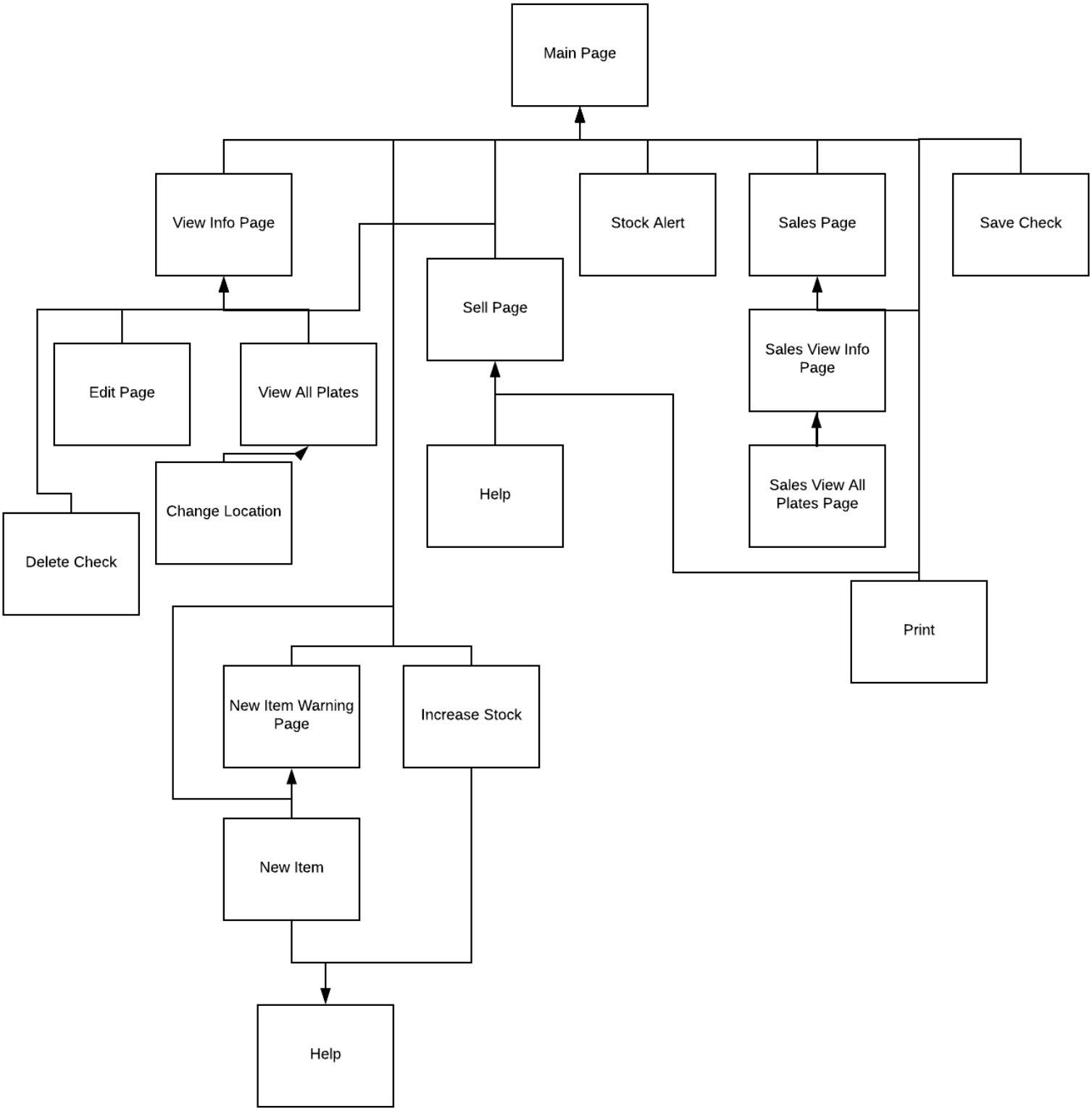
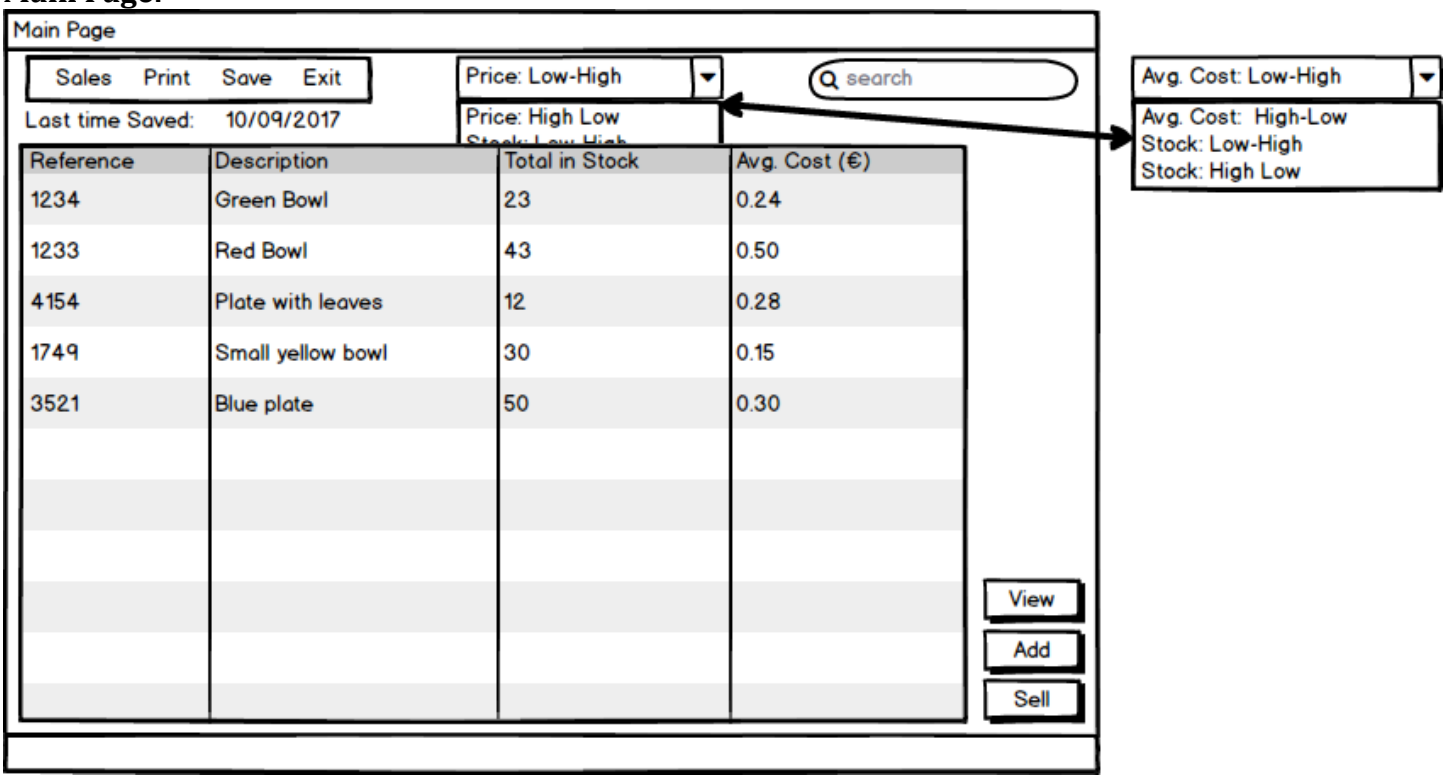
Criterion B: Design

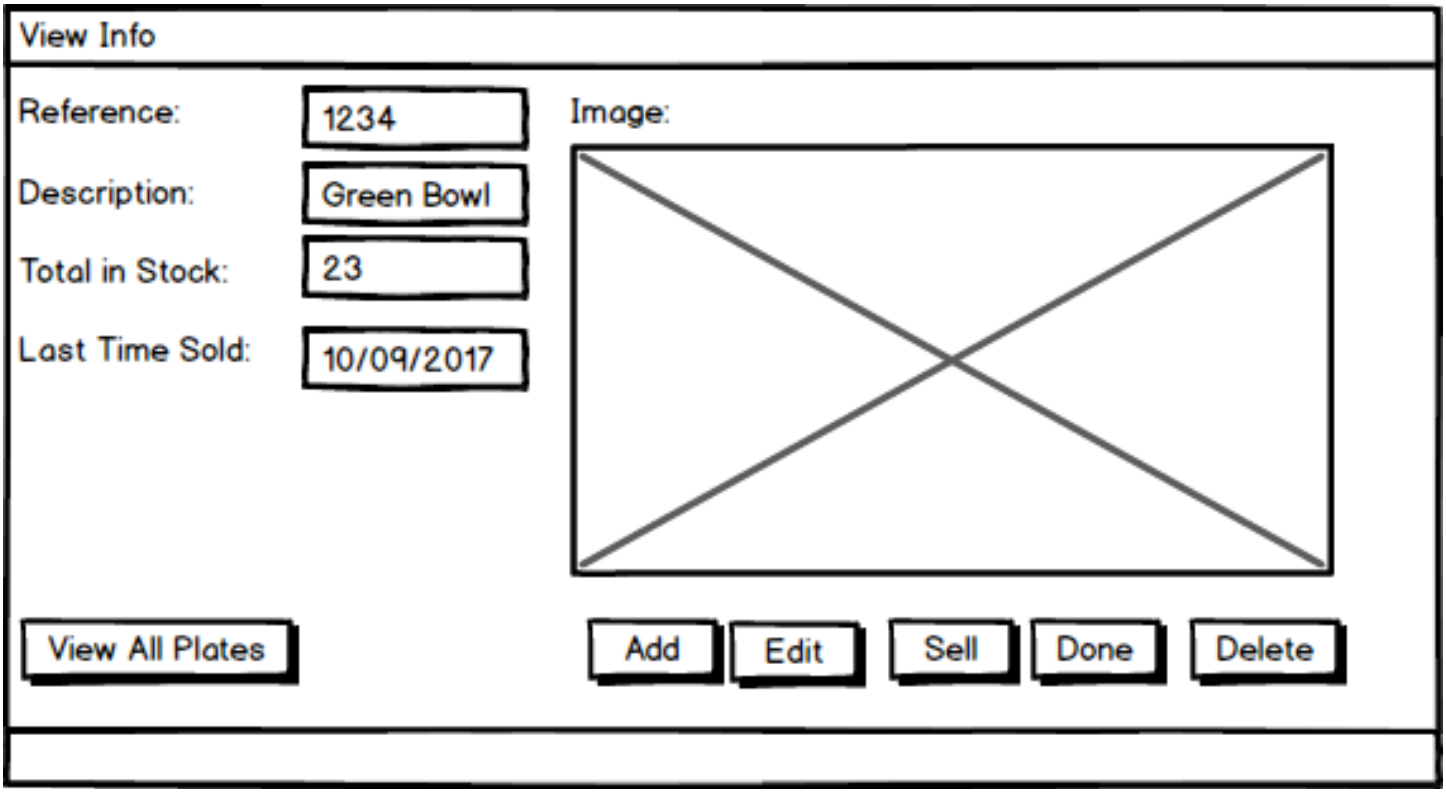
Overall Structure



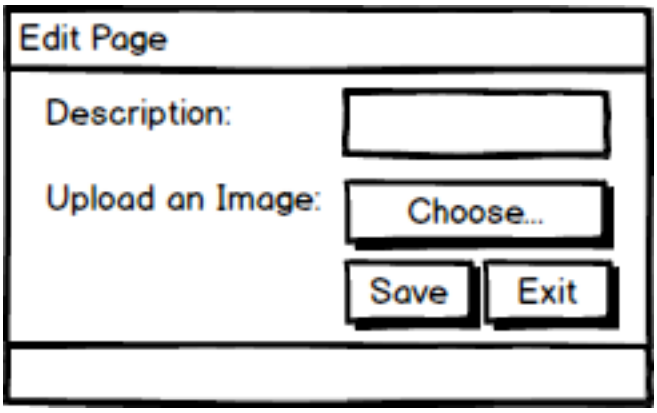
Wireframe
Main Page:



View Info:



Edit:



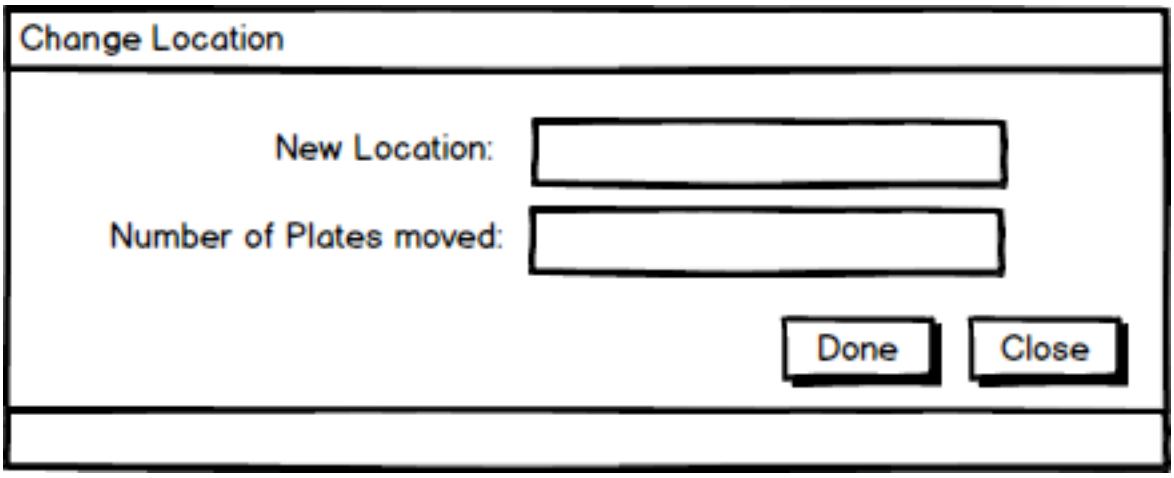
View All Plates:

View All Plates

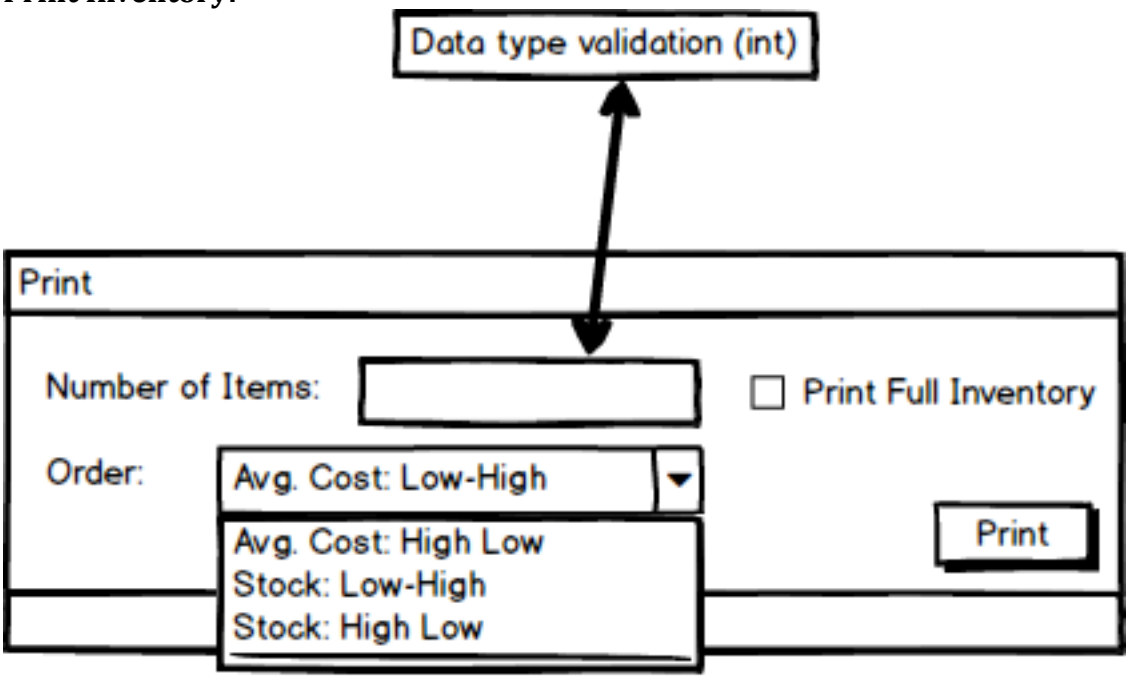
Amount	Cost	Location
23	.20	Madrid
23	.20	Lisbon
23	.30	Madrid

Change Location | Done

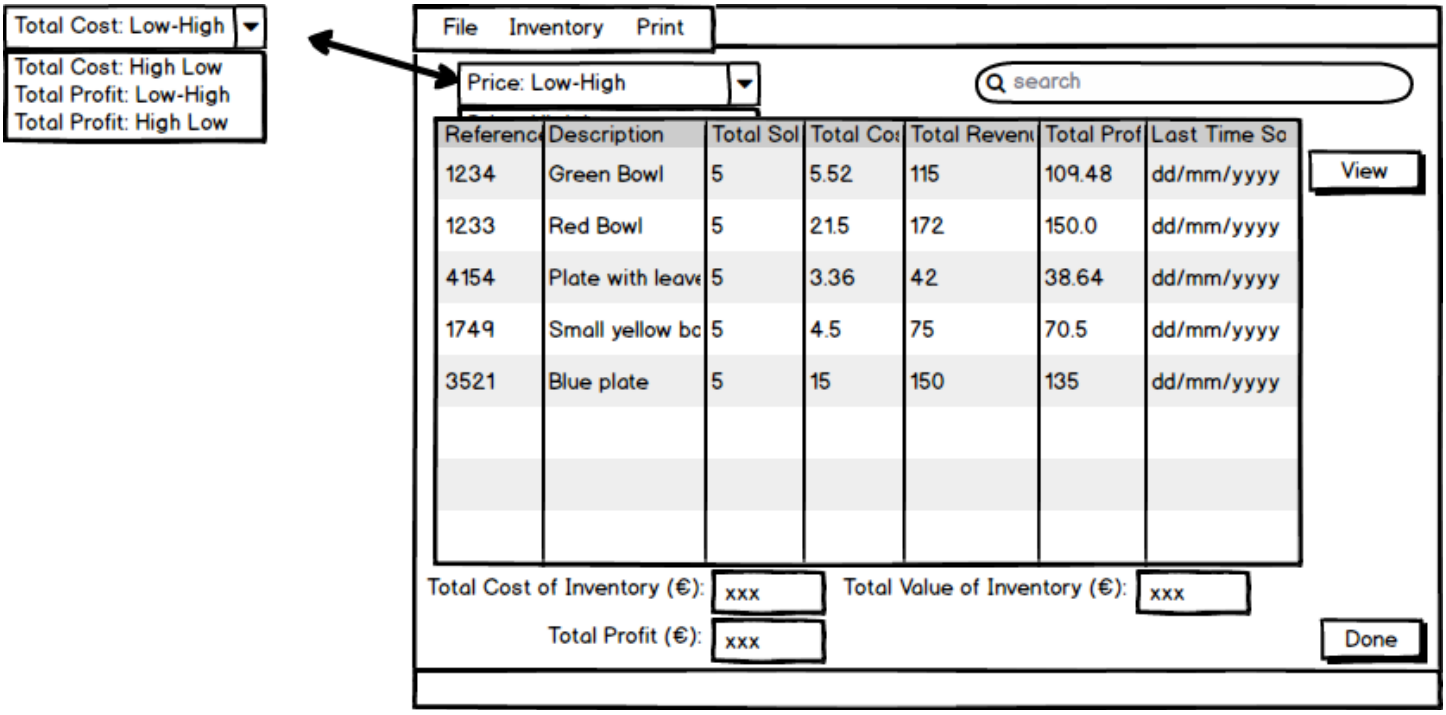
Change Location:



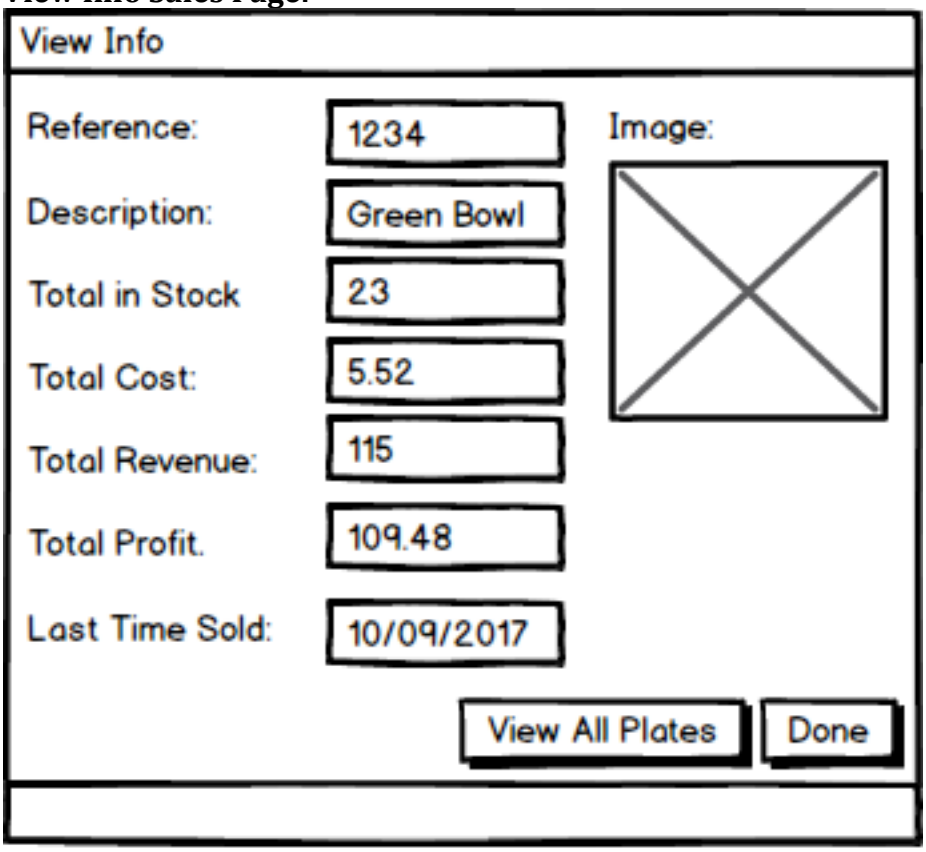
Print Inventory:



Sales Page:



View Info Sales Page:



View All Plates Sales:

View All Plates

Amount Sold	Price	Cost	Total Profit	Customer	Date
5	5	4	21	John Smith	20/01/2018
7	6	4	38	Catherine H.	20/11/2017
10	8	4	74	Jack R.	20/12/2017

Done

Print Sales:

Print

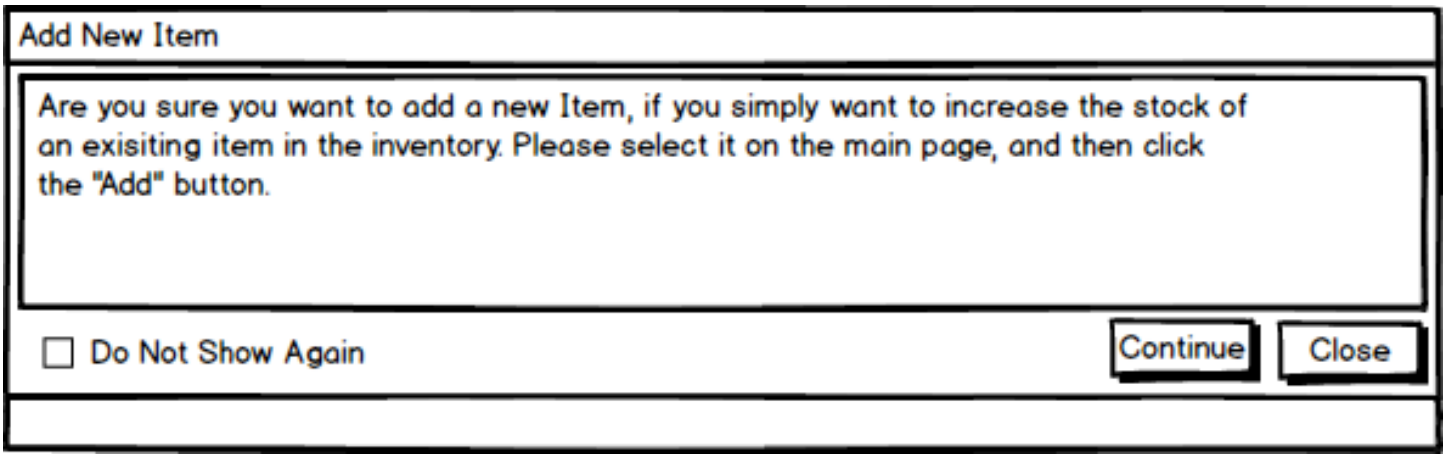
Select Start Date: / /

Select End Date: / /

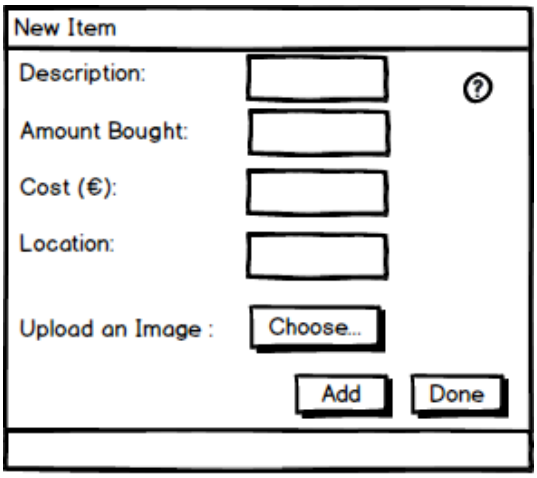
Print Full Sales

Print

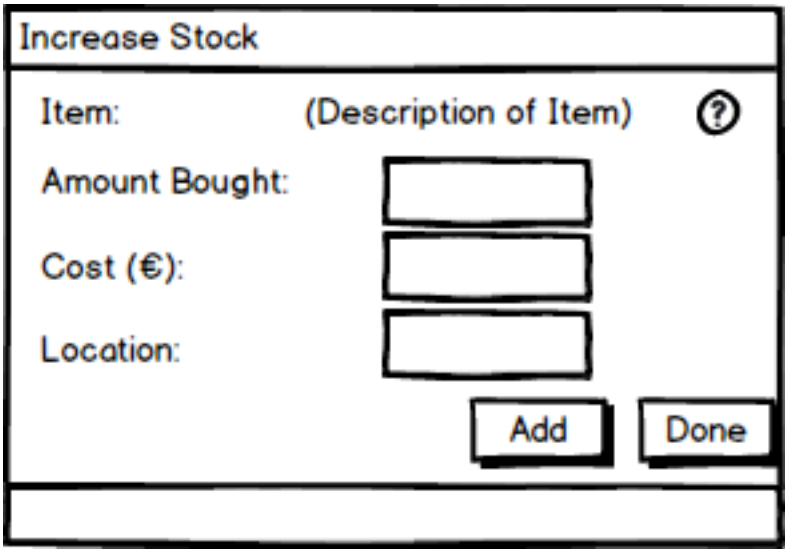
Add New Item Warning Page:



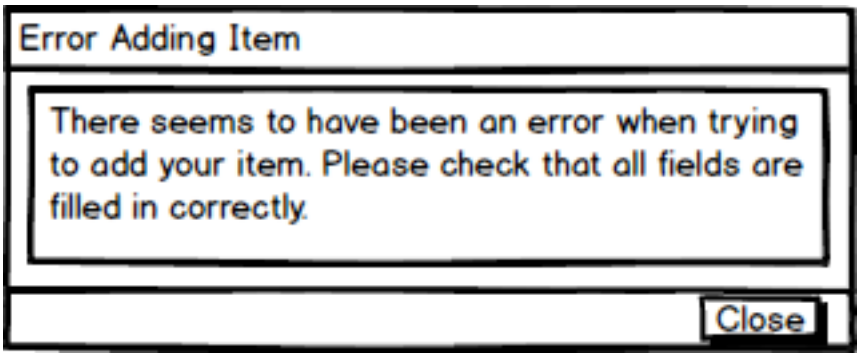
New Item:



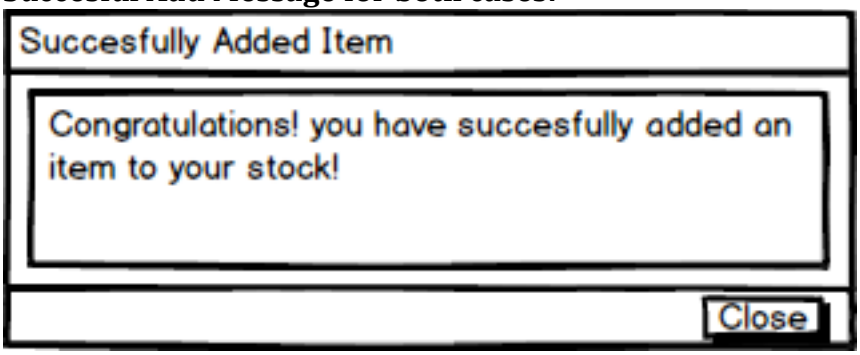
Increase Stock:



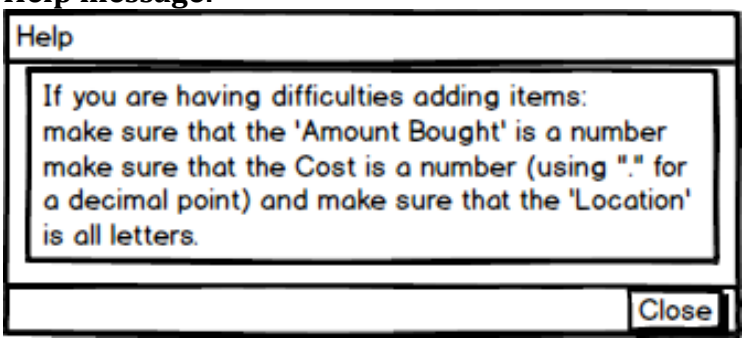
Error Message for both cases:



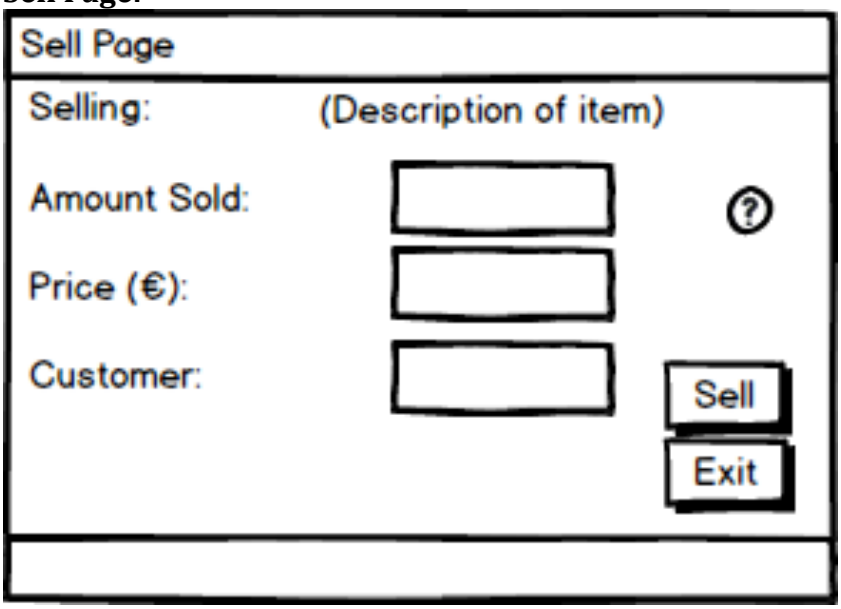
Succesful Add Message for both cases:



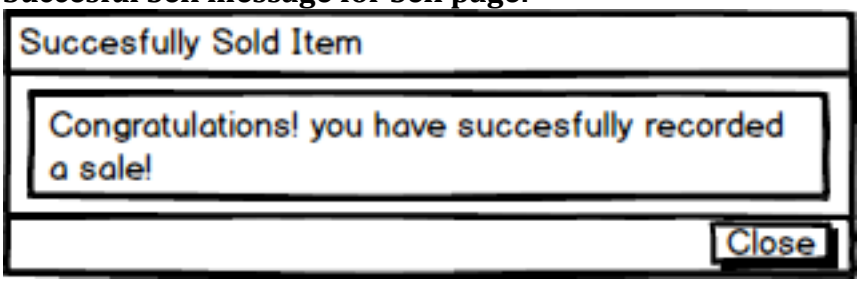
Help message:



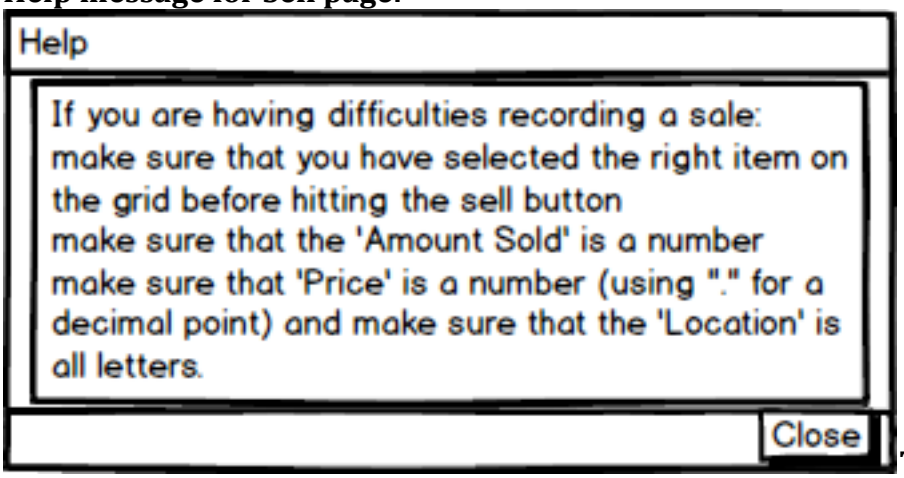
Sell Page:



Succesful Sell message for Sell page:

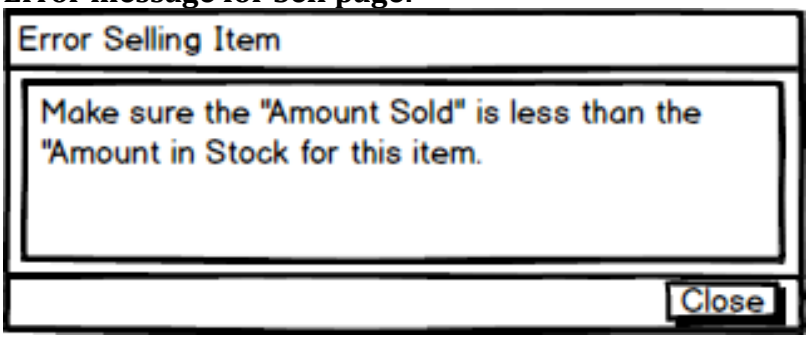


Help message for Sell page:

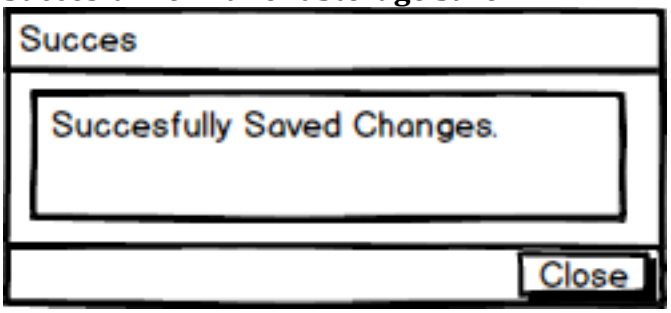


Tm

Error message for Sell page:



Succesful Permanent Storage Save:



Stock Alert:

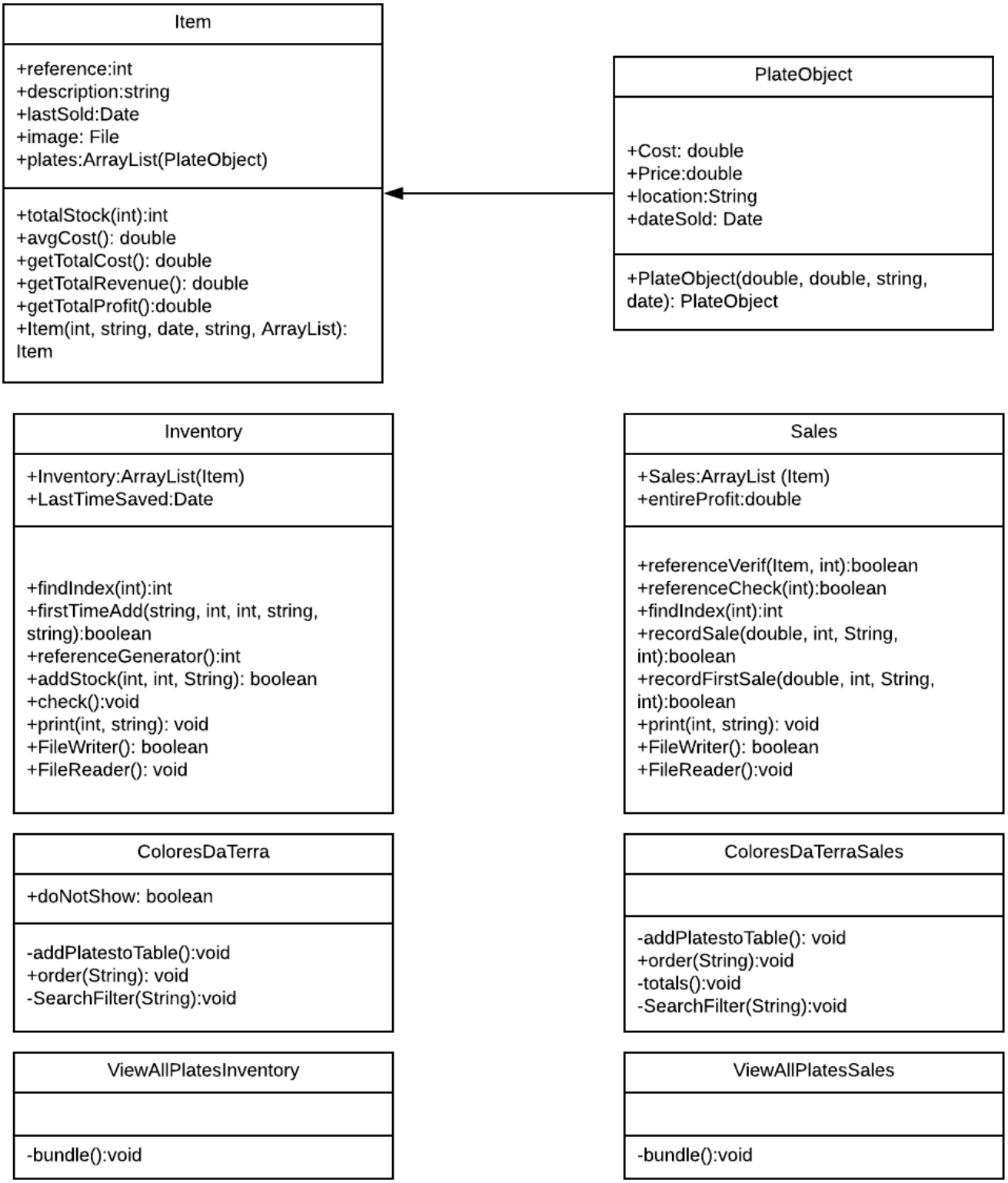
Stock Alert

The following objects have less than five items in stock:

Reference	Description	Total Stock	Avg. Cost
1000	xxxx	2	3.0
1001	yyyy	3	5.0
1002	zzzz	1	12.0

Close

UML:


Item
+reference:int
+description:string
+lastSold:Date
+image: File
+plates:ArrayList(PlateObject)
+totalStock(int):int
+avgCost(): double
+getTotalCost(): double
+getTotalRevenue(): double
+getTotalProfit():double
+Item(int, string, date, string, ArrayList): Item
PlateObject
+Cost: double
+Price:double
+location:String
+dateSold: Date
+PlateObject(double, double, string, date): PlateObject
Inventory
+Inventory:ArrayList(Item)
+LastTimeSaved:Date
+findIndex(int):int
+firstTimeAdd(string, int, int, string, string):boolean
+referenceGenerator():int
+addStock(int, int, String): boolean
+check():void
+print(int, string): void
+FileWriter(): boolean
+FileReader(): void
Sales
+Sales:ArrayList (Item)
+entireProfit:double
+referenceVerif(Item, int):boolean
+referenceCheck(int):boolean
+findIndex(int):int
+recordSale(double, int, String, int):boolean
+recordFirstSale(double, int, String, int):boolean
+print(int, string): void
+FileWriter(): boolean
+FileReader():void
ColoresDaTerra
+doNotShow: boolean
-addPlatestoTable():void
+order(String): void
-SearchFilter(String):void
ColoresDaTerraSales
-addPlatestoTable(): void
+order(String):void
-totals():void
-SearchFilter(String):void
ViewAllPlatesInventory
-bundle():void
ViewAllPlatesSales
-bundle():void

Example of an Object:

Object in Inventory

Reference: 1234
Description: "Green Bowl"
LastSold(yyyy-mm-dd): 09/28/17
ImageLocation: "/images/image.jpg"

ArrayList<PlateObject> plates = new ArrayList<PlateObject>

Cost: 0.50 Price: 0 Location: Madrid dateSold: null	Cost: 0.30 Price: 0 Location: Madrid dateSold: null	Cost: 0.60 Price: 0 Location: Madrid dateSold: null
Cost: 0.50 Price: 0 Location: Madrid dateSold: null	Cost: 0.50 Price: 0 Location: Lisbon dateSold: null	Cost: 0.42 Price: 0 Location:Lisbon dateSold: null

Object in Sales

Reference: 1234
Description: "Green Bowl"
LastSold(yyyy-mm-dd): 09/28/17
ImageLocation: "/images/image.jpg"

ArrayList<PlateObject> plates = new ArrayList<PlateObject>

Cost: 0.50 Price: 5 Location: John S. dateSold: 20/01/2018	Cost: 0.30 Price: 5 Location: John S. dateSold: 20/01/2018	Cost: 0.60 Price: 6 Location: Jack R. dateSold: 20/12/2017
Cost: 0.50 Price: 6 Location: Jack R. dateSold: 20/12/2017	Cost: 0.50 Price: 6 Location: Jack R. dateSold: 20/12/2017	Cost: 0.42 Price: 6 Location: Jack R. dateSold: 20/12/2017

Note: The location variable serves to store the location of where the plate is being stored if it is still in the inventory, but it serves to store the name of the buyer(customer) in the Sales database, as the plate's new location is with that customer.

Algorithms:

findIndex(int r**)**
REFERENCE = r
Loop X from 0 to Inventory.size()
    If(Inventory[X].reference = REFERENCE
        INDEX = X
    End if
End loop
Return X

totalStock(int a**)**
    I = findIndex(a)
    Return Inventory.get(I).plates.size();

avgCost()
TOTAL = 0
Loop X from 0 to plates.size()
    TOTAL = TOTAL + plates[X].cost
End loop
AVERAGE= TOTAL/plates.size()
Return AVERAGE

referenceGenerator()
REFERENCE = (random number between 1 and 10000)
X= 0
while(Inventory[X].reference = REFERENCE)
REFERENCE = (random number between 1 and 10000)
X = X +1
End loop
Return REFERENCE

stockCheck()
lowStock = new ArrayList
Loop X from 0 to Inventory.size()
If(Inventory[X].totalStock <5)
lowStock.add(Inventory[X])
end if
end loop

referenceVerif(int r, int a**)**
I = findIndex(r)
If(Inventory.get(i).totalStock()>a)
    Sale = true
    Else
    Sale =false
End if
Return sale

order(String s):
SORT = s

```
switch(SORT)
        case "Avg. Cost: Low-High":
        DisplayTable.sort(avgCost)

        case “Avg. Cost: High-Low":
        DisplayTable.sort(1-avgCost)

        case "Stock: Low-High":
        DisplayTable.sort(totalStock)

        case "Stock: High-Low":
        DisplayTable(1-totalStock)

End switch
```

order(String s**)** (for Sales)

```
SORT = s
Switch(SORT)
        Case “Total Cost: Low-High”
        DisplayTable.sort(totalStock)

        Case “Total Cost: High-Low”
        DisplayTable.sort(1-totalStock)

        Case “Total Profit: Low-High”
        DisplayTable.sort(totalProfit)

        Case “Total Profit: High-Low”
        DisplayTable.sort(1-totalProfit)

        Case “Last Time Sold: Low-High”
        DisplayTable.sort(lastSold – today())

        Case”Last Time Sold: High-Low”
        DisplayTable.sort(today()-lastSold)

End switch
```

firstTimeAdd(string D, int A, int C, string L, string I**)**

```
REFERENCE = referenceGenerator()
DESCRIPTION = D
AMOUNT = A
COST = C
LOCATION = L
IMAGE = I
ArrayList P = new ArrayList
Loop X from 0 to AMOUNT-1
        P.add(PlateObject(COST, 0, LOCATION,null)
End loop
Inventory[].add(Item(REFERENCE, DESCRIPTION, null, IMAGE, P)
```

addStock(int A, int C, string L, int R**)**

```
AMOUNT =A
COST =C
LOCATION = L
INDEX = findIndex(R)
Loop I from 0 to AMOUNT-1
    Inventory[INDEX].plates[].add(PlateObject(C, 0 ,L,null)
End loop
```

AddPlatestoTable()

MainPage:

```
Loop X from 0 to Inventory.size()
        rowData[0] = Inventory[X].reference
        rowData[1] = Inventory[X].description
        rowData[2] = Inventory[X].totalStock()
        rowData[3] = Inventory[X].avgCost()
        model.addRow(rowData);
end loop
```

SalesPage

```
Loop X from 0 to Sales.size()
  rowData[0] = Inventory[X].reference
        rowData[1] = Inventory[X].description
        rowData[2] = Inventory[X].getTotalCost()
        rowData[3] = Inventory[X].getTotalRevenue()
  rowData[4] = Inventory[X].getTotalProfit
  model.addRow(rowData);
end loop
```

View All Plates Inventory (bundle())

```
UNQ = new ArrayList()
UNQ.add(0)
Loop A from 0 to plates.size()
        COUNTUN = 0
        Loop K from 0 to plates.size()
                If(!(plates[A].cost = plates[UNQ[K]].cost AND plates[A].location =
                plates[UNQ[K]].location))
                        COUNTUN = COUNTUN+1
                End if
        End loop
        If(COUNTUN = UNQ.size())
                UNQ.add(A)
        End if
End loop
Loop X from 0 to UNQ.size()
        COST = plates[UNQ[X]].cost
        LOCATION = plates[UNQ[X]].location
        COUNT = 0
        Loop Y from 0 to plates.size()
                If(COST = plates[X].cost AND LOCATION = plates[X].location)
                COUNT = COUNT +1
```

```
            End if

    rowData[] = new Array[3]
    rowData[0] =COUNT
    rowData[1]= COST
    rowData[2]= LOCATION
    DisplayTable.add(rowData)
End loop
```

View All Plates Sales (bundle())

```
UNQ = new ArrayList()
UNQ.add(0)
DateFormat = "dd/MM/yyyy"
Loop A from 0 to plates.size()
            COUNTUN = 0
            Loop K from 0 to plates.size()
                If(!(plates[A].cost = plates[UNQ[K]].cost AND plates[A].price = plates[UNQ[K]] AND
                plates[A].location = plates[UNQ[K]].location AND plates[A].dateSold =
                plates[UNQ[K]].dateSold ))
                    COUNTUN = COUNTUN+1
                End if
            End loop
            If(COUNTUN = UNQ.size())
                UNQ.add(A)
            End if
End loop
Loop X from 0 to UNQ.size()
            COST = plates[UNQ[X]].cost
            PRICE = plates.[UNQ[X]].price
            LOCATION = plates[UNQ[X]].location
            DATE = plates.[UNQ[X]].dateSold
            COUNT = 0
            Loop Y from 0 to plates.size()
                If(plates[A].cost = plates[UNQ[K]].cost AND plates[A].price = plates[UNQ[K]] AND
                plates[A].location = plates[UNQ[K]].location AND plates[A].dateSold =
                plates[UNQ[K]].dateSold)
                    COUNT = COUNT +1
                    TOTALPROFIT = plates[Y].price – plates[Y].cost

                End if
            rowData[] = new rowData[6]
            rowData[0] = COUNT
            rowData[1] = PRICE
            rowData[2] = COST
            rowData[3]= TOTALPROFIT
            rowData[4] = LOCATION
            rowData[5] = DateFormat(DATE)
            DisplayTable.add(rowData)
    End loop
```

Change Location()

```
LOCATION = item.location
COST = item.location
NUMB = input
NEWLOCATION = input
Loop X from 0 to NUMB
        Loop Y from 0 to plates.size()
                If(plates[J].cost = cost AND plates[J].location =LOCATION
                        Plates[J].location = NEWLOCATION
                        Break
                End if
        End loop
End loop
```

recordFirstSale(double p, int a, String l , int r):

```
PRICE = p
AMOUNT = a
LOCATION = l
DATESOLD = today()
INVENTINDEX = Inventory.findIndex(r)
SALESINDEX = Sales.findIndex(r)
Loop COUNT from 0  to AMOUNT
        COST = Inventory[INVENTINDEX].plates[0].cost
        Sales[SALESINDEX].add(PlateObject(COST, PRICE, LOCATION,DATESOLD))
        Remove(Inventory[I].plates[0])
End loop
DateFormat = "dd/MM/yyyy"
Inventory[INVENTINDEX].lastSold = today()
DateFormat(Inventory[INVENTINDEX])
Sales[SALESINDEX].lastSold = today()
DateFormat(Sales[SALESINDEX])
```

recordSale(double p, int a, String l, int r):

```
PRICE = p
AMOUNT = a
LOCATION = l
DATESOLD = today()
INVENTINDEX = Inventory.findIndex(r)
ArrayList P = new ArrayList()
Loop COUNT from 0 to AMOUNT
   COST = Inventory[INVENTINDEX].plates[0].cost
   P.add(Plates(COST, PRICE, LOCATION,DATESOLD))
   Remove(Inventory[I].plates[0])
End loop
Inventory[INVENTINDEX].lastSold = today()
REFERENCE = Inventory[INVENTINDEX].reference
DESCRIPTION = Inventory[INVENTINDEX].description
Inventory[INVENTINDEX].lastSold = today()
IMAGE = Inventory[INVENTINDEX].image
SALES[].add(Item(REFRENCE, DESCRIPTION, LASTSOLD,IMAGE, P))
```

getTotalCost():

```
TOTALCOST = 0
loop X from 0 to plates.size()
        TOTALCOST = TOTALCOST + plates[X].cost
End loop
Return totalCost
```

getRevenue():redo

```
TOTALREV = 0
loop X from 0 to plates.size()
        TOTALREV = TOTALREV + plates[X].PRICE
End loop
Return TOTALREV
```

print(int I, String o):

- Use PdfWriter and Document libraries from iText, to print jTable to PDF

```
NUMBER = I
ORDER = 0
HEADING = "Colores Da Terra"
SUBTITLE = "Print out of" + NUMBER + "items in the order:" +ORDER
TABLE = new table("Reference", "Description", "Avg. Cost", "Total in Stock")
Loop X from 0 to NUMBER
TABLE.add(INVENTORY[X].reference)
TABLE.add(INVENTORY[X].description)
TABLE.add(INVENTORY[X].totalStock())
TABLE.add(INVENTORY[X].avgCost())
end loop
TABLE.add("")
TABLE.add("")
TABLE.add("")
TABLE.add("Total:")
TOTAL = 0
Loop Y from 0 to NUMBER
TOTAL = TOTAL+ INVENTORY[Y].totalStock()
End Loop
TABLE.add(TOTAL)
```

Printed Out PDF for Inventory:

Colores Da Terra

Print out of 15 objects in order: Stock: Low-High

Reference	Description	Total in Stock	Avg. Cost(€)	Total Cost(€)
3516	Red Plate1	1	14.0	14.0
1855	Yellow Bowl1	1	14.0	14.0
773	Red Plate1	1	14.0	14.0
7911	Yellow Bowl1	1	14.0	14.0
7141	Red Plate1	1	14.0	14.0
2335	Yellow Bowl1	1	14.0	14.0
9711	Red Plate1	1	14.0	14.0
8530	Yellow Bowl1	1	14.0	14.0
5984	Red Plate1	1	14.0	14.0
564	Yellow Bowl1	1	14.0	14.0
8327	Red Plate1	1	14.0	14.0
2032	Yellow Bowl1	1	14.0	14.0
5617	Red Plate1	1	14.0	14.0
5358	Yellow Bowl1	1	14.0	14.0
2645	Red Plate1	1	14.0	14.0
			Total:	**210.0**

print(date b, date e) (Sales)

```
START = b
END = e
PROVI = newArraylist
DateFormat = "dd/MM/yyyy"
Loop X from 0 to sales.size()
    CONTAINS = false
    Loop Y from 0 to sales[X].plates.size()
    If(Sales[X].plates[Y].dateSold.after(START) AND Sales[X].plates[Y].dateSold.before(END))
        CONTAINS = true
        Break
    End if
    End loop
    If(CONTAINS = true)
        PROVP = new Arraylist
        Loop Z from 0 to Sales[X].plates.size()
        If(Sales[X].plates[Z].dateSold.after(START) AND Sales[X].plates[Z].dateSold.before(END))
            PROVP.add(Sales[X].plates[Z])
        End if
        End loop
        PROVI.add( new Item(Sales[X].reference, Sales[X].description, Sales[X].lastSold,
        Sales[X].image, PROVP ))
HEADER = "Colores Da Terra Sales'
SUBTITLE = "Print out of objects between" +START+"and"+END
TABLE = new TABLE(Reference, Description, Total Sold, Total Cost, Total Revenue, Total Profit)
Loop X from 0 to PROVI.size()
```

```
        TABLE.add(PROVI[X].reference)
        TABLE.add(PROVI[X].description)
        TABLE.add(PROVI[X].plates.size())
        TABLE.add(PROVI[X].getTotalCost())
        TABLE.add(PROVI[X].getTotalRevenue())
        TABLE.add(PROVI[X].getTotalProfit())
End loop
TOTALCOST = 0
TOTALREV = 0
TOTALPROFIT = 0
Loop Y from 0 to PROVI.size()
        TOTALCOST = TOTALCOST + PROVI[Y].getTotalCost()
        TOTALREV = TOTALREV +PROVI[Y].getTotalRevenue()
        TOTALPROFIT = TOTALPROFIT + PROVI[Y].getTotalProfit()
End loop
TABLE.add("")
TABLE.add("")
TABLE.add("Total:")
TABLE.add(TOTALCOST)
TABLE.add(TOTALREV)
TABLE.add(TOTALPROFIT)
```

Printed out PDF for Sales:

Colores Da Terra Sales

Print out of objects between 01/12/2017 - 31/12/2017

Reference	Description	Total Sold	Total Cost	Total Revenue	Total Profit
8488	Red Bowl	1	2.00	6.00	4.00
8332	Yellow Bowl	1	2.00	6.00	4.00
1974	Red Bowl	1	2.00	6.00	4.00
735	Yellow Bowl	1	2.00	6.00	4.00
8655	Red Bowl	1	2.00	6.00	4.00
1604	Yellow Bowl	1	2.00	6.00	4.00
2516	Red Bowl	1	2.00	6.00	4.00
3301	Yellow Bowl	1	2.00	6.00	4.00
725	Red Bowl	1	2.00	6.00	4.00
1634	Yellow Bowl	1	2.00	6.00	4.00
5739	Red Bowl	1	2.00	6.00	4.00
2965	Yellow Bowl	1	2.00	6.00	4.00
2871	Red Bowl	1	2.00	6.00	4.00
812	Yellow Bowl	1	2.00	6.00	4.00
6830	Red Bowl	1	2.00	6.00	4.00
1533	Yellow Bowl	1	2.00	6.00	4.00
1525	Red Bowl	1	2.00	6.00	4.00
3176	Yellow Bowl	1	2.00	6.00	4.00
5516	Red Bowl	1	2.00	6.00	4.00
9911	Yellow Bowl	1	2.00	6.00	4.00
3589	Red Bowl	1	2.00	6.00	4.00
7150	Yellow Bowl	1	2.00	6.00	4.00
		Total:	**44.00**	**132.00**	**88.00**

FileWriter():

```
SV=lastSaved
Print(SV)
```

```
Sv=""
Loop X from 0 to Inventory.size()
        If(Inventory[X].lastSold = null)
            If(Inventory[X].image = null)
                SV = Inventory[X].reference +","+Inventory[X].description++","+" "+","+" "
            Else
                SV = Inventory[X].reference +","+Inventory[X].description+","+"
                "+","+Inventory[X].image.toPath()
                End if
        Else
            If(Inventory[X].image = null)
                SV = Inventory[X].reference
                +","+Inventory[X].description++","+Inventory[X].lastSold+","+" "
            Else
                SV = Inventory[X].reference +","+Inventory[X].description+","+
                Inventory[X].lastSold+","+Inventory[X].image.toPath()
        End if
        Loop Y from 0 to Inventory[X].size()
                SV = SV + "," + Inventory[X].plates[Y].cost +"," + Inventory[X].plates[Y].price + "," +
                Inventory[X].plates[Y].location
        End loop
 Print(SV)
 End loop
```

FileReader():

- Elements are separated by commas, use string split method to get different elements. Elements are created and stored in arraylists, the algorithm is the same for Sales

```
TEMP = readLine()
While(!(TEMP = null))
    STORE[] = TEMP.split(",")
    LASTSAVED = STORE[0]
    REFERENCE= STORE[1]
    DESCRIPTION = STROE[2]
    If(STORE[3] = " ")
        LASTSOLD = null
    Else
        LASTSOLD = STORE[3]
    End if
    If(STORE[4] = " ")
        IMAGE = null
    Else
        IMAGE = new File(STORE[4])
    End if
P = new ArrayList
Loop X from 0 to STORE.size() (X =X +3 every loop)
COST = STORE[X]
PRICE = STORE[X+1]
LOCATION = STORE[X+2]
P.add(new PlateObject(COST, PRICE, LOCATION, null))
End loop
Inventory.add(new Item(REFERENCE,DESCRIPTION,LASTSOLD,IMAGE,P)
```

TEMP = readLine()

Sample .txt file for Inventory:

SaveInventory.txt — Edited

```
06/01/2018
1000,xxxx1, , ,14.0,0.0,Madrid
1001,yyyy1, , ,14.0,0.0,MLisbon
1002,xxxx2, , ,13.0,0.0,Madrid,13.0,0.0,Madrid
1003,yyyy2, , ,13.0,0.0,MLisbon,13.0,0.0,MLisbon
1004,xxxx3, , ,12.0,0.0,Madrid,12.0,0.0,Madrid,12.0,0.0,Madrid
1005,yyyy3, , ,12.0,0.0,MLisbon,12.0,0.0,MLisbon,12.0,0.0,MLisbon
1006,xxxx4, , ,11.0,0.0,Madrid,11.0,0.0,Madrid,11.0,0.0,Madrid,11.0,0.0,Madrid
1007,yyyy4, , ,11.0,0.0,MLisbon,11.0,0.0,MLisbon,11.0,0.0,MLisbon,11.0,0.0,MLisbon
1008,xxxx5, , ,10.0,0.0,Madrid,10.0,0.0,Madrid,10.0,0.0,Madrid,10.0,0.0,Madrid
1009,yyyy5, , ,10.0,0.0,MLisbon,10.0,0.0,MLisbon,10.0,0.0,MLisbon,10.0,0.0,MLisbon
1020,qwerty, ,/Users/felipedelclaux/Desktop/IMG_2345.JPG,0.4,0.0,Lisbon,0.4,0.0,Lisbon
```

Sample .txt file for Sales:

SaveSales

```
3711,1234,01/04/2018, ,2.0,4.0,Jack,01/02/2018,2.0,4.0,Jack,01/02/2018,2.0,4.0,Jack,
01/03/2018,2.0,4.0,Jack,01/04/2018,2.0,4.0,Jack,01/04/2018,2.0,4.0,Jack,
01/04/2018,2.0,4.0,Jack,01/04/2018,2.0,4.0,Jack,01/04/2018,2.0,6.0,John Smith,
01/04/2018,2.0,6.0,John Smith,01/04/2018,2.0,6.0,John Smith,01/04/2018,2.0,6.0,John Smith,
01/04/2018,2.0,7.0,Catherine H.,01/04/2018,2.0,7.0,Catherine H.,
01/04/2018,2.0,7.0,Catherine H.,01/04/2018,2.0,7.0,Catherine H.,01/04/2018
8748,qwer,02/04/2018, ,2.0,6.0,Ms. Smith,02/01/2018,2.0,6.0,Ms. Smith,
02/01/2018,2.0,6.0,Ms. Smith,02/01/2018,2.0,6.0,Ms. Smith,02/01/2018,2.0,6.0,Ms. Smith,
02/01/2018,2.0,6.0,Ms. Smith,02/01/2018,2.0,6.0,Ms. Smith,02/01/2018,2.0,6.0,Ms. Smith,
02/01/2018
```

Word Count: 49