Test Suite Design

For

RESTAURANT AUTOMATION SYSTEM

By
Ankit Kumar Gupta
12CS10006
Gaurav Kumar
12CS10020

Table of Contents

1. Introduction:

The Primary goal of this software is to simulate the Restaurant automation system. This test plan consist of the test steps needed to be performed along with workflow how these test will be executed. We have excluded those test steps which are not applicable to this specific problem.

1.1 Purpose

The purpose of this test plan is to check all the information and variables governing the functioning of this software. Identify the functional requirements. Validate the Graphical User Interface. Validate all the random parameters generated by this system. Identify the System Requirements (JVM etc.).

1.2 scope

This test plan will consist of Unit testing Black box testing and White box testing, Integration testing and Function testing (to validate the functional requirements given in the SRS).

2. Target Test Items

Following are the different items to be tested:

2.1 Unit Testing

Unit testing consist of all the different units of this software. This includes all three classes (Main frame, manager and clerk) and the Graphical User Interface. Basically, only the methods inside different classes will be tested according to this test plan. Some of those functions are following.

1. Validate Login
2. Generate sales receipt and expenses data
3. Generate sales statics
4. View Menu and Update Menu
5. Generate purchase order
6. Enter sales details
7. Issue Ingredients

2.2 Integration Testing

There is only one module present in this system. Since different classes communicate with each other, it is necessary to test all classes after integration with respect to a common interface. So after checking whether all units are bugs free. Integrate them and test whether they after Integrating works as expected or not.

2.3 Function Testing

This approach tests whether given software fulfills the functional requirements of the SRS. Following are the functions we have for testing:

2.4 User Interface Testing

There can be two approaches for this test. First, we can test the GUI manually by clicking all buttons and validating output. Secondly, we can write an automated script which can be run to validate all the GUI components. There are many software available for this purpose e.g. Abbot. But we will not adopt automated testing for this problem.

3. Testing Approach

Testing approach defines the required methodology for carrying out different tests we described above.

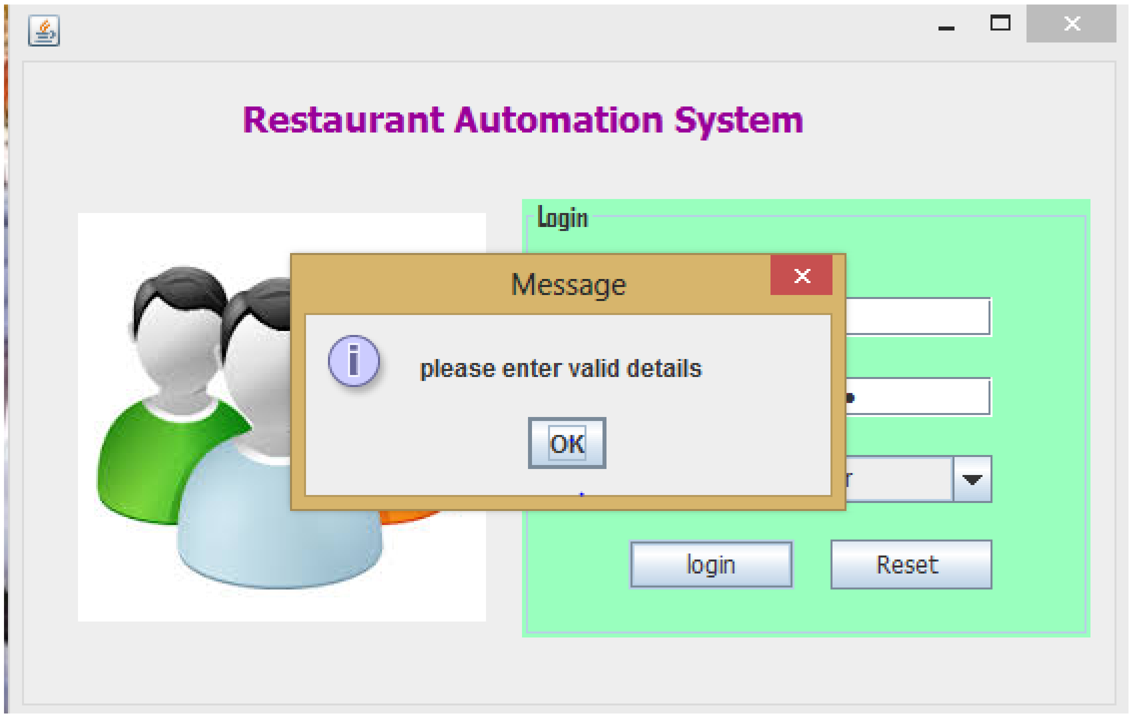
3.1 Black Box Testing

We will do Boundary Value Analysis on the all the significant variables or parameter of a method.

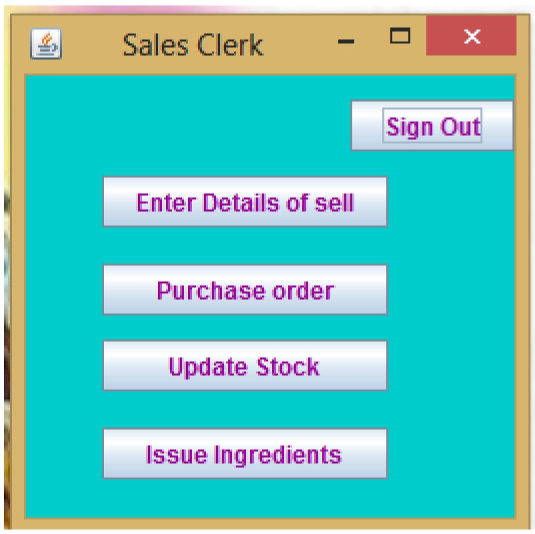
3.1.1 BBT for Login Case

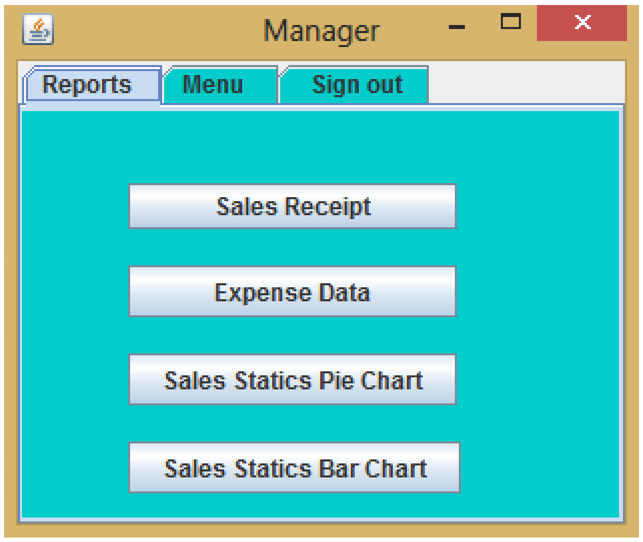
Equivalence Class	Input		Expected Output
	User Name	Password	
Invalid Input	Abcde	ececvs	Pop up showing error message
	manager	afegthg	Pop up showing error message
	Sgfyg54	clerk	Pop up showing error message
Valid Input	manager	manager	Manger frame
	clerk	clerk	Clerk frame

Actual Output:



please enter valid details





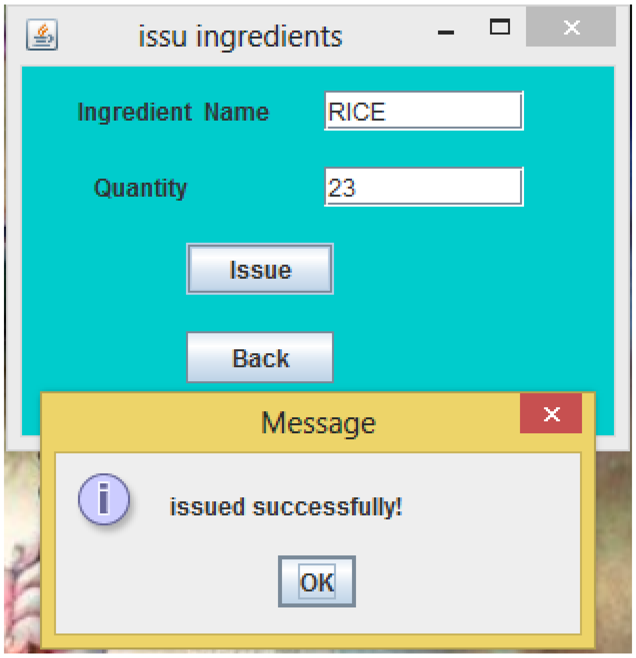
3.1.2 BBT for Issue Ingredients

Equivalence Class	Input		Expected Output
	Ingredient name	Quantity	
Invalid Ingredient	abcdef	43	Pop up showing error message
Valid Ingredient (but not enough quantity)	PANEER	a > current quantity	Pop up showing error message
Valid Ingredient (enough quantity)	PANEER	a < current quantity	Ingredient Issue Successful

Actual Outputs:

It is shown in the following diagrams.

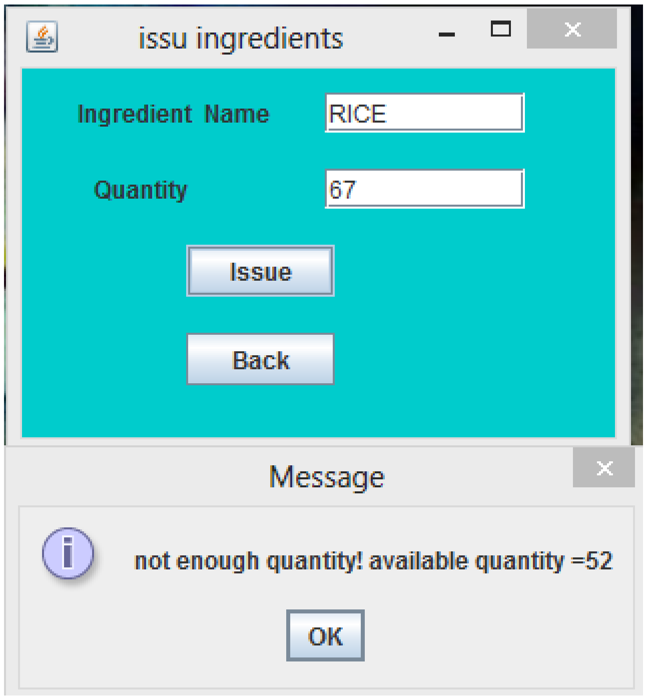
1st diagram corresponds to Valid Ingredient (enough quantity), 2nd Valid Ingredient (enough quantity) and 3rd corresponds to Invalid Ingredient name.


issu ingredients

Ingredient Name: RICE

Quantity: 23

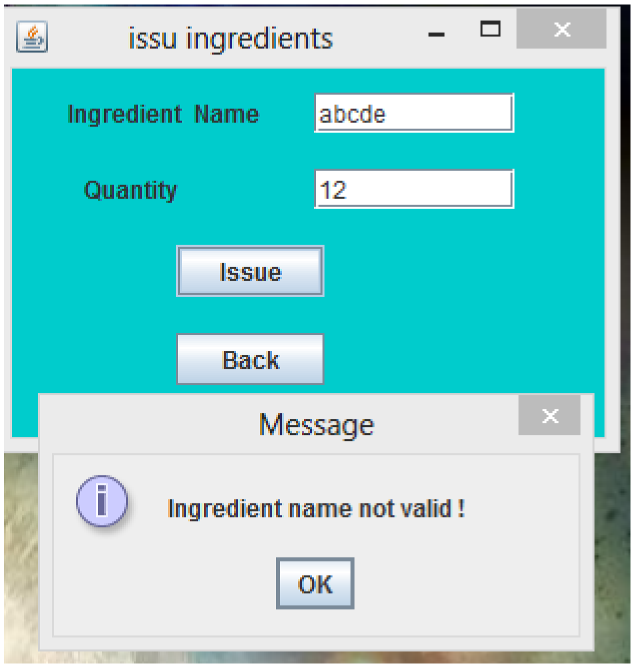
Message: issued successfully!


issu ingredients

Ingredient Name: RICE

Quantity: 67

Message: not enough quantity! available quantity =52


issu ingredients

Ingredient Name: abcde

Quantity: 12

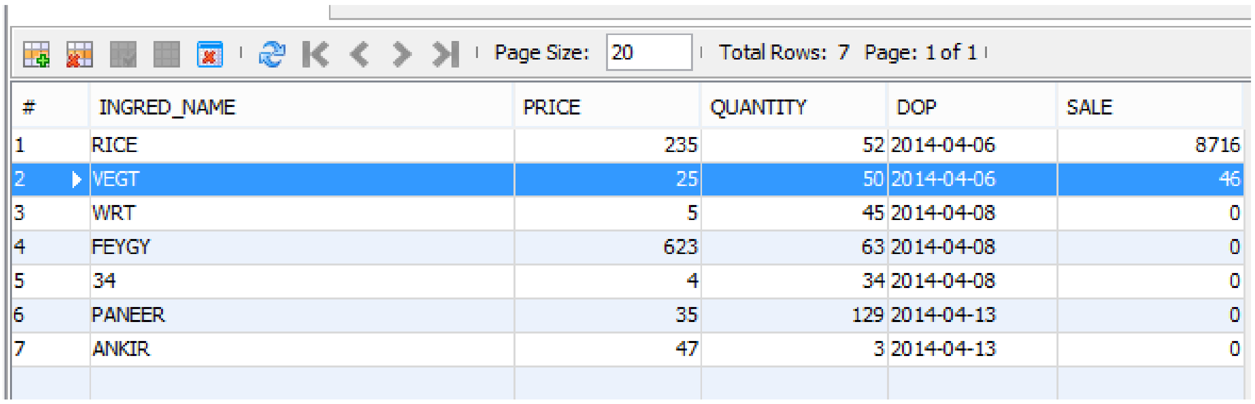
Message: Ingredient name not valid !

3.1.3 BBT for Purchase Order Generation

Check whether correct purchase orders are getting generated or not. I.e. check whether all Ingredients having stock less than threshold are there in purchase order generated by system.

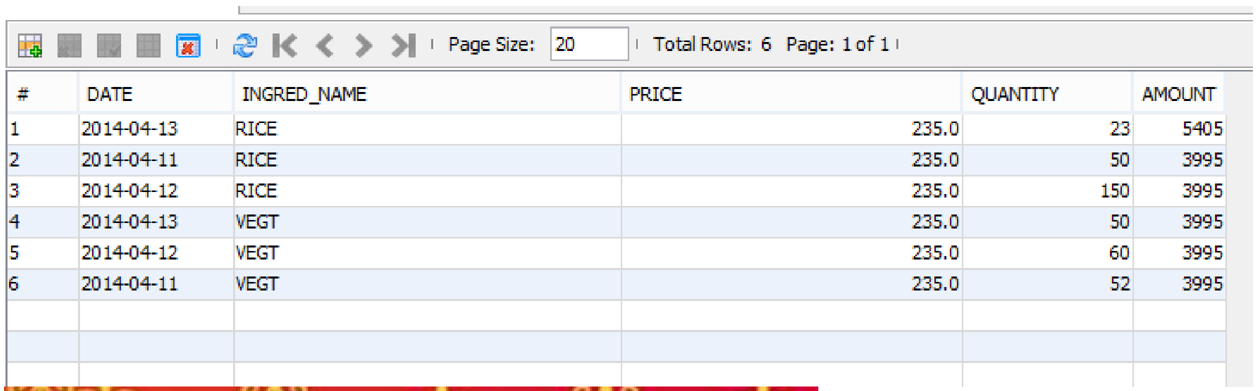
As you can see current stock for Rice and VEGT is less than threshold which is average of last 3 days consumption. Which according to following diagrams are 74, 54 respectively. So purchase order for RICE and VEGT is generated.

1st figure is of current stock, 2nd of consumption and 3rd is of purchase order generated by system.



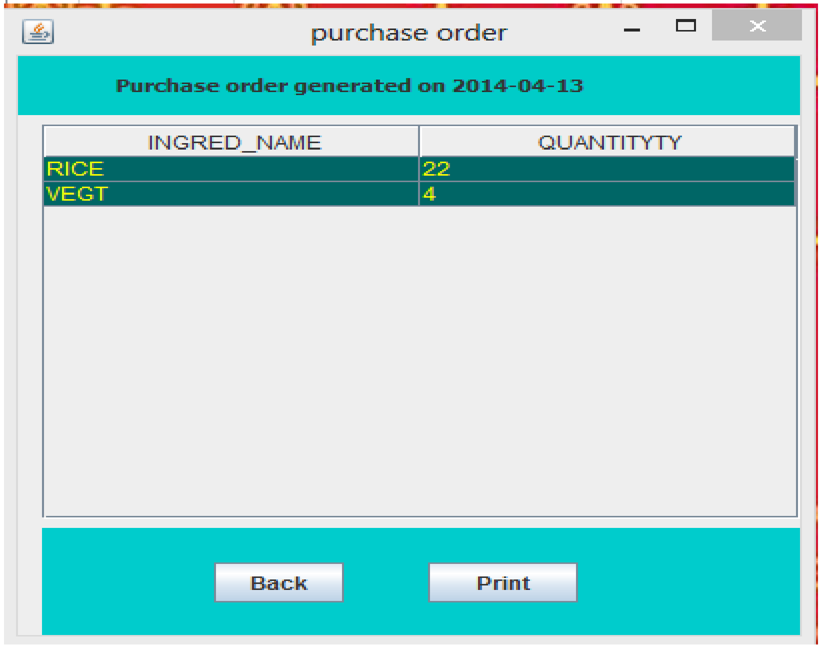
Page Size: 20 | Total Rows: 7 Page: 1 of 1

#	INGRED_NAME	PRICE	QUANTITY	DOP	SALE
1	RICE	235	52	2014-04-06	8716
2	VEGT	25	50	2014-04-06	46
3	WRT	5	45	2014-04-08	0
4	FEYGY	623	63	2014-04-08	0
5	34	4	34	2014-04-08	0
6	PANEER	35	129	2014-04-13	0
7	ANKIR	47	3	2014-04-13	0



Page Size: 20 | Total Rows: 6 Page: 1 of 1

#	DATE	INGRED_NAME	PRICE	QUANTITY	AMOUNT
1	2014-04-13	RICE	235.0	23	5405
2	2014-04-11	RICE	235.0	50	3995
3	2014-04-12	RICE	235.0	150	3995
4	2014-04-13	VEGT	235.0	50	3995
5	2014-04-12	VEGT	235.0	60	3995
6	2014-04-11	VEGT	235.0	52	3995



purchase order

Purchase order generated on 2014-04-13

INGRED_NAME	QUANTITYTY
RICE	22
VEGT	4

3.1.4 BBT for threshold calculation

Check whether threshold value generated by system is consistent with the manually calculated by taking average of last three day stock.

In testing of purchase order generation threshold value calculate manually is as follows:

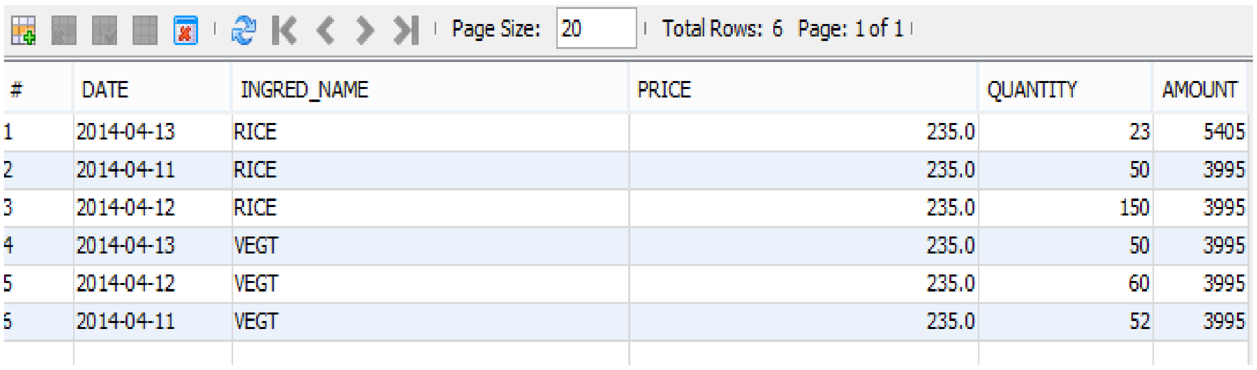
For Rice = (23+150+50)/3 = 74

For VEGT = (50+60+52)/3 = 54

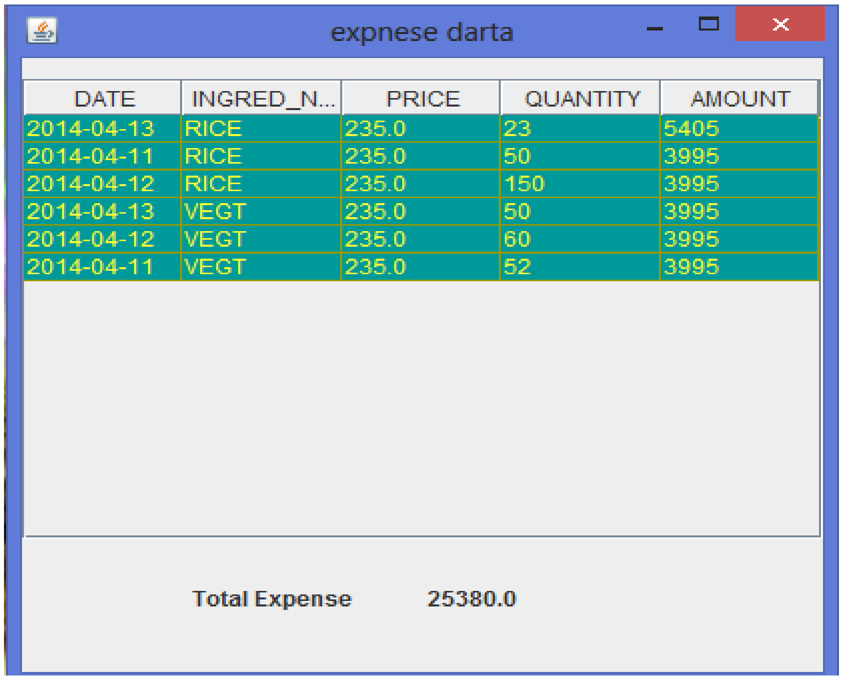
Which is consistent with as generated by system which can be verified by purchase order generated.

3.1.5 BBT for expenses data

Check whether money expenses on purchase of ingredients and other items are calculated correctly or not. This can be done by calculating all expenses manually and matching with the output of software, if both are same there is no bug in this part.



#	DATE	INGRED_NAME	PRICE	QUANTITY	AMOUNT
1	2014-04-13	RICE	235.0	23	5405
2	2014-04-11	RICE	235.0	50	3995
3	2014-04-12	RICE	235.0	150	3995
4	2014-04-13	VEGT	235.0	50	3995
5	2014-04-12	VEGT	235.0	60	3995
6	2014-04-11	VEGT	235.0	52	3995



DATE	INGRED_N...	PRICE	QUANTITY	AMOUNT
2014-04-13	RICE	235.0	23	5405
2014-04-11	RICE	235.0	50	3995
2014-04-12	RICE	235.0	150	3995
2014-04-13	VEGT	235.0	50	3995
2014-04-12	VEGT	235.0	60	3995
2014-04-11	VEGT	235.0	52	3995

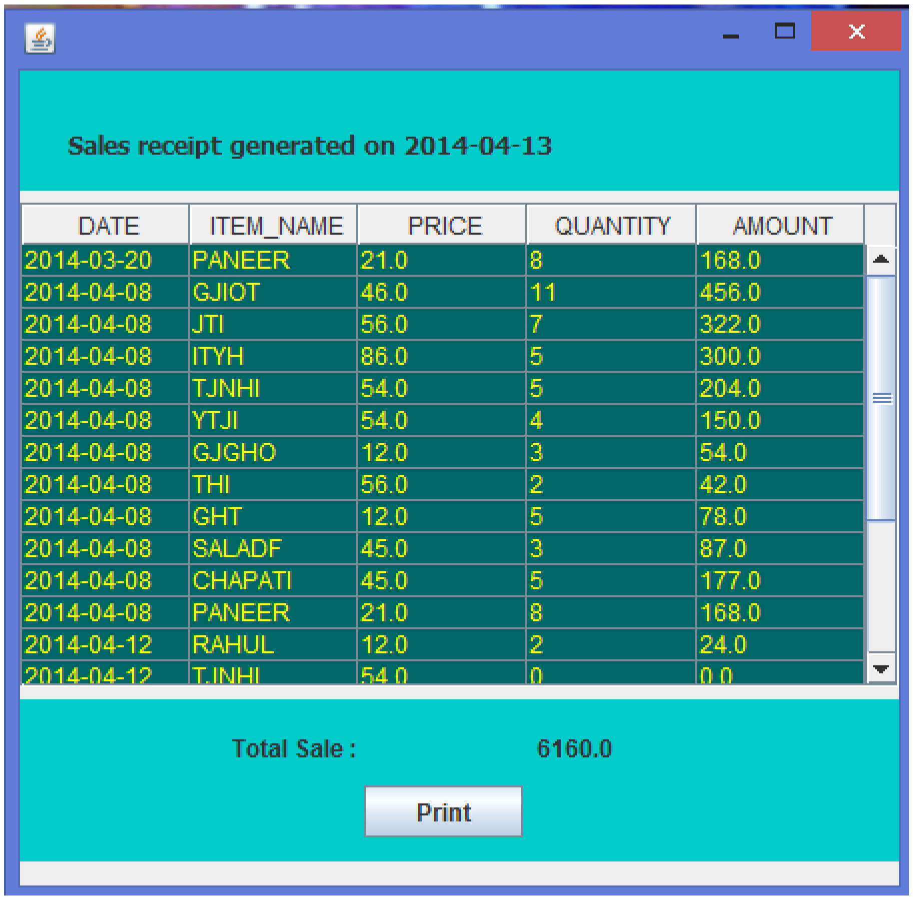
Total Expense 25380.0

3.1.6 BBT for sales receipt

It checks whether the generated sales receipts is correct or not. It checks weather the sales receipt contains correct information of quantity, money or not. And whether it prints the sales receipts or not.

As we can see in the following diagrams, it only generates sales which are less than a month old from sales receipt generation date and it is also able to print this sales receipt.

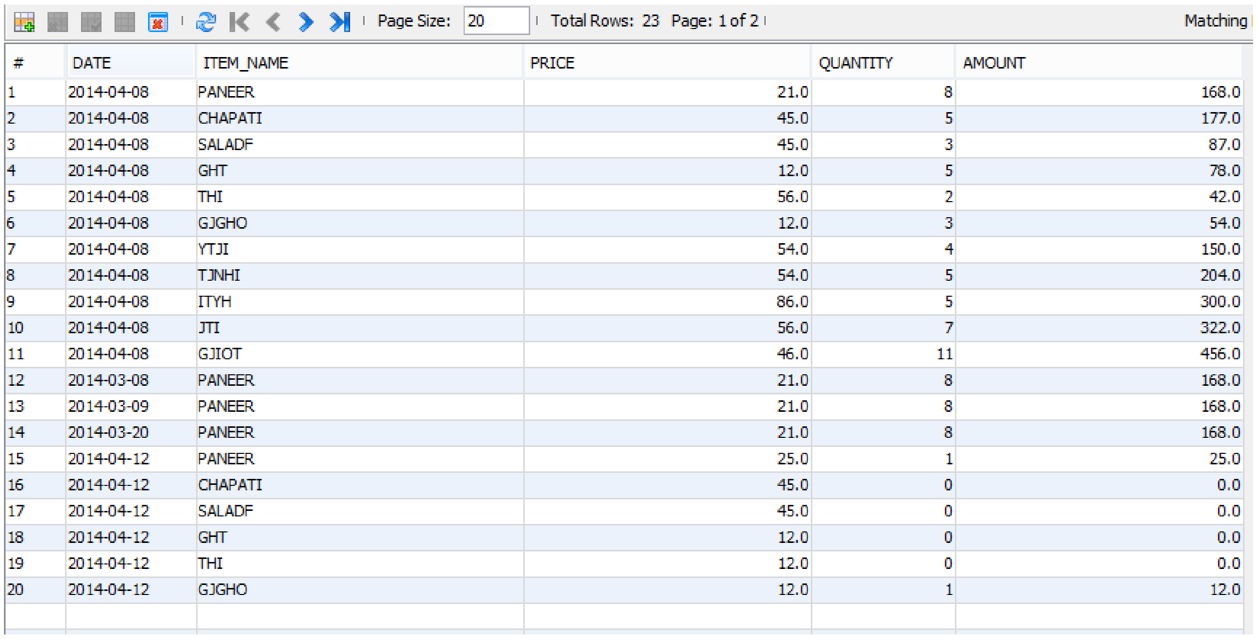
1st diagram is of sales receipt generated, 2nd of sale saved in database and 3rd is of sale receipt generated.



Sales receipt generated on 2014-04-13

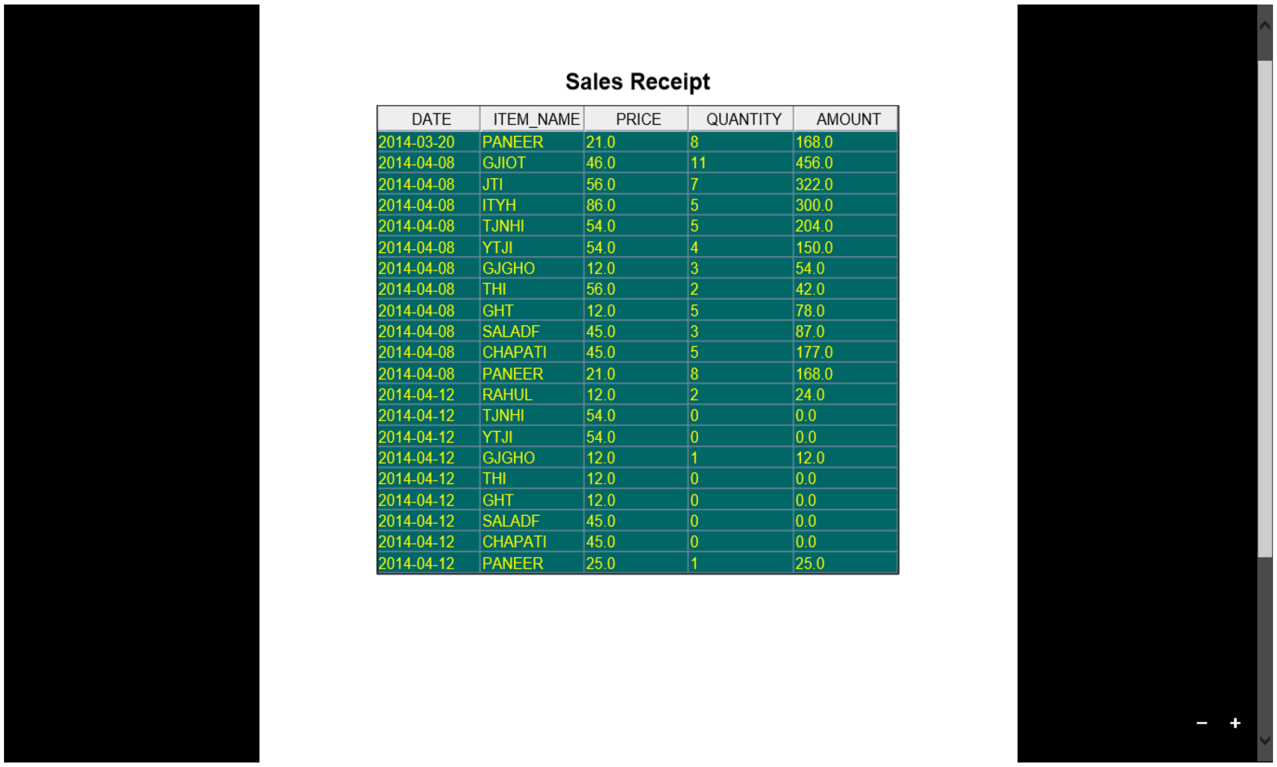
DATE	ITEM_NAME	PRICE	QUANTITY	AMOUNT
2014-03-20	PANEER	21.0	8	168.0
2014-04-08	GJIOT	46.0	11	456.0
2014-04-08	JTI	56.0	7	322.0
2014-04-08	ITYH	86.0	5	300.0
2014-04-08	TJNHI	54.0	5	204.0
2014-04-08	YTJI	54.0	4	150.0
2014-04-08	GJGHO	12.0	3	54.0
2014-04-08	THI	56.0	2	42.0
2014-04-08	GHT	12.0	5	78.0
2014-04-08	SALADF	45.0	3	87.0
2014-04-08	CHAPATI	45.0	5	177.0
2014-04-08	PANEER	21.0	8	168.0
2014-04-12	RAHUL	12.0	2	24.0
2014-04-12	TJNHI	54.0	0	0.0

Total Sale : 6160.0



Page Size: 20 Total Rows: 23 Page: 1 of 2 Matching

#	DATE	ITEM_NAME	PRICE	QUANTITY	AMOUNT
1	2014-04-08	PANEER	21.0	8	168.0
2	2014-04-08	CHAPATI	45.0	5	177.0
3	2014-04-08	SALADF	45.0	3	87.0
4	2014-04-08	GHT	12.0	5	78.0
5	2014-04-08	THI	56.0	2	42.0
6	2014-04-08	GJGHO	12.0	3	54.0
7	2014-04-08	YTJI	54.0	4	150.0
8	2014-04-08	TJNHI	54.0	5	204.0
9	2014-04-08	ITYH	86.0	5	300.0
10	2014-04-08	JTI	56.0	7	322.0
11	2014-04-08	GJIOT	46.0	11	456.0
12	2014-03-08	PANEER	21.0	8	168.0
13	2014-03-09	PANEER	21.0	8	168.0
14	2014-03-20	PANEER	21.0	8	168.0
15	2014-04-12	PANEER	25.0	1	25.0
16	2014-04-12	CHAPATI	45.0	0	0.0
17	2014-04-12	SALADF	45.0	0	0.0
18	2014-04-12	GHT	12.0	0	0.0
19	2014-04-12	THI	12.0	0	0.0
20	2014-04-12	GJGHO	12.0	1	12.0

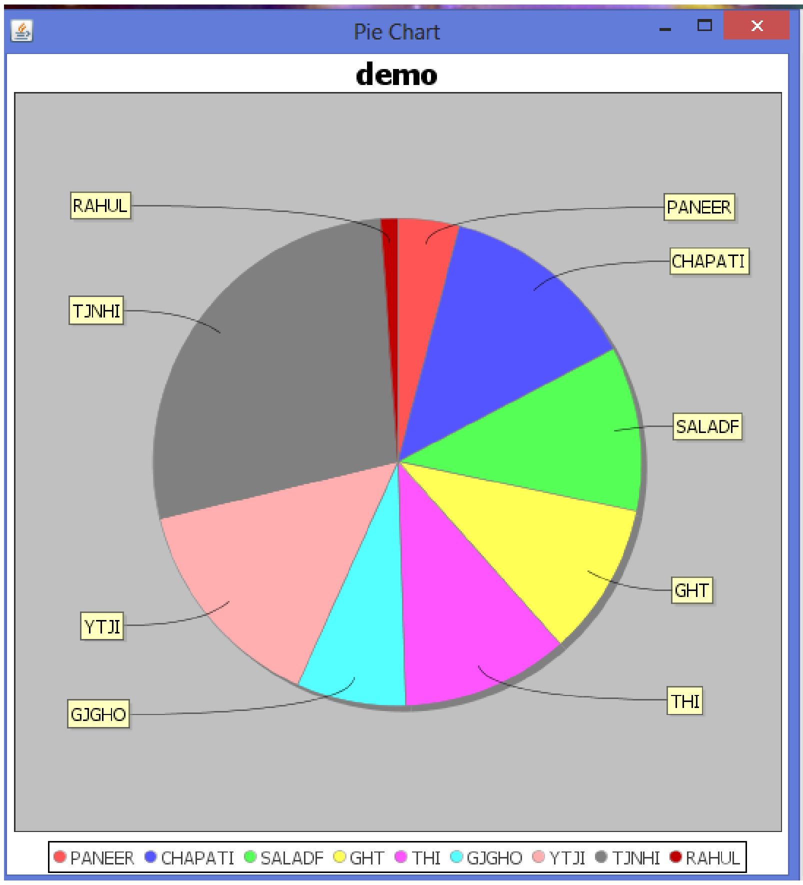


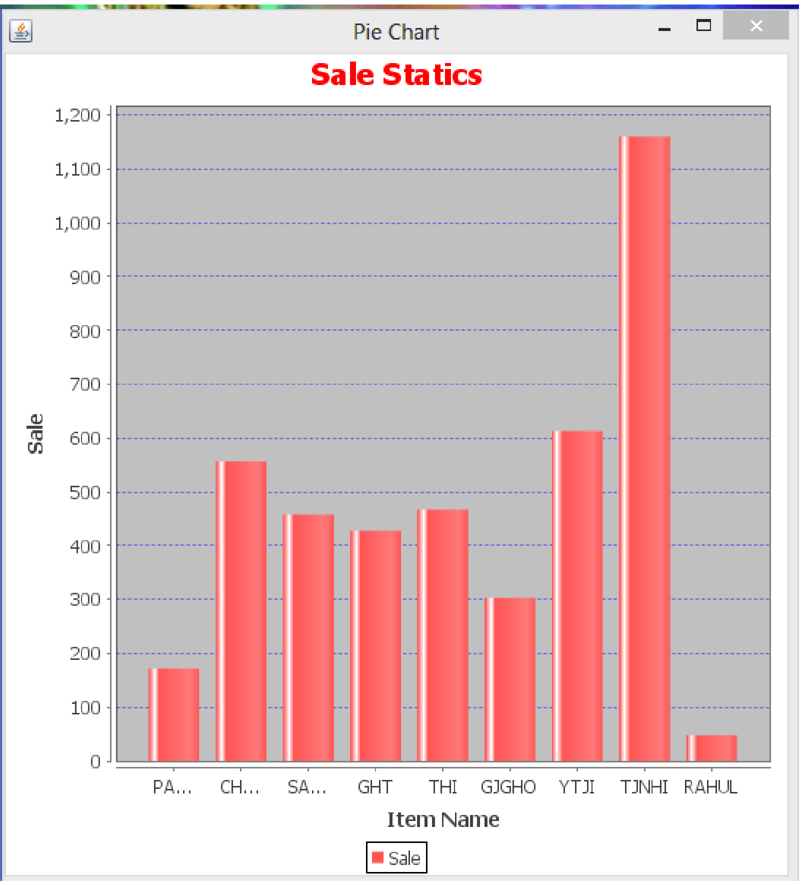
Sales Receipt

DATE	ITEM_NAME	PRICE	QUANTITY	AMOUNT
2014-03-20	PANEER	21.0	8	168.0
2014-04-08	GJIOT	46.0	11	456.0
2014-04-08	JTI	56.0	7	322.0
2014-04-08	ITYH	86.0	5	300.0
2014-04-08	TJNHI	54.0	5	204.0
2014-04-08	YTJI	54.0	4	150.0
2014-04-08	GJGHO	12.0	3	54.0
2014-04-08	THI	56.0	2	42.0
2014-04-08	GHT	12.0	5	78.0
2014-04-08	SALADF	45.0	3	87.0
2014-04-08	CHAPATI	45.0	5	177.0
2014-04-08	PANEER	21.0	8	168.0
2014-04-12	RAHUL	12.0	2	24.0
2014-04-12	TJNHI	54.0	0	0.0
2014-04-12	YTJI	54.0	0	0.0
2014-04-12	GJGHO	12.0	1	12.0
2014-04-12	THI	12.0	0	0.0
2014-04-12	GHT	12.0	0	0.0
2014-04-12	SALADF	45.0	0	0.0
2014-04-12	CHAPATI	45.0	0	0.0
2014-04-12	PANEER	25.0	1	25.0

3.1.7 BBT for statistical report

Statistical report contains statistics of every item sold. It checks whether the statistical report is consistent with actual data.

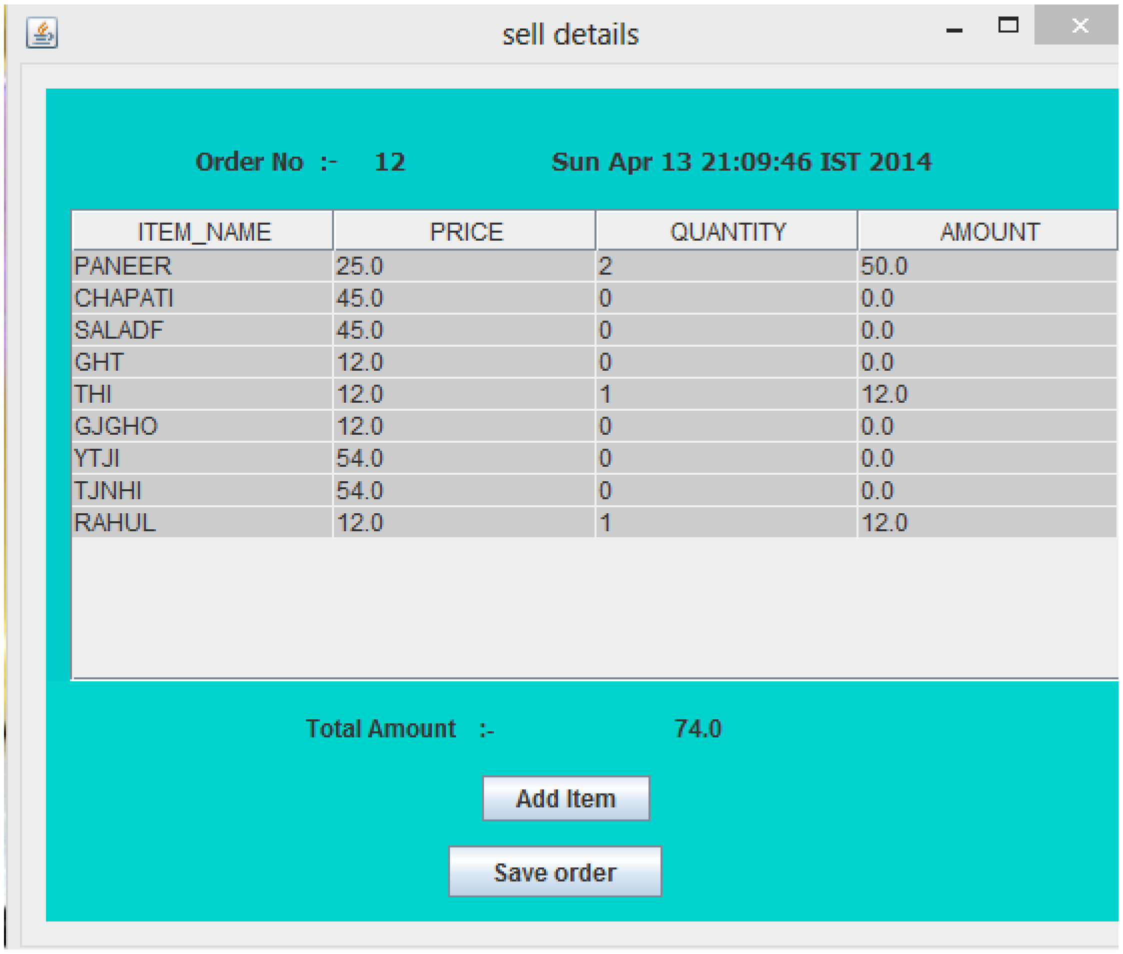




3.1.8 BBT for generate bills

Check whether bill contains all the ordered items and total cost of items are consistent with the actual cost.

As you can see all things are correct.



Order No :- 12 Sun Apr 13 21:09:46 IST 2014

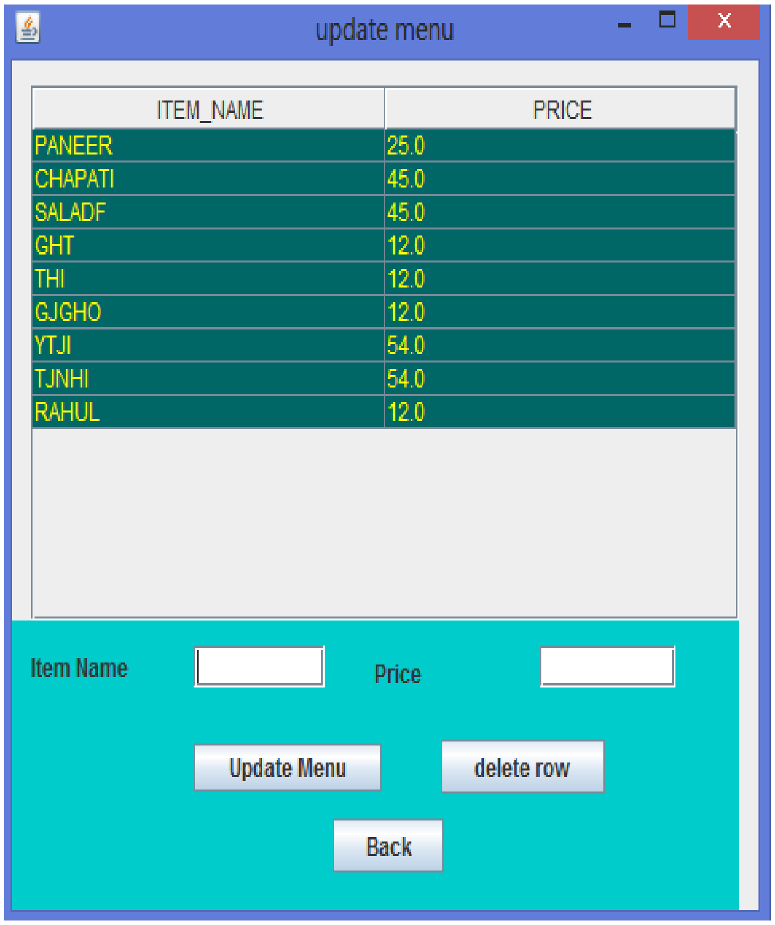
ITEM_NAME	PRICE	QUANTITY	AMOUNT
PANEER	25.0	2	50.0
CHAPATI	45.0	0	0.0
SALADF	45.0	0	0.0
GHT	12.0	0	0.0
THI	12.0	1	12.0
GJGHO	12.0	0	0.0
YTJI	54.0	0	0.0
TJNHI	54.0	0	0.0
RAHUL	12.0	1	12.0

Total Amount :- 74.0

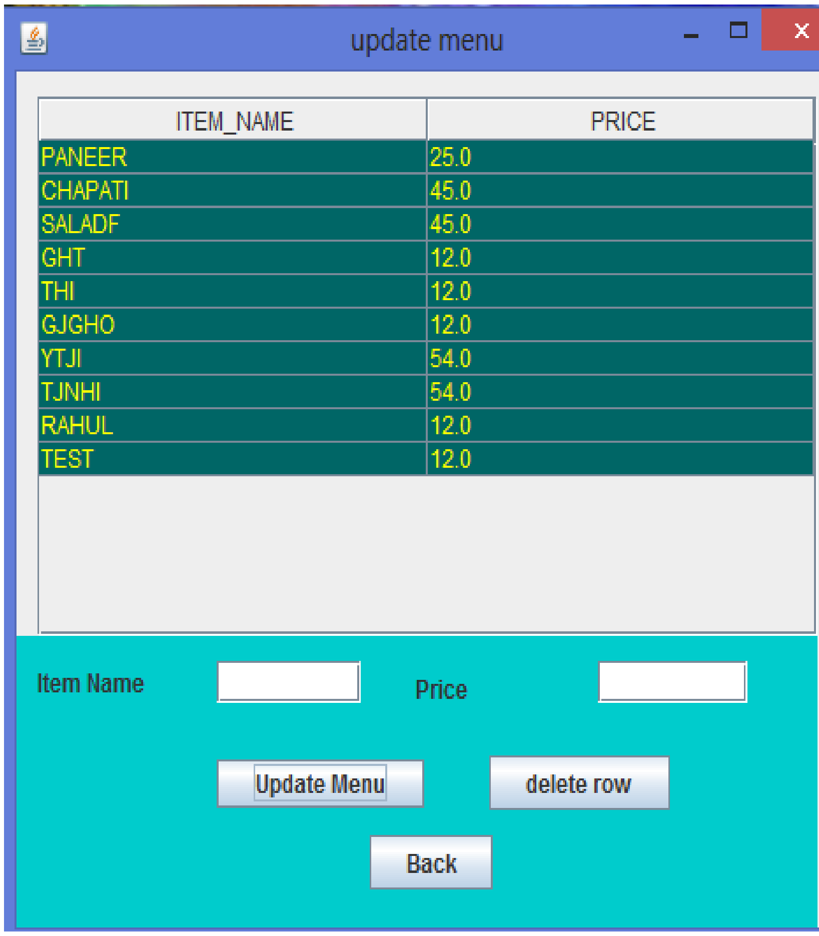
3.1.9 BBT for update menu

Whenever a new item is added the menu should update and should contains the new item. It checks weather the new items is added to the menu list correctly or not.

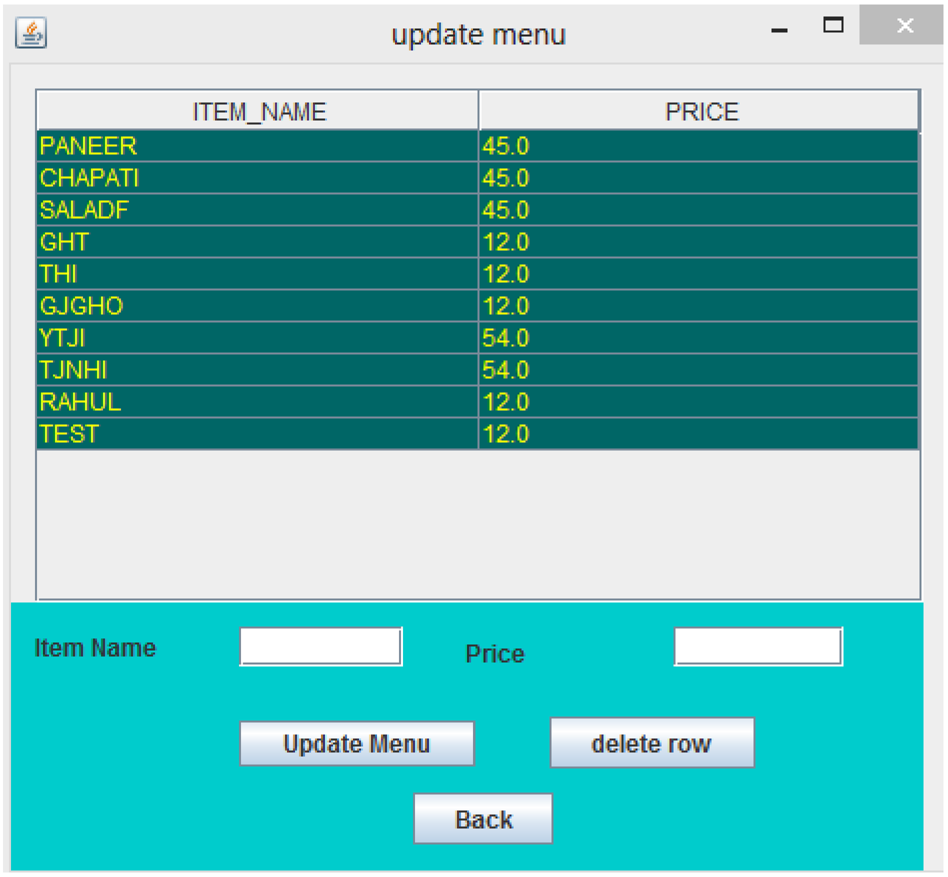
As we can see Test Item is added to menu. And for price change price of PANEER is changed from 25 to 45 after update.



ITEM_NAME	PRICE
PANEER	25.0
CHAPATI	45.0
SALADF	45.0
GHT	12.0
THI	12.0
GJGHO	12.0
YTJI	54.0
TJNHI	54.0
RAHUL	12.0



ITEM_NAME	PRICE
PANEER	25.0
CHAPATI	45.0
SALADF	45.0
GHT	12.0
THI	12.0
GJGHO	12.0
YTJI	54.0
TJNHI	54.0
RAHUL	12.0
TEST	12.0

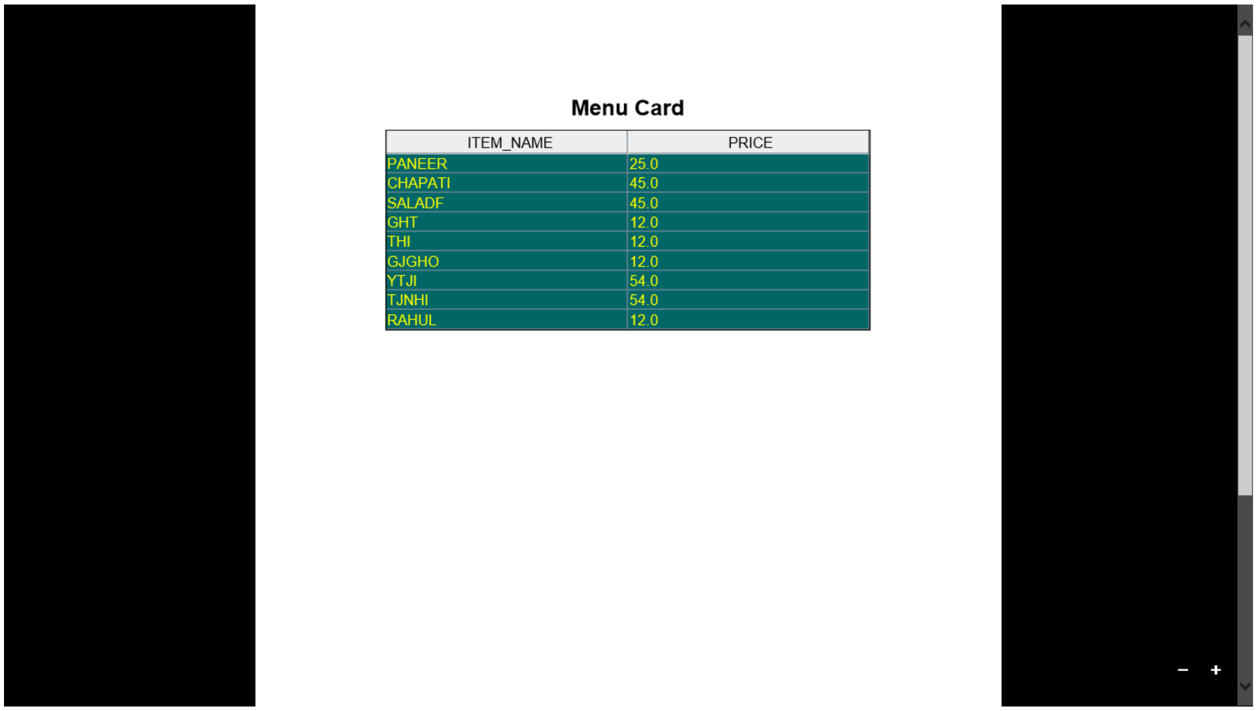


ITEM_NAME	PRICE
PANEER	45.0
CHAPATI	45.0
SALADF	45.0
GHT	12.0
THI	12.0
GJGHO	12.0
YTJI	54.0
TJNHI	54.0
RAHUL	12.0
TEST	12.0

3.1.10 BBT for print menu

It prints the menu item.so in this section we check whether the It contains all the menu items or some items are missing. If all items are there then there I no bug.

As shown in the following it correctly prints the menu.



Menu Card

ITEM_NAME	PRICE
PANEER	25.0
CHAPATI	45.0
SALADF	45.0
GHT	12.0
THI	12.0
GJGHO	12.0
YTJI	54.0
TJNHI	54.0
RAHUL	12.0

3.2 White Box Testing

3.2.1 WBT for Login Case

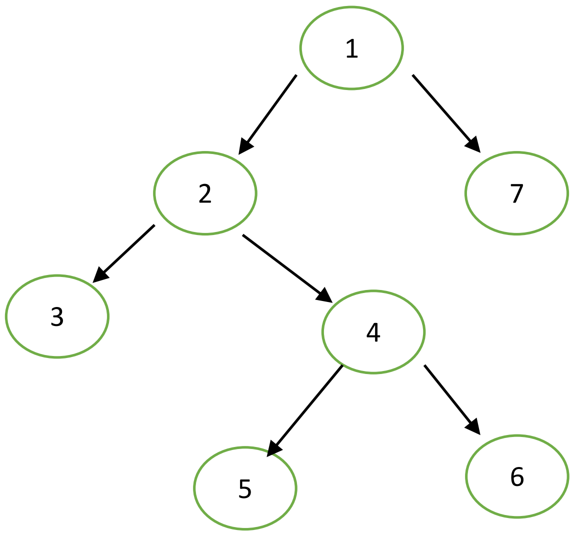
Check whether all loops are executed or not.

Test ID		Tester Name	Ankit Gupta	Test Date	13/04/2014
Class Name	Login	Method Name	loginActionPerformed	File Name	Login.java

Code Segment

```
 1.  if(!us.equals("") && !ps.equals(""))
      {
  2.    if(us.equals("manager") && ps.equals("manager") && type.getSelectedIndex()==0)
       {
   3.      new Manager().setVisible(true);
          user.setText("");
          password.setText("");
          type.setSelectedIndex(0);
          dispose();
       }
 else
       {
  4.      if(us.equals("clerk") && ps.equals("clerk") && type.getSelectedIndex()==1)

    5.    {  new Clerk().setVisible(true);
           user.setText("");
           password.setText("");
           type.setSelectedIndex(0);
           dispose();
         }

   Else
 {
   6.      JOptionPane.showMessageDialog(rootPane,” please enter valid details");
           user.setText("");
           password.setText("");
           type.setSelectedIndex(0);

       }
    }
    else
    {
7.      JOptionPane.showMessageDialog(rootPane, “please fill in details");

    }
```

Path Diagram



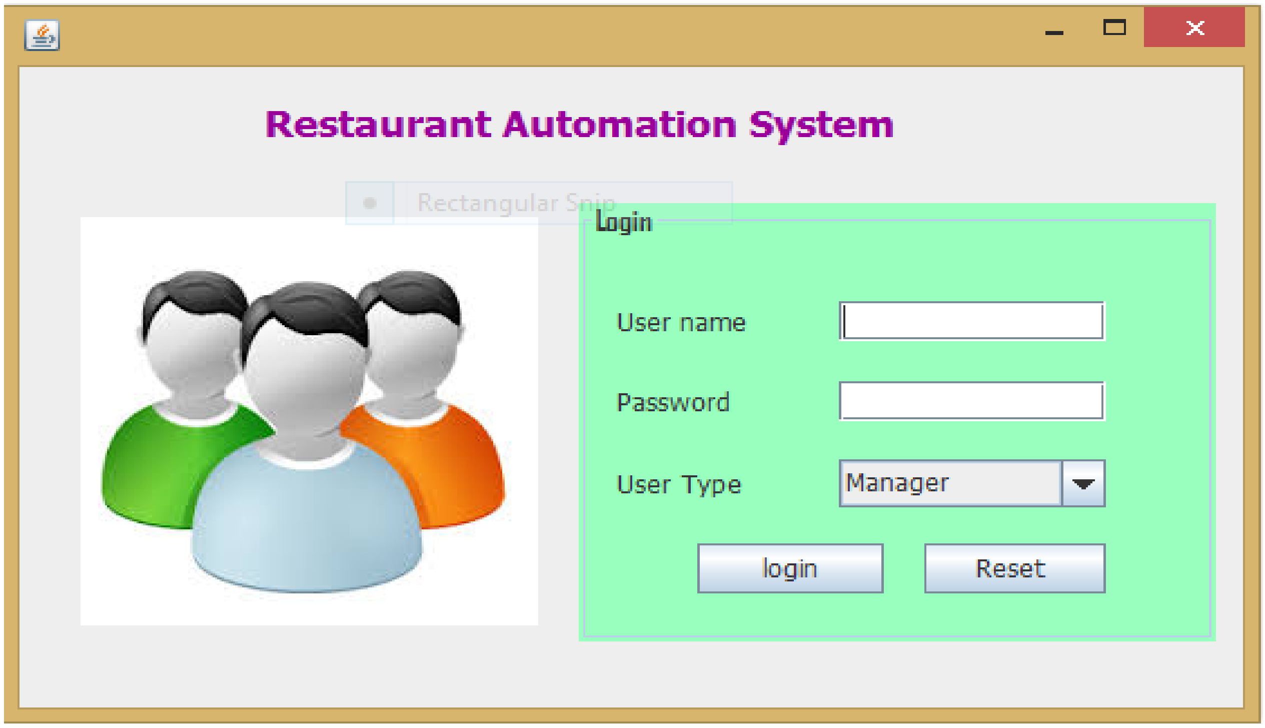
Path 1	**1-> 7**
Path 2	**1-> 2-> 3**
Path 3	**1-> 2-> 4-> 5**
Path 4	**1-> 2-> 4-> 6**

Path	**Input**		**Expected output**
	User Name	**Password**	
Path 1	blank	blank	Error Message
Path 2	manager	manager	Manager frame
Path 3	clerk	clerk	Clerk frame
Path 4	abcdef	abcdeg	Error Message

3.3 Integration Testing

This testing is done when software is combined and it is tested that all components, which are combined, are interacting with each other properly. Each module/component that is being integrated into the software is tested by integration testing. Unit testing is a pre-requisite for integration testing. One thing that we will incorporate in this testing is Synchronization. It is only issue we have to take care while combining different classes/Threads. We will deal with Synchronization separately in the next segment of test plan.

3.4 User Interface Testing



To test the User Interface, each function of the GUI must be tested either manually or by automatic techniques. We will describe here only the manual techniques.

3.4.1 Main Window Testing

What is tested?	All those components which are interacting with the user (Jtest fields ,Jbuttons)
Inputs	User fills login details and clicks login button
Expected outputs	On invalid inputs error message and on valid inputs user in redirected to his account.
Actual Output	Output matches with the expected output.

3.4.2 Manager Frame Testing

What is tested?	All those components which are interacting with the user (Jtest fields , Jbuttons)
Inputs	Manager clicks on different buttons.
Expected outputs	Corresponding frame is shown.
Actual Output	Output matches with the expected output.

3.4.3 Clerk Frame Testing

What is tested?	All those components which are interacting with the user (Jtest fields ,Jbuttons)
Inputs	User fills login details and clicks login button

Expected outputs	on invalid inputs error message and on valid inputs user in redirected to his account.
Actual Output	output matches with the expected output.

3.5 Synchronization Testing using Automated Testing

Synchronization is the biggest issue in the design of this software. There are different approaches for automated testing. To test the synchronization, we can Use a specific technique in which we add some code of our own. This piece of code adds an extra thread in the system which actually keeps track of different synchronization variables. NOTE: Here, we are NOT modifying any source code.
There are softwares available which can generate these scripts. But here, we will avoid this approach. Because it is not a big issue for our system RAS.

We can generalize the state of this system by Definite Finite Automata (DFA).Coming to the automated testing.