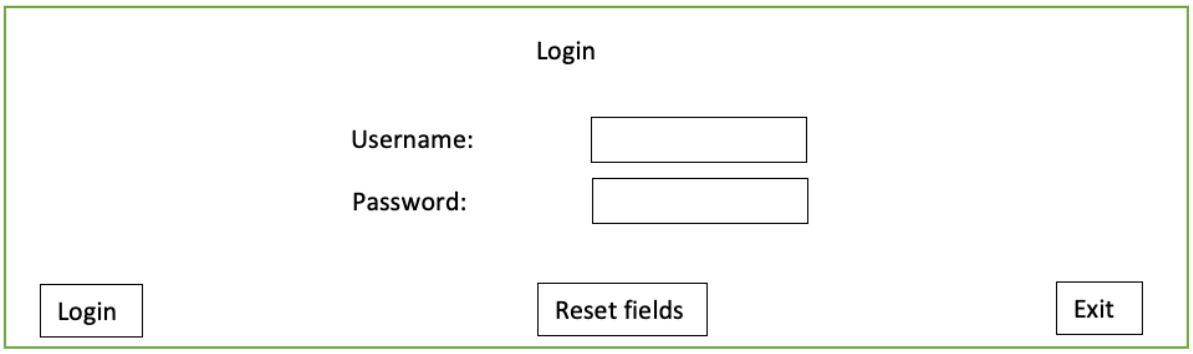
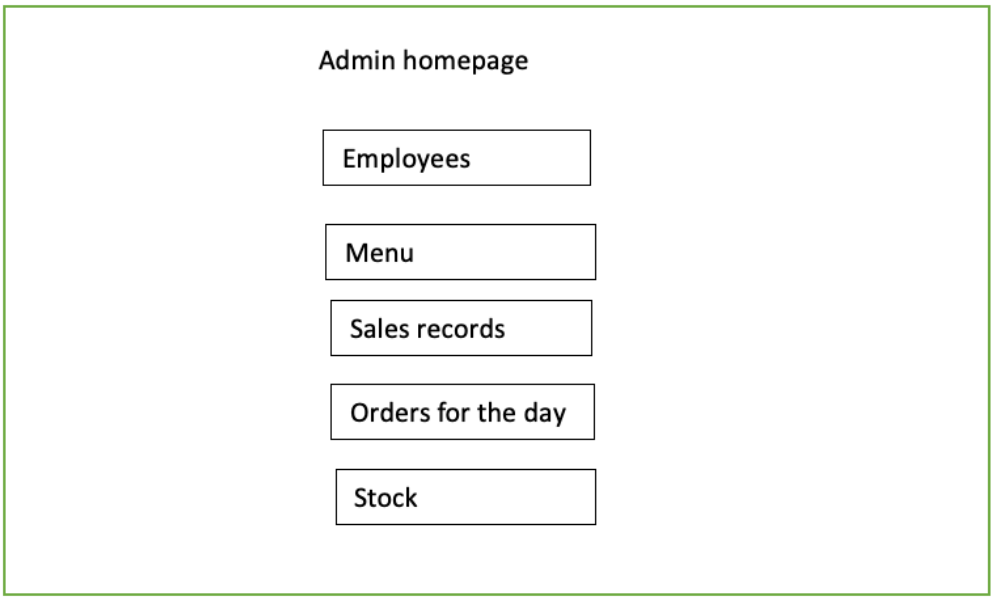
Criterion B: Design

Design of panels (graphical user interface)

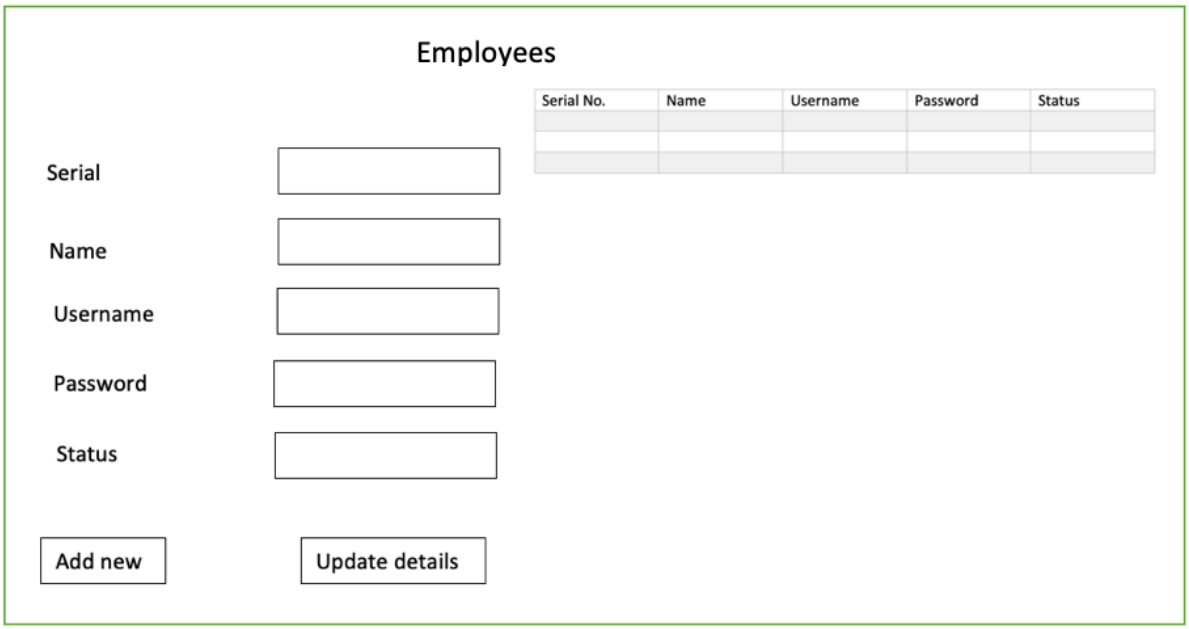
1. Login window:



2. Admin homepage



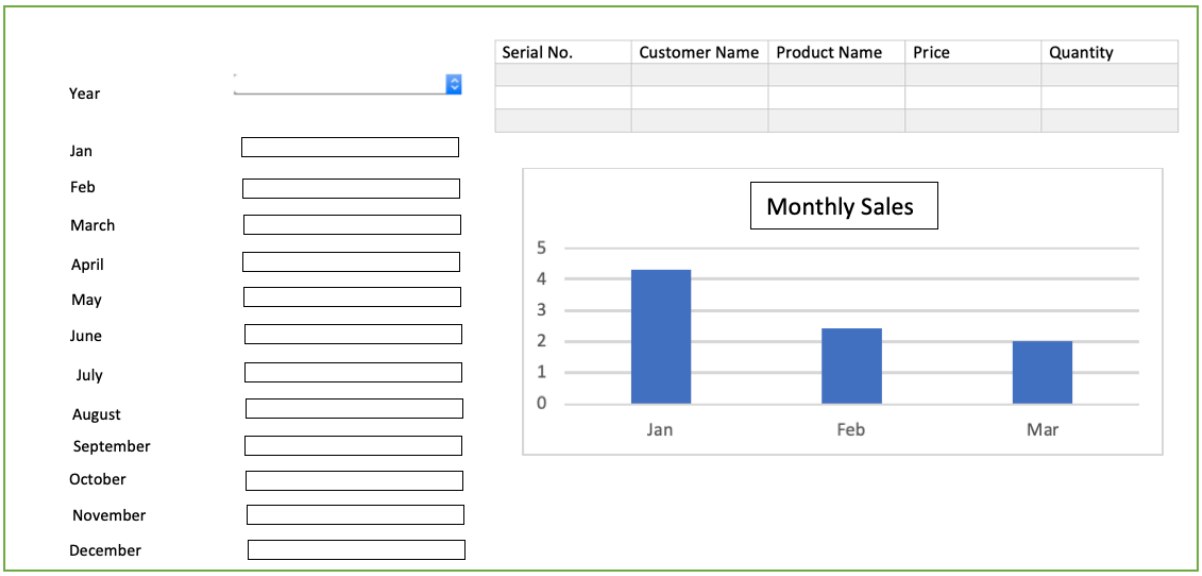
3. Employees



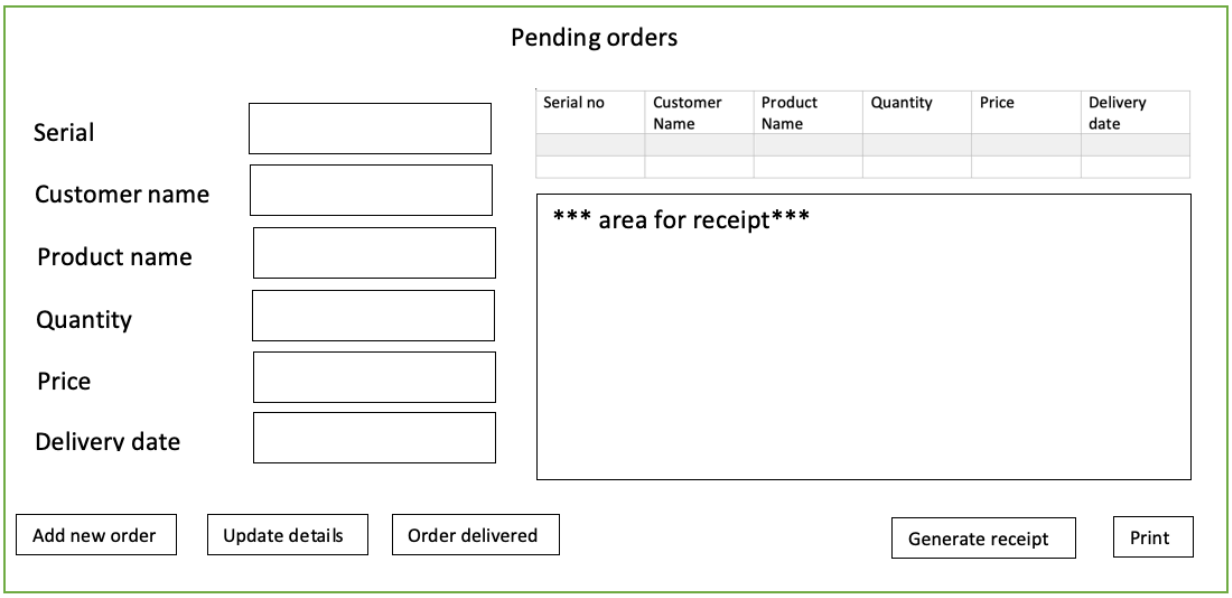
4. Menu

5. Past orders

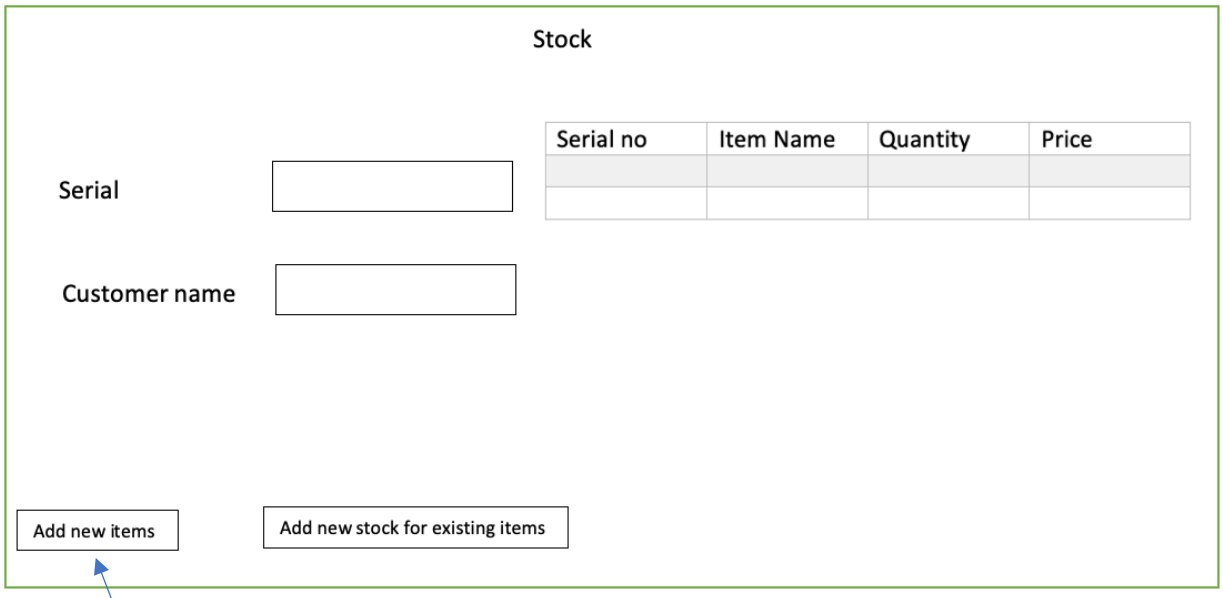
6. Statistics for sales (the graph will contain bars for all months)



7. Pending orders

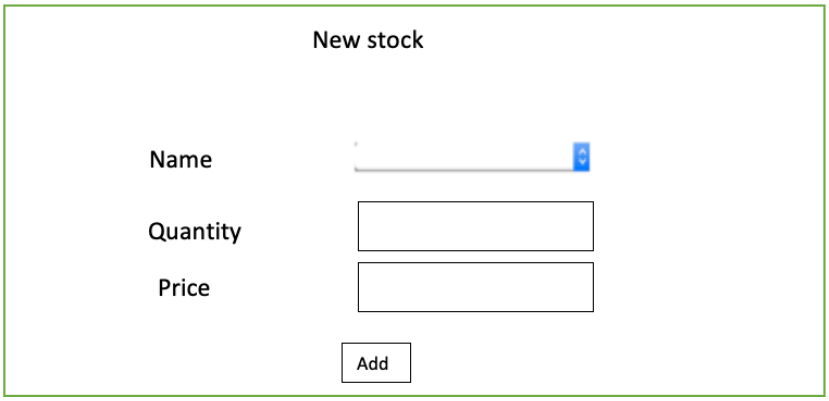


8. Stock

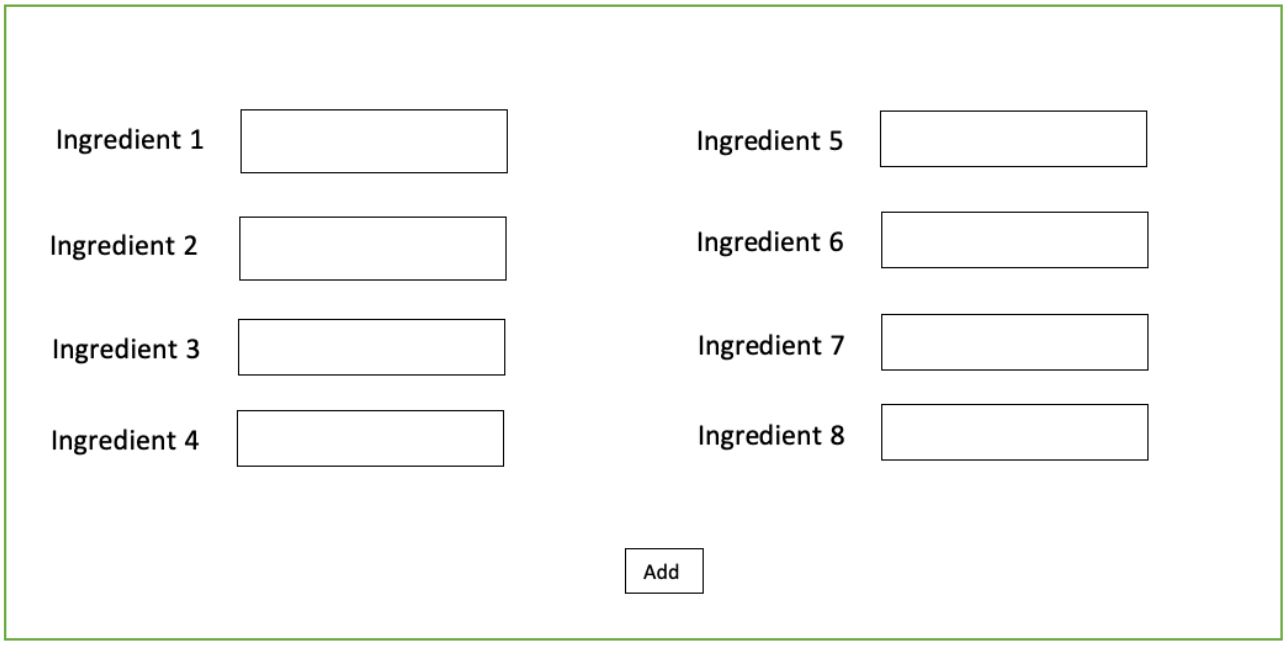


Clicking on the add new items button will lead to the page shown on the right:

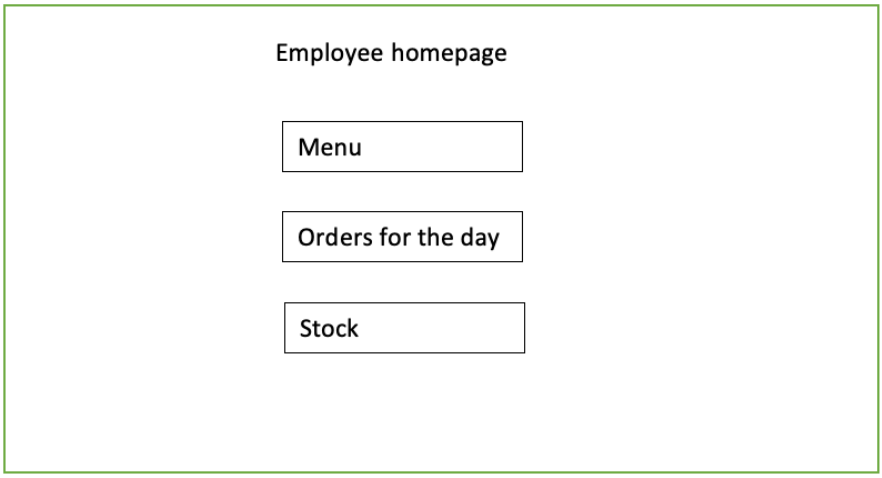
Clicking on the add new stock for existing items button will lead to the page shown on the right:



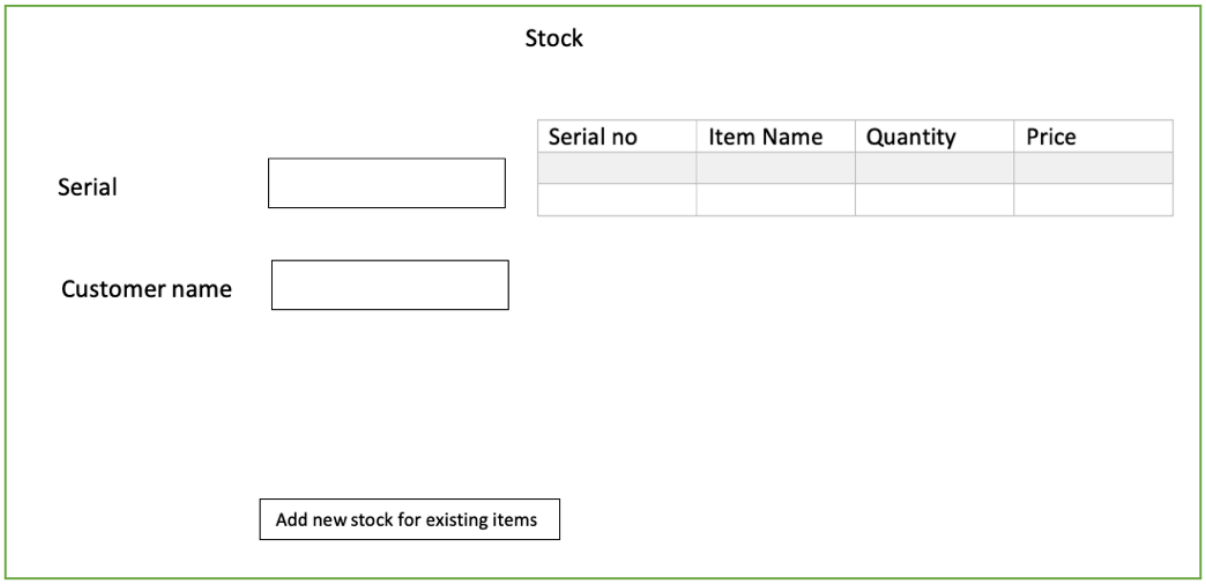
9. Ingredients page: when adding a new item in the menu, the program will ask for the amount of ingredients for a standard size. This will be achieved through the following page (names of ingredients later):



10. Employee homepage: the following is the homepage for employees. They will be redirected to the same pages as the admin, but they will only be able to view them and not make any changes.

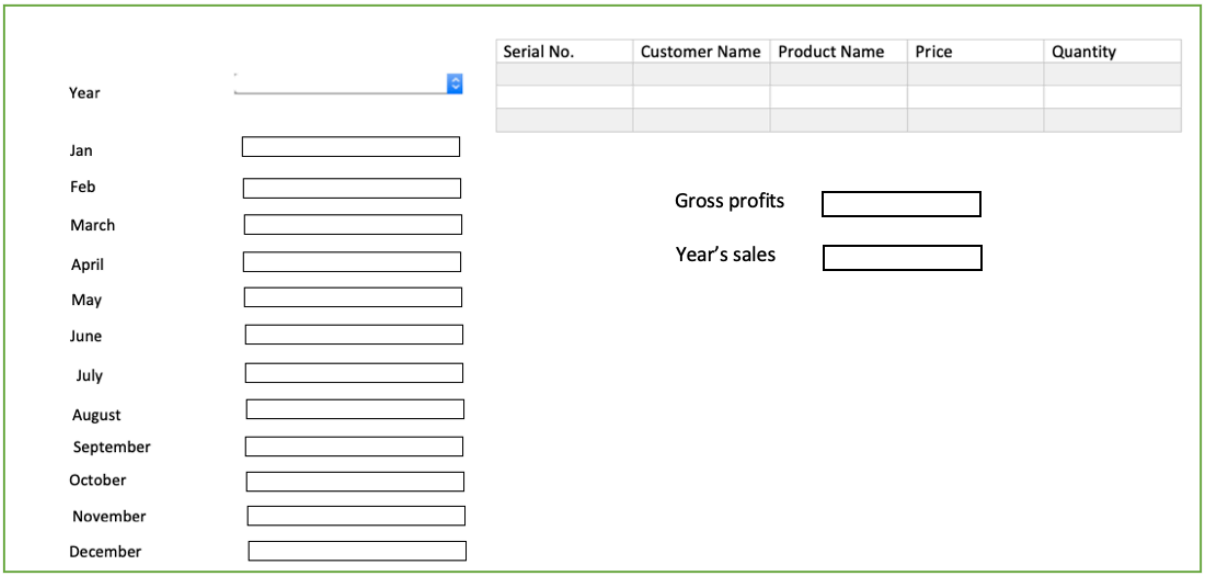


The prototypes shown above were the first version, the client wanted me to change the stock page[1], she did not want an add new stock page as she wanted any new item to come up in an update. The only stock items she wanted were included and the following page was made:



Hence, the new items page is void.

Another change the client wanted was that the graph should be on a new frame[2], instead of being plotted on the same frame. So, the following design was shown to the client:



[1] Refer to Appendix 2: Prototype interview
[2] Refer to Appendix 2: Prototype interview

UML diagrams for each class:

AddIngredients

- add: javax.swing.JButton
- back: javax.swing.JButton
- bp: javax.swing.JTextField
- bs: javax.swing.JTextField
- cm: javax.swing.JTextField
- colour: javax.swing.JTextField
- cp: javax.swing.JTextField
- essence: javax.swing.JTextField
- flour: javax.swing.JTextField
- jLabel1: javax.swing.JLabel
- jLabel10: javax.swing.JLabel
- jLabel11: javax.swing.JLabel
- jLabel3: javax.swing.JLabel
- jLabel4: javax.swing.JLabel
- jLabel5: javax.swing.JLabel
- jLabel6: javax.swing.JLabel
- jLabel7: javax.swing.JLabel
- jLabel8: javax.swing.JLabel
- jLabel9: javax.swing.JLabel
- jPanel1: javax.swing.JPanel
- oil: javax.swing.JTextField
- sugar: javax.swing.JTextField
- vinegar: javax.swing.JTextField

- void: cpActionPerformed(java.awt.event.ActionEvent: evt)
- void: addActionPerformed(java.awt.event.ActionEvent: evt)
- void: flourKeyTyped((java.awt.event.KeyEvent: evt)
- void: cmKeyTyped(java.awt.event.KeyEvent: evt)
- void: oilKeyTyped(java.awt.event.KeyEvent: evt)
- void: bpKeyTyped(java.awt.event.KeyEvent: evt)
- void: sugarKeyTyped(java.awt.event.KeyEvent: evt)
- void: cpKeyTyped(java.awt.event.KeyEvent: evt)
- void: vinegarKeyTyped(java.awt.event.KeyEvent: evt)
- void: colourKeyTyped(java.awt.event.KeyEvent: evt)
- void: backActionPerformed(java.awt.event.ActionEvent: evt)
- void: flourMouseEntered(java.awt.event.MouseEvent: evt)
- void: cmMouseEntered(java.awt.event.MouseEvent: evt)
- void: oilMouseEntered(java.awt.event.MouseEvent: evt)
- void: bpMouseEntered(java.awt.event.MouseEvent: evt)
- void: bsMouseEntered(java.awt.event.MouseEvent: evt)
- void: essenceMouseEntered(java.awt.event.MouseEvent: evt)
- void: sugarMouseEntered(java.awt.event.MouseEvent: evt)
- void: cpMouseEntered(java.awt.event.MouseEvent: evt)
- void: vinegarMouseEntered(java.awt.event.MouseEvent: evt)
- void: colourMouseEntered(java.awt.event.MouseEvent: evt)
- void: flourMouseExited(java.awt.event.MouseEvent: evt)
- void: cmMouseExited(java.awt.event.MouseEvent: evt)
- void: oilMouseExited(java.awt.event.MouseEvent: evt)
- void: bpMouseExited(java.awt.event.MouseEvent: evt)
- void: bsMouseExited(java.awt.event.MouseEvent: evt)
- void: essenceMouseExited(java.awt.event.MouseEvent: evt)
- void: sugarMouseExited(java.awt.event.MouseEvent: evt)
- void: cpMouseExited(java.awt.event.MouseEvent: evt)
- void: vinegarMouseExited(java.awt.event.MouseEvent: evt)
- void: colourMouseExited(java.awt.event.MouseEvent: evt)
- void: main(String: args[])

AddStock

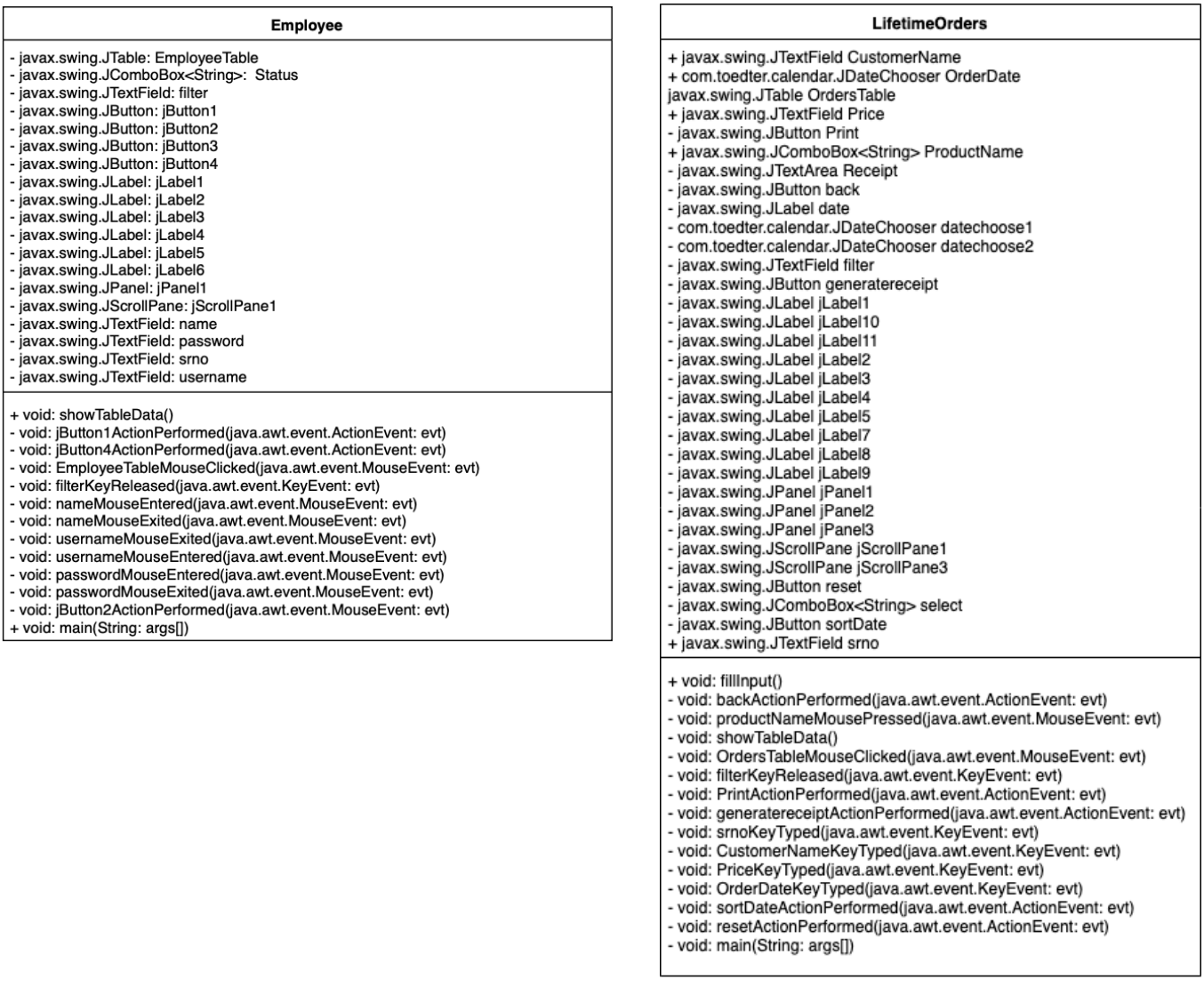
- javax.swing.JButton: Add
- javax.swing.JButton: jButton1
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JPanel: jPanel1
- javax.swing.JTextField: name
- javax.swing.JTextField: price
- javax.swing.JTextField: quantity

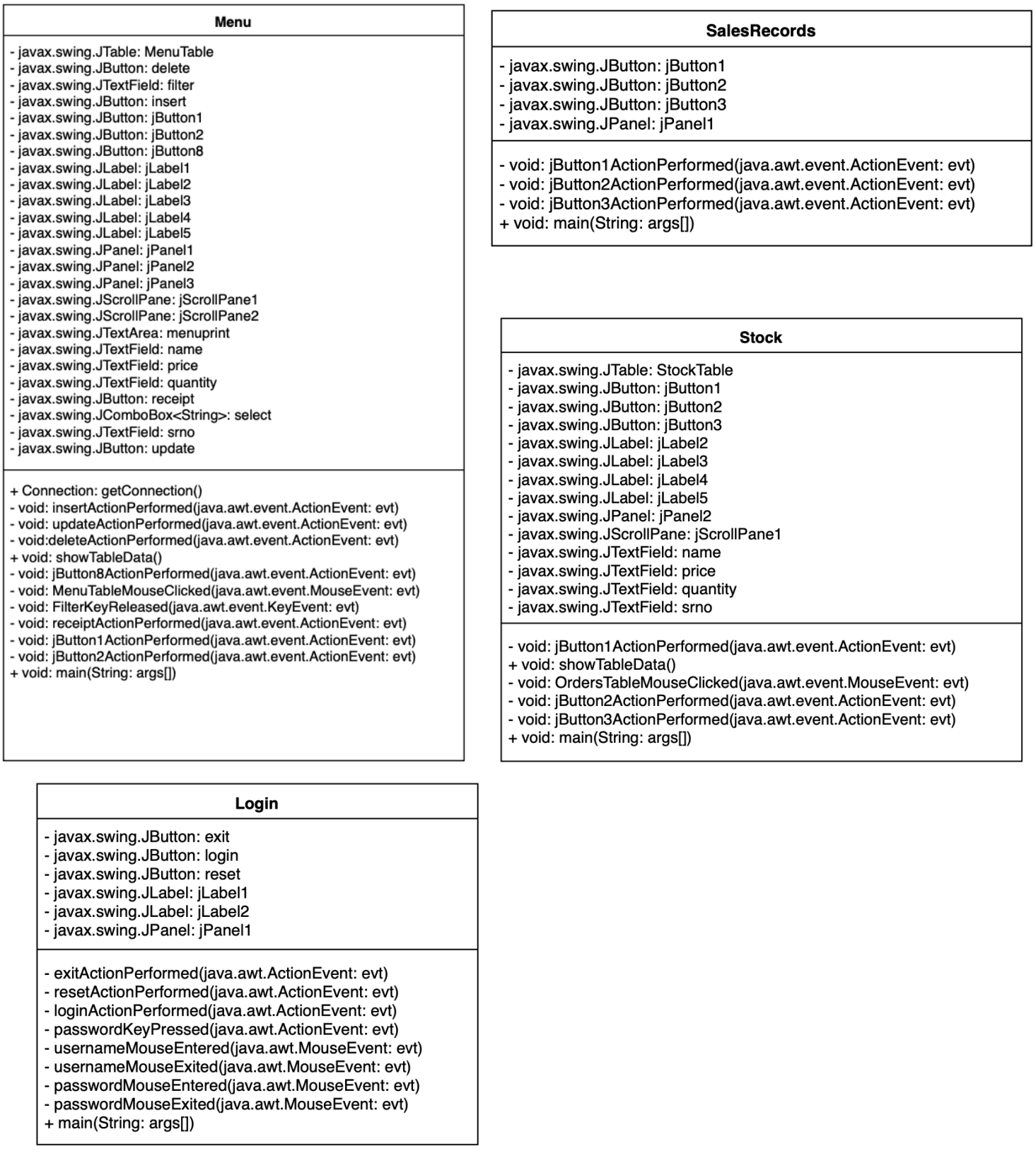
- void: AddActionPerformed(java.awt.event.ActionEvent: evt)
- void jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
- void: main(String: args[])

EditingForm

+ javax.swing.JTextField: CustomerName
- javax.swing.JButton: Delete
+ com.toedter.calendar.JDateChooser: DeliveryDate
  javax.swing.JTable: OrdersTable
+ javax.swing.JTextField: Price
- javax.swing.JButton: Print
+ javax.swing.JComboBox<String>: ProductName
- javax.swing.JTextArea: Receipt
- javax.swing.JButton: Update
- javax.swing.JButton: back
- javax.swing.JLabel: date
- javax.swing.JTextField: filter
- javax.swing.JButton: generatereceipt
- javax.swing.JButton: getprice
- javax.swing.JButton: insert
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JLabel: jLabel7
- javax.swing.JLabel: jLabel8
- javax.swing.JMenu: jMenu1
- javax.swing.JMenuItem: jMenuItem1
- javax.swing.JPanel: jPanel1
- javax.swing.JPanel: jPanel2
- javax.swing.JPanel: jPanel3
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JScrollPane: jScrollPane3
+ javax.swing.JTextField: quantity
- javax.swing.JComboBox<String>: select
+ javax.swing.JTextField: srno
- String: rprice
- String: rcustomername
- String: rproductname
- String: rdate
- javax.swing.JPanel: jPanel
- javax.swing.JScrollPane: jScrollPane2
- javax.swing.JButton: receipt
- javax.swing.JButton: back1
- javax.swing.JButton: generatereceipt1
- javax.swing.JButton: jButton3

+ String: getRPrice()
+ void: setRPrice(String: rprice)
+ String: getRCustomerName()
+ void: setRCustomerName(String: rcustomername)
+ String: getRProductName()
+ void: setRProductName(String: rproductname)
+ String: getRDate()
+ void: setRDate(String: rdate)
+ boolean: checkInputs()
- void: fillInput()
- void: UpdateActionPerformed(java.awt.event.ActionEvent: evt)
- void: insertActionPerformed(java.awt.event.ActionEvent: evt)
- void: DeleteActionPerformed(java.awt.event.ActionEvent: evt)
+ void: showTableData()
- void: backActionPerformed(java.awt.event.ActionEvent: evt)
- void: OrdersTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: filterKeyReleased(java.awt.event.KeyEvent: evt)
- void: generatereceiptActionPerformed(java.awt.event.ActionEvent: evt)
- void: printActionPerformed(java.awt.event.ActionEvent: evt)
- void: ProductNameMousePressed(java.awt.event.ActionEvent: evt)
- void: PriceMouseEntered(java.awt.event.MouseEvent: evt)
- void: PriceMouseExited(java.awt.event.MouseEvent: evt)
- void: CustomerNameMouseEntered(java.awt.event.MouseEvent: evt)
- void: CustomerNameMouseExited(java.awt.event.MouseEvent: evt)
- void: quantityMouseEntered(java.awt.event.MouseEvent: evt)
- void: quantityMouseExited(java.awt.event.MouseEvent: evt)
- void: getPriceActionPerformed(java.awt.event.ActionEvent: evt)
- void: main(String: args[])


Employee
- javax.swing.JTable: EmployeeTable
- javax.swing.JComboBox<String>: Status
- javax.swing.JTextField: filter
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton3
- javax.swing.JButton: jButton4
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JLabel: jLabel6
- javax.swing.JPanel: jPanel1
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JTextField: name
- javax.swing.JTextField: password
- javax.swing.JTextField: srno
- javax.swing.JTextField: username
+ void: showTableData()
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton4ActionPerformed(java.awt.event.ActionEvent: evt)
- void: EmployeeTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: filterKeyReleased(java.awt.event.KeyEvent: evt)
- void: nameMouseEntered(java.awt.event.MouseEvent: evt)
- void: nameMouseExited(java.awt.event.MouseEvent: evt)
- void: usernameMouseExited(java.awt.event.MouseEvent: evt)
- void: usernameMouseEntered(java.awt.event.MouseEvent: evt)
- void: passwordMouseEntered(java.awt.event.MouseEvent: evt)
- void: passwordMouseExited(java.awt.event.MouseEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.MouseEvent: evt)
+ void: main(String: args[])
LifetimeOrders
+ javax.swing.JTextField CustomerName
+ com.toedter.calendar.JDateChooser OrderDate
javax.swing.JTable OrdersTable
+ javax.swing.JTextField Price
- javax.swing.JButton Print
+ javax.swing.JComboBox<String> ProductName
- javax.swing.JTextArea Receipt
- javax.swing.JButton back
- javax.swing.JLabel date
- com.toedter.calendar.JDateChooser datechoose1
- com.toedter.calendar.JDateChooser datechoose2
- javax.swing.JTextField filter
- javax.swing.JButton generatereceipt
- javax.swing.JLabel jLabel1
- javax.swing.JLabel jLabel10
- javax.swing.JLabel jLabel11
- javax.swing.JLabel jLabel2
- javax.swing.JLabel jLabel3
- javax.swing.JLabel jLabel4
- javax.swing.JLabel jLabel5
- javax.swing.JLabel jLabel7
- javax.swing.JLabel jLabel8
- javax.swing.JLabel jLabel9
- javax.swing.JPanel jPanel1
- javax.swing.JPanel jPanel2
- javax.swing.JPanel jPanel3
- javax.swing.JScrollPane jScrollPane1
- javax.swing.JScrollPane jScrollPane3
- javax.swing.JButton reset
- javax.swing.JComboBox<String> select
- javax.swing.JButton sortDate
+ javax.swing.JTextField srno
+ void: fillInput()
- void: backActionPerformed(java.awt.event.ActionEvent: evt)
- void: productNameMousePressed(java.awt.event.MouseEvent: evt)
- void: showTableData()
- void: OrdersTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: filterKeyReleased(java.awt.event.KeyEvent: evt)
- void: PrintActionPerformed(java.awt.event.ActionEvent: evt)
- void: generatereceiptActionPerformed(java.awt.event.ActionEvent: evt)
- void: srnoKeyTyped(java.awt.event.KeyEvent: evt)
- void: CustomerNameKeyTyped(java.awt.event.KeyEvent: evt)
- void: PriceKeyTyped(java.awt.event.KeyEvent: evt)
- void: OrderDateKeyTyped(java.awt.event.KeyEvent: evt)
- void: sortDateActionPerformed(java.awt.event.ActionEvent: evt)
- void: resetActionPerformed(java.awt.event.ActionEvent: evt)
- void: main(String: args[])


Menu
- javax.swing.JTable: MenuTable
- javax.swing.JButton: delete
- javax.swing.JTextField: filter
- javax.swing.JButton: insert
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton8
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JPanel: jPanel1
- javax.swing.JPanel: jPanel2
- javax.swing.JPanel: jPanel3
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JScrollPane: jScrollPane2
- javax.swing.JTextArea: menuprint
- javax.swing.JTextField: name
- javax.swing.JTextField: price
- javax.swing.JTextField: quantity
- javax.swing.JButton: receipt
- javax.swing.JComboBox<String>: select
- javax.swing.JTextField: srno
- javax.swing.JButton: update
+ Connection: getConnection()
- void: insertActionPerformed(java.awt.event.ActionEvent: evt)
- void: updateActionPerformed(java.awt.event.ActionEvent: evt)
- void:deleteActionPerformed(java.awt.event.ActionEvent: evt)
+ void: showTableData()
- void: jButton8ActionPerformed(java.awt.event.ActionEvent: evt)
- void: MenuTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: FilterKeyReleased(java.awt.event.KeyEvent: evt)
- void: receiptActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])
SalesRecords
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton3
- javax.swing.JPanel: jPanel1
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton3ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])
Stock
- javax.swing.JTable: StockTable
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton3
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JPanel: jPanel2
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JTextField: name
- javax.swing.JTextField: price
- javax.swing.JTextField: quantity
- javax.swing.JTextField: srno
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: showTableData()
- void: OrdersTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton3ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])
Login
- javax.swing.JButton: exit
- javax.swing.JButton: login
- javax.swing.JButton: reset
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JPanel: jPanel1
- exitActionPerformed(java.awt.ActionEvent: evt)
- resetActionPerformed(java.awt.ActionEvent: evt)
- loginActionPerformed(java.awt.ActionEvent: evt)
- passwordKeyPressed(java.awt.ActionEvent: evt)
- usernameMouseEntered(java.awt.MouseEvent: evt)
- usernameMouseExited(java.awt.MouseEvent: evt)
- passwordMouseEntered(java.awt.MouseEvent: evt)
- passwordMouseExited(java.awt.MouseEvent: evt)
+ main(String: args[])

TotalSales

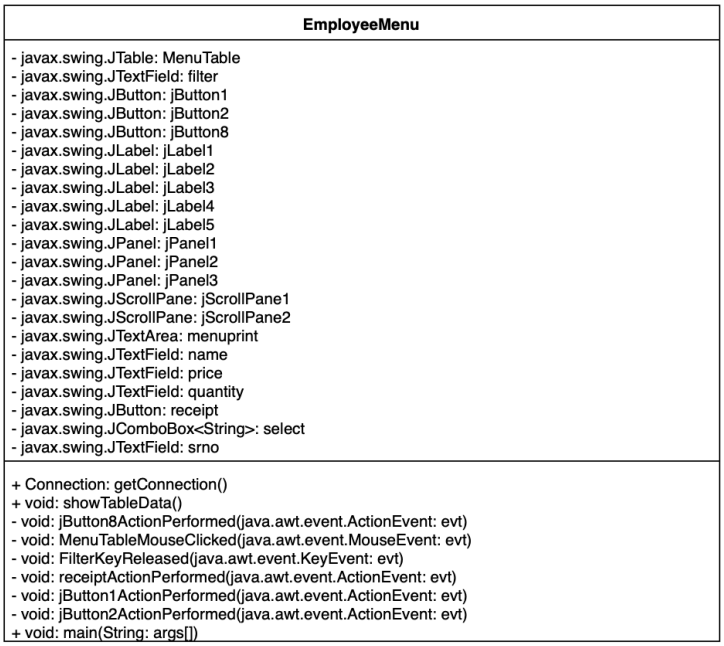
- javax.swing.JTable: OrdersTable
- javax.swing.JTextField: apr
- javax.swing.JTextField: aug
- javax.swing.JTextField: dec
- javax.swing.JTextField: feb
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton3
- javax.swing.JButton: jButton4
- javax.swing.JButton: jButton5
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel10
- javax.swing.JLabel: jLabel11
- javax.swing.JLabel: jLabel12
- javax.swing.JLabel: jLabel13
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JLabel: jLabel6
- javax.swing.JLabel: jLabel7
- javax.swing.JLabel: jLabel8
- javax.swing.JLabel: jLabel9
- javax.swing.JPanel: jPanel1
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JTextField: jTextField6
- javax.swing.JTextField: jan
- javax.swing.JTextField: july
- javax.swing.JTextField: jun
- javax.swing.JTextField: mar
- javax.swing.JTextField: may
- javax.swing.JButton: msales
- javax.swing.JTextField: nov
- javax.swing.JTextField: oct
- javax.swing.JTextField: profits1
- javax.swing.JTextField: sept
- javax.swing.JTextField: yearly

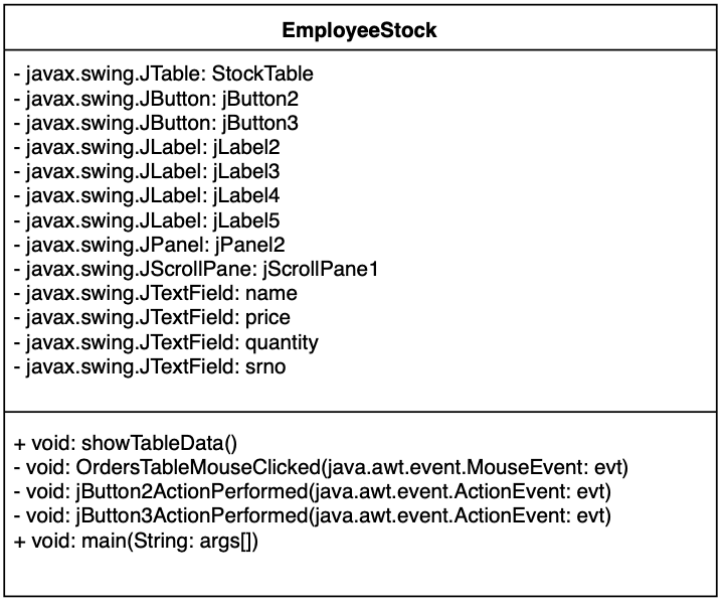
+ void: showTableData()
- void: msalesActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton3ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton4ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])

EmployeeEditingForm

+ javax.swing.JTextField: CustomerName
+ com.toedter.calendar.JDateChooser: DeliveryDate
- javax.swing.JTable: OrdersTable
+ javax.swing.JTextField: Price
+ javax.swing.JComboBox<String>: ProductName
- javax.swing.JButton: back
- javax.swing.JLabel: date
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JLabel: jLabel7
- javax.swing.JLabel: jLabel8
- javax.swing.JPanel: jPanel1
- javax.swing.JPanel: jPanel2
- javax.swing.JScrollPane: jScrollPane1
+ javax.swing.JTextField: quantity
- javax.swing.JComboBox<String>: select
+ javax.swing.JTextField: srno

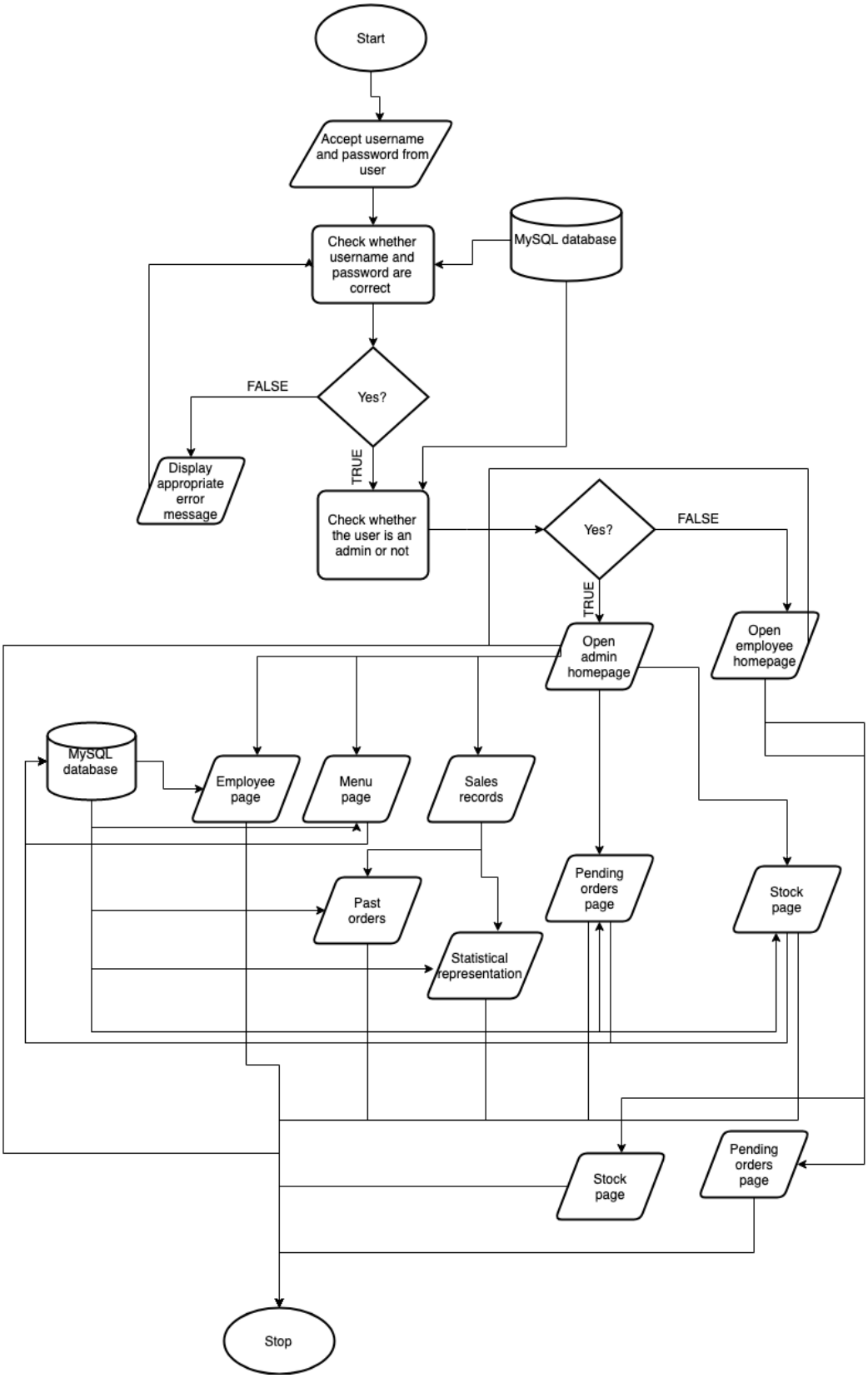
+ void: checkUser()
+ void: fillInput()
- void: CustomerNameMouseExited(java.awt.event.MouseEvent: evt)
- void: PriceMouseExited(java.awt.event.MouseEvent: evt)
- void: CustomerNameMouseEntered(java.awt.event.MouseEvent: evt)
- void: PriceMouseExited(java.awt.event.MouseEvent: evt)
- void: backActionPerformed(java.awt.event.ActionEvent: evt)
- void: OrdersTableMouseClicked(java.awt.event.MouseEvent: evt)


EmployeeMenu
- javax.swing.JTable: MenuTable
- javax.swing.JTextField: filter
- javax.swing.JButton: jButton1
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton8
- javax.swing.JLabel: jLabel1
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JPanel: jPanel1
- javax.swing.JPanel: jPanel2
- javax.swing.JPanel: jPanel3
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JScrollPane: jScrollPane2
- javax.swing.JTextArea: menuprint
- javax.swing.JTextField: name
- javax.swing.JTextField: price
- javax.swing.JTextField: quantity
- javax.swing.JButton: receipt
- javax.swing.JComboBox<String>: select
- javax.swing.JTextField: srno
+ Connection: getConnection()
+ void: showTableData()
- void: jButton8ActionPerformed(java.awt.event.ActionEvent: evt)
- void: MenuTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: FilterKeyReleased(java.awt.event.KeyEvent: evt)
- void: receiptActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton1ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])

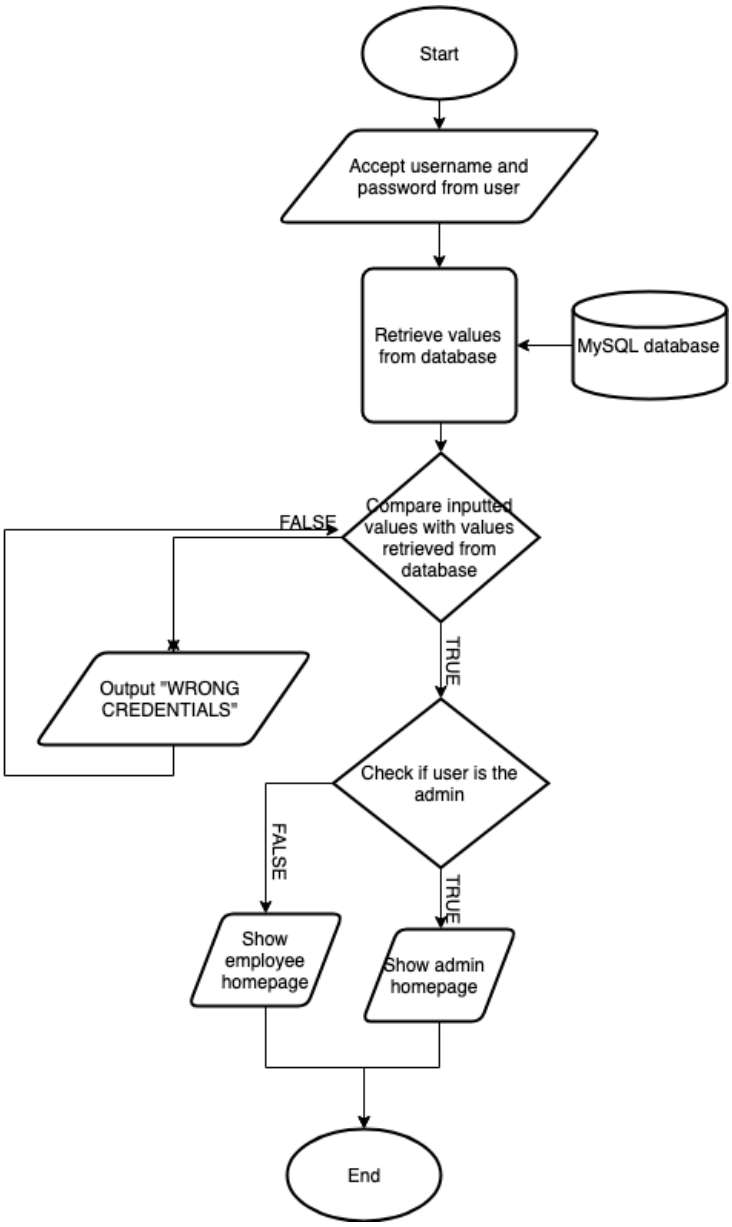

EmployeeStock
- javax.swing.JTable: StockTable
- javax.swing.JButton: jButton2
- javax.swing.JButton: jButton3
- javax.swing.JLabel: jLabel2
- javax.swing.JLabel: jLabel3
- javax.swing.JLabel: jLabel4
- javax.swing.JLabel: jLabel5
- javax.swing.JPanel: jPanel2
- javax.swing.JScrollPane: jScrollPane1
- javax.swing.JTextField: name
- javax.swing.JTextField: price
- javax.swing.JTextField: quantity
- javax.swing.JTextField: srno
+ void: showTableData()
- void: OrdersTableMouseClicked(java.awt.event.MouseEvent: evt)
- void: jButton2ActionPerformed(java.awt.event.ActionEvent: evt)
- void: jButton3ActionPerformed(java.awt.event.ActionEvent: evt)
+ void: main(String: args[])

Flowcharts depicting various processes

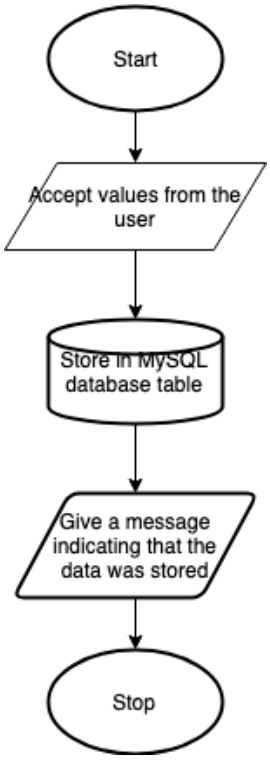
1. The entire product:



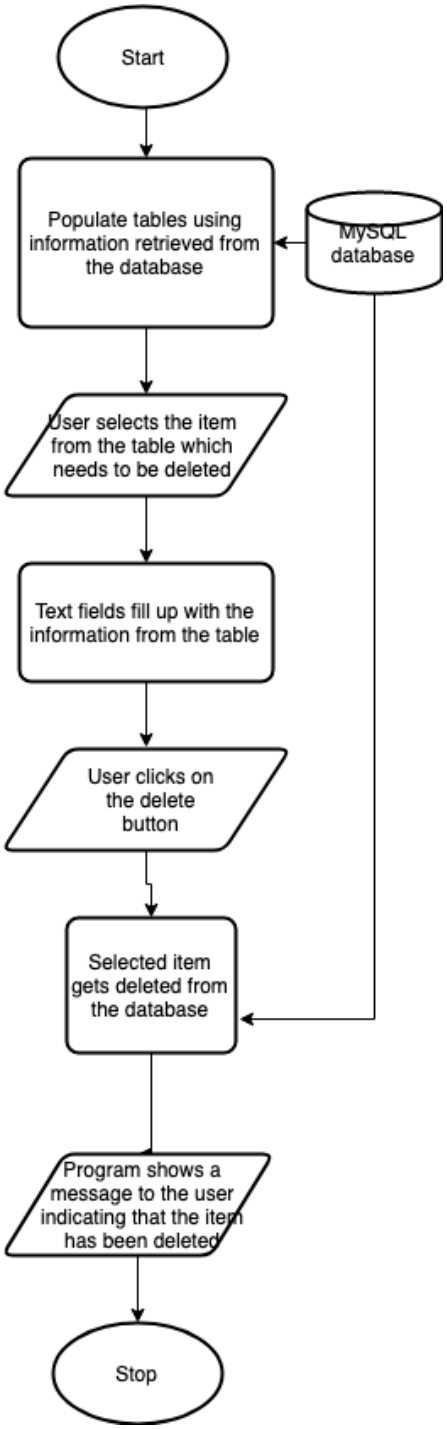
2. Login:


Start
Accept username and password from user
Retrieve values from database
MySQL database
Compare inputted values with values retrieved from database
FALSE
TRUE
Output "WRONG CREDENTIALS"
Check if user is the admin
FALSE
TRUE
Show employee homepage
Show admin homepage
End

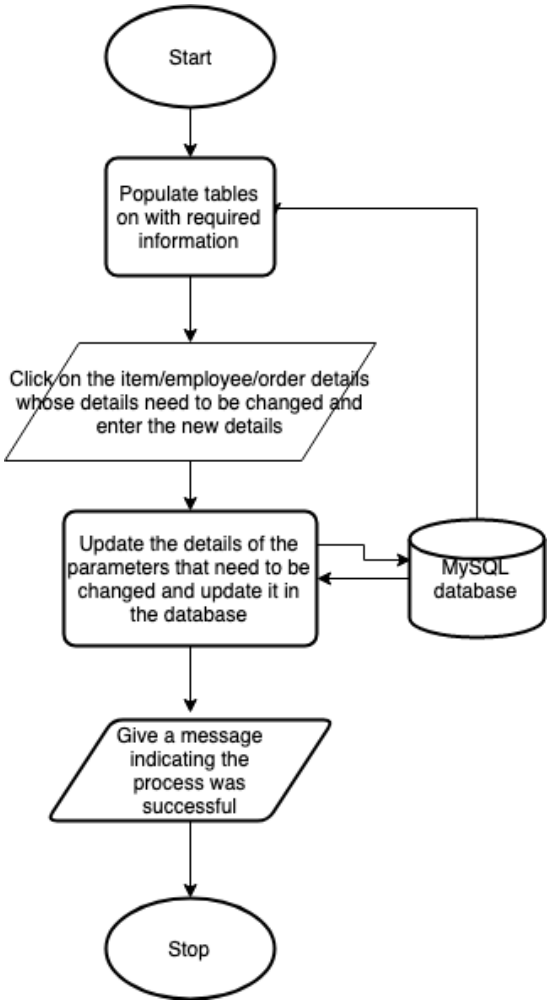
3. Adding a new employee/item in the menu/order/stock table:



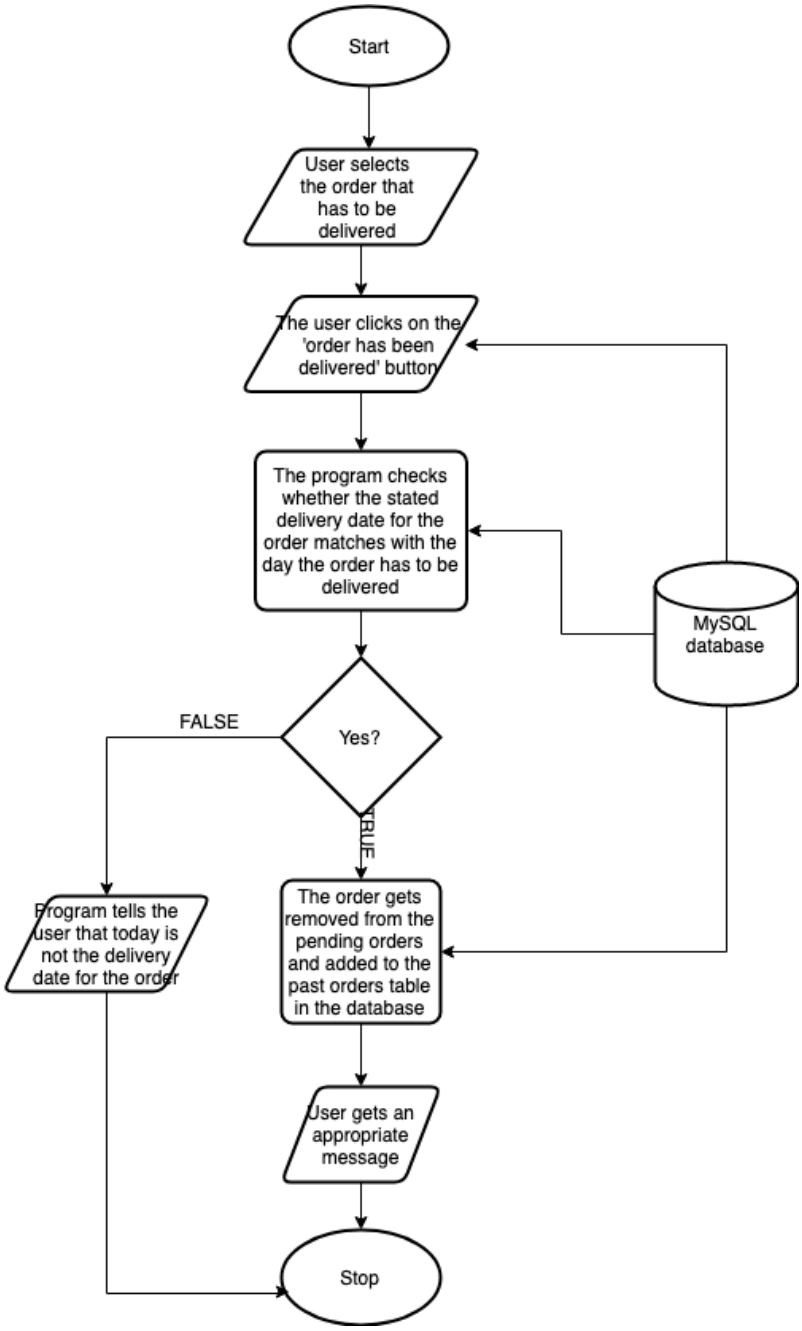
4. Updating details for an item/order in any database table:



5. Deleting items from the menu:



6. Order gets delivered



Functionality of each class:

1. **Login**: The class which accepts the username and password and directs the user according to his/her job description. For example, an admin would be redirected to the admin homepage and an employee would be redirected to the employee homepage.
2. **Admin**: The admin homepage, which consists of buttons which lead to various pages.
3. **Employees**: Shows the details of the different employees, including the admin. The class is connected to the database and consists of different fields: name, username, password and status. The class has a table which displays the information of the employees and can be searched using different parameters. It can also be sorted using different parameters. This class is only visible to the admin.
4. **Menu**: Shows the entire menu card. Includes the following fields: name, quantity and price. It has a table showing the information of the items in the menu and can be searched and sorted using different parameters. This class is visible to both the employee and admin, however only the admin has permission to edit anything.
5. **SalesRecords**: This class consists of buttons to call the LifetimeOrders and TotalSales (statistical representation) class. This class is only visible to the admin.
6. **LifetimeOrders**: This class shows the past orders in a table. It has the ability to generate a receipt for any order and the items can be searched or sorted using different parameters but no information can be edited. This class is only visible to the admin.
7. **TotalSales**: This class shows the monthly sales for a given year along with gross profits for the year. It also displays a graph for the given year. This class is only visible to the admin.
8. **EditingForm**: This is the main class of the product. Like the other classes, this is connected to the database and hence, the admin can add new orders, update details for existing orders and mark off orders as delivered. Like the previous classes, the orders are displayed in a table which can be sorted and searched using different parameters. This class is visible to both the employee and admin but only the admin has the ability to edit anything in the class.

9. **Stock1**: This class shows the different stock levels for various items that are required in the day-to-day operations of the business. It has buttons which allow the admin to add new items to the stock list or add new stock for items already present in the list. This only visible to the admin and the employee, but only the admin can add new stock.
10. **AddStock**: This class allows the admin to add new items in the stock list. The fields in this class are: name, quantity and price.
11. **ExistingStock**: This class allows the admin to add new stock for existing items.
12. **EmployeePage**: This class allows has the buttons to redirect the employee to the following pages: pending orders, menu and stock.
13. **EmployeeEditingForm**: The pending orders page for the employee where he/she is able to view all the pending orders.
14. **EmployeeStock**: The page where employees are able to view stock present in the database.
15. **EmployeeMenu**: The employee's menu page. Here, he/she is able to view the menu card.

Test plan

Sr. no/success criterion	**Type of testing**	**What to test**	**Input**	**Expected outcome**
1/a	Unit testing	Login page	Normal inputs- "admin" "employee1" Abnormal inputs- "klasgj"	The program should only accept correct credentials and redirect the user according to their role in business. If the user is the admin, he/she should see the admin homepage and if the user is an employee, he/she should see the employee homepage.
2/b	Unit testing	Buttons in admin homepage	N/A	The buttons should redirect the admin to the indicated page.
3/c(I-IV)	Functional testing	Adding values to the database.	Normal input (ex. Table: menu) - "Chocolate truffle cake" as the name - "2000" as the price Abnormal inputs:	The values should get added to the required table in the database. Any employee added should get added to the employee table, any item in the menu should get added to the menu page, and so on.

			- "1500te" as the price The user enters any item that is already present in the table.	
4/d(I-III)	Functional testing	Editing values already present in the database.	Normal input (ex. Table: menu) - "Chocolate truffle cake" as the name - "2000" as the price Abnormal inputs: - "1500te" as the price	The values that are edited should get reflected in the required database table. Any employee's information edited should get reflected in the employee table, any menu item's information edited should get reflected in the menu table, and so on.
5/e	Functional testing	Deleting an item from the menu table.	N/A	The selected item should get deleted from the required database table.
6/f	Functional testing	Keeping track of delivered orders.	N/A	Once an order gets delivered, the program should move the order from the pending orders table to the past orders table.
7/g-1	Functional testing	Searching the employee table using various parameters.	Normal inputs: Serial No: 1,2,3,4,5 Name: admin, employee1 Username: admin2, employee3 Status: working, left Abnormal inputs: Serial no: e, f, g, h Name: 123, thg, aDmin	The program should show the required results when the inputs return a result.
8/g-2	Functional testing	Searching the menu table using various parameters.	Normal inputs: Serial No: 1,2,3,4,5 Name: chocolate truffle cake Price: 1500 Abnormal inputs: Serial no: e, f, g, h Name: 123, thg, aDmin	The program should show the required results when the inputs return a result.
9/g-3	Functional testing	Searching the past orders table using	Normal inputs: Serial No: 1,2,3,4,5 Name: chocolate truffle cake	The program should show the required results when the inputs return a result.

		various parameters.	Price: 1500 Abnormal inputs: Serial no: e, f, g, h Name: 123, thg, aDmin	
10/g-3	Functional testing	Displaying the orders in the past orders table between two dates.	N/A	The program should display all the orders that were delivered between the two date ranges.
11/g-4	Functional testing	Searching the pending orders table using various parameters.	Normal inputs: Serial No: 1,2,3,4,5 Name: chocolate truffle cake Price: 1500 Abnormal inputs: Serial no: e, f, g, h Name: 123, thg, aDmin	The program should show the required results when the inputs return a result.
12/h	Functional testing	Graphing	N/A	The program should plot a graph of the monthly sales of the year that the user selects.
13/i	Functional testing	Printing	N/A	The program should be able to print out a menu card or the receipt for any order.
14/j	Functional testing	Low stock popup message	N/A	The program should display a popup to the admin upon login if the stock for any item is low.
15/k	Unit testing	Employee page	N/A	The employee should be able to view the following pages but not be able to edit any information: - Stock - Menu - Orders for the day