Section 5 - Post Prototype Refinement of Design

Table of contents:

Third party

The two third parties I will be choosing will be my Computer Science teacher Mr Coyles and my brother, who is in my Computer Science class, Dean Logan. The reason for this is due to the fact that both are familiar with the objectives of my prototype and have experience programming.

Feedback

1. Mr Coyles:
    - 1.1. Add CustID to the booking table to link customers to bookings.
    - 1.2. Addition of a calendar widget for the return date and loan date would aid user friendliness. There should be examples of these online you could integrate.
    - 1.3. If you plan to keep the separate window for adding/creating, then a close button would be nice along with a window title.
    - 1.4. Consider adding a more flexible search function. For example, a partial record search or similar. This could be achieved by a multi table query (inner joins) and use of the OR operator in an SQL statement.
    - 1.5. Consider using a try catch block for times when a user has not selected a record in the treeview.
2. Dean Logan:
    - 2.1. Having a separate window for adding a record is just bad design especially when the system is already using tabs. Just create a new tab for the user to enter this information.
    - 2.2. When you quit the main menu any other windows that are open will not close.
    - 2.3. When you hit the edit, add, delete and search buttons multiple times lots of windows open.
    - 2.4. When you first go onto the system make the welcome text bigger to fill the unused space.
    - 2.5. Change the colour to something better than grey.
    - 2.6. Some sort of pop up to the user informing them that the email has been sent. Just so the user knows that the email has definitely been sent rather than just hoping it has.
    - 2.7. Just a small detail which would be nice is changing where it says tk on the top to something more meaningful.
    - 2.8. A customer does not need to read the bill table. This is too complex, it would be better if this is taken out.
    - 2.9. Just storing the offer code is enough. You do not need to provide a discount and email customers offer codes and both of these would be too complex.
    - 2.10. The user should not be able to delete their own record as this could cause problems.
    - 2.11. Receptionists do not need to view the bill table as this is breaching the customer’s privacy.

Discussion of feedback

1. Mr Coyles
   1.1. I will implement the first suggestion made as this allows for the sending of the emails to the specific customer rather than to a random email. This will also allow me to include the customer's name within the email to make it more personal.
   1.2. To add the calendar widgets an external library will need to be downloaded as in the tkinter library these are not supported. This would improve my input as it will be easier for the user to enter the data. This will also assist in validation as the user will not be able to choose whether they enter dd/mm/yy or mm/dd/yy. The user will also not be able to make such mistakes as leaving out a '/', adding another '/' or adding '\' instead of '/'.
   1.3. Adding a close button and a title to the 'pop-up' window would assist the user as if they change their mind they will be able to close the window. Adding a title for the window will make my system more user-friendly as the user will be able to quickly identify what the purpose of each window. The table name of the table they are editing/adding to will be included in the title as this will also aid user-friendliness.
   1.4. Instead of replacing my current search function I will simply add an advanced search button. Within this advanced search they will be able to select any field within the table they are viewing and the system will return all the records which match this result. This will allow the user to search for information which a person is more likely to need to remember, for example a customer's name.
   1.5. The problem described in the final point is when a record is not selected and the 'Edit' button is selected an error will occur. A solution to this problem will be to and a try catch block which will stop the error from occurring. This will also allow a message box to be displayed explaining to the user that they need to select a record from the table.
2. Dean Logan
   2.1. The reason for the separate windows is due to the fact that I believe this is a better design than opening a new tab in the notebook. The reason for this is because the notebook is used to store the tables and therefore the user may be confused to see an add tab to appear beside the tabs that are used to select the table. Another disadvantage of using this approach is that the user cannot look at the table they are adding/editing too beside the entry forms. Using separate windows allows the user to close the window if they change their mind while using tabs instead this is not possible Taking into account these disadvantages, I decided to create pop-up windows instead of using this approach as I believe that this is not 'bad design' but a better design than that, that was suggested. Another method of input instead of 'pop-up' windows is simply clearing the screen and replacing this with the entry widgets. Although this method is better than the tab method as a back button could be used but it still has the disadvantage of not being able to see the table when adding/editing a record and this is why I will be implementing the 'pop-up' window for input and not these other two methods.

2.2. For the second point that was made I will need to close all windows that are open when the quit button is selected and not just the main window. This will further increase user-friendliness as when the user wants to exit the program all the windows will close and they don't need to close each tab.

2.3. The third point is another issue with my system which will need to be fixed as when this is done the user will not know what table each of the 'pop-up' windows belong too and this will cause confusion. This can be fixed by preventing the user from opening multiple 'pop-up' windows by closing the older one.

2.4. The unused space will be filled by a welcome message and a short explanation to the user on how to use the system and a brief description of the function of each button. This will increase user-friendliness as the user can read this and will clear up any ambiguity the system may have. I may add an extra button above the 'Quit' button which will return the user to this window.

2.5. I will not be changing the colour of my system due to the fact that I personally like the colour of the system. When picking the colour of my system I wanted to pick a colour which would result in the black text being clearly shown and when used in the dark would not strain the user's eyes. I believe the grey colour which I have chosen achieves these aims.

2.6. Point 6 is another great point as this will again aid user-friendliness. The reason it will do this as already stated in point 6 is that the user will know that an email should have been sent.

2.7. As stated already when discussing Mr Coyles' points above I will be changing the name of the 'pop-up' windows for the reasons discussed above.

2.8. I would have to agree with the feedback given here. This would be too complex to implement and I do not have the time to implement this functionality.

2.9. Once again I agree with this assessment as this would require creating a system to validate these offer codes and another system to create the offer codes. As I have a limited time on this project I do not believe doing this would be feasible and add little to the system. By keeping the 'offer code' as a field it is possible to add this functionality in my system if more time is given.

2.10. This is once again another good point but causes issues whether I implement this or not. If the user is able to delete their record as a customer this would result in the cancellation of the booking which the customer may not want but if the customer does want this to occur they would have to contact the hotel which is a nuisance. I have decided to do what is suggested as this has the added benefit of not allowing staff to delete their record which they should not be able to do, due to the fact that if they are deleting the account they are leaving the hotel and therefore need to inform the manager.

2.11. I will also listen to this advice as I believe, due to privacy concerns for the customer.

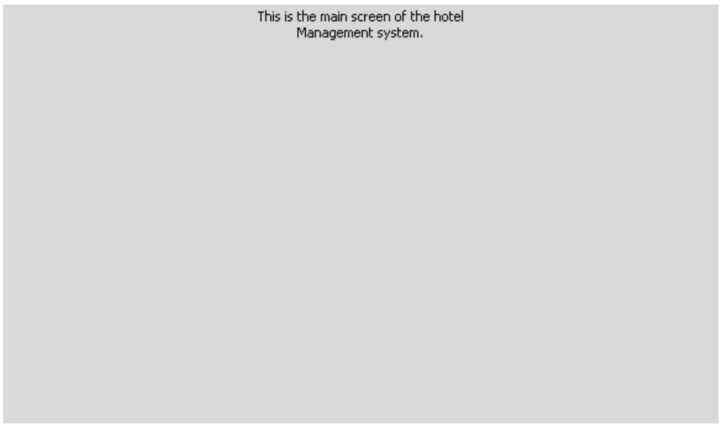
Changes to input/output

Email message box: (2.6)

As message boxes are controlled by tkinter I will only be able to change the message. I will be using an ‘info’ message box with the message ‘A confirmation email has been sent’.

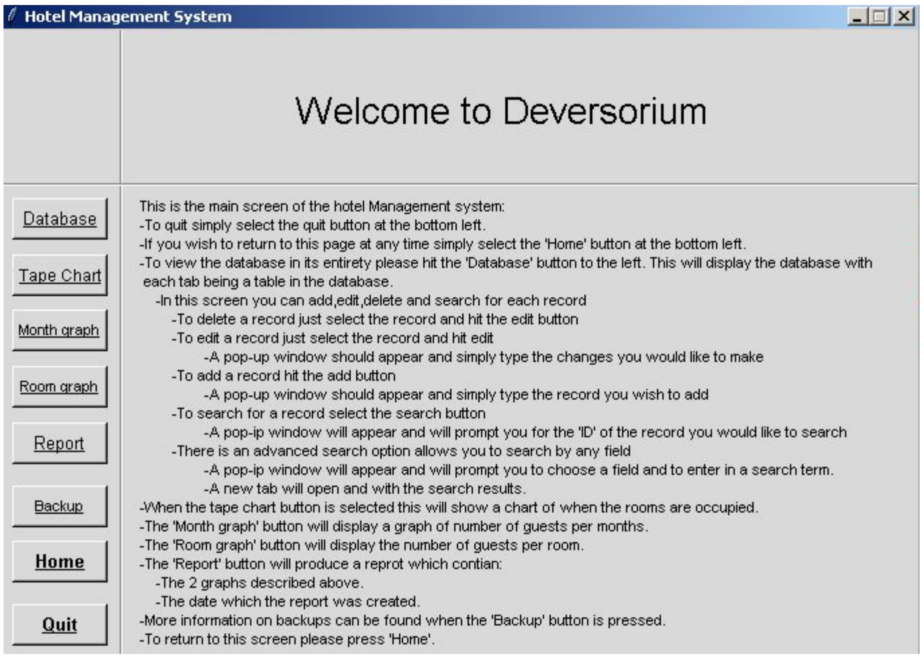
Welcome message: (2.4)

Currently my welcome message is very basic and therefore I plan to improve this by explaining what each system does.
Old welcome message (for management):



This is the main screen of the hotel
Management system.

New welcome message (for management):



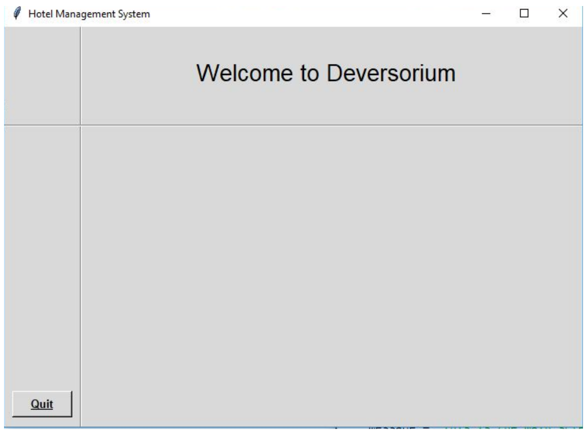
Welcome to Deversorium

```
This is the main screen of the hotel Management system:
-To quit simply select the quit button at the bottom left.
-If you wish to return to this page at any time simply select the 'Home' button at the bottom left.
-To view the database in its entirety please hit the 'Database' button to the left. This will display the database with
 each tab being a table in the database.
   -In this screen you can add,edit,delete and search for each record
      -To delete a record just select the record and hit the edit button
      -To edit a record just select the record and hit edit
            -A pop-up window should appear and simply type the changes you would like to make
      -To add a record hit the add button
            -A pop-up window should appear and simply type the record you wish to add
      -To search for a record select the search button
            -A pop-ip window will appear and will prompt you for the 'ID' of the record you would like to search
      -There is an advanced search option allows you to search by any field
            -A pop-ip window will appear and will prompt you to choose a field and to enter in a search term.
            -A new tab will open and with the search results.
-When the tape chart button is selected this will show a chart of when the rooms are occupied.
-The 'Month graph' button will display a graph of number of guests per months.
-The 'Room graph' button will display the number of guests per room.
-The 'Report' button will produce a reprot which contian:
   -The 2 graphs described above.
   -The date which the report was created.
-More information on backups can be found when the 'Backup' button is pressed.
-To return to this screen please press 'Home'.
```

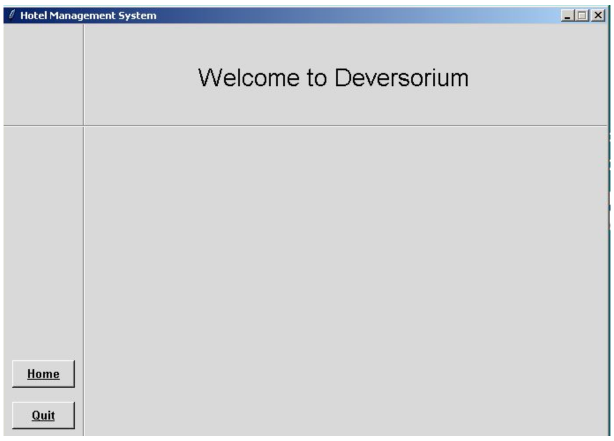
Main menu screen design (2.4)

I have decided to add a home button so the user can go back to the welcome screen.

Old Main menu:

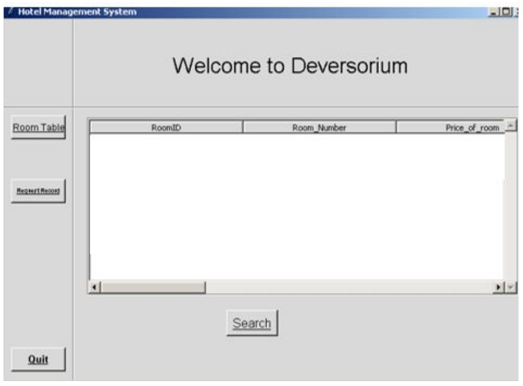


New Main menu:

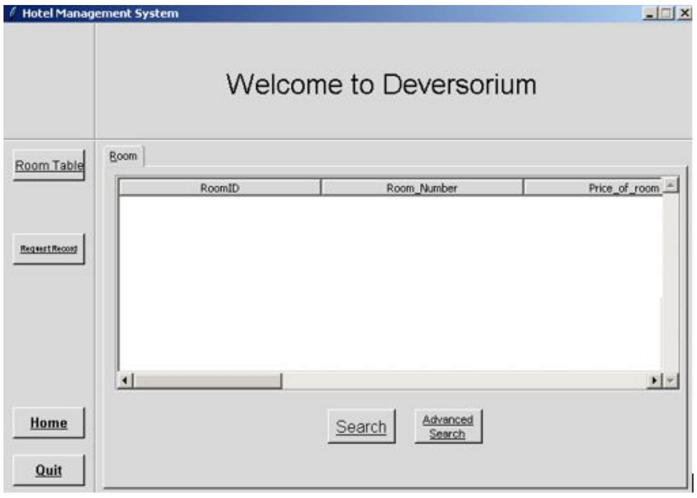


Read and Write permission and Read Only screen design screen design: (1.4)

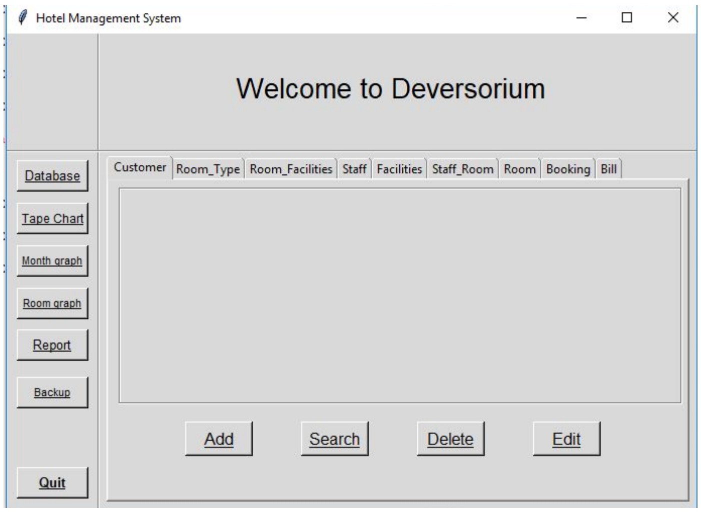
I have decided to change the 'Read and Write permission and Read Only permission' screen designs as the 'Search' button so it is more centred.
Old Read Only Permission:



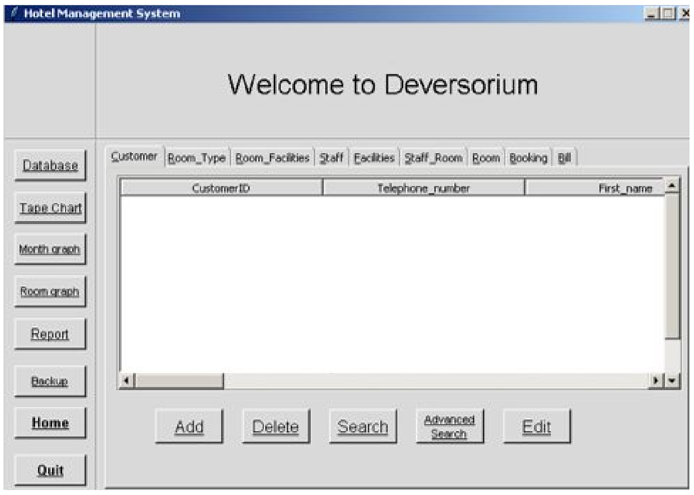
New Read Only Permission:



Old Read and Write Permission:

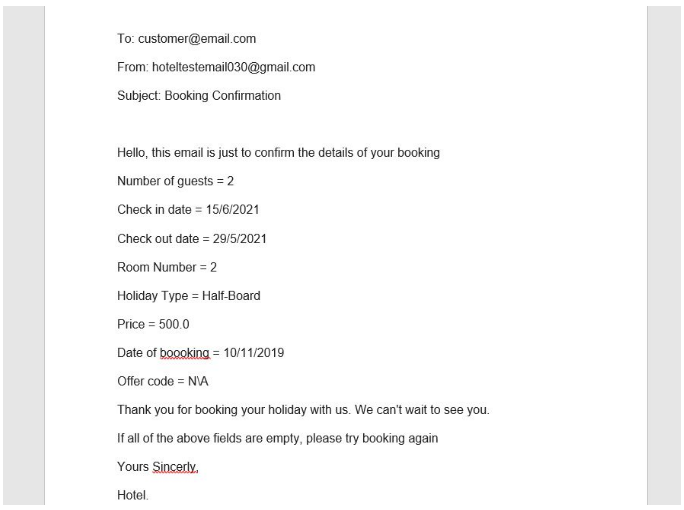


New Read and Write permission:



Email: (1.1)

To make the email more personal I have decided to include the first name and the surname of the customer that has booked the holiday.
Old email:



To: customer@email.com

From: hoteltestemail030@gmail.com

Subject: Booking Confirmation

Hello, this email is just to confirm the details of your booking

Number of guests = 2

Check in date = 15/6/2021

Check out date = 29/5/2021

Room Number = 2

Holiday Type = Half-Board

Price = 500.0

Date of boooking = 10/11/2019

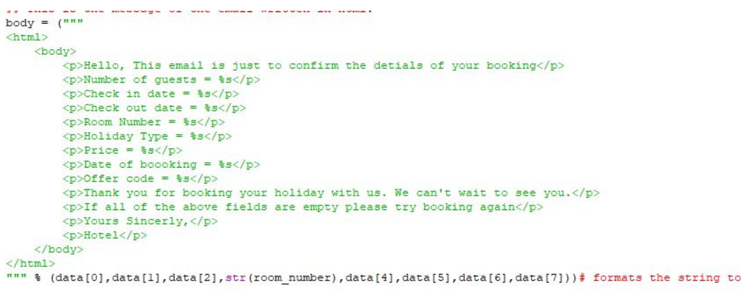
Offer code = N\A

Thank you for booking your holiday with us. We can't wait to see you.

If all of the above fields are empty, please try booking again

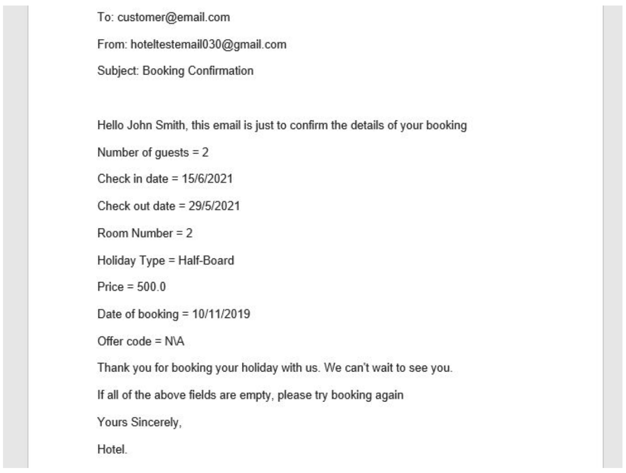
Yours Sincerly,

Hotel.



```
body = ("""
<html>
    <body>
        <p>Hello, This email is just to confirm the detials of your booking</p>
        <p>Number of guests = %s</p>
        <p>Check in date = %s</p>
        <p>Check out date = %s</p>
        <p>Room Number = %s</p>
        <p>Holiday Type = %s</p>
        <p>Price = %s</p>
        <p>Date of boooking = %s</p>
        <p>Offer code = %s</p>
        <p>Thank you for booking your holiday with us. We can't wait to see you.</p>
        <p>If all of the above fields are empty please try booking again</p>
        <p>Yours Sincerly,</p>
        <p>Hotel</p>
    </body>
</html>
""" % (data[0],data[1],data[2],str(room_number),data[4],data[5],data[6],data[7]))# formats the string to
```

New email:



To: customer@email.com

From: hoteltestemail030@gmail.com

Subject: Booking Confirmation

Hello John Smith, this email is just to confirm the details of your booking

Number of guests = 2

Check in date = 15/6/2021

Check out date = 29/5/2021

Room Number = 2

Holiday Type = Half-Board

Price = 500.0

Date of booking = 10/11/2019

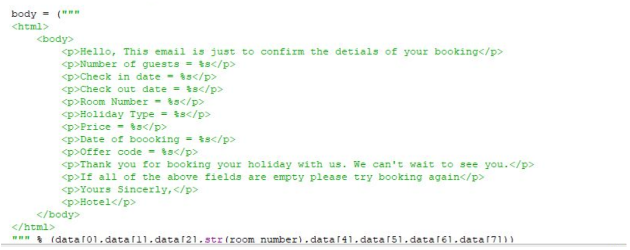
Offer code = N\A

Thank you for booking your holiday with us. We can't wait to see you.

If all of the above fields are empty, please try booking again

Yours Sincerely,

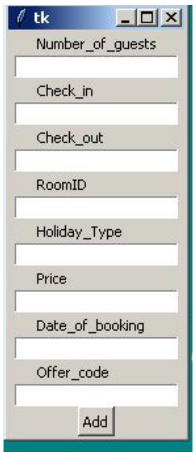
Hotel.



```
body = ("""
<html>
    <body>
        <p>Hello, This email is just to confirm the detials of your booking</p>
        <p>Number of guests = %s</p>
        <p>Check in date = %s</p>
        <p>Check out date = %s</p>
        <p>Room Number = %s</p>
        <p>Holiday Type = %s</p>
        <p>Price = %s</p>
        <p>Date of boooking = %s</p>
        <p>Offer code = %s</p>
        <p>Thank you for booking your holiday with us. We can't wait to see you.</p>
        <p>If all of the above fields are empty please try booking again</p>
        <p>Yours Sincerly,</p>
        <p>Hotel</p>
    </body>
</html>
""" % (data[0],data[1],data[2],str(room_number),data[4],data[5],data[6],data[7]))
```

‘Pop-up’ windows: (1.2, 1.3 and 2.7)

Old ‘pop-up’ windows:



Number_of_guests

Check_in

Check_out

RoomID

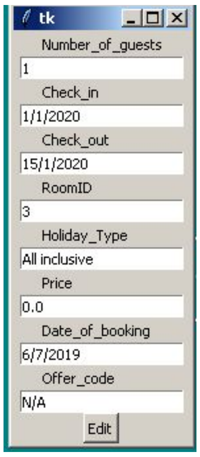
Holiday_Type

Price

Date_of_booking

Offer_code

Add



Number_of_guests: 1

Check_in: 1/1/2020

Check_out: 15/1/2020

RoomID: 3

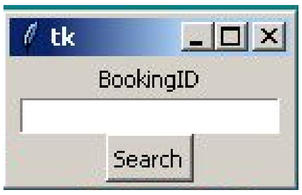
Holiday_Type: All inclusive

Price: 0.0

Date_of_booking: 6/7/2019

Offer_code: N/A

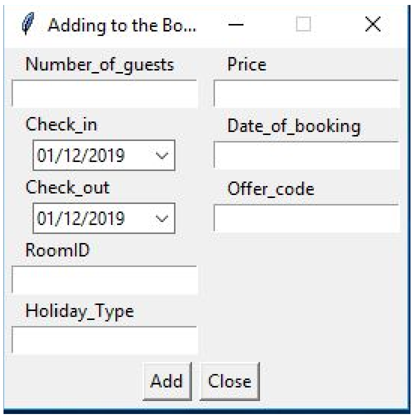
Edit



BookingID

Search

New ‘pop-up’ windows:


Adding to the Bo...

Number_of_guests

Price

Check_in: 01/12/2019

Date_of_booking

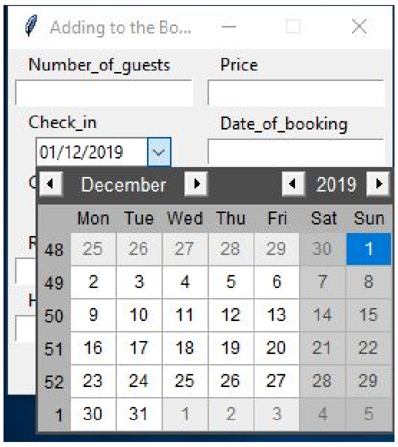
Check_out: 01/12/2019

Offer_code

RoomID

Holiday_Type

Add Close


Adding to the Bo...

Number_of_guests

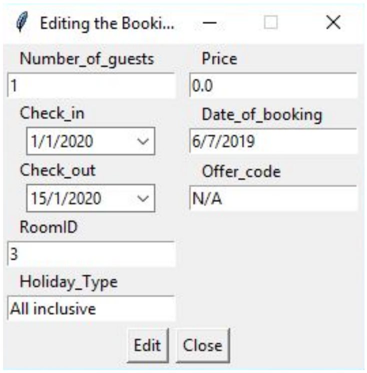
Price

Check_in: 01/12/2019

Date_of_booking

December 2019

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
48	25	26	27	28	29	30	1
49	2	3	4	5	6	7	8
50	9	10	11	12	13	14	15
51	16	17	18	19	20	21	22
52	23	24	25	26	27	28	29
1	30	31	1	2	3	4	5


Editing the Booki...

Number_of_guests: 1

Price: 0.0

Check_in: 1/1/2020

Date_of_booking: 6/7/2019

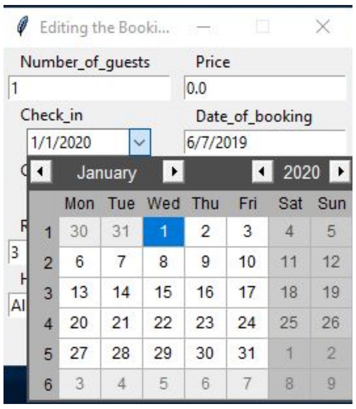
Check_out: 15/1/2020

Offer_code: N/A

RoomID: 3

Holiday_Type: All inclusive

Edit Close


Editing the Booki...

Number_of_guests: 1

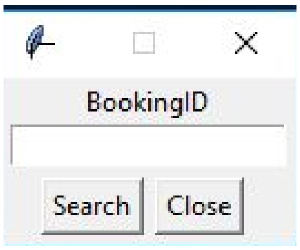
Price: 0.0

Check_in: 1/1/2020

Date_of_booking: 6/7/2019

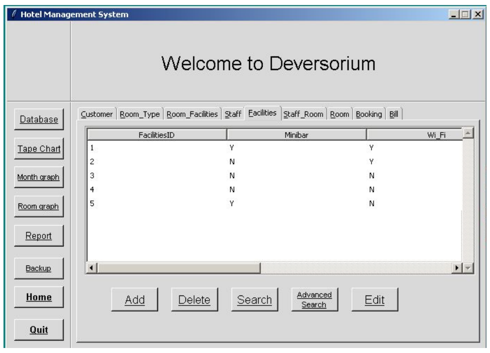
January 2020

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	30	31	1	2	3	4	5
2	6	7	8	9	10	11	12
3	13	14	15	16	17	18	19
4	20	21	22	23	24	25	26
5	27	28	29	30	31	1	2
6	3	4	5	6	7	8	9

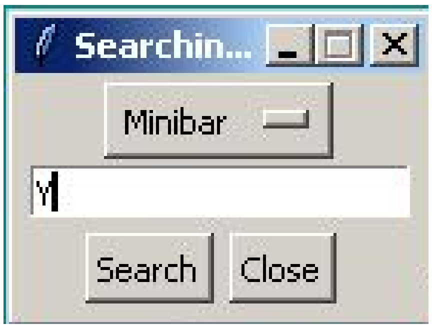

BookingID

Search Close

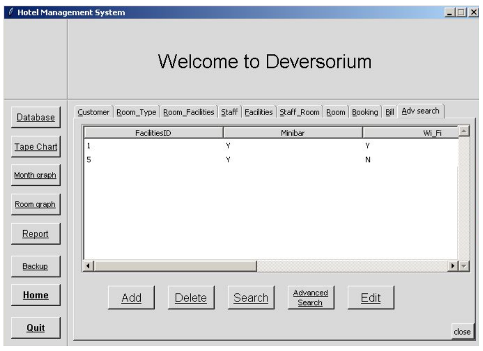
Advanced search (Manager screen): (1.4)



FacilitiesID	Minibar	Wi_Fi
1	Y	Y
2	N	Y
3	N	N
4	N	N
5	Y	N



Y



FacilitiesID	Minibar	Wi_Fi
1	Y	Y
5	Y	N

Changes to the data structure

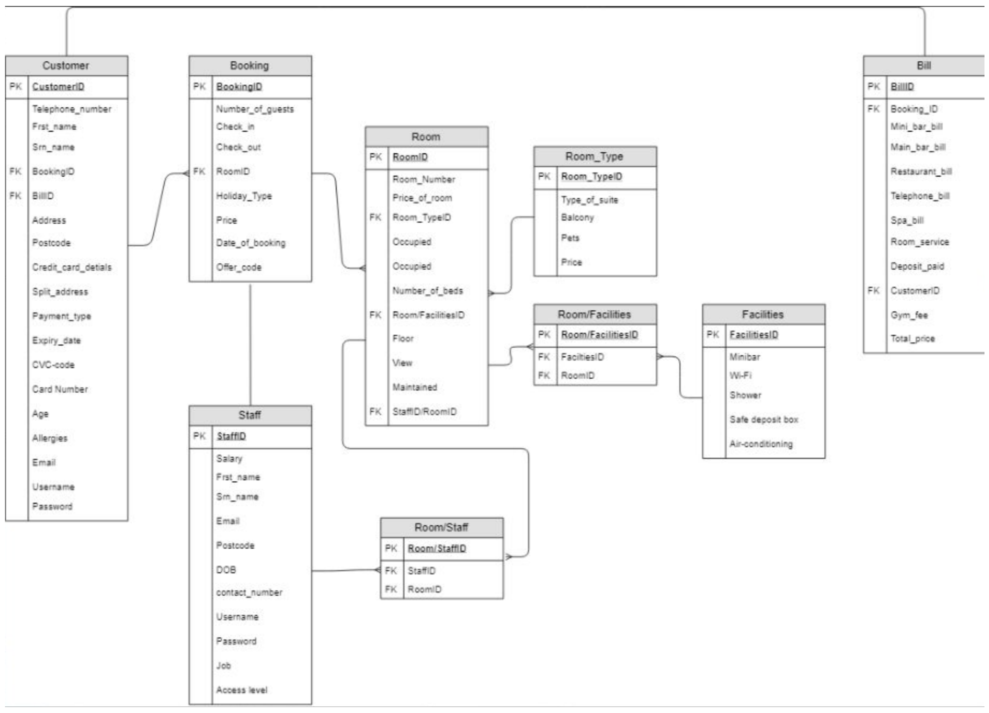
ER diagram: (1.1)

As suggested by Mr Coyles I will add ‘CustomerID’ to the booking table. This will allow me to send emails to a specific customer when a holiday is booked. There are several changes to the data structure that I am making after creating the database and realising that there are easier ways to do something or realising mistakes made. A change which I will have to make is by changing the ‘Age’ field in the customer table to ‘DOB’ (Date of Birth) as a customer’s age will change over time but their date of birth will not. The field ‘credit_card_detials’ will be removed as everything that this field could contain is within other fields. Another field within the customer table is the ‘split address' as there is already an ‘address’ field. I will also need to remove the Occupied field from the Room table as it does not include a date. To check if a room is occupied on a certain date I will need to check the booking’s table to see when a room will be occupied. The ‘access level’ field within the staff table can be removed as each job has a seperate access level and therefore this can be used instead.

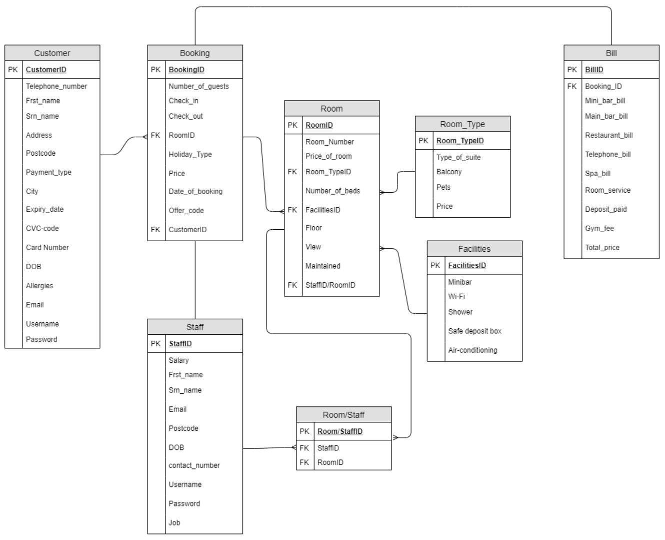
I am removing the room_facilities table as this linking table is not actually needed. There should be a one to many relationship between the facilities and room table as a set of facilities can be linked to many rooms but a room can only have one set of facilities. Therefore the ‘room_facilitiesID’ field in the room table will be replaced by the ‘facilitiesID’ field which is a foreign key linked to the facilities table.

The telephone number field will need to change its data type from an integer to a string. This is because when it is stored as an integer the first ‘0’ of the number is ignored by the system. If this field is stored as a string this problem will not occur. Another issue with the customer table is that the ‘BookingID’ should not be in this table as this results in a customer being unable to book a room more than once. The ‘billID’ field should also not be within the customer table as a bill is linked to a booking not a customer. As this is the case I am also removing ‘customerID’ from the bill table as these tables are already linked via the booking table. This is already done as the ‘bookingID’ is within the bill table. If this was linked to a customer then a customer can only have one bill and therefore one stay.

Old ER diagram:



Updated ER diagram:



Data dictionaries:

The changes made in these are to reflect the changes made in the ER diagram.

Old data dictionaries:

Customer:

Field	**Data Type**	**Data Format**	**Description**	**Example**	**Validation**
CustomerID	Integer		This is the primary key for this table.	1973	It should check if the data already exists in another field.
Telehone_number	Integer		This is the phone number that belongs to the customer.	075567826912	This will be a length check as all phone numbers have the same length.
Frst_name	String		This will be the first name of the customer.	Adam	This will be checked to see if it contains only alphabetic characters.
Surname	String		This will be the surname of the customer.	Logan	This will be checked to see if it contains only alphabetic characters.
BookingID	Integer		This is a forien key linked to the Booking table.	546	This will be checked if it exists within the Booking table.

BillID	Integer		This is a forien key linked to the Bill table.	245624	This will be checked if it exists within the Bill table.
Address	String	‘^\d+\s[A-z]+\s[A-z]+$’	This will be the home address of the customer.	16 Victoria Road	This will be checked to see if it fits the format mentioned in the ‘Data Format’ column.
Postcode	String	‘^[A-Z]{1,2}[0-9][A-Z0-9]? ?[0-9][A-Z]{2}$’.This will validate all UK postcodes with or without spaces but the letters will need to be upper case.	This will be the postcode of the customer.	BT78 9HJ	This will be checked to see if it fits the format mentioned in the ‘Data Format’ column.
Payment_type	String		This is how the customer is paying for their stay.	MaterCard	This field will be validated using a lookup check as there are only a certain amount of payment types the hotel can accept.
Card_Number	Integer		This will be the card number of the customer.	4111 1111 1111 1111	This will be checked using a type check as it will be checked if it is an integer.
Expiry_date	String (via the datetime library)	dd/mm/YY	This is the expiry date of the customer’s debit/credit card.	8/9/21	This will be checked to see if it fits the format mentioned in the ‘Data Format’ column. This will also be checked if the date is in the future.
CVC-code	Integer		This is the CVC-code of the customer’s card.	134	This will be checked using a type check as it will be checked if it is an integer.
City	String		This is the town/city/village where the customer lives.	Carrickfergus	A presence check will be used as the user will have entered a street name and therefore lives in a town, city or village.
Age	Integer		This will be the age of the customer.	38	This field will need to have a type check form of validation to check if it is an integer
Allergies	String		This will be a list of allergies the customer has.	nuts, shellfish	This will be a lookup check that will check a list of known allergies. ‘[milk,eggs, peanuts,tree

					nuts,fish,shellfish]’
email	“(^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$)”. This format will simply check if a character has been entered before and after the ‘@’ symbol and before and after the ‘.’.		This is the email of the customer	customer1@gmail.com	This field will have a format check to check if it is the correct format for an email.
Username	String		This will be the username of the customer which they will use to log onto the system.	alogan185	This will be checked to see if it matches any other usernames in the system.
Password	String		This will be the password of the customer which they will use to log onto the system.	Password123!	This field will be checked if it is above a certain amount of characters, if it contains uppercase and lowercase characters and if it contains alphanumeric characters.

Booking:

Field	**Data type**	**Data format**	**Description**	**Example**	**Validation**
BookingID	Integer		This is the primary key of the table.	123	This will be checked to make sure that the ‘BookingID’ entered does not already exist.
Number_of_guests	Integer		This is the number of guests per booking	5	The ‘number_of_guests’ field will need to have a type check form of validation to check if it is an integer.
Check_in	String (via the datetime library)	dd/mm/yy	This is the date at which the customer ‘checks in’	6/8/19	This will have a format check to check if they are in the correct format for a date.
Check_out	String (via the datetime library)	dd/mm/yy	This is the date at which the customer ‘checks out’	14/8/19	This will have a format check to check if they are in the correct format for a date.
RoomID	Integer		This is a forien key	441	This will be

			linked to the Room table.		checked if it exists within the Room table.
Holiday_Type	String		This stores whether the guest has booked an all inclusive stay or half board.	half board	This will have a lookup check. The options will be ‘all inclusive’, ‘half-board’,’full-board’,’Room Only’
Price	Float		This is how much the customer is being charged for their booking.	199.99	This field will be validated using a type check to check if it is a floating point or not
Date_of_booking	String (via the datetime library)	dd/mm/yy	This is the date the customer booked the room.	5/5/19	This will have a format check to check if they are in the correct format for a date.
Offer_code	String		This will be a code provided by the hotel that will allow the customer to get a discount of some sort.	658HFF9P6	A lookup check will be used to check if the offer code is valid.

Room:

Field	**Data type**	**Data format**	**Description**	**Example**	**Validation**
RoomID	Integer		This is the primary key of the table.	34153	This field must be checked if it is already exists within the table.
Room_Number	Integer		This is the number of the room.	111	This will be checked to see if it is an integer.
Price_of_room	Float		This is the price of the room per night.	400.00	This will be checked to see if it is a float.
Room_TypeID	Integer		This is a forien key linked to the Room_Type table.	43532	This will be checked if it exists within the Room_Type table.
Occupied	Boolean		This value will say if the room is occupied or not.	N	This will be checked to see if it is a boolean value.
Number_of_beds	Integer		This is the number of beds the room contains.	3	This will be checked to see if it is an integer.
Room/FacilitiesID	Integer		This is a forien key linked to the Room/Facilities table.	155	This will be checked if it exists within the Room/Facilities table.

Floor	Integer		This is the floor number of the room.	2	This will be checked to see if it is an integer.
View	String		This will store the type of view the room has.	Ocean	This will have a lookup check. The list that will be used is ‘Ocean’,’City’,’Garden’ and ‘Street’.
Maintained	Boolean		This will say if the room has been maintained i.e. has it been cleaned?,is everything working?	Y	This will be checked to see if it is a boolean value.
Staff/RoomID	Integer		This is a forien key linked to the StaffID/RoomID table.	2144	This will be checked if it exists within the StaffID/RoomID table.

New data dictionaries:

Customer:

Field	Data Type	Data Format	Description	Example	Validation
CustomerID	Integer		This is the primary key for this table.	1973	It should check if the data already exists in another field.
Telehone_number	String		This is the phone number that belongs to the customer.	075567826912	This will be a length check as all phone numbers have the same length.
Frst_name	String		This will be the first name of the customer.	Adam	This will be checked to see if it contains only alphabetic characters.
Surname	String		This will be the surname of the customer.	Logan	This will be checked to see if it contains only alphabetic characters.
BillID	Integer		This is a forien key linked to the Bill table.	245624	This will be checked if it exists within the Bill table.
Address	String	'^\d+\s[A-z]+\s[A-z]+$'	This will be the home address of the customer.	16 Victoria Road	This will be checked to see if it fits the format mentioned in the 'Data Format' column.
Postcode	String	'^[A-Z]{1,2}[0-9][A-Z0-9]??[0-9][A-Z]{2}$'.This will validate all UK postcodes with or without spaces but the letters will need to be upper case.	This will be the postcode of the customer.	BT78 9HJ	This will be checked to see if it fits the format mentioned in the 'Data Format' column.
Payment_type	String		This is how the customer is paying for their stay.	MaterCard	This field will be validated using a lookup check as there are only a certain amount of payment types the hotel can accept.
Card_Number	Integer		This will be the card number of the customer.	4111 1111 1111 1111	This will be checked using a type check as it will be checked if it is an integer.
Expiry_date	String (via the datetime library)	dd/mm/YY	This is the expiry date of the customer's debit/credit card.	8/9/21	This will be checked to see if it fits the format mentioned in the 'Data Format'

					column. This will also be checked if the date is in the future.
CVC-code	Integer		This is the CVC-code of the customer’s card.	134	This will be checked using a type check as it will be checked if it is an integer.
City	String		This is the town/city/village where the customer lives.	Carrickfergus	A presence check will be used as the user will have entered a street name and therefore lives in a town, city or village.
DOB	String (via the datetime library)	dd/mm/yyyy	This will be the date of birth of the customer. .	28/7/1975	This field will need to have a format check to check if it is a date.
Allergies	String		This will be a list of allergies the customer has.	nuts, shellfish	This will be a lookup check that will check a list of known allergies. ‘[milk,eggs, peanuts,tree nuts,fish,shellfish]’
email	“(^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$)”. This format will simply check if a character has been entered before and after the ‘@’ symbol and before and after the ‘.’.		This is the email of the customer	customer1@gmail.com	This field will have a format check to check if it is the correct format for an email.
Username	String		This will be the username of the customer which they will use to log onto the system.	alogan185	This will be checked to see if it matches any other usernames in the system.
Password	String		This will be the password of the customer which they will use to log onto the system.	Password123!	This field will be checked if it is above a certain amount of characters, if it contains uppercase and lowercase characters and if it contains alphanumeric characters.

Booking:

Field	**Data type**	**Data format**	**Description**	**Example**	**Validation**
BookingID	Integer		This is the primary key of the table.	123	This will be checked to make sure that the ‘BookingID’ entered does not already exist.
Number_of_guests	Integer		This is the number of guests per booking	5	The ‘number_of_guests’ field will need to have a type check form of validation to check if it is an integer.
Check_in	String (via the datetime library)	dd/mm/yy	This is the date at which the customer ‘checks in’	6/8/19	This will have a format check to check if they are in the correct format for a date.
Check_out	String (via the datetime library)	dd/mm/yy	This is the date at which the customer ‘checks out’	14/8/19	This will have a format check to check if they are in the correct format for a date.
RoomID	Integer		This is a forien key linked to the Room table.	441	This will be checked if it exists within the Room table.
Holiday_Type	String		This stores whether the guest has booked an all inclusive stay or half board.	half board	This will have a lookup check. The options will be ‘all inclusive’, ‘half-board’,’full-board’,’Room Only’
Price	Float		This is how much the customer is being charged for their booking.	199.99	This field will be validated using a type check to check if it is a floating point or not
Date_of_booking	String (via the datetime library)	dd/mm/yy	This is the date the customer booked the room.	5/5/19	This will have a format check to check if they are in the correct format for a date.
Offer_code	String		This will be a code provided by the hotel that will allow the customer to get a discount of some sort.	658HFF9P6	A lookup check will be used to check if the offer code is valid.
CustomerID	Integer		This is a forien key linked to the Customer table.	8968621	This will be checked if it exists within the Customer table.

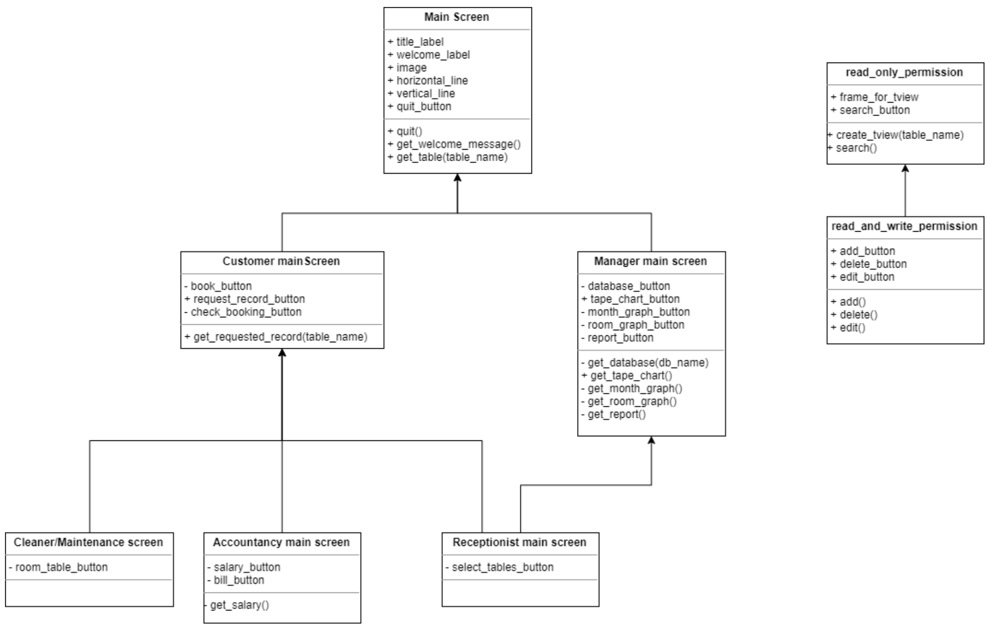
Room:

Field	**Data type**	**Data format**	**Description**	**Example**	**Validation**
RoomID	Integer		This is the primary key of the table.	34153	This field must be checked if it is already exists within the table.
Room_Number	Integer		This is the number of the room.	111	This will be checked to see if it is an integer.
Price_of_room	Float		This is the price of the room per night.	400.00	This will be checked to see if it is a float.
Room_TypeID	Integer		This is a forien key linked to the Room_Type table.	43532	This will be checked if it exists within the Room_Type table.
Number_of_beds	Integer		This is the number of beds the room contains.	3	This will be checked to see if it is an integer.
Room/FacilitiesID	Integer		This is a forien key linked to the Room/Facilities table.	155	This will be checked if it exists within the Room/Facilities table.
Floor	Integer		This is the floor number of the room.	2	This will be checked to see if it is an integer.
View	String		This will store the type of view the room has.	Ocean	This will have a lookup check. The list that will be used is ‘Ocean’,’City’,’Garden’ and ‘Street’.
Maintained	Boolean		This will say if the room has been maintained i.e. has it been cleaned?,is everything working?	Y	This will be checked to see if it is a boolean value.
Staff/RoomID	Integer		This is a forien key linked to the StaffID/RoomID table.	2144	This will be checked if it exists within the StaffID/RoomID table.

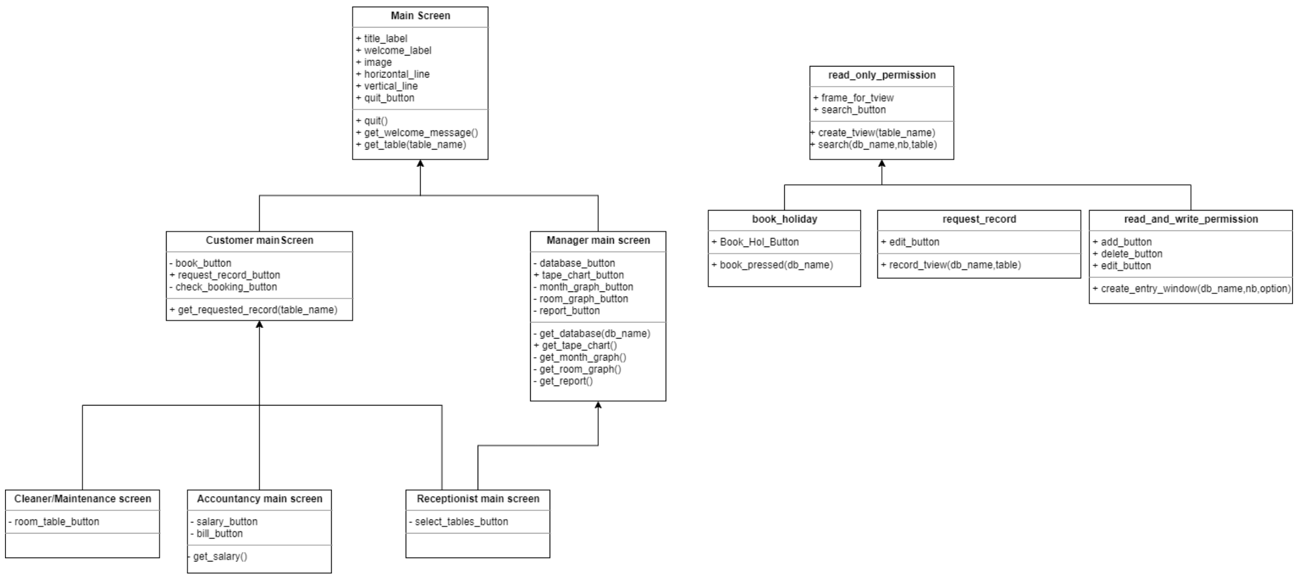
Class Diagram:

As I was implementing my class diagrams I realised that my class diagrams could be improved.

Old class diagrams:



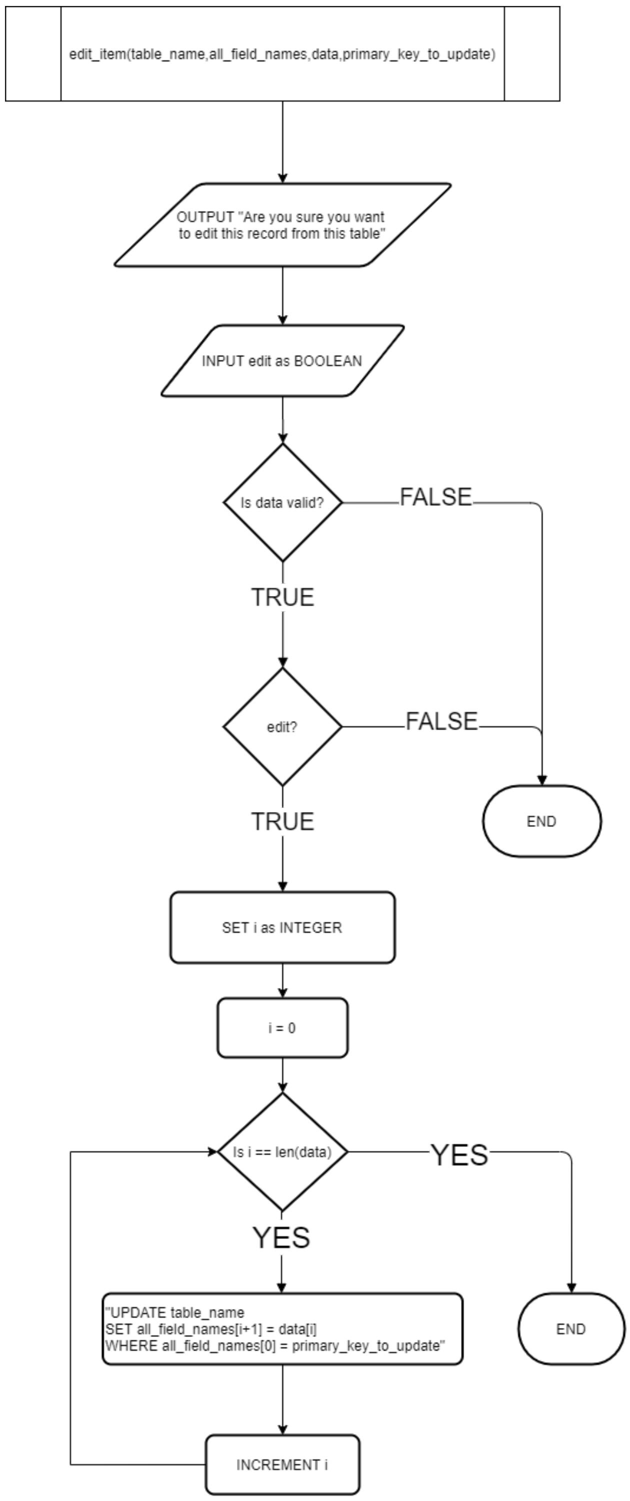
New class diagrams:



Processing Routines

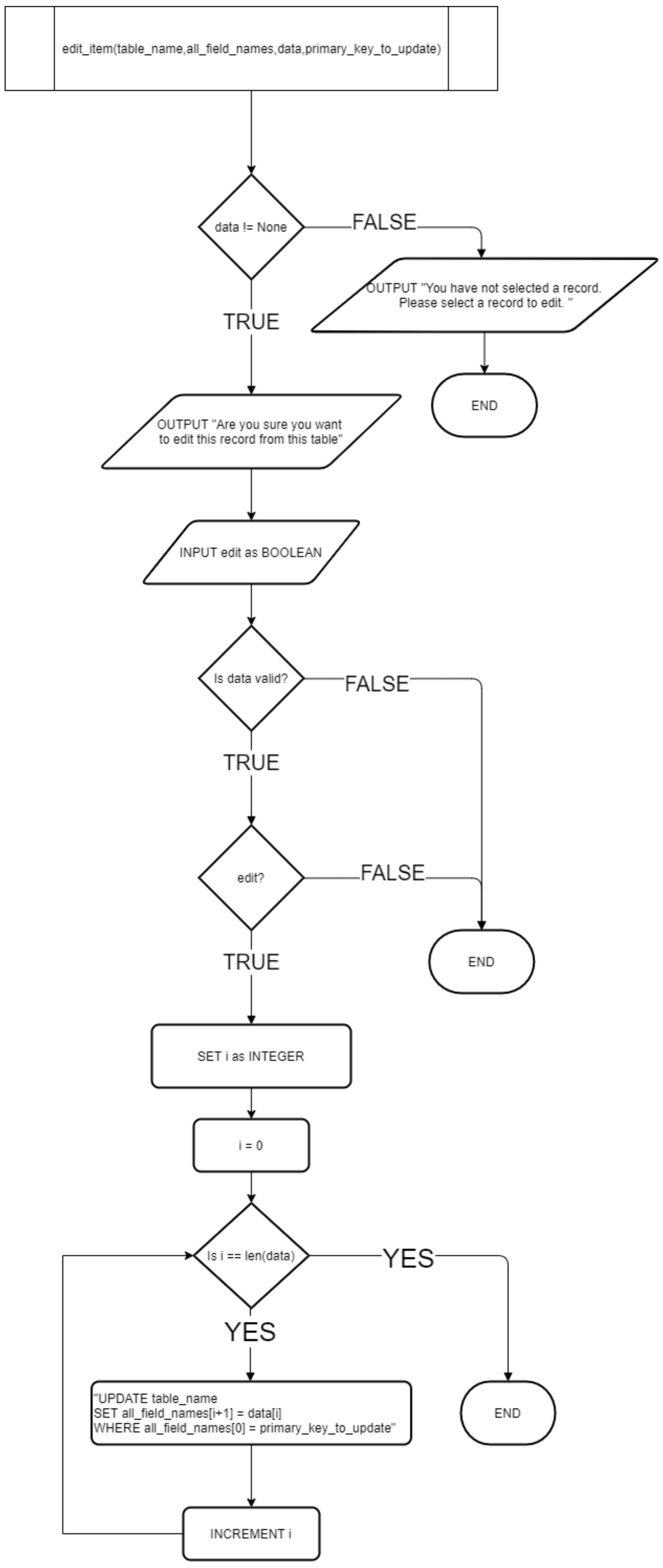
Edit

Old Edit Flowchart (1.5):

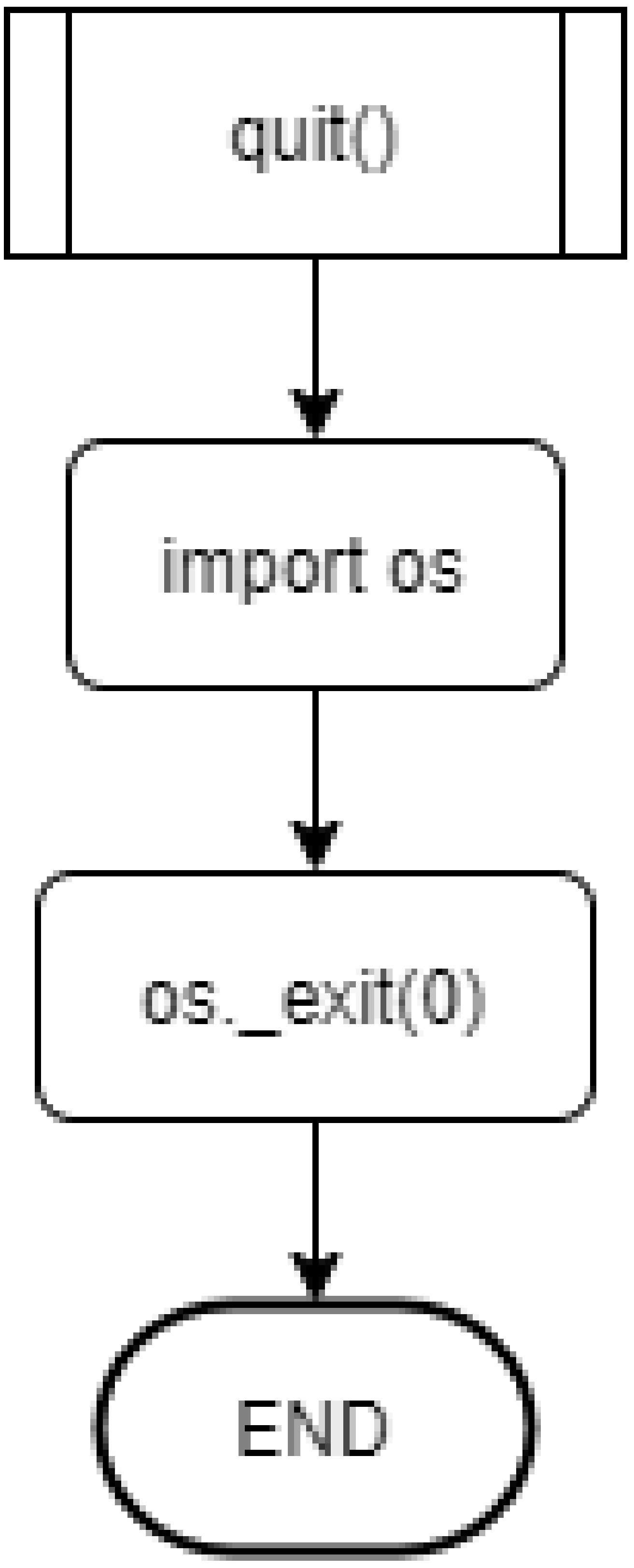


edit_item(table_name,all_field_names,data,primary_key_to_update)
OUTPUT "Are you sure you want to edit this record from this table"
INPUT edit as BOOLEAN
Is data valid?
FALSE
TRUE
edit?
FALSE
TRUE
END
SET i as INTEGER
i = 0
Is i == len(data)
YES
YES
"UPDATE table_name SET all_field_names[i+1] = data[i] WHERE all_field_names[0] = primary_key_to_update"
END
INCREMENT i

New Edit flowchart:

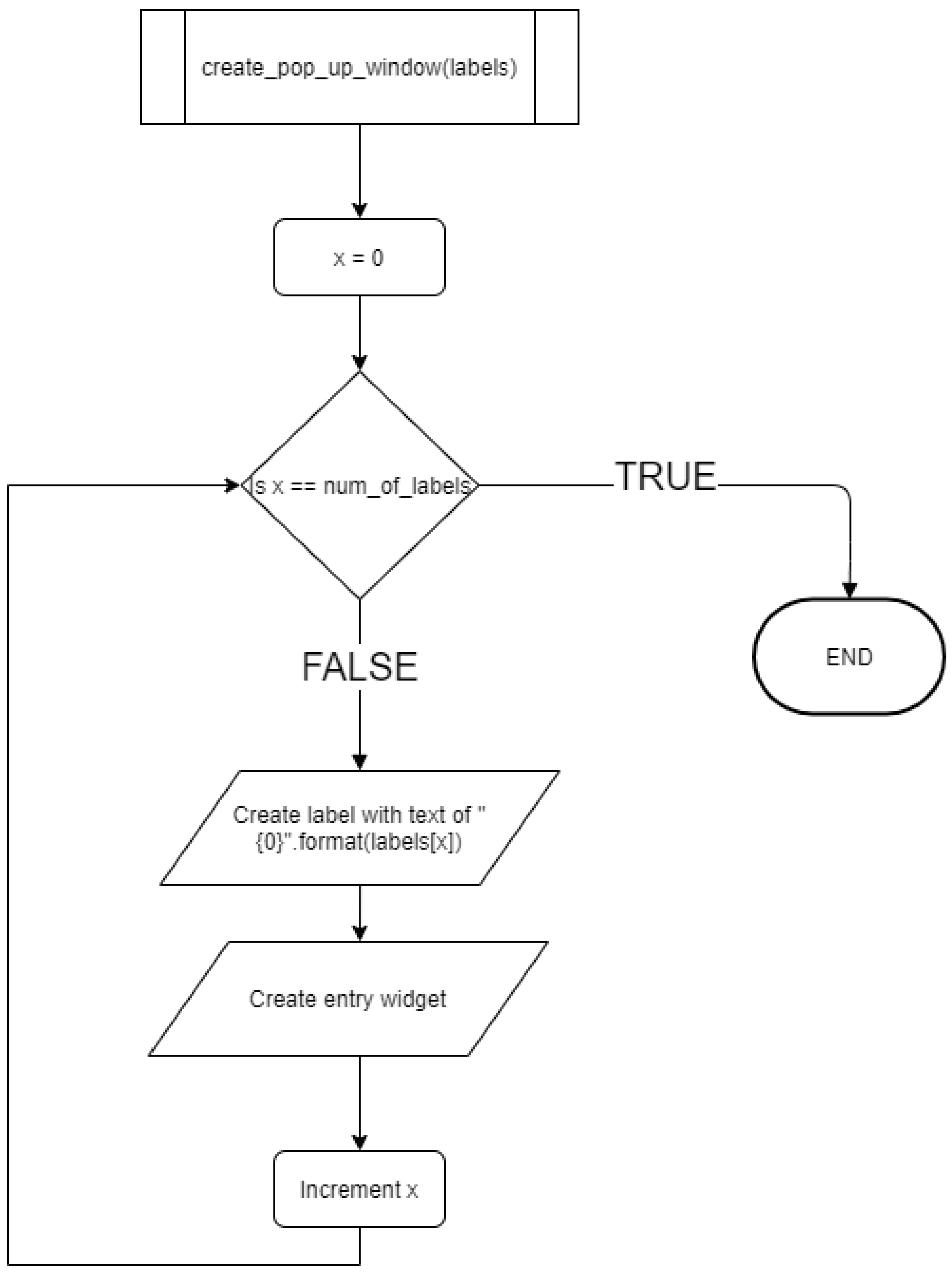

edit_item(table_name,all_field_names,data,primary_key_to_update)
data != None
FALSE
OUTPUT "You have not selected a record. Please select a record to edit. "
END
TRUE
OUTPUT "Are you sure you want to edit this record from this table"
INPUT edit as BOOLEAN
Is data valid?
FALSE
TRUE
edit?
FALSE
END
TRUE
SET i as INTEGER
i = 0
Is i == len(data)
YES
YES
END
"UPDATE table_name SET all_field_names[i+1] = data[i] WHERE all_field_names[0] = primary_key_to_update"
INCREMENT i

Quit (2.2):

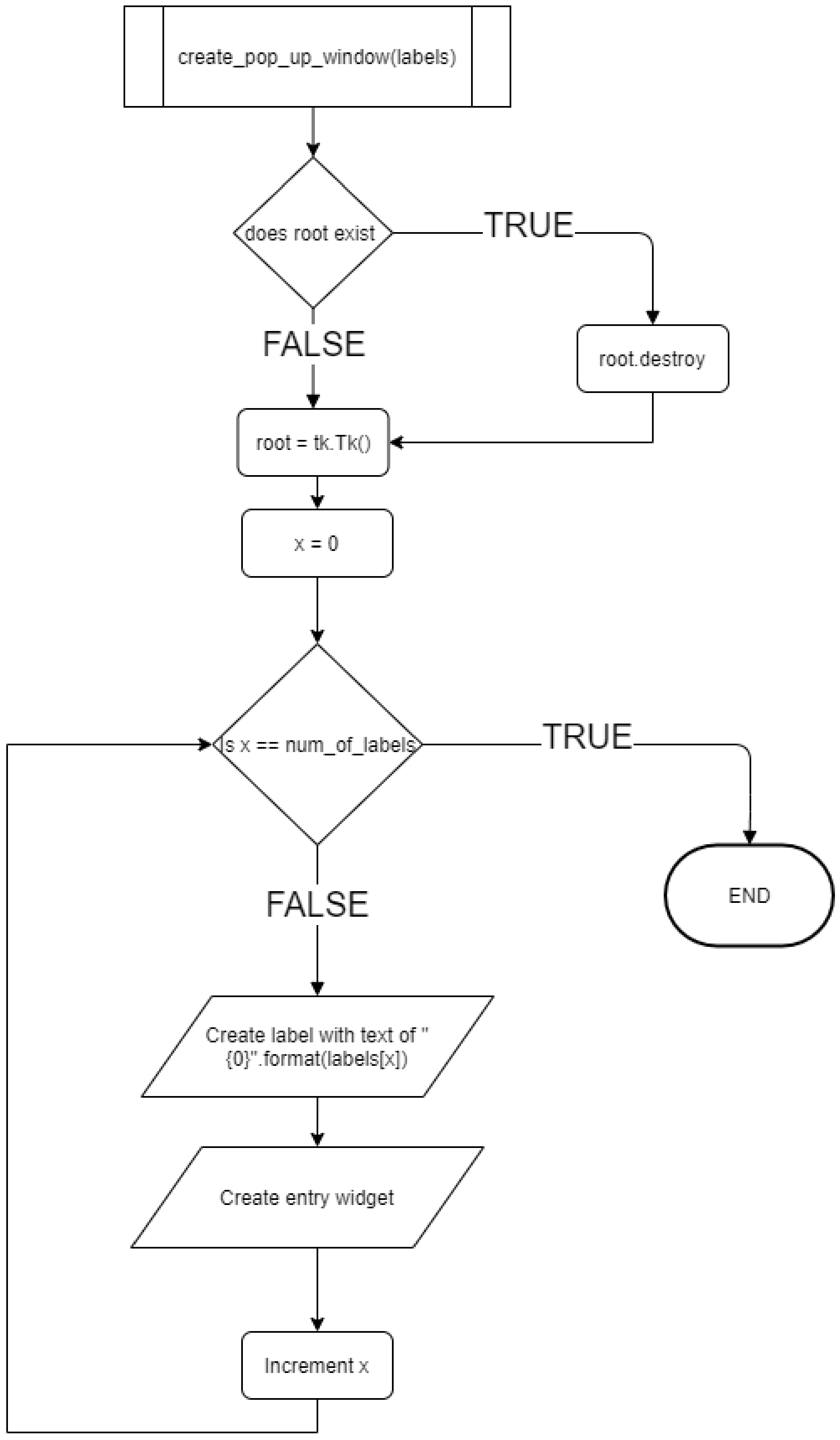


Creating pop-up window

Old Creating pop-up window (2.3)


create_pop_up_window(labels)
x = 0
Is x == num_of_labels
TRUE
END
FALSE
Create label with text of "{0}".format(labels[x])
Create entry widget
Increment x

New creating pop-up windows


create_pop_up_window(labels)
does root exist
TRUE
FALSE
root.destroy
root = tk.Tk()
x = 0
is x == num_of_labels
TRUE
FALSE
END
Create label with text of "{0}".format(labels[x])
Create entry widget
Increment x