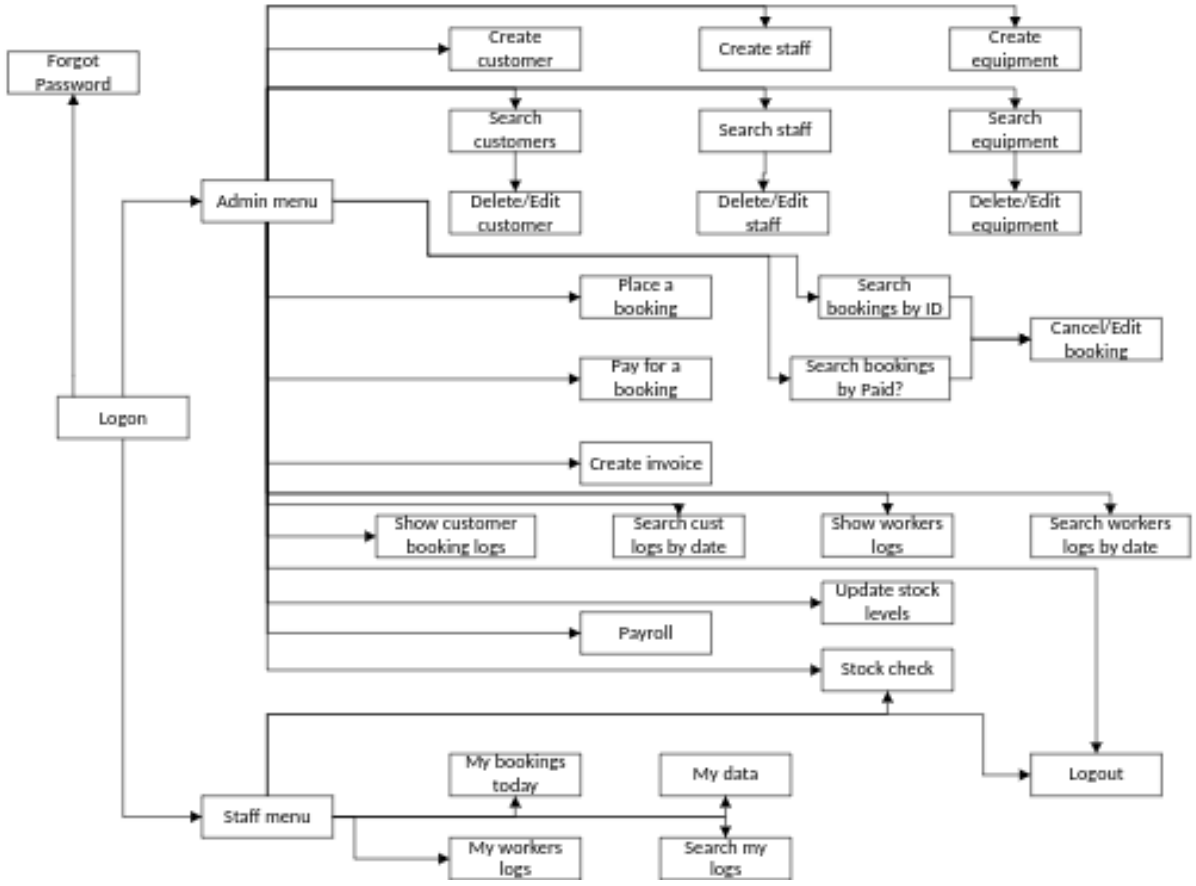
Design

Overview of the system

Below is the overview of access for the system:



For All Forms

Font: Calibri

Font size: 11

Heading size: 20

“Adding” menu colours: Yellow and golden

“Editing/Searching” menu colours: Dark and light pink

General menu colours: Dark and light blue

Other function menu colours: Dark and light green

DATA STRUCTURE TABLES

1. Customer table

File description - The file stores all customer related data who have registered when placing an order with Handy Man Hiring

File capacity - The number of items in the file increases as more customers register and can be decreased if a customer account is deleted

File access - Customers will have access to their own personal data and will be able to edit their data as the text file will be serially accessed. The secretary and manager will also have access to the customer files.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Customer ID **	String	6	AAQE05	Presence check; not equal to " "
Title	String	4	Miss	List check; "Miss","Mr","Mrs","Dr","Ms"
First name	String	15	John	
Surname	String	15	Tanner	
1st Address Line	String	20	117 Swansea Avenue	
2nd Address Line	String	20	Flat 202	
Postcode	String	8	SA11 1HL	
Town/City	String	15	Neath	
Phone Number	String	11	07951142805	Length check; len(num) = 11
Email	String	30	StockingM@ hotmail.com	Format check; like *@*.*
DoB	Date	10	12/10/2001	Format check; like **/**/****

Primary key = **

Foreign key = ^^

2. Staff table

File description - The file will hold the staff data of the workers at Handy Man Hiring

File capacity - The number of items increases/decreases depending on the number of current workers

File access – A staff members details can be accessed by the individual staff member but also by both the secretary and manager. The files can be edited as the text files will have serial access.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Staff ID **	String	6	WKPO39	Presence check; not equal to " "
Title	String	4	Dr	List check; "Miss","Mr","Mrs","Dr","Ms"
First name	String	15	Lucy	
Surname	String	15	Beckett	
1st Address Line	String	20	68 West Avenue	
2nd Address Line	String	20	Flat 304	
Postcode	String	8	SA12 6HF	
Town/City	String	15	Port Talbot	
Phone Number	String	11	07951334705	Length check; len(num) = 11
Email	String	30	Beckett998@gmail.com	Format check; like *@*.*
DoB	Date	10	12/04/1985	Format check; like **/**/****
Speciality	String	15	Welding	List check; like Welding, Woodwork, Plumbing, Carpentry, Electrician, Gardening, Roofing, Other

Primary key = **

Foreign key = ^^

3. Payroll table

File description - This file will hold the details of the payroll of a specific staff member. For each month, it will hold a list of all the staff member's booking's IDs, the number of hours they worked and the wage for the month

File capacity - The number of items depends on the number of staff members

File access – The file can be accessed by the individual worker, the secretary and the manager. It will also be able to be edited as it will be accessed serially. Staff can also access their own payroll data.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Payroll ID **	String	5	HE127	Presence check; not equal to " "
Staff ID ^^	String	5	WR934	
Month	Date	9	September	Length check; > 3 and <10
Hours worked in month	Integer	3	160	Range check; 30 to 200
Wage for the month	Real	4	£954.50	

Primary key = **

Foreign key = ^^

4. Bookings table

File description - The file will hold the details of all of the bookings that have been made by customers, past and current. It will be linked to the customer, staff and invoice files via the corresponding IDs. It will include job specific details such as the date and notes

File capacity - The number of files will increase as customers create bookings and will only ever decrease if a customer cancels a booking (it cannot be deleted after the booking was completed)

File access - The secretary, manager, staff members who are involved in the bookings will have access to the file. The files can be edited as they will be serially accessed. The files will also be able to be viewed by the customers who's booking it is.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Booking ID **	String	6	WK0393	Presence check; not equal to " "
Job type	String	11	Plumbing	List check; like Welding, Woodwork, Plumbing, Carpentry, Electrician, Welding, Gardening, Roofing, Other
Date	Date	10	18/09/2021	Format check; like **/**/****
Customer ID ^^	String	5	HS462	
Staff ID ^^	String	5	WK453	
Time taken (hours)	Real	2	4	Range check; >0.1 and <100
Price	Real	4	£300	
Tool details	String	300	Hedge trimmers, wheelbarrow, sickle	
Job specific notes	String	300	Key under plant pot, kitchen is first door to your right	

Primary key = **

Foreign key = ^^

5. Invoices table

File description - The file will hold all the bookings from a specific invoice which will be linked by the booking IDs. It will also include the total price of the invoice and whether it has been paid for (I.e. whether all the bookings have been paid for)

File capacity – The number of items will increase as more bundles of bookings are created

File access - The file can be accessed by the secretary, manager, and the customer who's invoice it is, can access the file. However, the customer can only edit whether the invoice has been paid for by paying for the bookings.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Invoice ID **	String	6	IP7321	Presence check; not equal to " "
Booking ID ^^	String	70	WK7483, GG4382, TW3234	Presence check; not equal to " "
Invoice paid?	Boolean	1	True	List check; True or False

Primary key = **

Foreign key = ^^

6. Equipment table

File description - The file holds the details of all the tools and equipment that the company possesses. Limits are set for what the maximum and minimum amounts of stock

File capacity - The number of items can be varied depending on how many different items Handy Man Hiring has. This is independent to the number of items in stock

File access - The file can be accessed by all the staff members, including the secretary and manager. The files can be edited as they are serially accessed.

Field Name	**Data Type**	**Data Size**	**Example**	**Validation**
Equipment ID **	String	4	23	Presence check; not equal to " "
Name	String	25	Shovel	Presence check; not equal to " "
Amount in stock	Integer	3	124	Range check; >0 and <1000
Reorder stock level	Integer	3	5	
Reorder stock amount	Integer	3	20	
Last test date	Date	10	03/04/2021	Format check; like **/**/****
Next test date	Date	10	03/10/2021	Format check; like **/**/****
Tested by	String	30	Jack Samuel	

Primary key = **

Foreign key = ^^

7. Password table

File description – The file holds the details of the staff members login details

File Capacity – Equal to the number of staff members

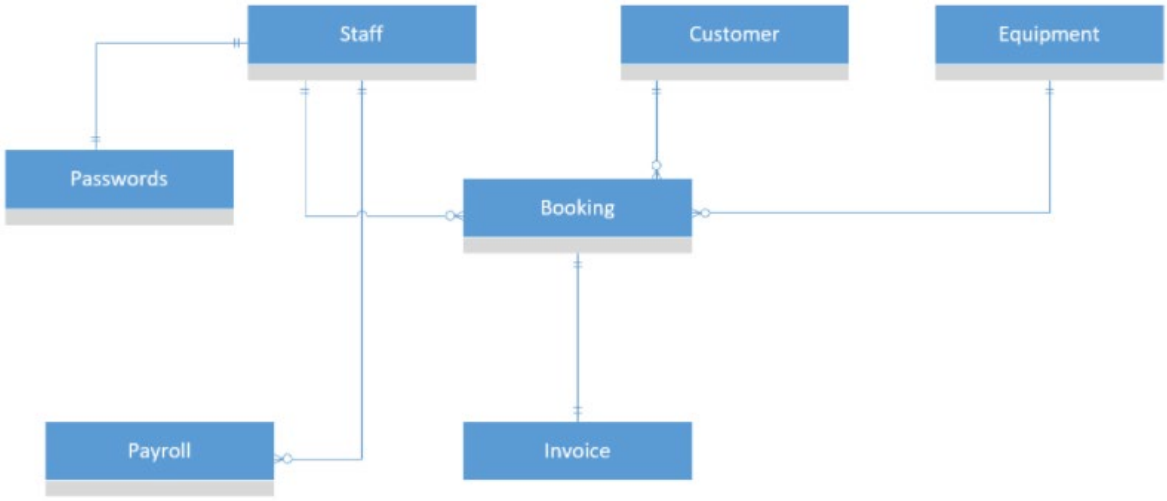
File Access - These will only be accessed by admin users. Staff members must request admin users to alter their password details if they wish so.

Field Name	Data Type	Data Size	Example	Validation
Password ID**	String	5	ER231	
Staff ID ^^	String	15	TY321	
Password	String	15	62149#	Presence check: Not equal to " "
Staff type	String	6	Worker	List check: "Worker", "Admin"
Clue	String	50	The name of your first dog	

Primary key = **

Foreign key = ^^

ER DIAGRAM



- one to one:
- one to many:

DATA ACCESS TYPES

Textfiles (such as Customers.txt, Staff.txt, Payroll.txt, Equipment.txt, Passwords.txt, and Bookings.txt) will have serial access meaning that the program will only be able to access them one at a time.

Arrays (such as ArrayCustomers, ArrayStaff, ArrayPayroll, ArrayEquipment, ArrayPasswords, and ArrayBookings) will have random access meaning that the program will be able to access and use multiple arrays at the same time.

ARRAYS

ArrayCustomers(CustomerID** char(6), Title, Fname, Sname, FAddress, SAddress, Postcode, Town, Phone, Email, DoB Date(10))

ArrayStaff(StaffID**, Title, Fname, Sname, FAddress, SAddress, Postcode, Town, Phone, Email, DoB Date(10), Speciality)

ArrayEquipment(EquipmentID** char(6), Name, Amount int(3), reorderLevel int(3), reorderAmount int(3), LTestDate date(10), NTestDate date(10), TestedBy)

ArrayPasswords(PasswordID**, StaffID^^, Password, StaffType, Clue)

ArrayBookings(BookingID**, JobType, Date date(10), CustomerID^^, StaffID^^, Time, Price, Tools, Notes)

ArrayPayroll(PayrollID**, StaffID^^, Month, HoursWorked, MonthsWage)

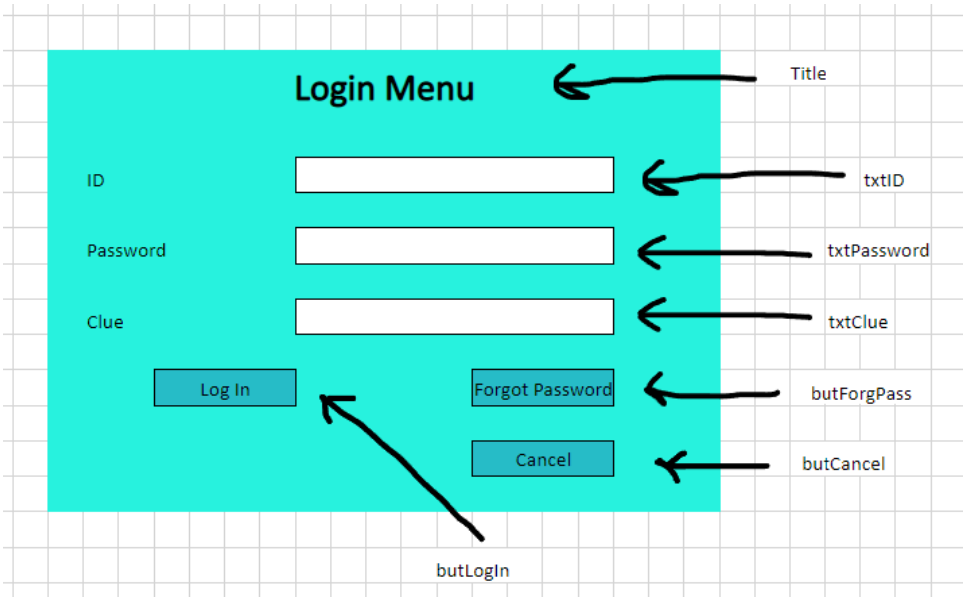
** = Primary key

^^ = Foreign key

FORM DESIGNS

Key: Green = Global variable

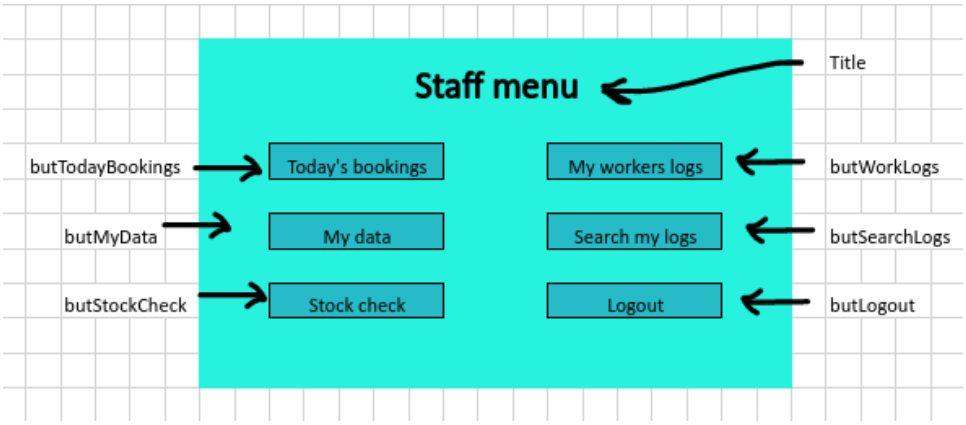
Login/Sign-up




```
Declare ID as string
Declare LoggedInID as global
Declare WorkerAccount as boolean = False
Declare password as string
Declare count as integer = 0
Load ArrayPassword[]

txtID = ID
txtPassword = Password

If butLogin clicked:
        While Count < 3:
                For x in ArrayPassword[x] :
                        If ID = ArrayPassword[x,1] and Password = ArrayPassword[x,2]:
                                If ArrayPassword[x,3] = “Worker” then
                                        LoggedInID = ID
                                        WorkerAccount = True
                                        Open Staff menu
                                Else
                                        Open Admin menu
                        Else
```

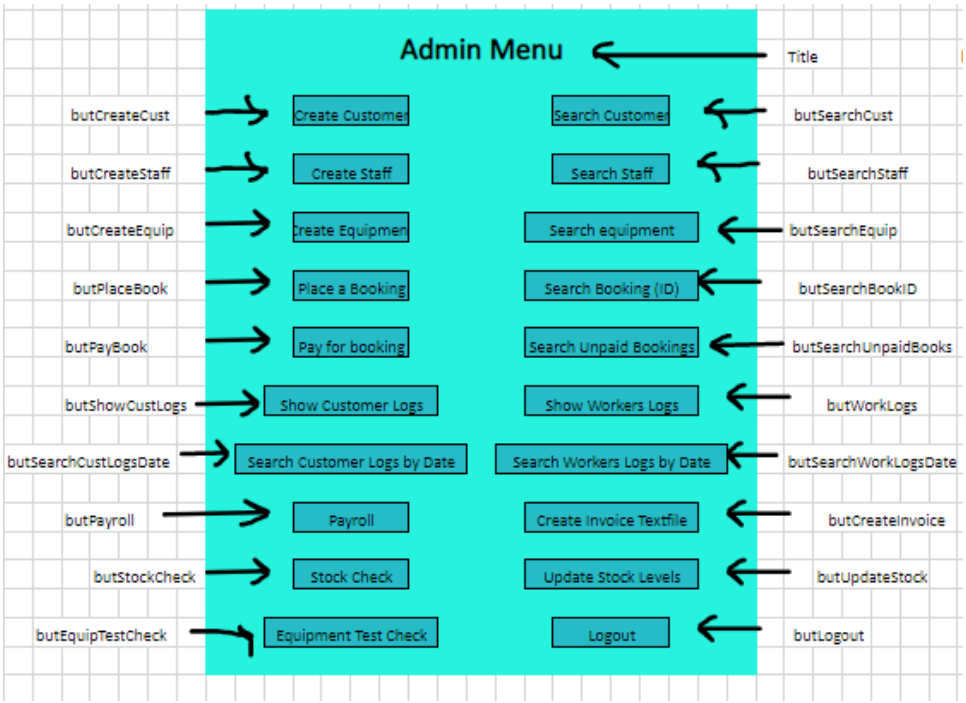
```
                                        Print “Incorrect username or password”
                                        Count = Count + 1
                                        Enter ID
                                        Enter Password
        Print “Sorry, you have run out of attempts”
        Close program

If butForgPass clicked:
For x in ArrayPassword[x]:
        If ID = ArrayPassword[x,1] then:
                TxtClue.text = ArrayPassword[x,4]
        Else
                Print(“Please enter a valid ID before we can provide you with your clue”)
```

Staff Menu




```
Declare TodayDate as Date = Today
Declare diff as integer

If butTodayBookings clicked:
        Open MyBookingsToday

If butMyData clicked:
        For x in ArrayStaff[x]:
                If ArrayStaff[x,0] = LoggedInID then:
                        Print ArrayStaff[x]

If butStockLevel clicked:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,1] < ArrayEquipment[x,2] then:
                        Diff = ArrayEquipment[x,2] - ArrayEquipment[x,1]
                        Print (ArrayEquipment[x,0] “is” diff “under the reorder stock value”)
                        Print (“Reorder” ArrayEquipment[x,3] “of this item”)

If butWorkLogs clicked:
        Open WorkersLogs

If butSearchLogs clicked:
        Open SearchWorkersLogs

If butLogout clicked:
        Close program
```

Admin Menu



Declare TodayDate as Date = Today

If butCreateCust clicked:
    Open AddCustomer

If butSearchCust clicked:
    Open SearchCustomer

If butCreateStaff clicked:
    Open AddStaff

If butSearchStaff clicked:
    Open SearchStaff

If butCreateEquip clicked:
    Open AddEquipment

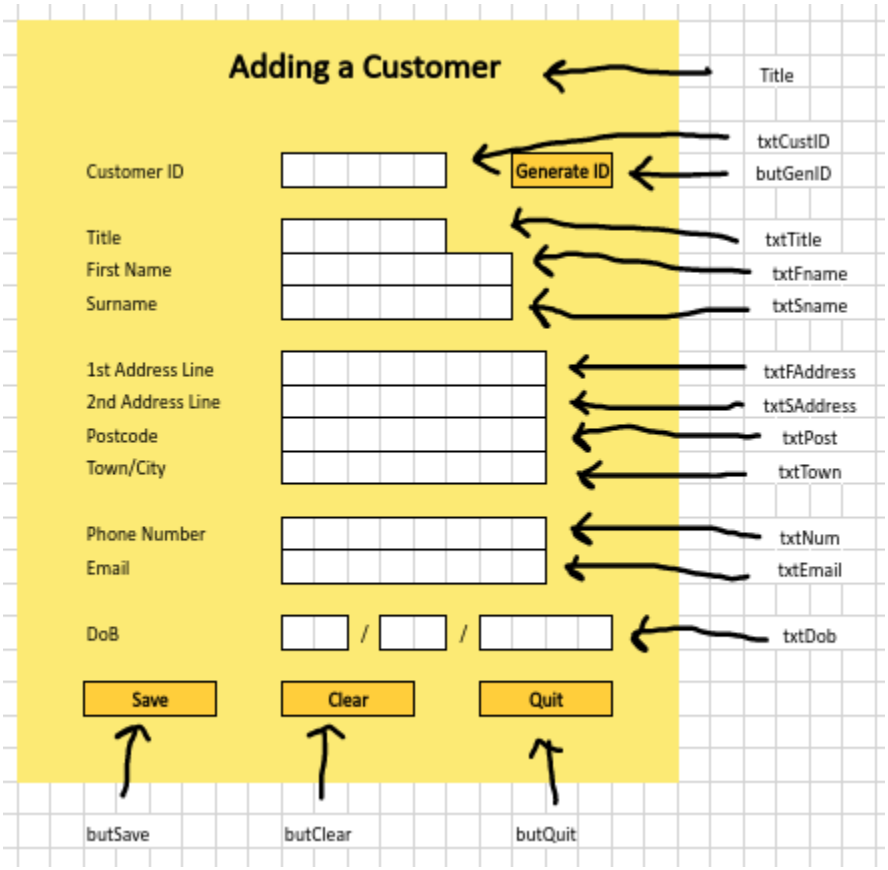
If butSearchEquip clicked:
    Open SearchEquipment

If butPlaceBook clicked:
    Open PlaceBooking

If butSearchBookID clicked:
    Open SearchBookingID

```
If butSearchUnpaidBooks clicked:
        Open UnpaidBooks

If butPayBook clicked:
        Open PayBooking

If butShowCustLogs clicked:
        Open CustomerLogs

If butWorkLogs clicked:
        Open WorkerLogs

If butSearchCustLogsDate clicked:
        Open SearchCustomerLogs

If butSearchWorkLogsDate clicked:
        Open SearchWorkersLogs

If butPayroll clicked:
        Open Payroll

If butCreateInvoice clicked:
        Open CreateInvoice

If butStockCheck clicked:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,1] < ArrayEquipment[x,2] then:
                        Diff = ArrayEquipment[x,2] - ArrayEquipment[x,1]
                        Print (ArrayEquipment[x,0] “is” diff “under the reorder stock value”)
                        Print (“Reorder” ArrayEquipment[x,3] “of this item”)

If butUpdateStock clicked:
        Open StockUpdate

If butEquipTestCheck clicked:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,6] > TodayDate then:
                        Print (ArrayEquipment[x,1] “requires testing”)
If butLogout clicked:
        Close Program
```

Adding a customer



(In loading the form)
Declare IDGenClick as boolean = False

If butGenID clicked:
If txtTitle =" " or txtFname = " " or txtSname = " " or txtFAddress = " " or txtSAddress = " " or txtPost = " " or txtTown = " " or txtNum = " " or txtEmail = " " or txtDob = " " then:
Messagebox ("Please fill out the rest of the form before generating ID")
Exit subroutine

Declare CustID as string
CustID = txtSname.text.substring(0,2).ToUpper & txtFname.text.substring(1,2).ToUpper & txtNum.text.substring(4,2)
TxtCustID.text = CustID

IDGenClick = True

If butSave clicked:
Declare Title as string
Declare Fname as string
Declare Sname as string
Declare FAddressLine as string

```
Declare SAddressLine as string
Declare Postcode as string
Declare TownCity as string
Declare Phone as string
Declare Email as string
Declare DoB as string

If IDGenClick =False then:
        Messagebox("Please generate an ID before saving")
        Exit subroutine

Messagebox("Please note that if you manually change your ID then it will not be saved and the
previously generated ID will be saved")
Messagebox("The ID that was generated is:", CustID)

Title = txtTitle.text
(List check)
If Title <> "Mr" and Title <> "Mrs" and Title <> "Miss" and Title <> "Ms" and Title <> "Mx" and Title <>
"Sir" and Title <> "Dr" and Title <> "Cllr" and Title <> "Lady" and Title <> "Lord" then:
        Print ("Incorrect title entered, correct titles; Mr, Mrs, Miss, Ms, Mx, Sir, Dr, Cllr, Lord, Lady.
Please try again)
        Exit subroutine

Fname = txtFname.text.substring(0,1).ToUpper & txtFname.text.substring(1, txtFname.text.length -1)
Sname = txtSname.text.substring(0,1).ToUpper & txtSname.text.substring(1, txtSname.text.length -1)

FAddressLine = FAddress.text
SAddressLine = SAddress.text
Postcode = txtPost.text
TownCity = txtTown.text

Phone = txtNum.text
(Length check)
If Len(Phone) <> 11 then:
        Print ("Incorrect phone number entered, please try again)
        Exit Subroutine

Email = txtEmail.text
(Format check)
Declare correct as boolean = Email like "*@*.*"
If correct = false then:
        Print ("Incorrect email format detected, please try again")
        Exit Subroutine

DoB = txtDoB.text
(Format check)
Declare correct2 as boolean = DoB like "**/**/****"
If correct2 = False then:
```

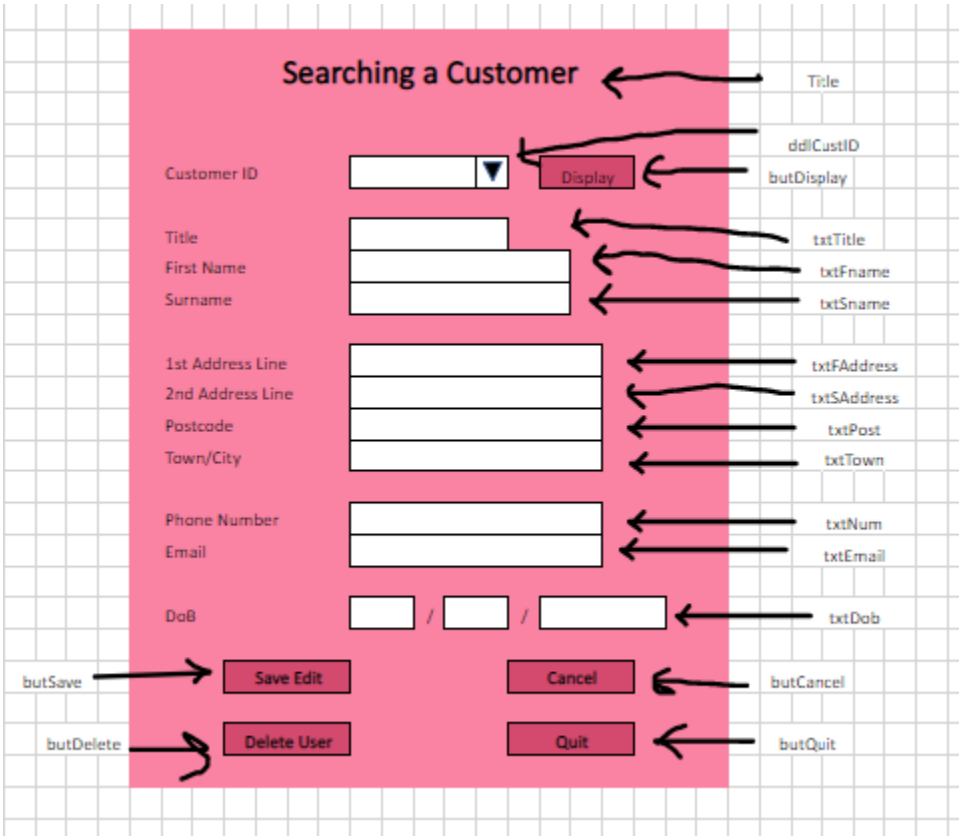
Print (“Incorrect Date of Birth format, must be ‘**/**/****’, please try again”)
Exit subroutine

Save data into Customers.docx

If butClear clicked:
TxtCustID.text = “ “
TxtTitle.text = “ “
TxtFname.text = “ “
TxtSname.text = “ “
txtFAddress.text = “ “
txtSAddress.text = “ “
TxtPost.text = “ “
TxtTown.text = “ “
TxtNum.text = “ “
TxtEmail.text = “ “
TxtDoB.text = “ “

If butQuit clicked:
Open AdminMenu

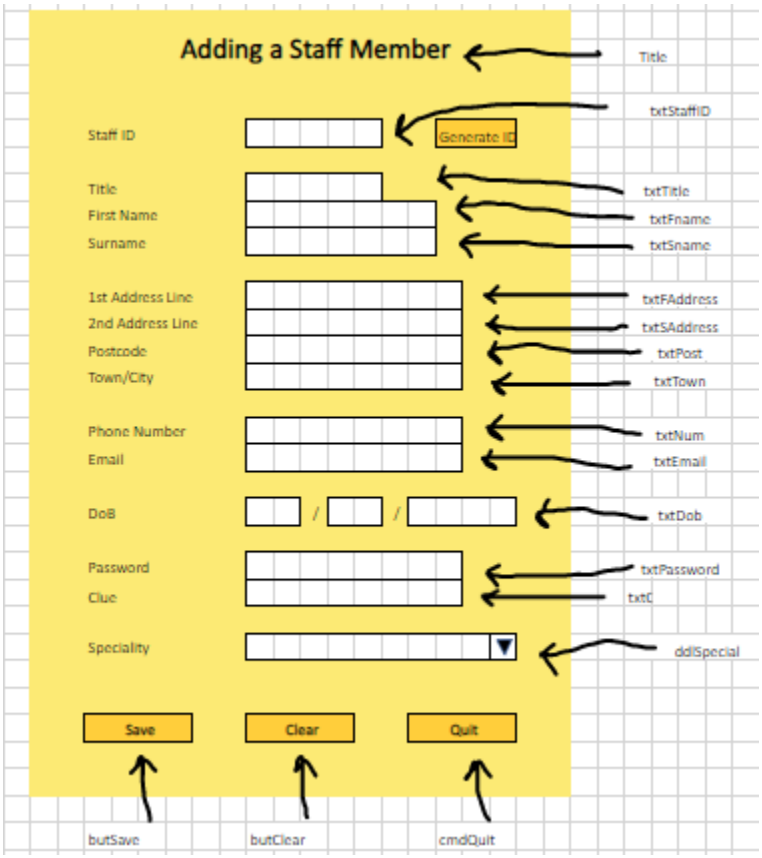
Search/Edit Customers




```
(In loading the form)
For x in ArrayCustomers[x]:
        DdlCustID.Items.Add(ArrayCustomers[x,0])

Declare Position as integer

If butDisplay clicked:
For x in ArrayCustomers[x]:
        If ddlCustID = ArrayCustomers[x,0] then:
                TxtTitle.text = ArrayCustomers[x,1]
                TxtFname.text = ArrayCustomers[x,2]
                TxtSname.text = ArrayCustomers[x,3]
                TxtFaddress.text = ArrayCustomers[x,4]
                TxtSaddress.text = ArrayCustomers[x,5]
                TxtPost.text = ArrayCustomers[x,6]
                TxtTown.text = ArrayCustomers[x,7]
                TxtNum.text = ArrayCustomers[x,8]
                TxtEmail.text = ArrayCustomers[x,9]
                TxtDoB.text = ArrayCustomers[x,10]
                Position = x
```

```
                    Exit for loop

If butSave clicked:
Load_customers()
Customers(Position).Title = txtTitle.text
Customers(Position).Fname = txtFname.text
Etc... for all textboxes + appropriate checks

Sort_Array()
Save_AllCustomers()

If butDelete clicked:
For x = position to numCustomers – 1
        ArrayCustomers[x,1] = ArrayCustomers[x+1,1]
        ArrayCustomers[x,2] = ArrayCustomers[x+1,2]
        ArrayCustomers[x,3] = ArrayCustomers[x+1,3]
        ArrayCustomers[x,4] = ArrayCustomers[x+1,4]
        ArrayCustomers[x,5] = ArrayCustomers[x+1,5]
        ArrayCustomers[x,6] = ArrayCustomers[x+1,6]
        ArrayCustomers[x,7] = ArrayCustomers[x+1,7]
        ArrayCustomers[x,8] = ArrayCustomers[x+1,8]
        ArrayCustomers[x,9] = ArrayCustomers[x+1,9]
        ArrayCustomers[x,10] = ArrayCustomers[x+1,10]

NumCustomers = numCustomers – 1
Save_allcustomers()

If butCancel clicked:
ddlCustID.text = " "
TxtTitle.text = " "
TxtFname.text = " "
TxtSname.text = " "
FAddress.text = " "
SAddress.text = " "
TxtPost.text = " "
TxtTown.text = " "
TxtNum.text = " "
TxtEmail.text = " "
TxtDoB.text = " "

If butQuit clicked:
Open AdminMenu
```

Adding a staff member



When loading the form:
DdlSpecial.Items.Add("Welding")
DdlSpecial.Items.Add("Woodwork")
DdlSpecial.Items.Add("Plumbing")
DdlSpecial.Items.Add("Carpentry")
DdlSpecial.Items.Add("Electrician")
DdlSpecial.Items.Add("Gardening")
DdlSpecial.Items.Add("Roofing")
DdlSpecial.Items.Add("Other")

Declare IDGenClick as boolean = False

If butGenID clicked:
If txtTitle.text = " " or txtFname.text = " " or txtSname = " " or txtFAddress.text = " " or txtSAddress.text = " " txtPost.text = " " or txtTown = " " or txtDob.text = " " txtPassword.text = " " or ddlSpecial.text = " " then:
Messagebox ("Please fill out the rest of the form before generating ID")
Exit subroutine

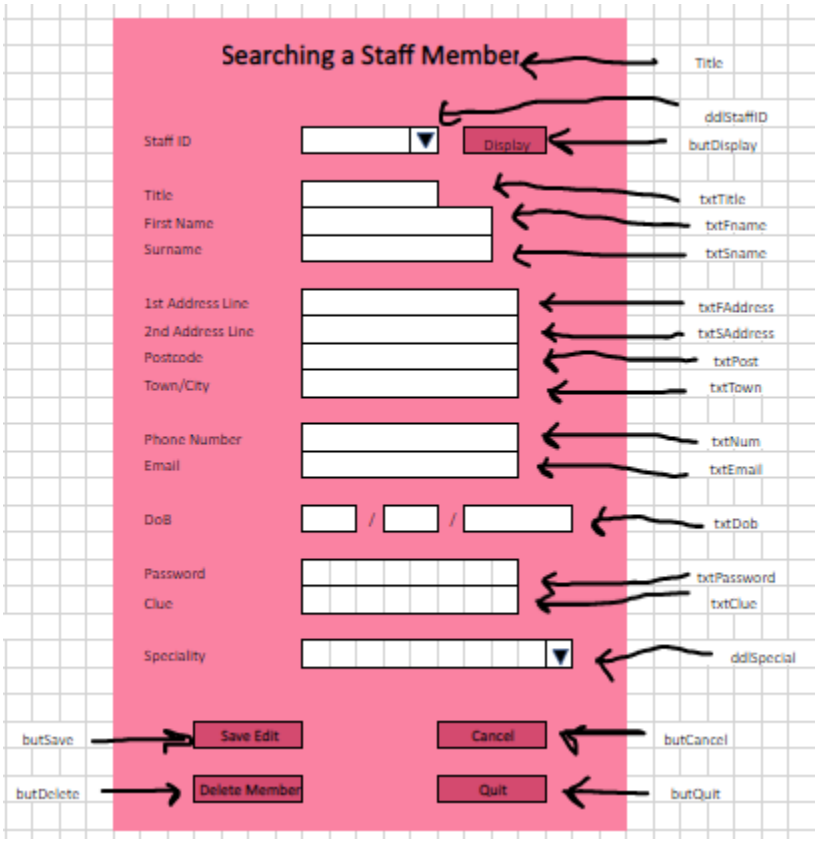
Declare StaffID as string
StaffID = txtSname.text.substring(0,2).ToUpper & txtFname.text.substring(1,2).ToUpper & txtNum.text.substring(4,2)

```
TxtStaffID.text = StaffID

IDGenClick = True

If butSave clicked:
Declare Title as string
Declare Fname as string
Declare Sname as string
Declare FAddressLine as string
Declare SAddressLine as string
Declare Postcode as string
Declare TownCity as string
Declare Phone as string
Declare Email as string
Declare DoB as string
Declare Password as string
Declare Clue as string
Declare Speciality as string

If IDGenClick =False then:
        Messagebox("Please generate an ID before saving")
        Exit subroutine

Messagebox("Please note that if you manually change your ID then it will not be saved and the
previously generated ID will be saved")
Messagebox("The ID that was generated is:", StaffID)

Title = txtTitle.text
(List check)
If Title <> "Mr" and Title <> "Mrs" and Title <> "Miss" and Title <> "Ms" and Title <> "Mx" and Title <>
"Sir" and Title <> "Dr" and Title <> "Cllr" and Title <> "Lady" and Title <> "Lord" then:
        Print ("Incorrect title entered, correct titles; Mr, Mrs, Miss, Ms, Mx, Sir, Dr, Cllr, Lord, Lady.
Please try again)
        Exit subroutine

Fname = txtFname.text.substring(0,1).ToUpper & txtFname.text.substring(1, txtFname.text.length -1)
Sname = txtSname.text.substring(0,1).ToUpper & txtSname.text.substring(1, txtSname.text.length -1)

FAddressLine = FAddress.text
SAddressLine = SAddress.text
Postcode = txtPost.text
TownCity = txtTown.text

Phone = txtNum.text
(Length check)
If Len(Phone) <> 11 then:
        Print ("Incorrect phone number entered, please try again)
        Exit Subroutine
```

```
Email = txtEmail.text
(Format check)
Declare correct as boolean = Email like “*@*.*”
If correct = false then:
        Print (“Incorrect email format detected, please try again”)
        Exit Subroutine

DoB = txtDoB.text
(Format check)
Declare correct2 as boolean = DoB like “**/**/****”
If correct2 = False then:
        Print (“Incorrect Date of Birth format, must be ‘**/**/****’, please try again)
        Exit Subroutine
Convert DoB from string to date

Password = txtPassword.text
Clue = txtClue.text
Speciality = ddlSpecial.text

Save data (except password, Clue) to Staff.docx
Save (Password, Clue) to Passwords.docx

TxtStaffID.text = “ “
TxtTitle.text = “ “
TxtFname.text = “ “
TxtSname.text = “ “
FAddress.text = “ “
SAddress.text = “ “
TxtPost.text = “ “
TxtTown.text = “ “
TxtNum.text = “ “
TxtEmail.text = “ “
TxtDoB.text = “ “
TxtPassword.text = “ “
TxtClue.text = “ “
TxtSpeciality.text = “ “

If butClear clicked:
TxtStaffID.text = “ “
TxtTitle.text = “ “
TxtFname.text = “ “
TxtSname.text = “ “
FAddress.text = “ “
SAddress.text = “ “
TxtPost.text = “ “
TxtTown.text = “ “
TxtNum.text = “ “
```

```
TxtEmail.text = “ “
TxtDoB.text = “ “
TxtPassword.text = “ “
TxtClue.text = “ “
TxtSpeciality.text = “ “

If butQuit clicked:
Open AdminMenu
```

Search/Edit a staff member



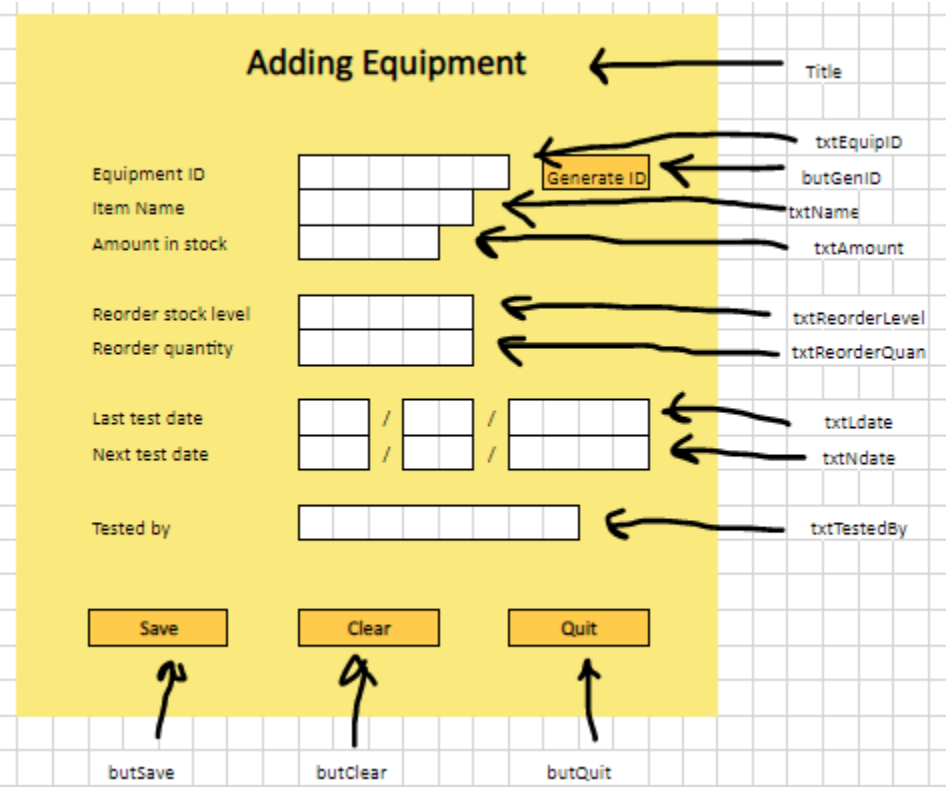

```
(In loading the form)
For x in ArrayStaff[x]:
        DdlStaffID.Items.Add(ArrayStaff[x,0])

If butDisplay clicked:
For x in ArrayStaff[x]:
        If ddlStaffID = ArrayStaff[x,0] then:
                TxtTitle.text = ArrayStaff[x,1]
                TxtFname.text = ArrayStaff [x,2]
                TxtSname.text = ArrayStaff [x,3]
                TxtFaddress.text = ArrayStaff [x,4]
                TxtSaddress.text = ArrayStaff [x,5]
                TxtPost.text = ArrayStaff [x,6]
                TxtTown.text = ArrayStaff [x,7]
                TxtNum.text = ArrayStaff [x,8]
                TxtEmail.text = ArrayStaff [x,9]
                TxtDoB.text = ArrayStaff [x,10]
                TxtSpecial.text = ArrayStaff[x,11]
                PositionX = x
                For y in ArrayPassword[y]:
                        If ArrayPassword[y,1] = ArrayStaff[x,0] then:
                                TxtPassword.text = ArrayPassword[y,2]
```

```
                                                    TxtClue.text = ArrayPassword[y,4]
                                                    PositionY = y
                                                    Exit For
                                Exit For

If butSave clicked:
Load_staff()
Load_Passwords()

Staff(PositionX).Title = txtTitle.text
List check on txtTitle.text

Staff(PositionX).Fname = txtFname.text
Etc... for all non-password related textboxes + appropriate checks

Staff(PositionY).Password = txtPassword.text
Staff(PositionY).clue = txtClue.text

Sort_ArrayStaff()
Sort_ArrayPasswords()
Save_AllStaff()
Save_AllPasswords()

If butDelete clicked:
For x = positionX to numStaff – 1
        ArrayStaff[x,1] = ArrayStaff [x+1,1]
        ArrayStaff[x,2] = ArrayStaff [x+1,2]
        ArrayStaff[x,3] = ArrayStaff [x+1,3]
        ArrayStaff[x,4] = ArrayStaff [x+1,4]
        ArrayStaff[x,5] = ArrayStaff [x+1,5]
        ArrayStaff[x,6] = ArrayStaff [x+1,6]
        ArrayStaff[x,7] = ArrayStaff [x+1,7]
        ArrayStaff[x,8] = ArrayStaff [x+1,8]
        ArrayStaff[x,9] = ArrayStaff [x+1,9]
        ArrayStaff[x,10] = ArrayStaff [x+1,10]
        ArrayStaff[x,11] = ArrayStaff [x+1,11]

For x = PositionY to numPasswords – 1
        ArrayPasswords[x,1] = ArrayPasswords[x+1,1]
        ArrayPasswords[x,1] = ArrayPasswords[x+1,1]

numStaff = numStaff – 1
Save_allcustomers()
NumPasswords = numPasswords – 1

If butCancel clicked:
ddlStaffID.text = " "
```

TxtTitle.text = “ “
TxtFname.text = “ “
TxtSname.text = “ “
FAddress.text = “ “
SAddress.text = “ “
TxtPost.text = “ “
TxtTown.text = “ “
TxtNum.text = “ “
TxtEmail.text = “ “
TxtDoB.text = “ “
TxtPassword.text = “ “
TxtSpeciality.text = “ “

If butQuit clicked:
Open AdminMenu

Adding equipment



(In loading the form)
Declare IDGenClick as boolean = False

If butGenID clicked:
If txtEquipID.text = " " or txtName.text = " " or txtAmount.text = " " or txtReorderLevel.text = " " or txtReorderQuan.text = " " or txtLdate.text = " " or txtNdate = " " then:
    Messagebox ("Please fill out the rest of the form before generating ID")
    Exit subroutine

Declare EquipID as string
EquipID = txtName.text.substring(2,1).ToUpper & txtName.text.substring(1,1).ToUpper & txtName.text.substring(0,1).ToUpper & txtReorderQuan.text.substring(0,2)
TxtEquipID.text = EquipID

IDGenClick = True

If butSave clicked:
Declare Name as string
Declare Amount as integer
Declare ReorderLvl as integer
Declare ReorderAmount as integer
Declare LTestDate as string
Declare NTestDate as string

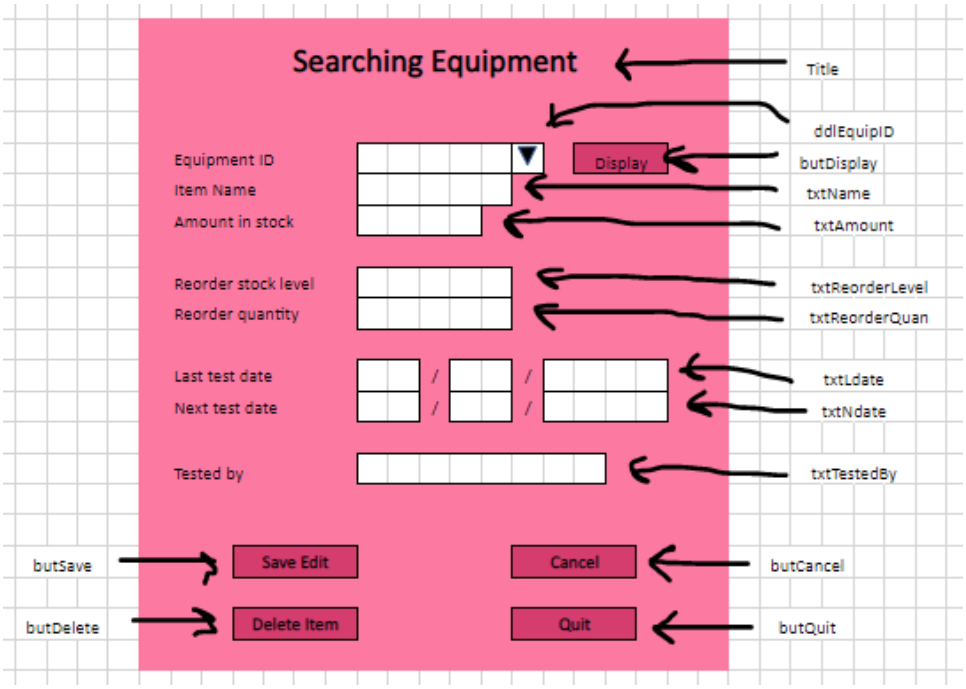
```
Declare TestedBy as string

If IDGenClick =False then:
        Messagebox("Please generate an ID before saving")
        Exit subroutine

Messagebox("Please note that if you manually change the ID then it will not be saved and the
previously generated ID will be saved")
Messagebox("The ID that was generated is:", EquipID)

Name = txtName.text
(Presence check)
If Name = " " then
        Print ("No item name has been entered, please try again")
        Exit Subroutine

Amount = txtAmount.text
(Range check)
If Amount > 1000 or Amount < 0 then:
        Print ("Amount of this item entered is out of range, please try again")
        Exit Subroutine

ReorderLvl = txtReorderLevel.text
ReorderAmount = txtReorderQuan.text

LTestDate = txtLDate.text
(Format check)
Declare correct as boolean = LTestDate like "**/**/****"
If correct = False then:
        Print ("Incorrect Last Test Date format, must be '**/**/****', please try again")
        Exit Subroutine
Convert LTestDate from string to date

NTestDate = txtNDate.text
(Format check)
Declare correct2 as boolean = NTestDate like "**/**/****"
If correct2 = False then:
        Print ("Incorrect Next Test Date format, must be '**/**/****', please try again")
        Exit Subroutine
Convert NTestDate from string to date

TestedBy = txtTestedBy.text

Save data to Equipment.docx

TxtEquipID.text = " "
TxtName.text = " "
TxtAmount.text = " "
```

TxtReorderLvl.text = “ “
TxtReorderAmount.text = “ “
TxtLTestDate.text = “ “
TxtNTestDate.text = “ “
TxtTestedBy.text = “ “

If butClear clicked:
TxtEquipID.text = “ “
TxtName.text = “ “
TxtAmount.text = “ “
TxtReorderLvl.text = “ “
TxtReorderAmount.text = “ “
TxtLTestDate.text = “ “
TxtNTestDate.text = “ “
TxtTestedBy.text = “ “

If butQuit clicked:
Open AdminMenu

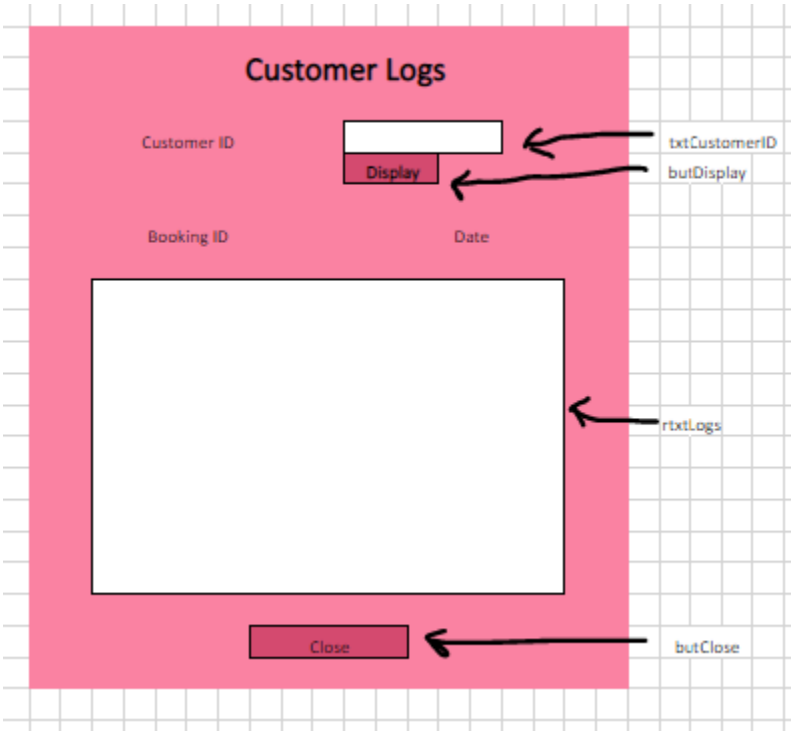
Search/Edit equipment




```
(In loading the form)
For x in ArrayEquipment[x]:
        DdlEquipID.Items.Add(ArrayEquipment[x,0])

If butDisplay clicked:
For x in ArrayEquipment[x]:
        If ddlEquipID = ArrayEquipment[x,0] then:
                TxtName.text = ArrayEquipment[x,1]
                TxtAmount.text = ArrayEquipment[x,2]
                TxtReorderLevel.text = ArrayEquipment[x,3]
                TxtReorderQuan.text = ArrayEquipment[x,4]
                TxtLdate.text = ArrayEquipment[x,5]
                TxtNdate.text = ArrayEquipment[x,6]
                TxtTestedby.text = ArrayEquipment[x,7]
                Position = x
                Exit for loop


If butSave clicked:
Load_Equipment()
Equipment(Position).Name= txtName.text
Equipment(Position).Amount = txtAmount.text
Etc... for all textboxes + appropriate checks

Equipment(Position).LDate = txtLDate.text converted as Date
Equipment(Position).NDate = txtNDate.text converted as Date
```

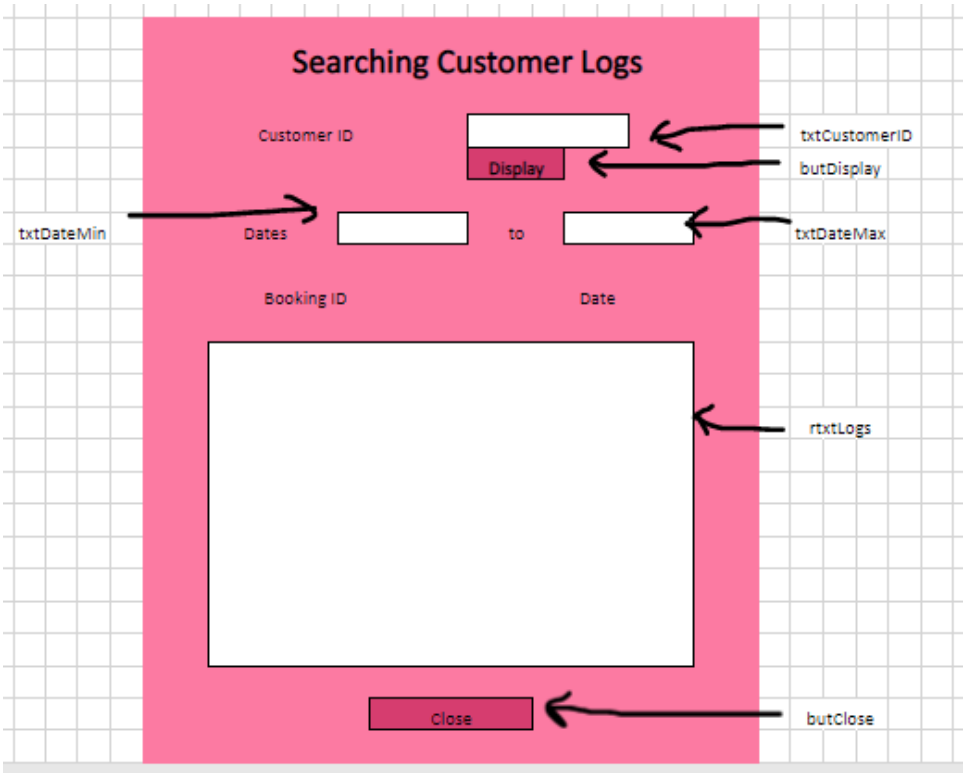
```
Sort_ArrayEquipment()
Save_AllEquipment()


If butDelete clicked:
For x = position to numEquipment – 1
        ArrayEquipment[x,1] = ArrayEquipment[x+1,1]
        ArrayEquipment[x,2] = ArrayEquipment[x+1,2]
        ArrayEquipment[x,3] = ArrayEquipment[x+1,3]
        ArrayEquipment[x,4] = ArrayEquipment[x+1,4]
        ArrayEquipment[x,5] = ArrayEquipment[x+1,5]
        ArrayEquipment[x,6] = ArrayEquipment[x+1,6]
        ArrayEquipment[x,7] = ArrayEquipment[x+1,7]

NumEquipment = numEquipment – 1
Save_allequipment()

If butCancel clicked:
Open AdminMenu

If butQuit clicked:
Open AdminMenu
```

Customer Logs



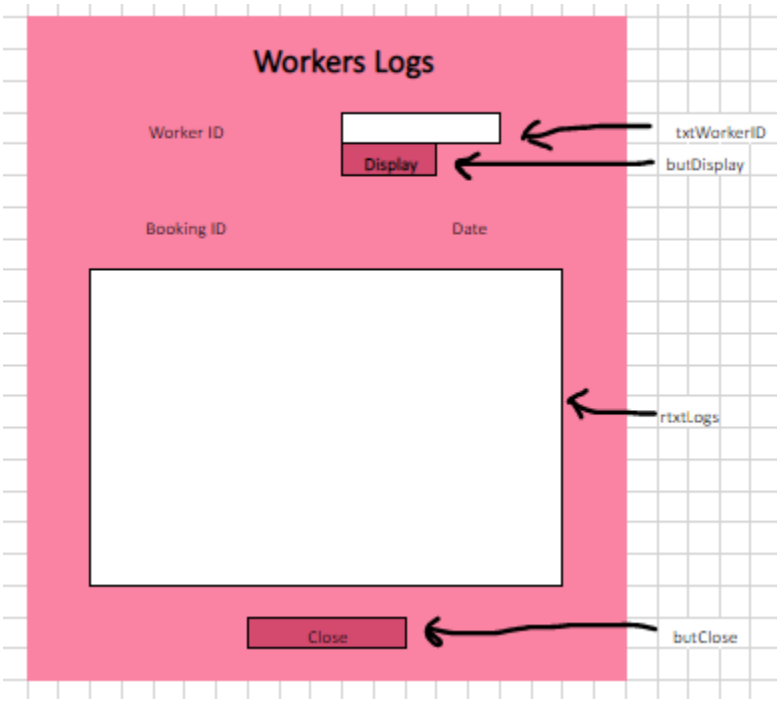

```
If butDisplay clicked:
Load_Bookings()
Declare BookingsList as string = ""

For x in BookingsArray:
        If Booking(x).CustomerID = txtCustomerID.text then:
                BookingsList = BookingsList & Booking(x) & returnButton.click(3)

Output(BookingsList)

If butClose clicked:
Open AdminMenu
```

Searching Customer Logs



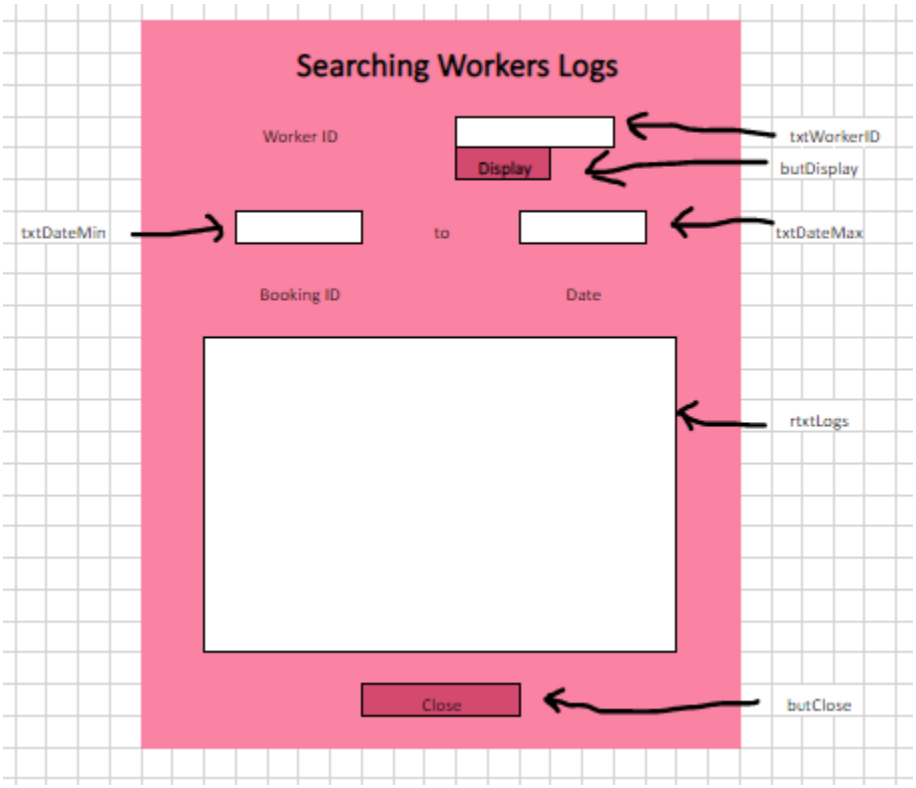

```
If butDisplay clicked:
Load_Bookings()
Declare BookingsList as string = ""

For x in BookingsArray:
        If Booking(x).CustomerID = txtCustomerID.text and Booking(x).Date > txtMindate.text and
Booking(x).Date < txtMaxdate.text then:
                BookingsList = BookingsList & Booking(x) & returnButton.click(3)

Output(BookingsList)

If butClose clicked:
Open AdminMenu
```

Workers Logs




```
If butDisplay clicked:
Load_Bookings()
Declare BookingsList as string = ""

For x in BookingsArray:
        If Booking(x).StaffID = txtWorkerID.text then:
                BookingsList = BookingsList & Booking(x) & returnButton.click(3)

Output(BookingsList)

If butClose clicked:
If WorkerAccount = True then:
        Open StaffMenu
Else:
        Open AdminMenu
```

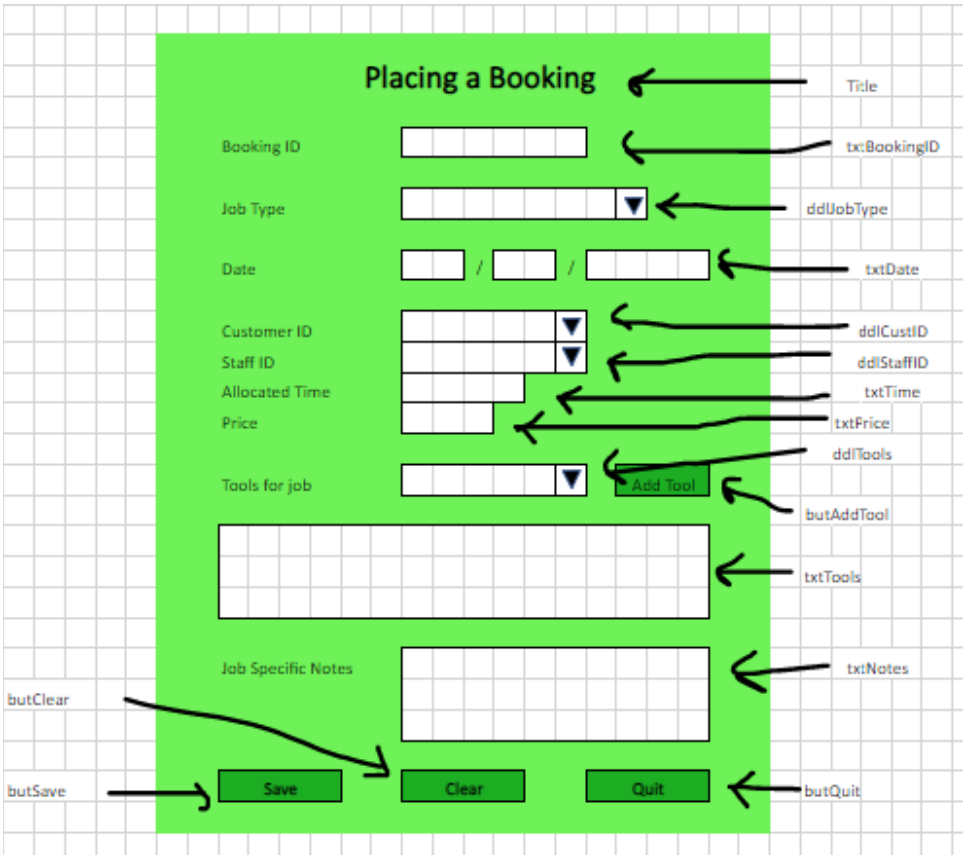
Searching Workers Logs




```
(In loading the form)
'Checking if the user is a worker and, if so, printing their ID into the textbox
If WorkerAccount = True then:
        TxtWorkerID.text = LoggedInID

If butDisplay clicked:
'Checking if the user is a worker and, if so, using their ID for the searches
Declare WorkerID as string
If WorkerAccount = True then:
        TxtWorkerID.text = LoggedInID

WorkerID = txtWorkerID.text

'Declaring a data structure for logs of all the workers
Declare Data Structure AllWorkerLogsStructure
        Declare AllWorkerID as string
        Declare AllBookingID as string
        Declare AllDate as string

'Declaring an array for the data structure of the logs of all the workers
Declare AllLogs(0 to 999999) as AllWorkerLogsStructure

'Declaring a data structure for the searched customer's booking details
Declare Data Structure WorkerLogsStructure
```

```
        Declare BookingID as string
        Declare Date as string

'Declaring an array for the searched customer's booking details
Declare Logs(0 to 999999) as WorkerLogsStructure

Load Bookings.docx = file
Declare line as string
Declare NumJobs as integer = 0

'Loading the booking details from the textfile into the array
Do
        Line = file.readline()
        AllLogs(NumJobs).AllWorkerID = (StaffID.file).trim
        AllLogs(NumJobs).AllBookingID = (BookingID.file).trim
        AllLogs(NumJobs).AllDate = (Date.file).trim
        NumJobs = NumJobs + 1
Until(file.EndOfStream)

'Check which Booking's Staff IDs match the entered Worker ID
Declare matched as integer = 0
Declare DateMin as string = txtDateMin.text
Declare DateMax as string = txtDateMax.txt
CODE TO CONVERT TO DATE

For x = 0 to NumJobs in AllLogs
        If AllWorkerID[x] = WorkerID then:
                If AllDate[x] > DateMin and AllDate[x] < DateMax then:
                        BookingID[matched] = AllBookingID[x]
                        Date[matched] = AllDate[x]
                        Matched = matched + 1
        Next x

'Save the details of the searched workers booking IDs and booking dates into a text file
Save into SearchedWorkerLogs.docx as (BookingID & "," & Date)

'Output text file through the rich text box
RtxtLogs.Loadfile("SearchedWorkerLogs.txt")

If butClose clicked:
If WorkerAccount = True then:
        Open StaffMenu
Else:
        Open AdminMenu
```

Placing a booking




```
(In loading the form)
DdlJobType.Items.Add("Welding")
DdlJobType.Items.Add("Woodwork")
DdlJobType.Items.Add("Plumbing")
DdlJobType.Items.Add("Carpentry")
DdlJobType.Items.Add("Electrician")
DdlJobType.Items.Add("Welding")
DdlJobType.Items.Add("Gardening")
DdlJobType.Items.Add("Roofing")
DdlJobType.Items.Add("Other")

For x in ArrayCustomers[x]:
        DdlCustID.Items.Add(ArrayCustomers(x,0))

For x in ArrayEquipment[x]:
        DdlTools.Items.Add(ArrayEquipment(x,1))

If butAddTool clicked:
TxtTools.text = txtTools.text + "," + ddlTools.text
DdlTools.text = " "

If butSave clicked:
```

```
Declare BookingID as string
Declare JobType as string
Declare Date as string
Declare CustomerID as string
Declare StaffID as string
Declare Time as real
Declare Price as real
Declare Equipment as string
Declare Notes as string

BookingID = txtBookingID.text
(Presence check)
If bookingID = " " then:
        Print "No booking ID was entered, please try again"
        End Subroutine

JobType = ddlJobType.text
(List check)
If JobType <> "Welding" and JobType <> "Woodwork" and JobType <> "Plumbing" and JobType <>
"Carpentry" and JobType <> "Electrician" and JobType <> "Welding" and JobType <> "Gardening" and
JobType <> "Roofing" and JobType <> "Other" then:
        Print "Invalid job type entered, please try again"
        End subroutine

Date = txtDate.text
(Format check)
Declare correct as boolean = Date like "**/**/****"
If correct = false then:
        Print "Incorrect date format entered, must be '**/**/****', please try again"
        End subroutine
Convert Date from string to date

CustomerID = ddlCustID.text
StaffID = ddlStaffID.text

Time = txtTime.text
(Range check)
If Time < 0.2 or Time > 100 then:
        Print "Invalid allocated time entered, please try again"
        End subroutine

Price = "£" + txtPrice.text

Equipment = txtTools.text
Notes = txtNotes.text

Save data to Bookings.docx
```

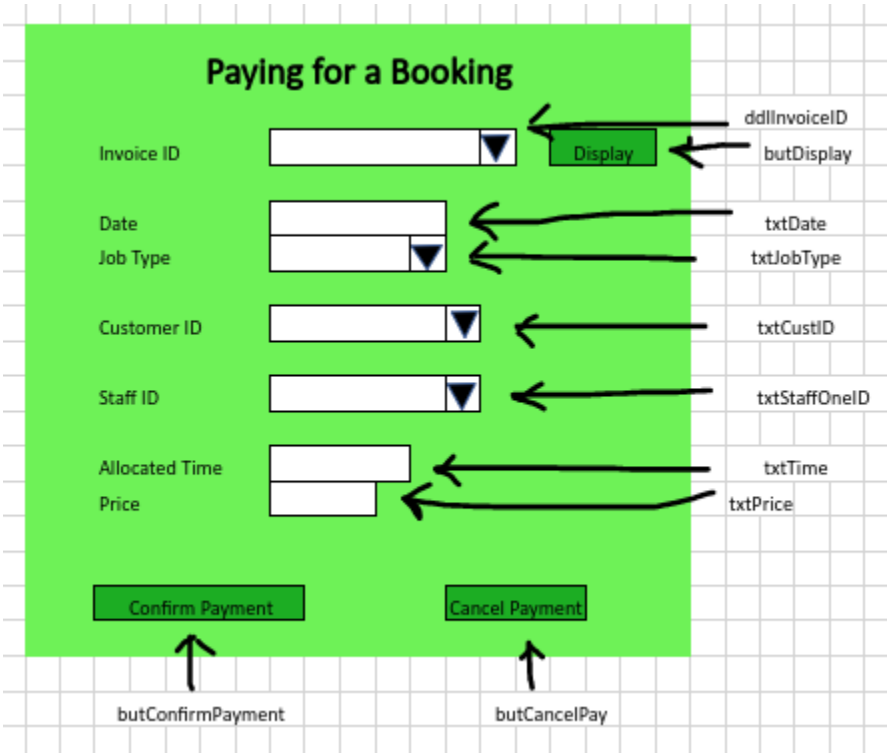
TxtBooking.text = “ “
DdlJobType.text = “ “
TxtDate.text = “ “
DdlCustomerID.text = “ “
DdlStaffID.text = “ “
TxtTime.text = “ “
TxtTools.text = “ “
TxtNotes.text = “ “

If butClear clicked:

TxtBooking.text = “ “
DdlJobType.text = “ “
TxtDate.text = “ “
DdlCustomerID.text = “ “
DdlStaffID.text = “ “
TxtTime.text = “ “
TxtTools.text = “ “
TxtNotes.text = “ “

If butQuit clicked:
Open AdminMenu

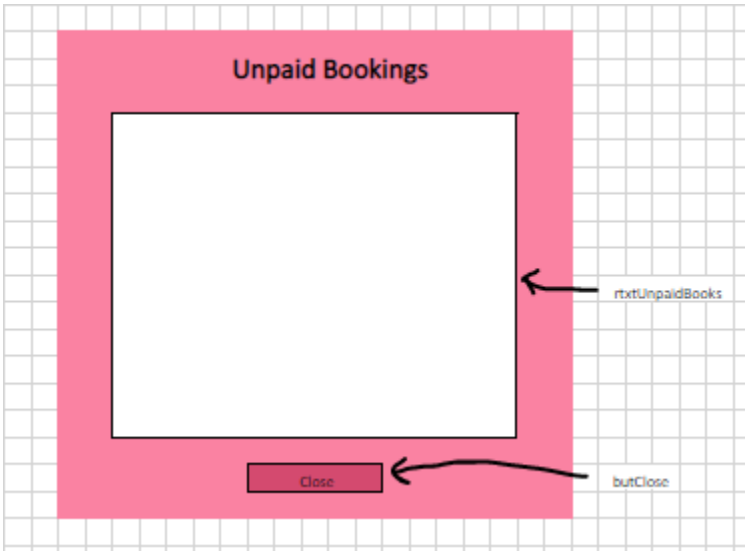
Paying for a booking




```
(Loading the array)
For x in ArrayInvoices[x]:
        DdlInvoiceID.Items.Add(ArrayInvoice[x,0])

If butDisplay clicked:
Declare InvoiceID as string
Declare BookingID as string

InvoiceID = ddlBookingID.text

For x in ArrayInvoices[x]:
        If ArrayInvoices[x,0] = InvoiceID then:
                For y in ArrayBookings[y]:
                        If ArrayBookings[y,0] = ArrayInvoices[x,1] then:
                                TxtJobType.text = ArrayBookings[y,1]
                                TxtDate.text = ArrayBookings[y,2]
                                TxtCustID.text = ArrayBookings[y,3]
                                TxtStaffID.text = ArrayBookings[y,4]
                                TxtTimeTaken.text = ArrayBookings[y,5]
                                TxtPrice.text = ArrayBookings[y,6]
        Else
                Print “No invoice was found with this ID”
                End subroutine
```

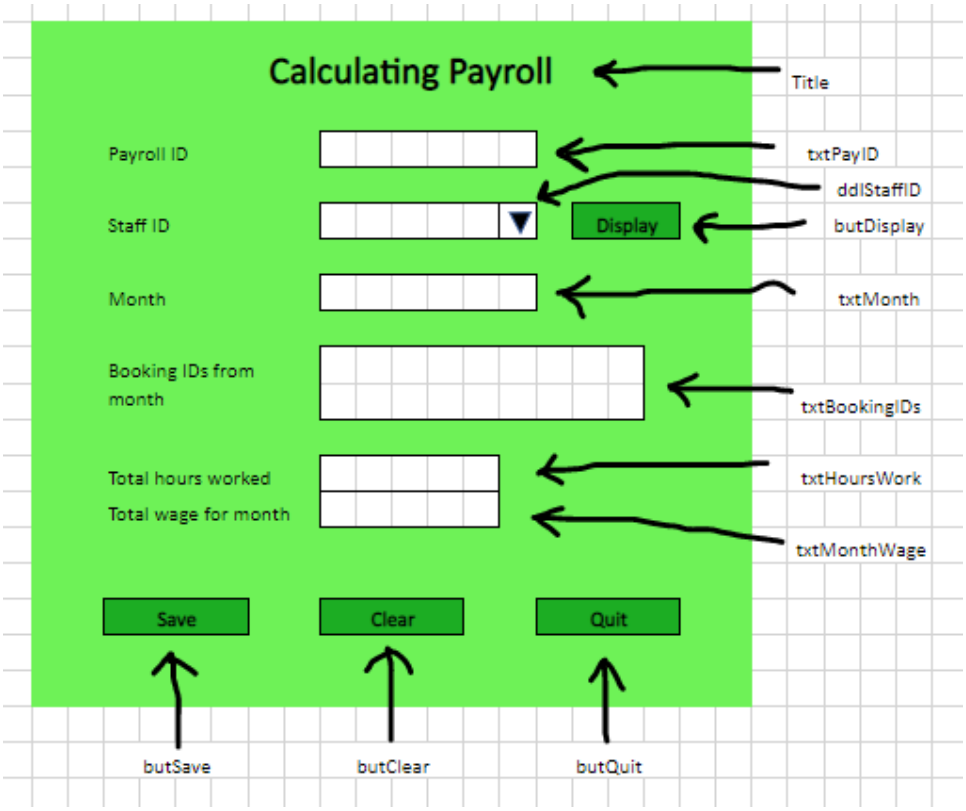
```
If butConfirmPayment clicked:
InvoiceID = ddlBookingID.text
For x in ArrayInvoices[x]:
        If InvoiceID = ArrayInvoices[x,0] then:
                ArrayInvoices[x,2] = True
        Else
                Print “Invalid invoice ID entered, please try again”
                End subroutine

If butCancelPay clicked:
Open AdminMenu
```

Searching bookings by Paid?




```
In loading form:
Load_Bookings()
Declare BookingsList as string = ""

For x in BookingsArray:
        If Booking(x).Paid = False then:
                BookingsList = BookingsList & Booking(x) & returnButton.click(3)

Output(BookingsList)

If butClose clicked:
Open AdminMenu
```

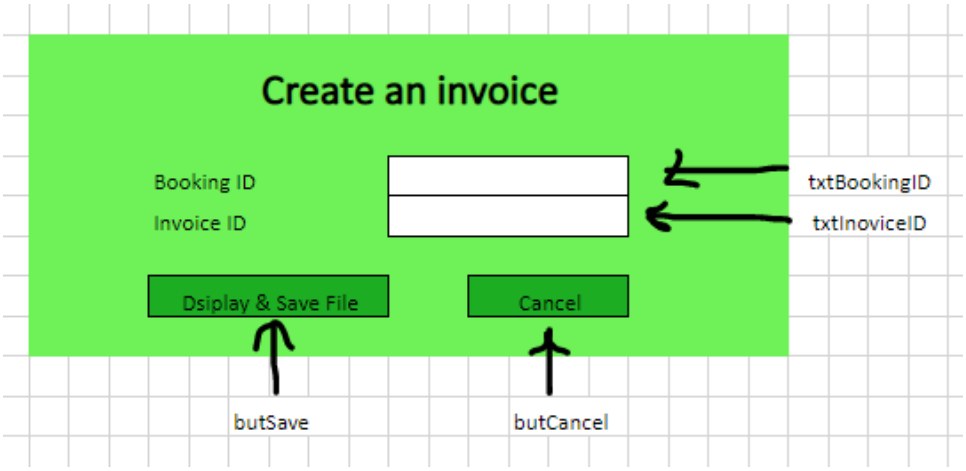
Calculating payroll




```
(Loading the form)
Declare PayRate as float

For x in ArrayStaff[x]:
        DdlStaffID.Items.Add(ArrayStaff[x,0])

If butDisplay clicked:
Txtmonth.text = now.month
TxtpayrollID.text = txtmonth.text & ddlstaffID.text

For x = 0 to numbooking – 1
        If ArrayBookings(x).staffID = ddlstaffID.text then
                TxtBookingIDs.text = txtBookingIDs.text + “,” + ArrayBookings(x).date
                TxtHoursWork.text = txtHoursWork.text + ArrayBookings(x).time
        End if
Next x

Inputbox(“Please enter the workers wage”) = PayRate

TxtMonthWage.text = txtHoursWork.text * PayRate

If butSave clicked:
If ddlstaffID.text = “” then
```

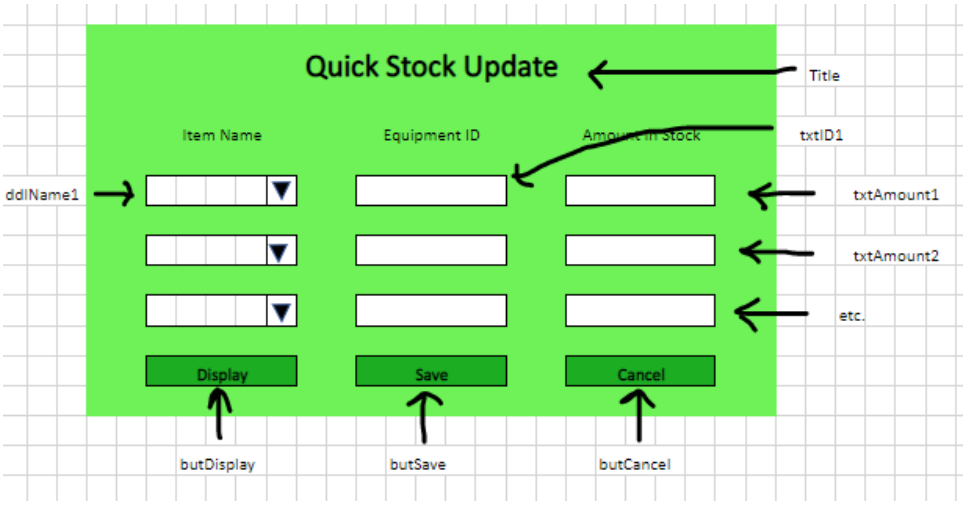
```
            Output("Please enter staff ID")
            Exit sub
End if

Save data to a new textfile called (txtPayrollID).text

If butClear clicked:
TxtPayID.text = " "
TxtStaffID.text = " "
TxtMonth.text = " "
TxtBookingIDs.text = " "
TxtHoursWork.text = " "
TxtMonthWage.text = " "

If butQuit clicked:
Open AdminMenu
```

Creating an invoice




```
If butSave clicked:
Declare InvoiceID as string
Declare BookingID as string
Declare Paid as boolean

InvoiceID = txtInvoiceID.text
BookingID = txtBookingID.text
Paid = False

Mycomputer.files.open(DIR$(InvoiceID.txt))

If butCancel clicked:
Open AdminMenu
```

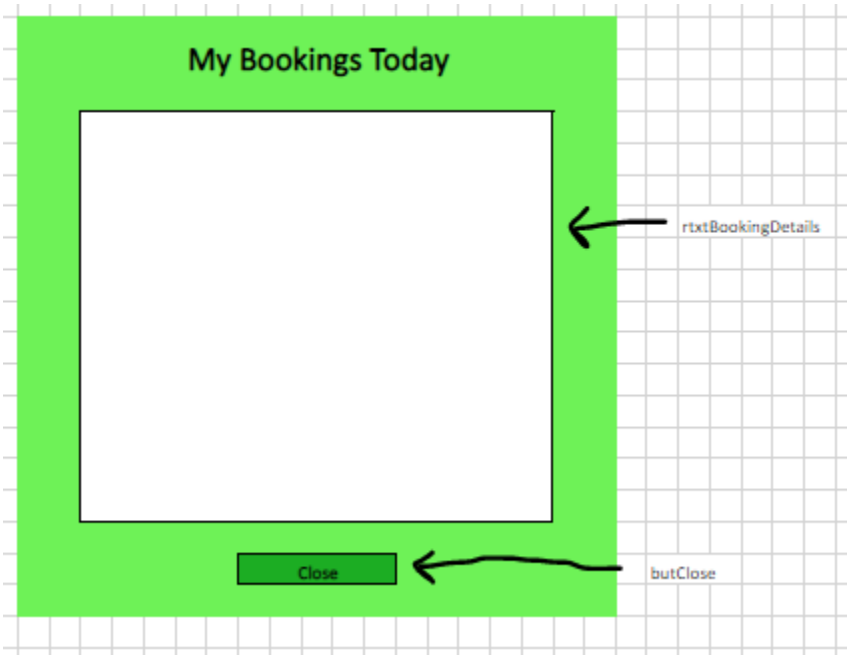
Quick stock update




```
(Loading the form)
For x in ArrayEquipment[x]:
        DdlName1.Items.Add(ArrayEquipment[x,1])
        DdlName2.Items.Add(ArrayEquipment[x,1])
        DdlName3.Items.Add(ArrayEquipment[x,1])

If butDisplay clicked:
If ddlName1.text = ddlName2.text or ddlName1.text = ddlName3.text or ddlName2.text =
ddlName3.text then:
        Print “An item has been entered more than once, please change this and try again”
        Exit subroutine

If ddlName1.text <> “ “ then:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,1] = ddlName1.text then:
                        TxtID1.text = ArrayEquipment[x,0]
                        TxtAmount1.text = ArrayEquipment[x,2]
                        Position1 = x

If ddlName2.text <> “ “ then:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,1] = ddlName2.text then:
                        TxtID2.text = ArrayEquipment[x,0]
                        TxtAmount2.text = ArrayEquipment[x,2]
                        Position2 = x

If ddlName3.text <> “ “ then:
        For x in ArrayEquipment[x]:
                If ArrayEquipment[x,1] = ddlName3.text then:
                        TxtID3.text = ArrayEquipment[x,0]
                        TxtAmount3.text = ArrayEquipment[x,2]
```

```
                                        Position3 = x

If butSave clicked:
Declare ID1 as string = txtID1.text
Declare ID2 as string = txtID2.text
Declare ID3 as string = txtID3.text
Declare Amount1 as integer = txtAmount1.text
Declare Amount2 as integer = txtAmount2.text
Declare Amount 3 as integer = txtAmount3.text

If txtAmount1 <> “” then
        If amount1 > 1000 or amount1 < 0 then
                Output(“Amount 1 must be <1000 and >0”)
                Exit sub
        End if
        ArrayEquipment[position1,2] = amount1
End if

If txtAmount2 <> “” then
        If amount2 > 1000 or amount2 < 0 then
                Output(“Amount 2 must be <1000 and >0”)
                Exit sub
        End if
        ArrayEquipment[position2,2] = amount2
End if

If txtAmount3 <> “” then
        If amount3 > 1000 or amount3 < 0 then
                Output(“Amount 3 must be <1000 and >0”)
                Exit sub
        End if
        ArrayEquipment[position3,2] = amount3
End if

Output(“NOTE: Equipment ID cannot be edited”)

If butCancel clicked:
Print “No data has been changed”
```

A worker's bookings for today




```
If butDisplay clicked:
Load_Bookings()
Declare BookingsList as string = ""

For x in BookingsArray:
        If Booking(x).StaffID = LoggedInID and Booking(x).date = today then:
                BookingsList = BookingsList & Booking(x) & returnButton.click(3)

Output(BookingsList)

If butClose clicked:
Open StaffMenu
```