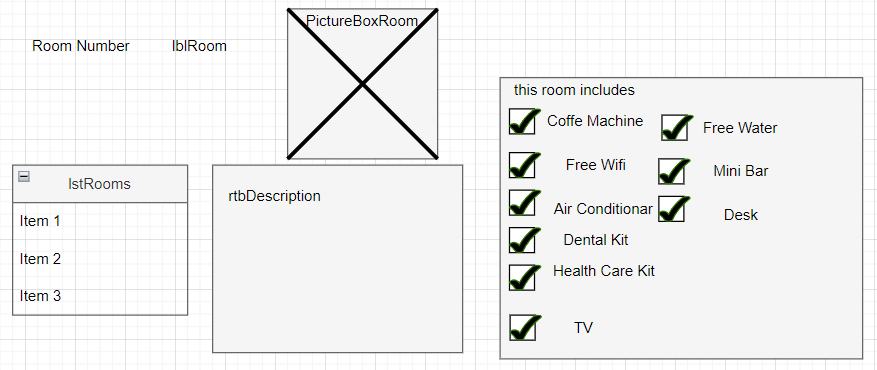
A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer screen

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A grid with a number of squares

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

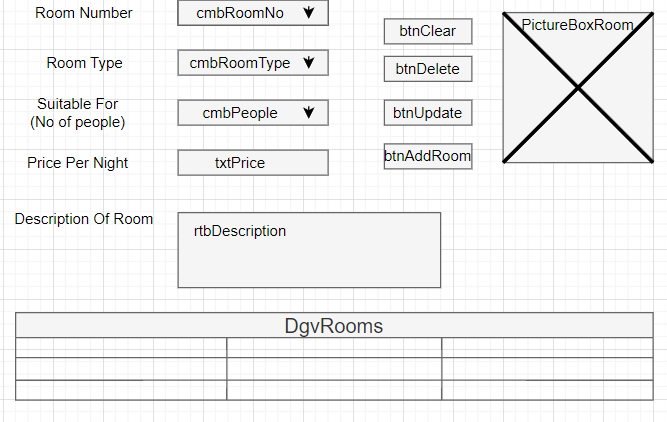
Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A diagram of a diagram

Description automatically generated

A screenshot of a computer screen

Description automatically generatedA screenshot of a login form

Description automatically generated