Automated Order Management System for … “City Mart”

T.M.Savindu Pasingtha NG/HNDCSE/24/11



**Assignment Cover Sheet**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | | **Module Number and Title** | |
| Higher National Diploma in Computing & Software Engineering | | Introduction to OOP- SEC4207 | |
| **Student Name & No.** | | **Assessor** | |
| T.M.Savindu Pasingtha NG/HNDCSE/24/11 | | Mr. Sanaka Perera | |
| **Hand out date** | | | **Submission Date** |
| 21/11/2020 | | | 17/10/2020 |
| **Assessment type**  Coursework | **Duration/Length of**  **Assessment Type**  Report and Demo | | **Weighting of Assessment**  100% |

|  |
| --- |
| **Learner declaration** |
| I am T.M.Savindu Pasingtha NG/HNDCSE/24/11, certify that the work submitted for this assignment is my own and research sources are fully acknowledged. |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Marks Awarded** | | | |  | | First assessor | |  | |  | | IV marks | |  | |  | | Agreed grade | |  | |  | | Signature of the assessor |  | Date |  |  | |

**Feedback Form**

**International College of Business & Technology**

**Module/Title:**

**Student:** T.M. Savindu Pasingtha

**Assessor:**

**Assignment:**

**Strong features of your work:**

**Areas for improvement:**

**Marks Awarded:**

# ACKNOLEDGMENT

In preparation of my assignment, I had to take the help and guidance of some respected persons, who deserve my deepest gratitude. As the completion of this assignment gave me much pleasure, I would like to show my gratitude **Mr. Sanaka Perera**, our lecturer, on ICBT Campus for giving me good guidelines for assignment throughout numerous consultations. I would also like to expand my gratitude to all those who have directly and indirectly guided me in making this assignment.

# ABSTRACT

This assignment comes under Object Oriented Programming skill for module at HNDCSE program. This assignment mainly focus, “CITY MART” wholesaler to develop a new Automated Order Management System. This system very help full to improve their business.

To completely this assignment firstly I discussed about what data is and what the java programming language object oriented concepts and it means explain with proper examples. So I clearly identify where to we can use OOP concept in this system project. In the second task I provide proper UML diagram for the above mentioned requirements. Its include User case diagram, Class diagram and Sequence diagram. Third task I develop the system for “CITY MART” based on the UML designs. Not only that I provide source code and interface screenshots with proper explanation.

In fourth task I mentioned proper testing technics, and after I tested this system. So below I included proper test plan and proper test cases. Finally I discussed what is the weakness and strength in this system. Not only that I included proper references and GANTT chats to this assignment.

# INTRODUCTION

This assignment mainly focused to create Automated Order Management System for “CITY MART” using Java programing language. So, object oriented concepts use to build this system. Not only has that how to UML diagrammed techniques using build proper system discuss in this assignment.

So I included this system to, multi user login system and bellow I mention main three **actors** activities in the system.

* **Manager** – Creating and managing user accounts, view stock details, view sales details, view suppliers, view request and approve requests.
* **Stock Keeper** – Create and manage items, view stock details and create purchase requests.
* **Cashier/Accountant** – Generate purchase order.

That all functions are properly work in the system. I used attractive user interfaces to implement this system. MYSQL Database provided the all backend database tasks in XAMPP server. There for any one can quickly data management task in the system such as logging, registering, purchases order and others.

I think this system can easy to use and improve their business every day.

# Table of Contents

[ACKNOLEDGMENT 4](#_Toc56807719)

[ABSTRACT 5](#_Toc56807720)

[INTRODUCTION 6](#_Toc56807721)

[TABLE OF FIGURE 10](#_Toc56807722)

[TASK 1 11](#_Toc56807723)

[What is the OOP Concept? 11](#_Toc56807724)

[Object 11](#_Toc56807725)

[Java data types Data types 11](#_Toc56807726)

[Access modifiers 11](#_Toc56807727)

[Return data type 12](#_Toc56807728)

[Methods or functions 12](#_Toc56807729)

[Class 13](#_Toc56807730)

[Constructor method 13](#_Toc56807731)

[Inheritance 13](#_Toc56807732)

[Polymorphism 14](#_Toc56807733)

[ Method Overloading 14](#_Toc56807734)

[ Method Overriding 14](#_Toc56807735)

[Casting 17](#_Toc56807736)

[Abstract 17](#_Toc56807737)

[Encapsulation 17](#_Toc56807738)

[“This” Key word 18](#_Toc56807739)

[“Super” Key word 18](#_Toc56807740)

[TASK 2 19](#_Toc56807741)

[User Case Diagram 19](#_Toc56807742)

[Class Diagram for City Mart Automation Order System 20](#_Toc56807743)

[TASK 3 21](#_Toc56807744)

[Home Page 21](#_Toc56807745)

[Cashier Page 21](#_Toc56807746)

[Manager page 22](#_Toc56807747)

[Store-Keeper page 22](#_Toc56807748)

[Login page 23](#_Toc56807749)

[User Account Registration Page 23](#_Toc56807750)

[Request-Order page 24](#_Toc56807751)

[Request View Page 24](#_Toc56807752)

[Approved-Request View Page 25](#_Toc56807753)

[Purchase-Order Page 25](#_Toc56807754)

[Store-Manager Page 26](#_Toc56807755)

[Store-View 26](#_Toc56807756)

[Supplier-Mange Page 27](#_Toc56807757)

[TASK 4 28](#_Toc56807758)

[TEST PLAN 28](#_Toc56807759)

[TEST CASES 31](#_Toc56807760)

[1. Home Page 31](#_Toc56807761)

[2. Login Authentication 33](#_Toc56807762)

[3. Login with invalid data 34](#_Toc56807763)

[4. User Registration 35](#_Toc56807764)

[5. User Registration Delete 37](#_Toc56807765)

[6. User Registration Update 40](#_Toc56807766)

[7. Store-Item Entry 42](#_Toc56807767)

[8. Store-Item Update 43](#_Toc56807768)

[9. Store-Item Delete 44](#_Toc56807769)

[10. Store view data 45](#_Toc56807770)

[11. Supplier Entry 46](#_Toc56807771)

[12. Supplier Update 47](#_Toc56807772)

[13. Supplier Delete 48](#_Toc56807773)

[14. Generate Purchase Order (PO) 51](#_Toc56807774)

[15. Create a Purchase Requisition (PR) 53](#_Toc56807775)

[16. Approve 54](#_Toc56807776)

[17. Request Delete 56](#_Toc56807777)

[18. Approve the Request 58](#_Toc56807778)

[CONCLUTION 60](#_Toc56807779)

[GANTT 61](#_Toc56807780)

[References 63](#_Toc56807781)

# TABLE OF FIGURE

[Figure 1-Access modifiers 12](#_Toc56517530)

[Figure 2-Access Modifires 12](#_Toc56517531)

[Figure 3-Inheritance types 14](#_Toc56517532)

[Figure 4-Casting 17](#_Toc56517533)

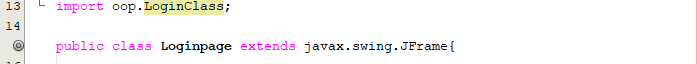
# TASK 1

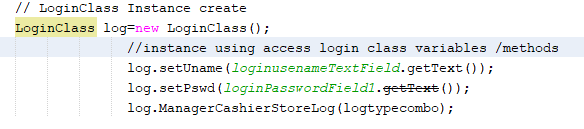
## What is the OOP Concept?

Object oriented programming is a computer programming model that organizes software design around data or objects rather than functions and logic. (Rouse, 2020) In this part we can identify major topics such as object, classes, Inheritance, Polymorphism, Abstract and Encapsulation. When we learn OOP concepts, these concepts depend on below terms. Such as coupling, cohesion, association and composition. Now we consider what the OOP concepts is one by one by one and describe bellow.

## Object

Any entity that has state and behavior is known as object. Such chair, car, pen, keyboard and others. Object can be create as instance of class.





## Java data types Data types

We can identify main two type of Data types.

**Primitive** data types specifies the size and type of variables values, and it has no additional methods. Such as, Boolean, char, byte, short, Int, long, float and double. Not only that, class, interface, arrays, strings and other belong to **Non-primitive** data types. (Anon., 2020)

## Access modifiers

Access modifiers in java specific the accessibility or scope of a field, method, constructor or class.so, we can change the access level of fields, constructors, methods and class by applying the modifier on it. We can see main four type of access modifiers. Such as, **public**, **private**, **protected** and **default**. (Anon., 2011)

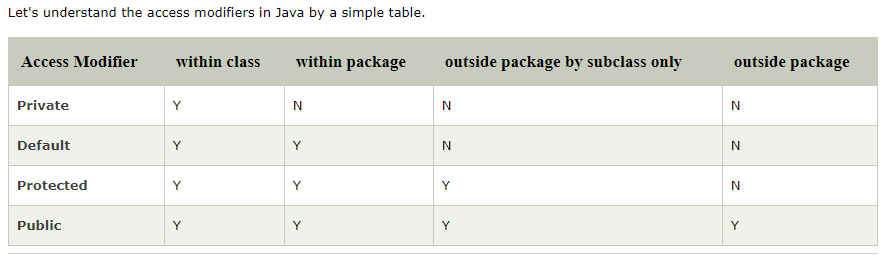


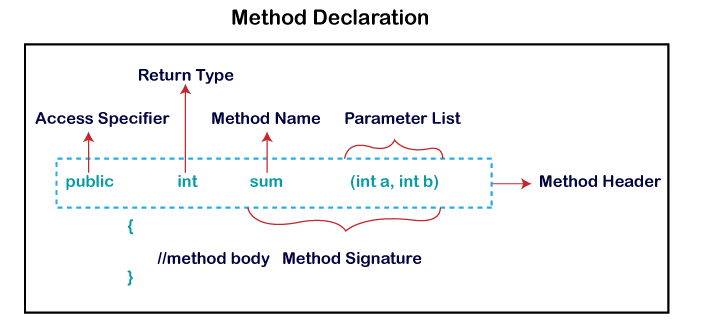
Figure 1-Access modifiers

Not only that we can identify Non-access-Modifier as a **static** key word. (Anon., 2011)

## Return data type

Most time return data type use when we implement the method or function. Return data type used to identify the returning value data type (void mean is no return value and other data type’s int-string-double-float and others or some class object -function.) in the method.

## Methods or functions

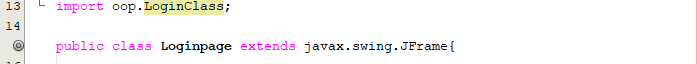
Figure 2-Access Modifires

Especially the method haven’t a return value, so we should use **Void** key word as a return data type.

## Class

A class is an extensible program-code-template or blueprint or prototype for creating objects. Class have object’s behaviors or states and Methods or fields.

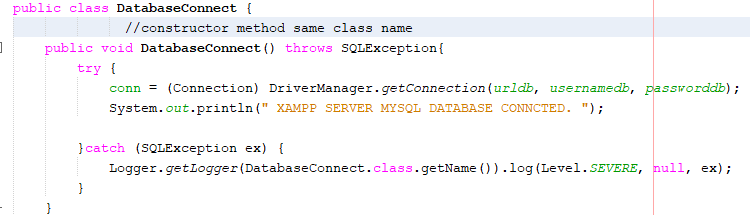
**Access Modifier** **class** **name of the class** {// your body of code here}



## Constructor method

Constructor method is a same class name to same method name. Constructor method to have no return types. Constructor methods run time compiling. Not only that java constructor cannot be abstract, static, final and synchronized. (Anon., 2020)

Class constructor { constructor ( ) {} }



## Inheritance

Inheritance in java is a mechanism in which one object requires all the properties and behaviors of a parent object. So, when it’s using code reusability and when we use run time polymorphism we can use methods overriding. In the OOP have main five type of inheritance, Such as single, Multilevel, hierarchical, hybrid, and multiple inheritance. But in the java programming not supported to multilevel inheritance. So its problem avoided using the **interfaces** concept. It’s like as a java class. “**Extends”** key word use to inherit the some sub class to parent class. When we use interfaces inheritance to use **Implemen**t key word.

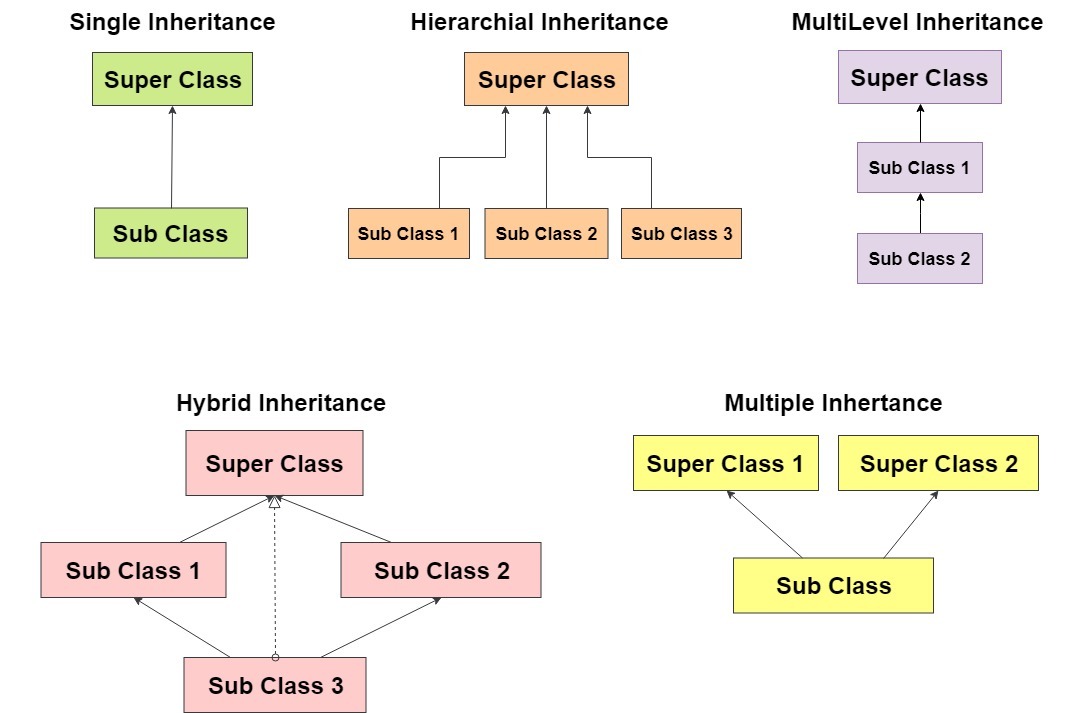
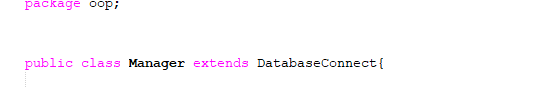


Figure 3-Inheritance types



## Polymorphism

Polymorphism in java is a concept by which we can perform a single action in different ways. (Many forms). We can see two types of polymorphism such as statics and dynamic polymorphism. Compile time polymorphism to tell **static**. So, this concept using **Method Overloading**.

In the run time polymorphism to tell **dynamic**, in this concept use **Method Overriding**.

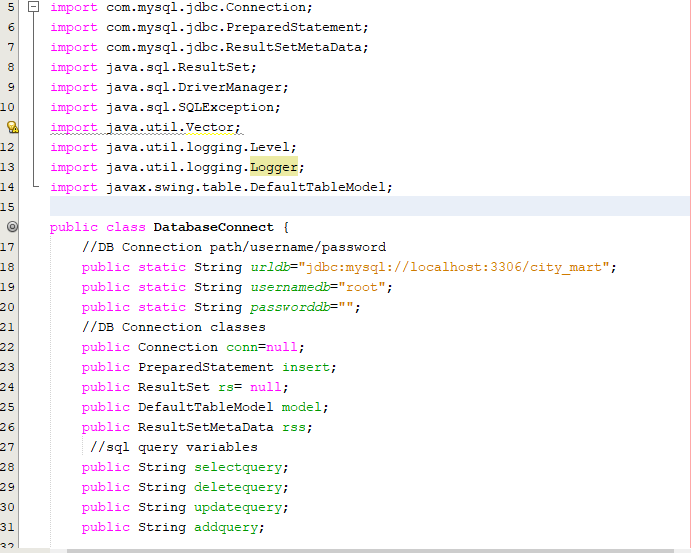
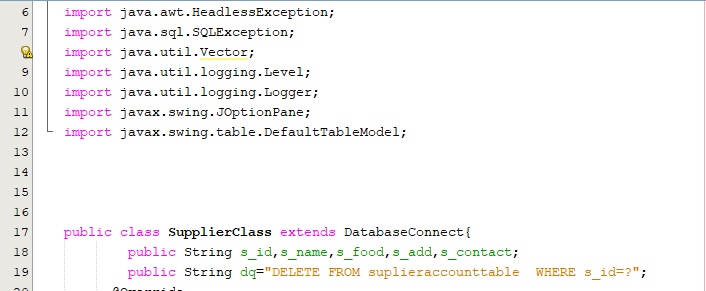
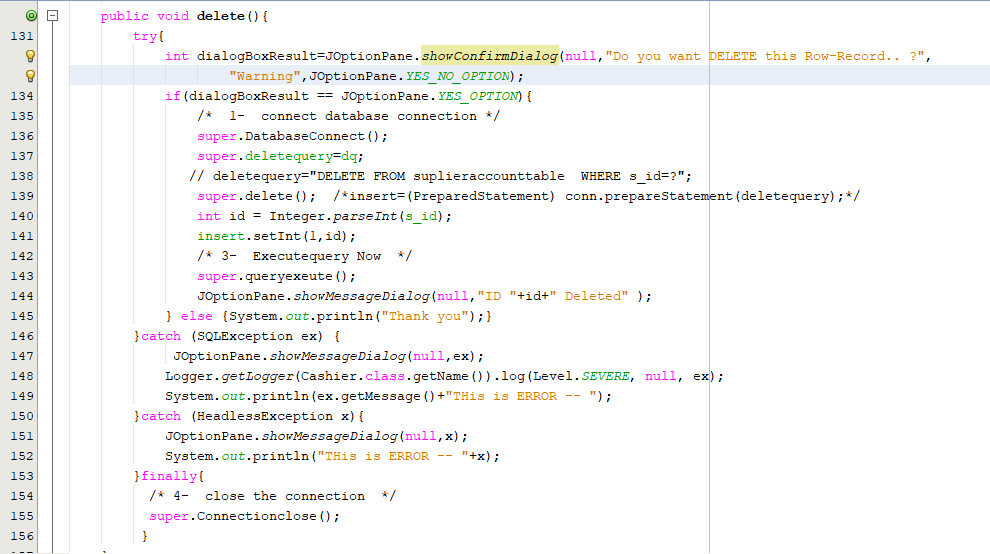
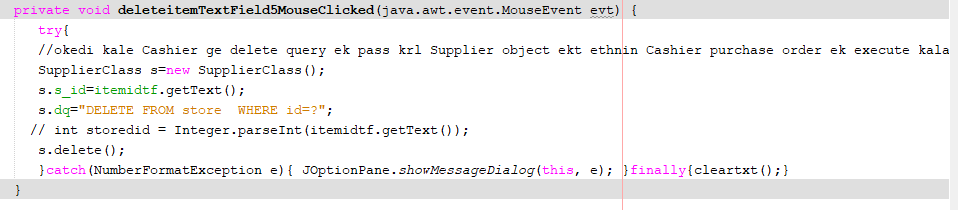
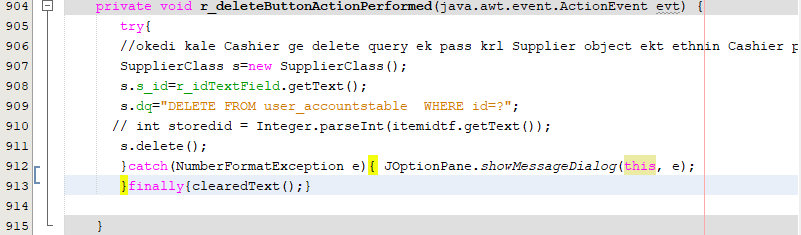
### Method Overloading

Create a same name with multiple methods in same class body. So, do this should all the methods return data type should same and all methods should have difference parameters. This concept to called method overloading. (Anon., 2020)

### Method Overriding

When inherited class methods change in the child class body using creating new parent class method name to same method. Not only should that it be same parameters in the method. (Anon., 2020)

* You should know private, statics, final and final class methods cannot be an override.

## Casting

#### *Up casting*

When there are inherited parent class object put to sub class object, this concept to call up casting. Therefore when we can access the sup class methods and variable using calling through the parent class object.

#### *Down casting*

When there are inherited sub/child class object put to the parent class object, this concept to call down casting. Therefore when we can access the parent class methods and variable using calling through the sub class object.

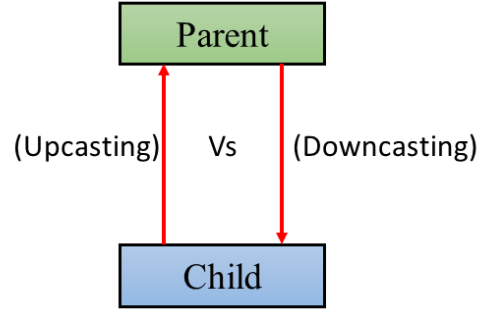


Figure 4-Casting

## Abstract

Abstract mean is we cannot write codes. So, some time we can will be identify some methods and class we cannot write codes. So, its class and methods to us call as abstract class or abstract methods. So, when we implement the abstract class or method firstly we should the put “**abstract**”

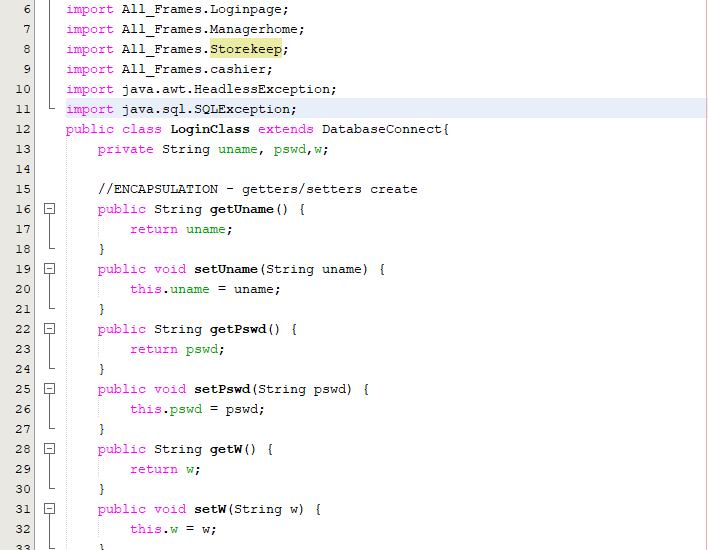
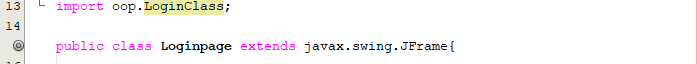
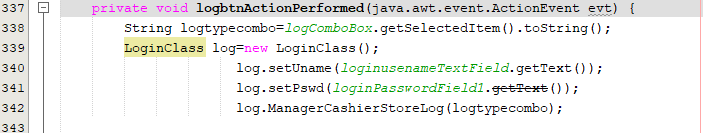
Keyword before the class keyword or method name.

* You should know, abstract class cannot generate an objects in this class using. Because we cannot generate constructor method in the abstract class. But we can inherited that abstract class and after we can access the sub class through the parent (abstract) class methods and variables. Not only that we can override the other methods in the abstract class.
* Abstract methods access to have another way is that all inherited class should be an abstract. So this task to do have another way. It is an **interfaces** concept. Interface like as a classes. (Anon., 2020)

## Encapsulation

Encapsulation in java is a mechanism of wrapping the data variables and code acting on the data (method) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as **data hiding**.

* Firstly we declare the **variables** with **private access modifier**. After generate **getters** and **setters** with **public** access modifier to, put values in to variable and get the value in inside the variable.

## “This” Key word

This key word use to access current class variables and methods.

Ex- this. Variable; this. Method ();

## “Super” Key word

Super key word use to access Parent or super class variables and methods.

Ex- super.variable; super.Methods ();

# TASK 2

## User Case Diagram



In this diagram I show all the this automation order system functions and who is the it’s do. So I identified main three actors as “manager, cashier and supplier” and their tasks.

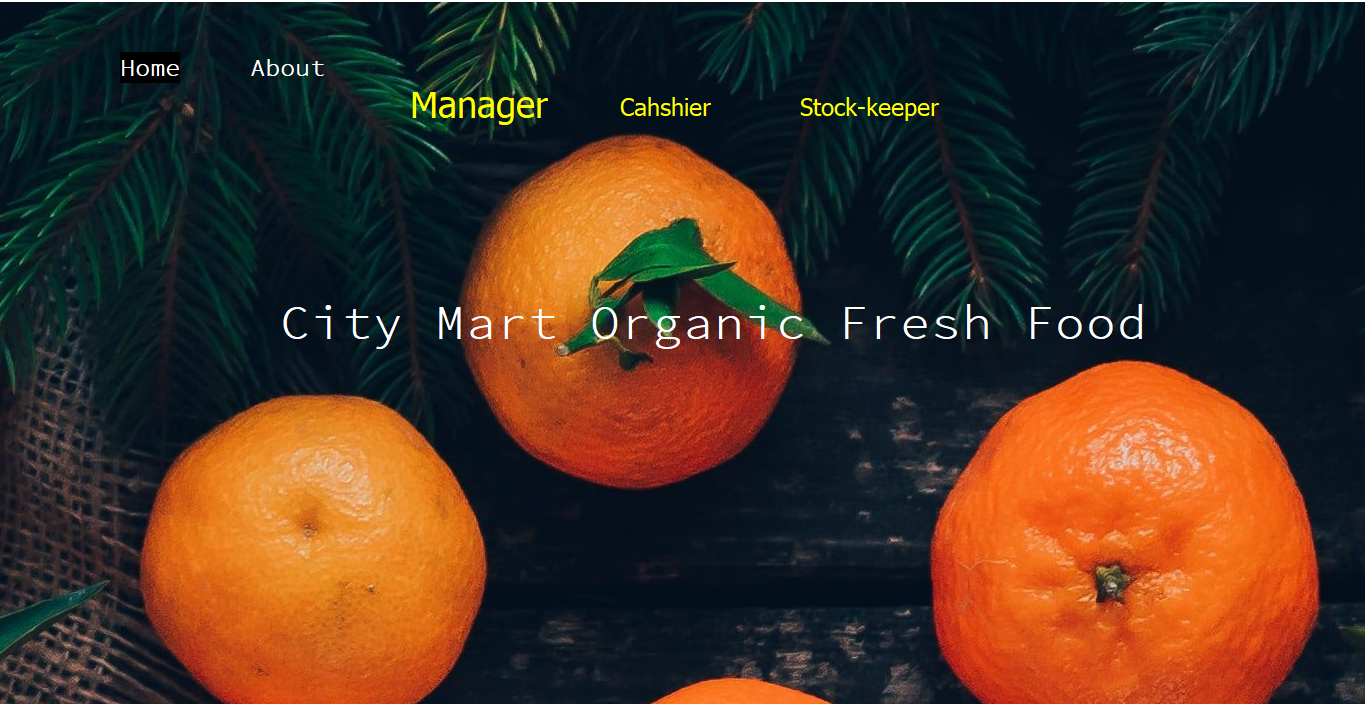
## Class Diagram for City Mart Automation Order System

Assumptions

* I thought all DatabseConnectionClass with other all the class have “COMPOSITION” relationship.
* Not only that I thought DatabseConnectionClass multiplicity have “one to many” with other classes.
* DatabseConnectionClass have Add, delete, update methods inherit and override to other all the classes. Because I could identify that methods common to all the classes. Not only have that in the Java between normal classes can’t multiple inheritance.
* GetModelDataToJTable methods to use return the DefaultTableModel class object. So this objects after the cached in design pages “jTable” to send all the model data to view rows and Colums in the Database Table.

# TASK 3

## Home Page



This is a Home page or Landing window page of System. This page have Home, Manager, Cashier, Stock-keeper buttons. It’s to use me all the label to set mouse click events to like as to work buttons.

When clicked the buttons we carry to the login page. If login success open new windows page as a Manager, Cashier, Stock-keeper home pages. Middle of the page have label “city mart fresh food”, so we can mouse move through it then change it font color.

## Cashier Page



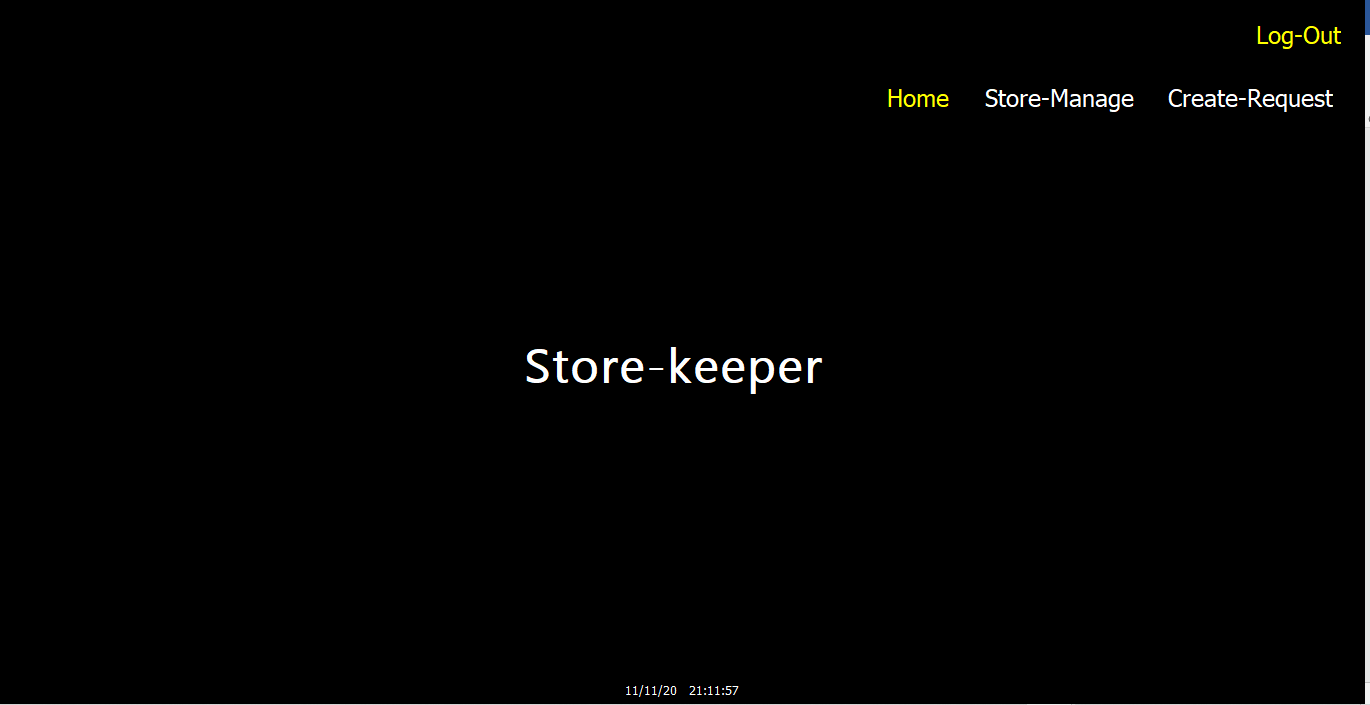
Cashier can view the Approve-request, sales view and purchase-order. I think when purchasing order cashier to helpful to check this order is approve or not. Then he or she can purchase the order within bill.

## Manager page



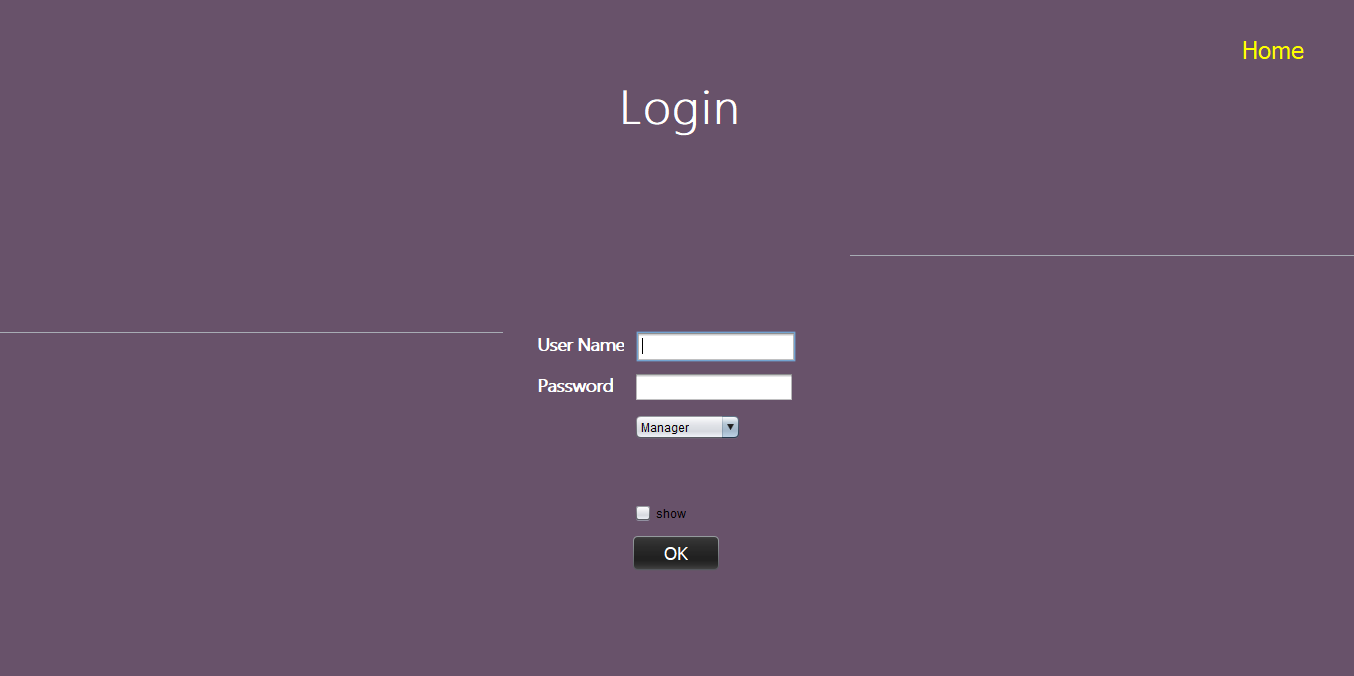
Manager can approve the order, user accounts manage, Suppliers mange, sales and stock view access. .

## Store-Keeper page



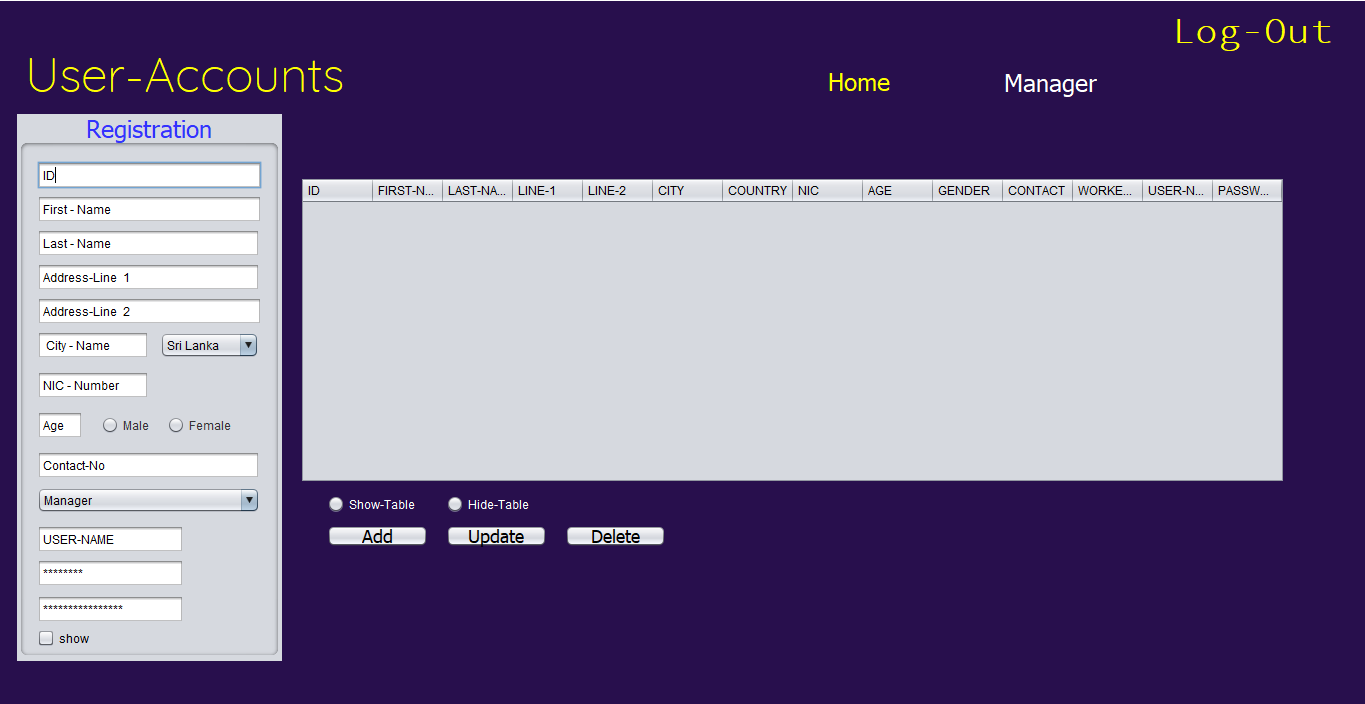
Stock–keeper can create request and he can stack manage.

## Login page



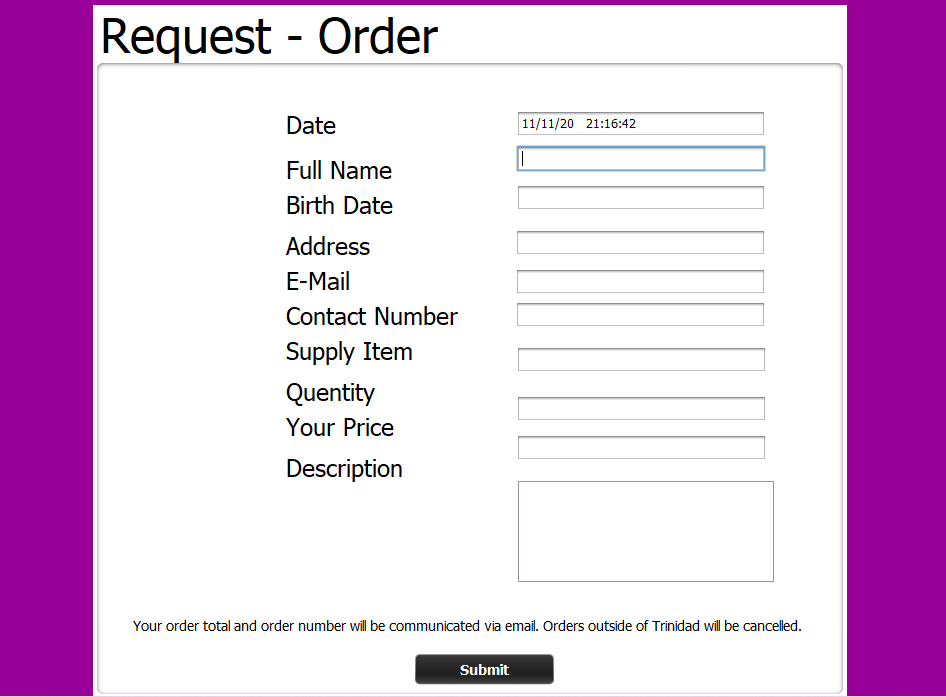
Login page have multi user login facility. When some user want to login to the system first he should enter he or she username, password and about what is her work position. When click radio button we can hide user name and password. Finally, clicked the “ok” button, its true the login details open the new work space windows. (Go to cashier/manager/ stock-keeper home pages). Else, refresh the login page.

## User Account Registration Page



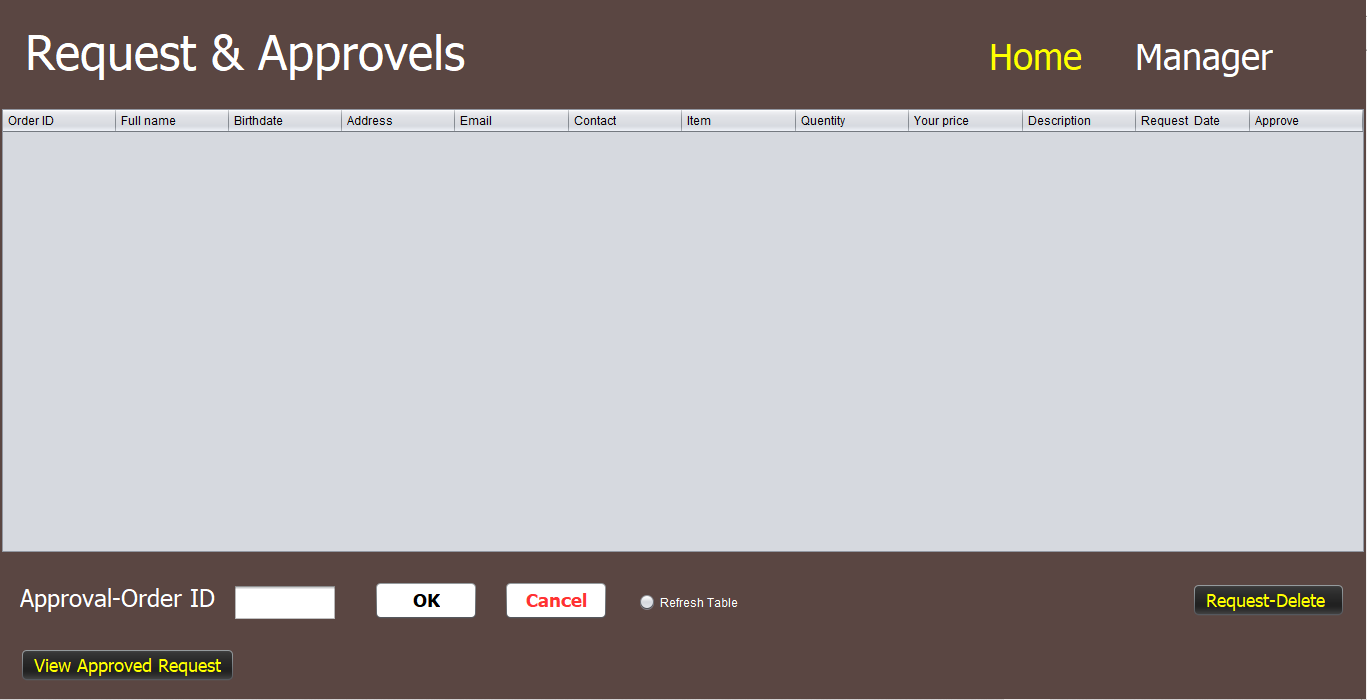
In the user account page, manager can add, delete, updates the users to system. Such ass manager, cashier or store-keeper. In this time all the users’ username and password generated to login the system. Additionally “show-Table and Hide-Table” button using manager can control the table data.

## Request-Order page



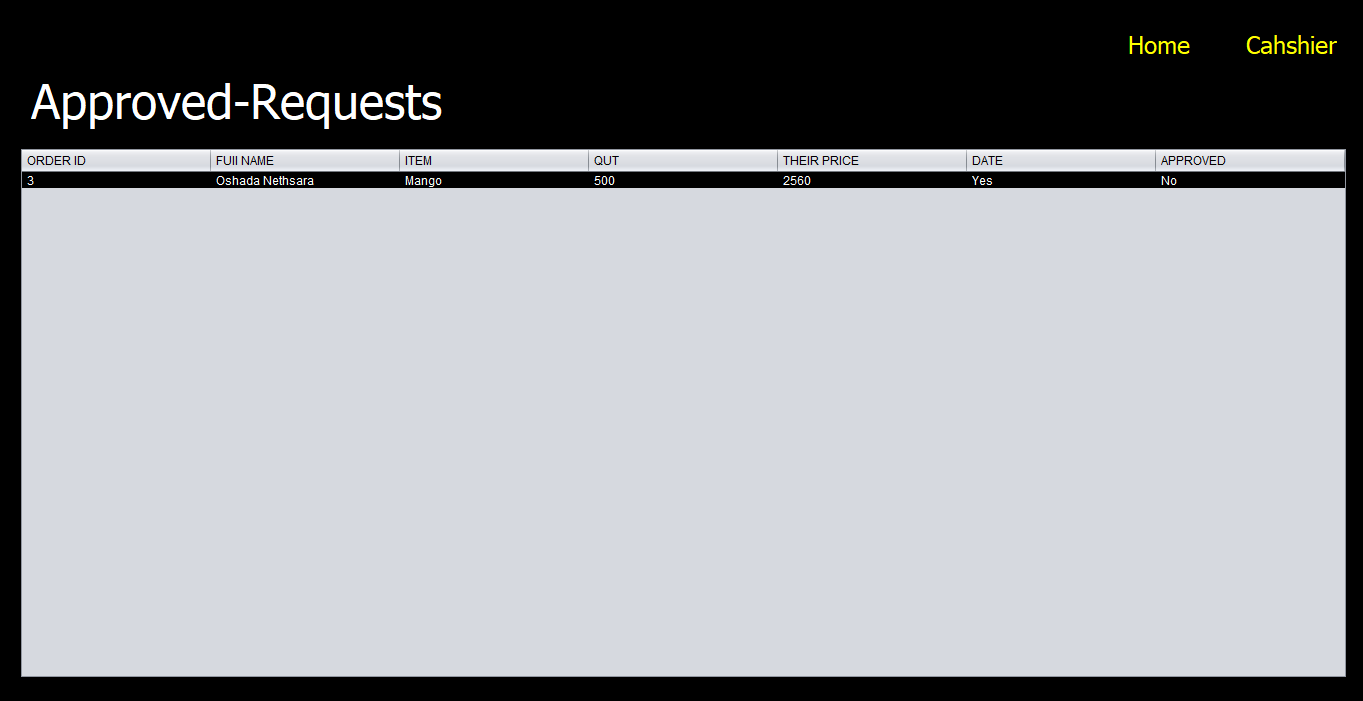
This part doing stock keeper. In this page some supplier can requites their orders after filling the form.

## Request View Page



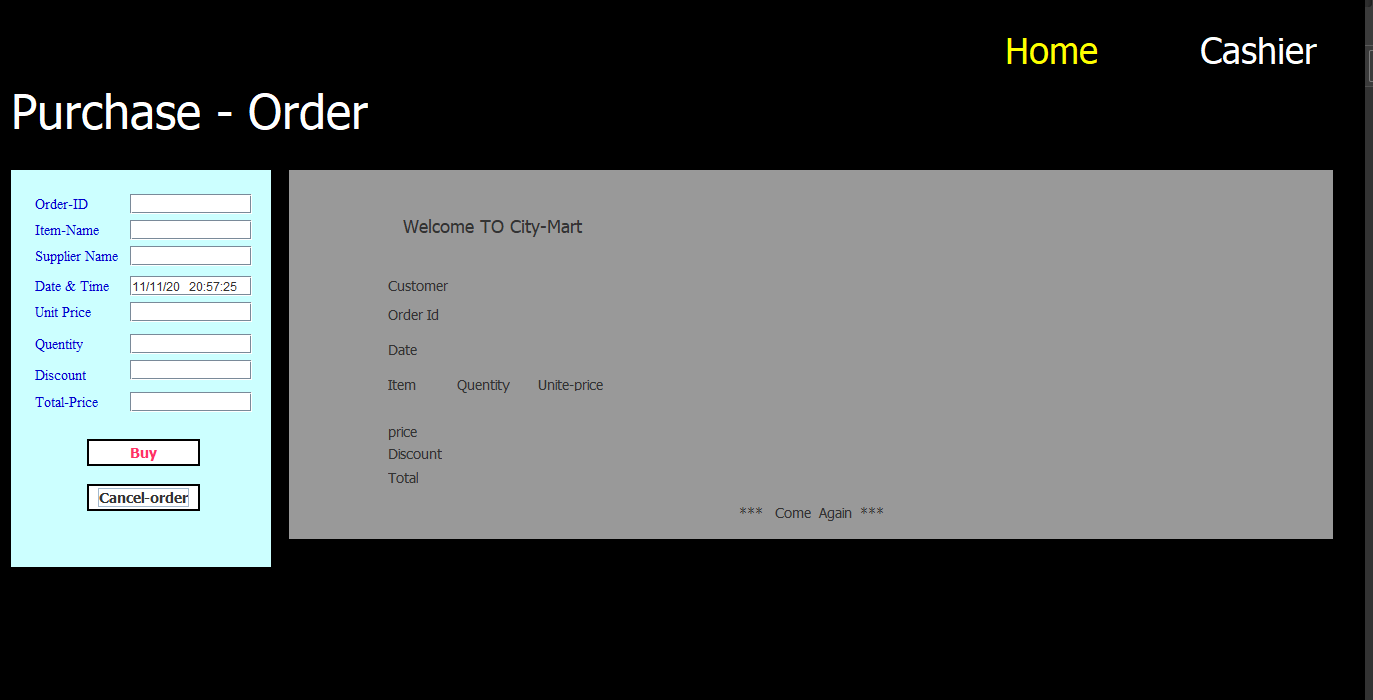
This part doing in Manger. In this page he can show requests and then he or she can Approved or not and delete the requests. Not only can that manager view the only approved request clicking “view approved request” button.

## Approved-Request View Page



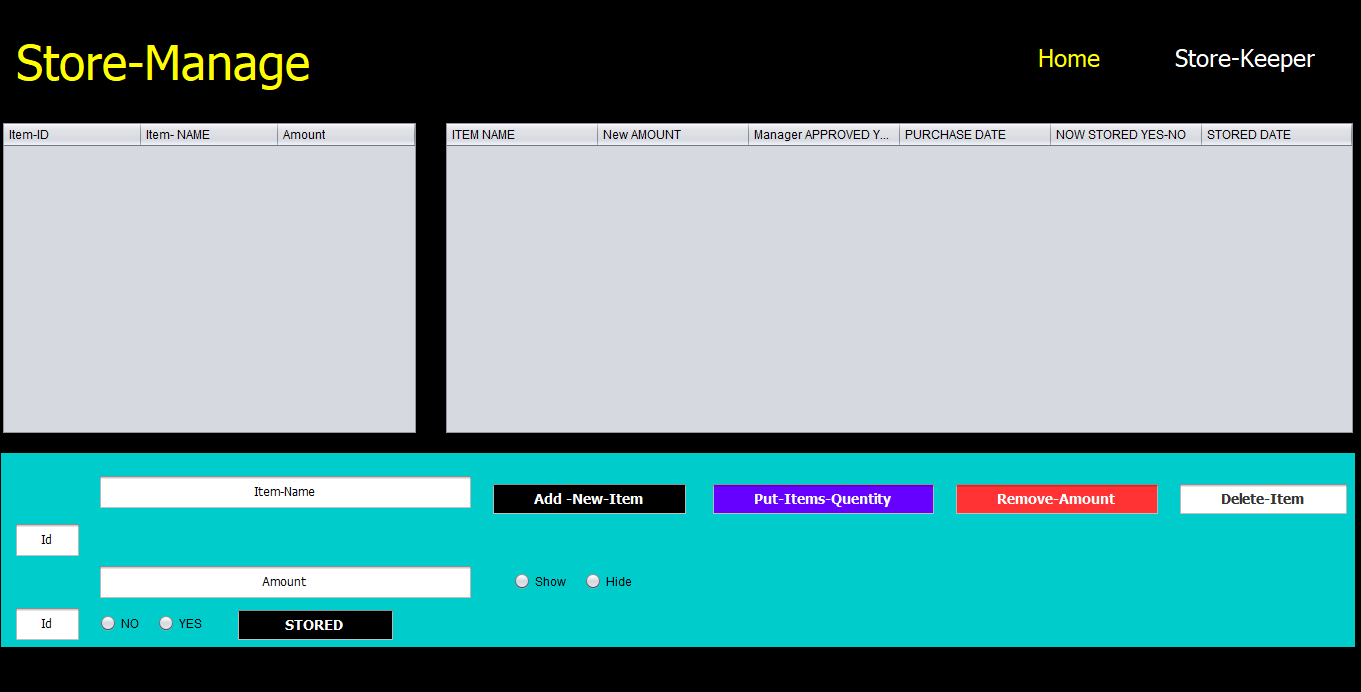
In this page provide some specific details with view the only approved request to stock keeper and cashier. So this details want to when purchasing order and when store manage.

## Purchase-Order Page



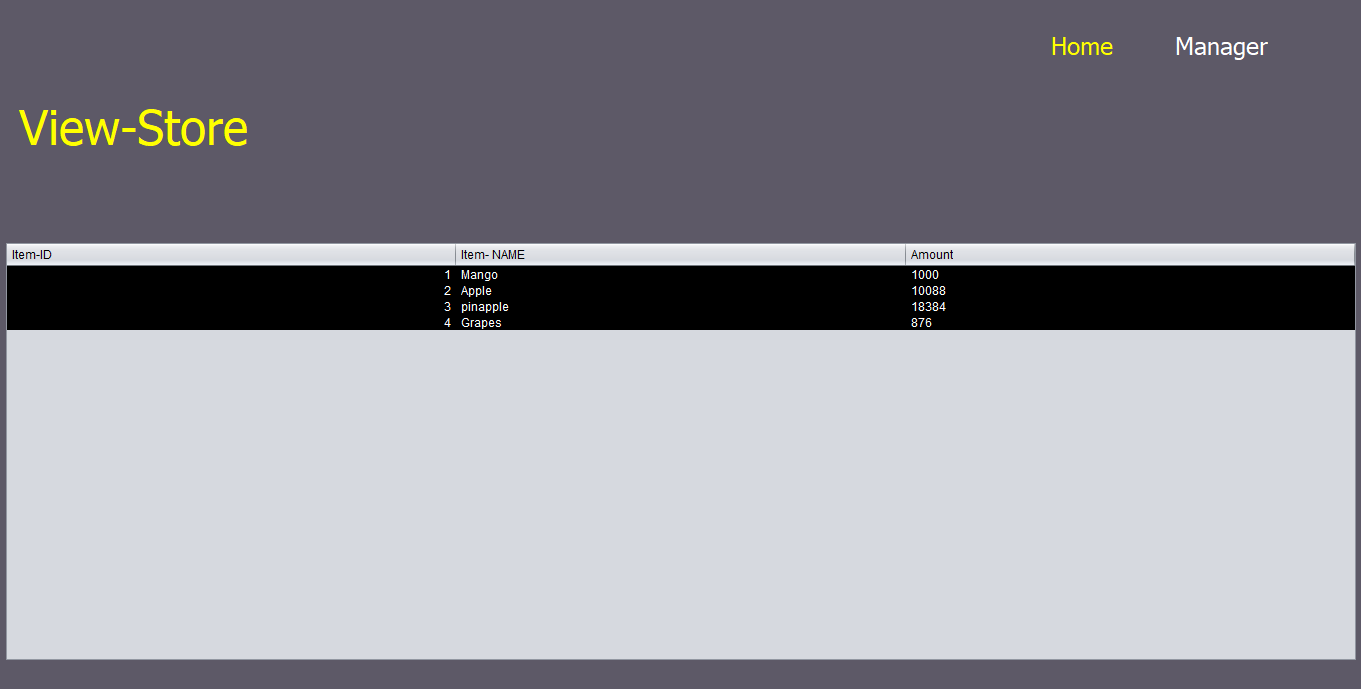
This page only can access cashier. He or she can purchased the approved orders and issue the bill. So cashier should enter above information.

## Store-Manager Page



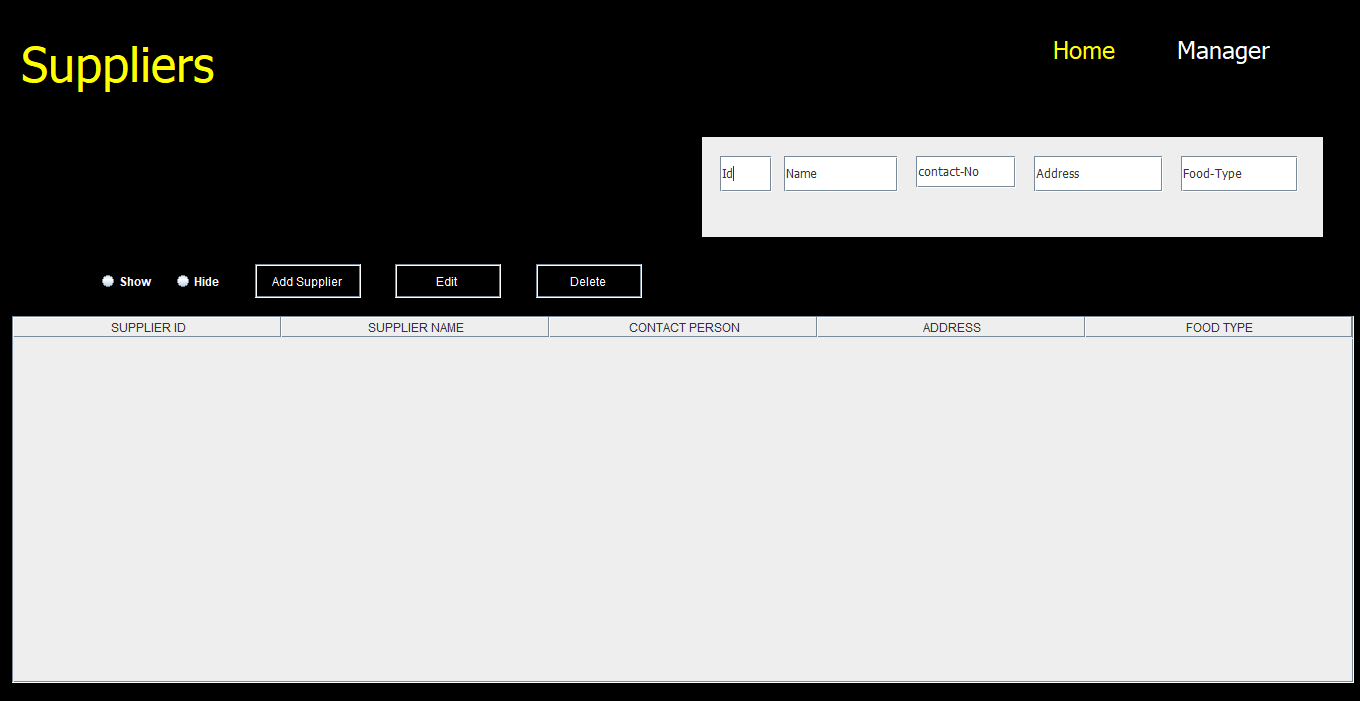
All the approved orders can storing to the data base. Right side table view the app the approved type yes orders details and left side table using view all the inserted store item data. This page provide new item add, that item update or delete , that item amount only add or remove facility and store keeper can order item after the store that order update as a stored “yes or no”.

## Store-View



This page can access manager to view how about the these days item store.

## Supplier-Mange Page



This page provide to system to add about who is the over company suppliers details. Not only that manager can add, delete and update facilities to handle the supplier database table.

# TASK 4

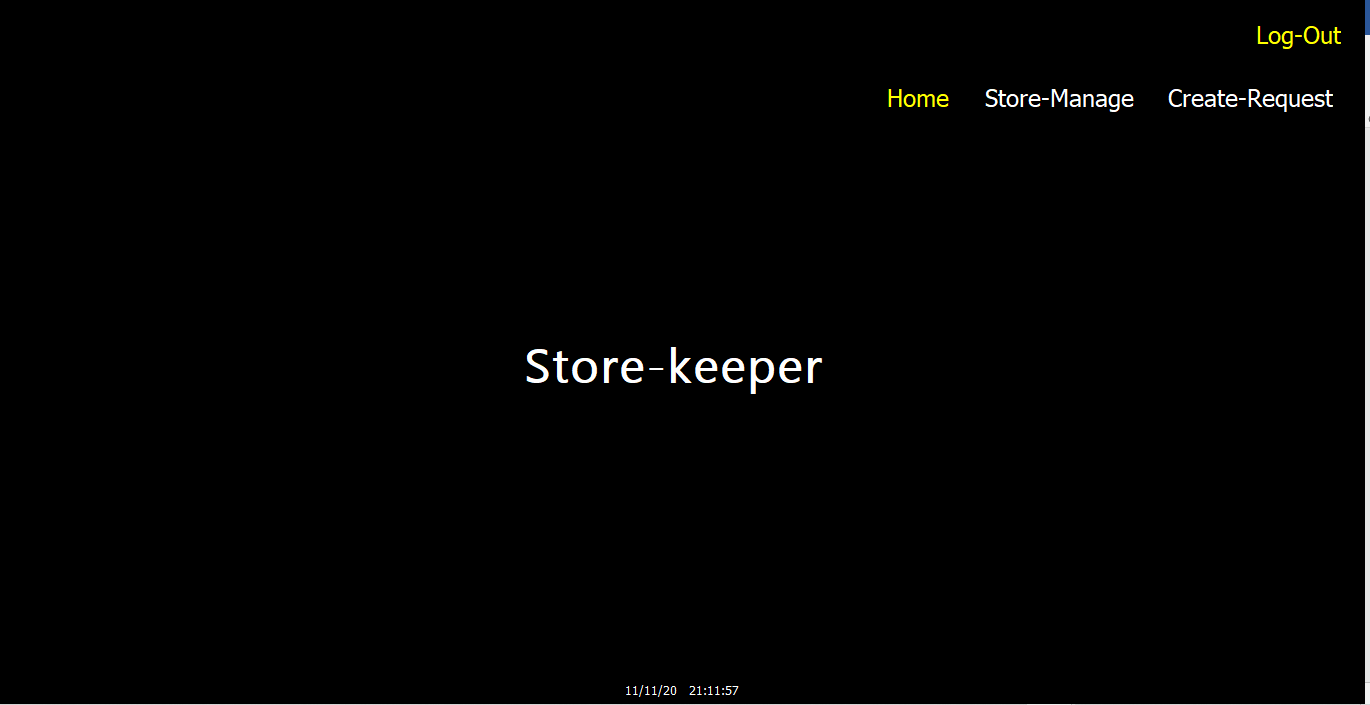
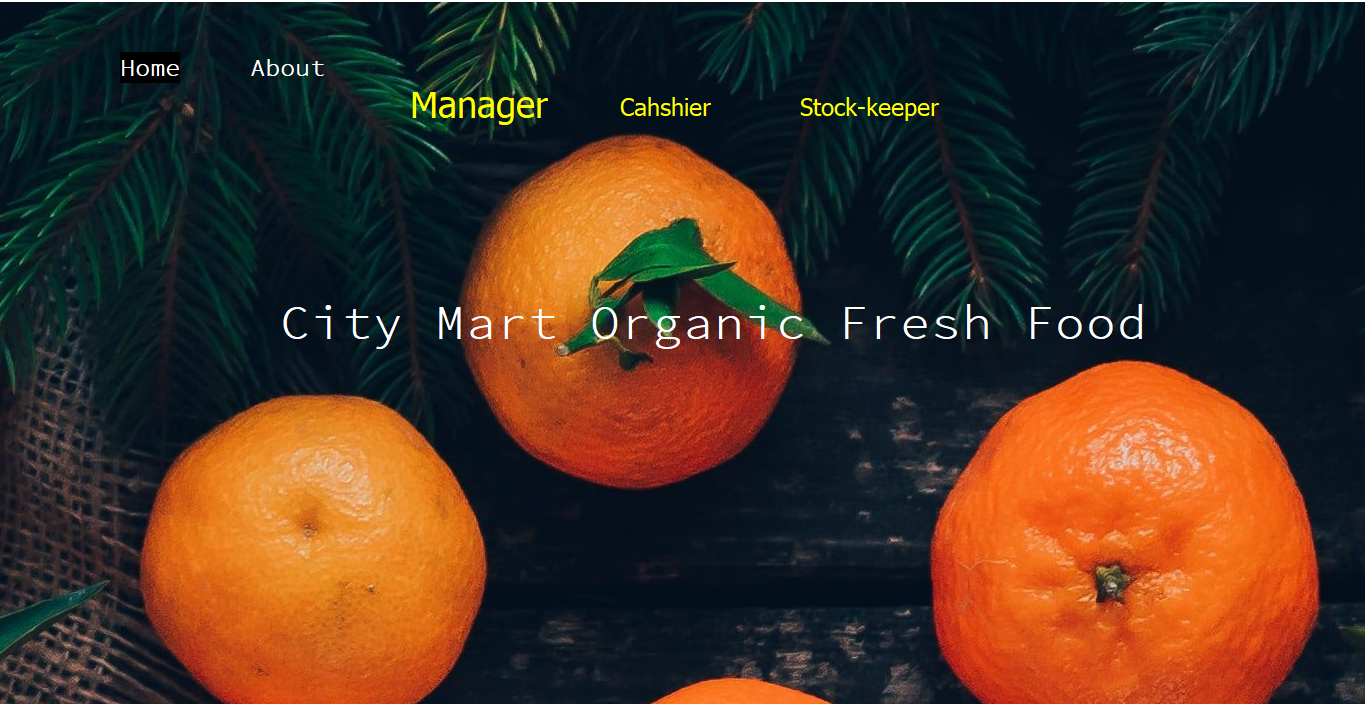
## TEST PLAN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case number** | | **Test case name** | **Steps** | **Expected result** |
| TC 2 | | Login Testing |  |  |
|  | TC2.1 | Check Login with Valid Data | Input valid work type, Username and password | “..Login Successfully..” message view |
|  | TC2.2 | Check Login with invalid data | Input invalid work type, Username and password | “Incorrect User name or Password” message view and then sql exception error message view |
| TC1 | | User registration Testing |  |  |
|  | TC1.1 | Check Data Add to user registration | Entering the data in all the text fields then add button click. | “Added successfully” message view |
|  | TC1.2 | Check User account registration data update | Click the table row or then enter new values to text fields. | “Update successfully” message view |
|  | TC1.3 | Check User account registration data Delete | Click the table row or then enter ID value to text fields. | View Message Dialog box with Yes No option and then view “Deleted successfully” message view |
| TC6 | | Store-Manage testing |  |  |
|  | TC6.1 | Check store-item entry. | Entering the data in all the text fields then add button click. | “ID 4 to Add Grapes 876” like as message view. |
|  | TC6.2 | Check store-item update | New data input to change old data in the record | “ID 3 Pineapple Updated” like as message view |
|  | TC6.3 | Check store-item id delete | Click the table row or then enter ID value to text field. | “Do you want to delete this row record?” message and view “ID 4 Deleted” like as message |
|  | TC6.4 | Check store-item View page | Click view button | Open new window and Show the table data |
| TC4 | | Supplier Data Manage |  |  |
|  | TC4.1 | Check the create new supplier | Entering the data in all the text fields then add button click. | “Added successfully” message view |
|  | TC4.2 | Check the supplier data update | Table row click or after new data insert to text fields. | View “Updated Successfully” message |
|  | TC4.3 | Check the supplier data delete | Click the table row or then enter ID value to text field. | “Do you want to delete this row record?” message and view “ID 4 Deleted” like as message |
|  | TC4.4 | Check the supplier data view. | Clicked the show button | Table data view. |
| TC3 | | Create purchase requisitions |  |  |
|  | TC3.1 | Check the request-order data entering. | Entering the data in all the text fields then Submit button click. | “Added successfully” message view. |
| TC5 | TC5.1 | Generate purchase order checking | Entering the data in all the text fields then Submit button click | View the with entered data. |
| TC4 | | Approval Manage |  |  |
|  | TC4.1 | Check the Approve “Yes” | Click the table row or enter request ID value and after click “OK” button. | View as message “ID 5 APPROVED “Yes” |
|  | TC4.2 | Check the Approve Cancel as “No” | Click the table row or enter request ID value and after click “CANCEL” button. | View as message “ID 5 APPROVED “No” |
| TC8 | | Check the Home page buttons | Clicking the all the buttons | Go to the login page |

## TEST CASES

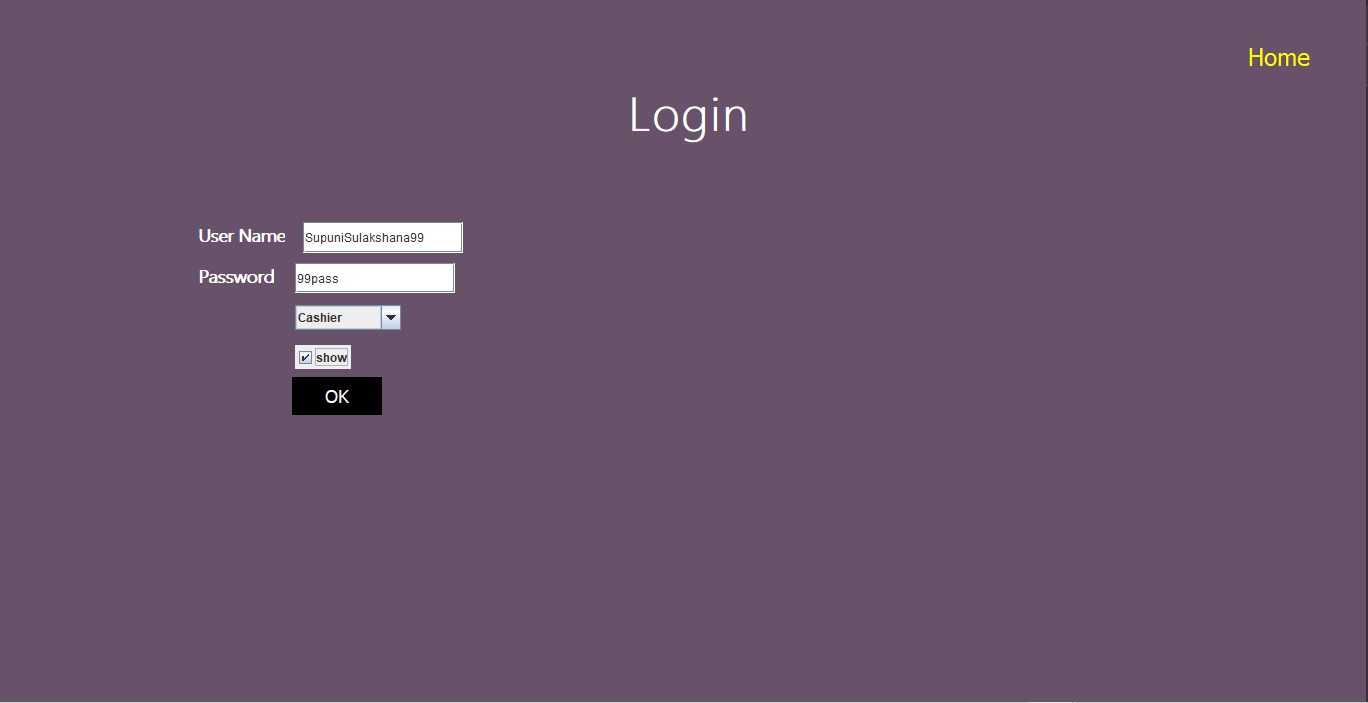
### Home Page

|  |  |
| --- | --- |
| **Test Case NO** | 08 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Home page Buttons mouse click, move events |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. Clicked on “Home” 2. Clicked on “Manger” 3. Clicked on “Cashier” 4. Clicked on “Stock-keeper” 5. Mouse move on “City Mart Organic Fresh Food” |
| **Expected Results** | 1. Refresh the Home page 2. Go to “Login Page” page 3. Go to “Login Page” page 4. Go to “Login Page” page 5. Change the “City Mart Organic Fresh Food” label colour white to green |
| **Actual Results** | All Home page items accurately working in the system. |
| **Pass/Fail** | pass |
| **Conclusion** | Expected actual output. |

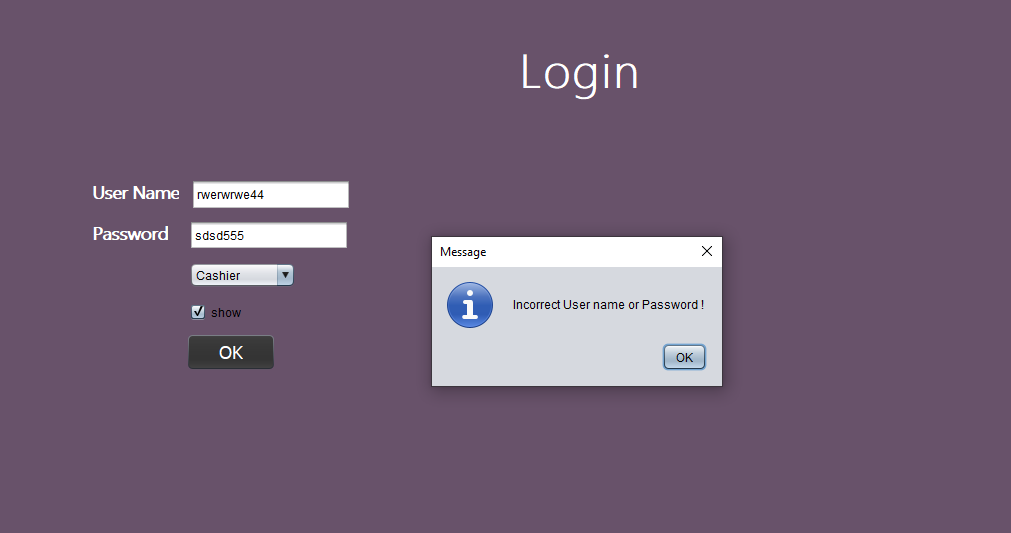
|  |  |
| --- | --- |
| **Test Case NO** | 02 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Login Page with valid Data |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. User name= “supunisulakshana99” 2. Password= “99pass” 3. Select “combo box item” = “Cashier” 4. Check box = enabled 5. Mouse move the “Login” label |
| **Expected Results** | Login Successful message and open “Cashier” Page |
| **Actual Results** | Login Successful message and open “Cashier” Page |
| **Pass/Fail** | pass |
| **Conclusion** | Expected actual output. |

### Login Authentication



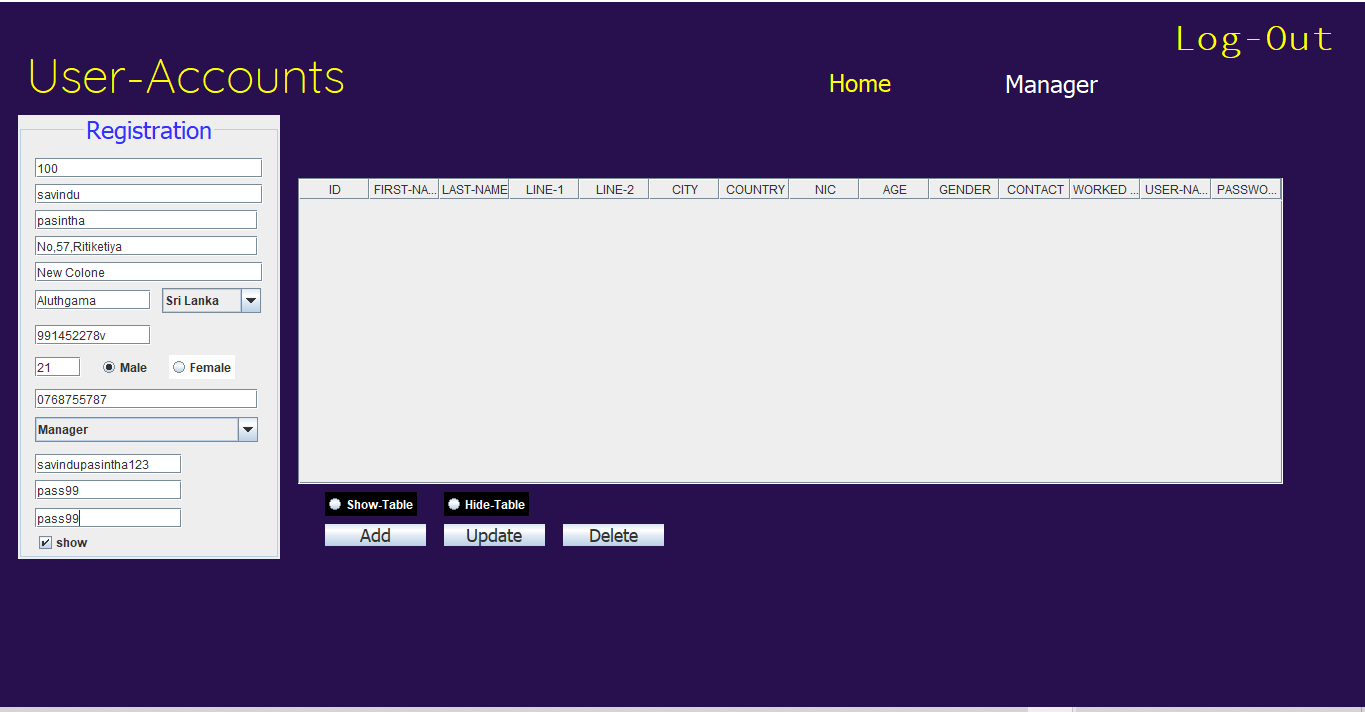
### Login with invalid data

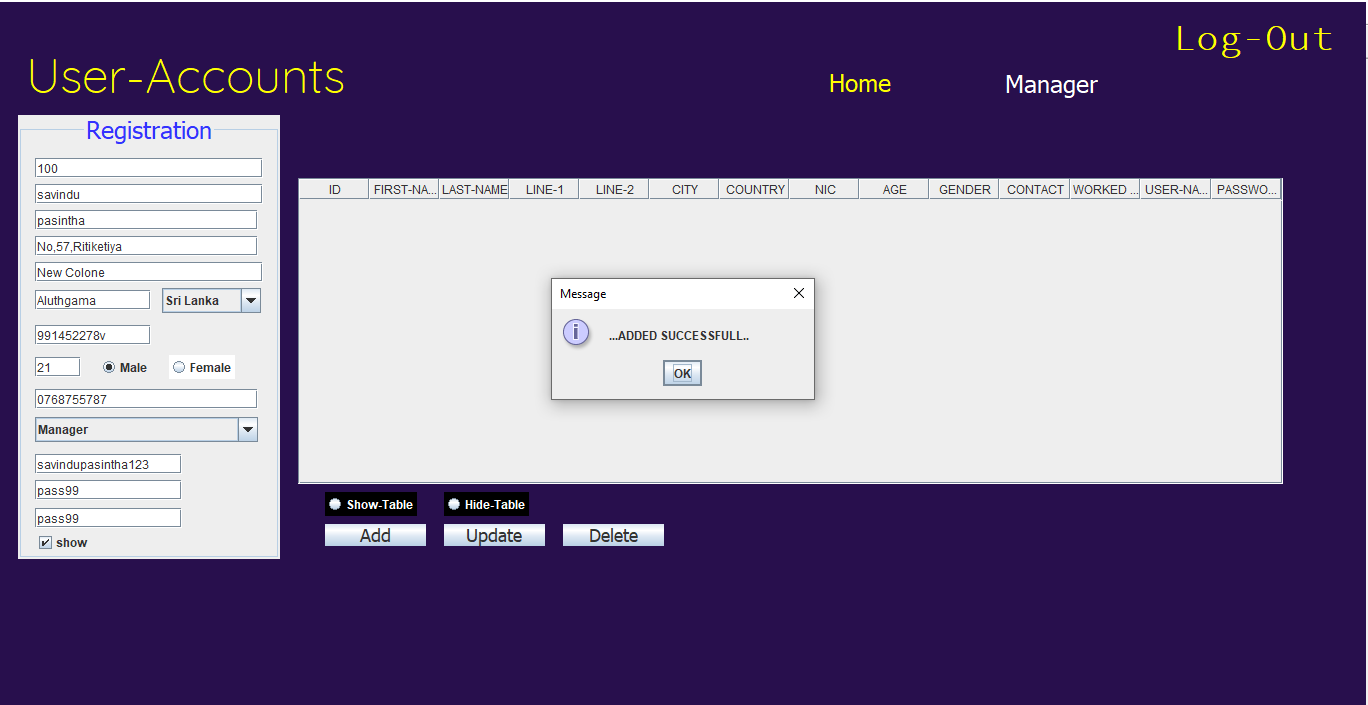
|  |  |
| --- | --- |
| **Test Case NO** | 02 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Login Page with invalid Data |
| **OBJECTIVE**  ID | 2 |
| **Test Data** | 1. User name= “rwerwrwe44” 2. Password= “sdsd555” 3. Select “combo box item” = “Cashier” |
| **Expected Results** | Invalid username or password message and Refresh “Login” Page |
| **Actual Results** | Login fail and direction again login page. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

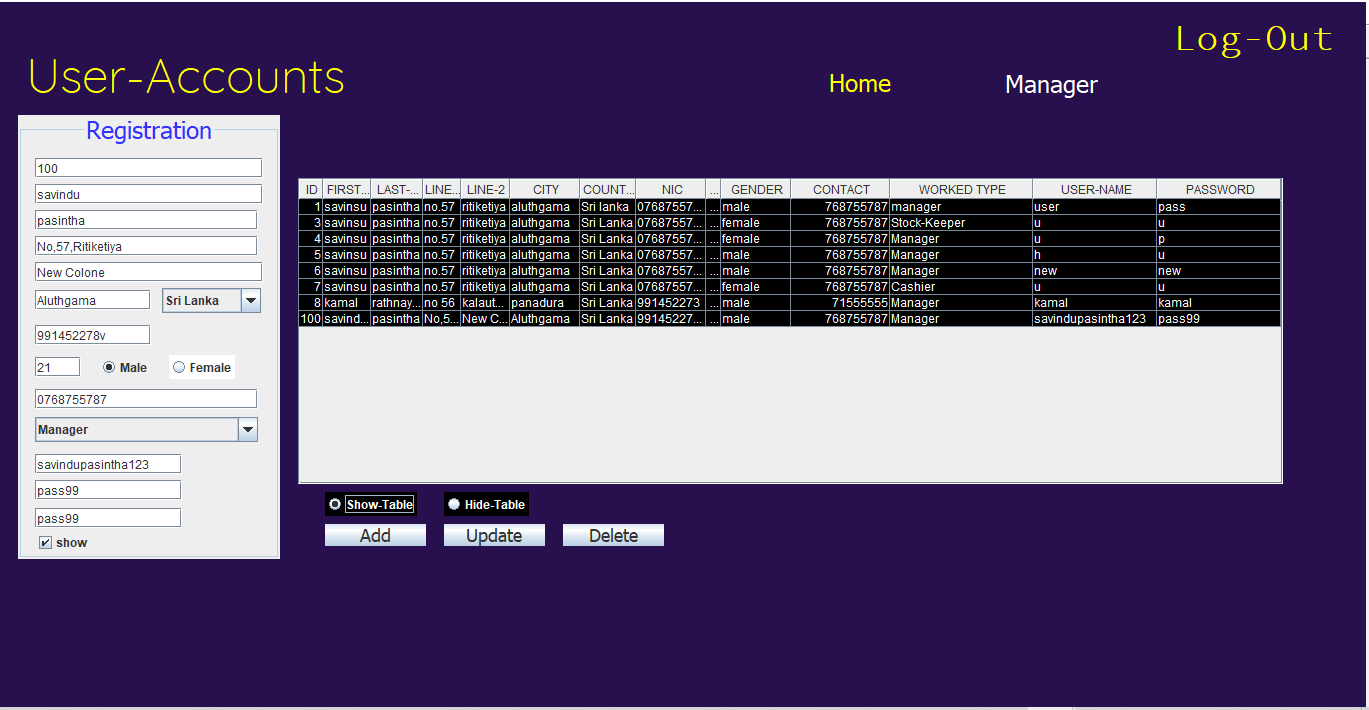


### User Registration

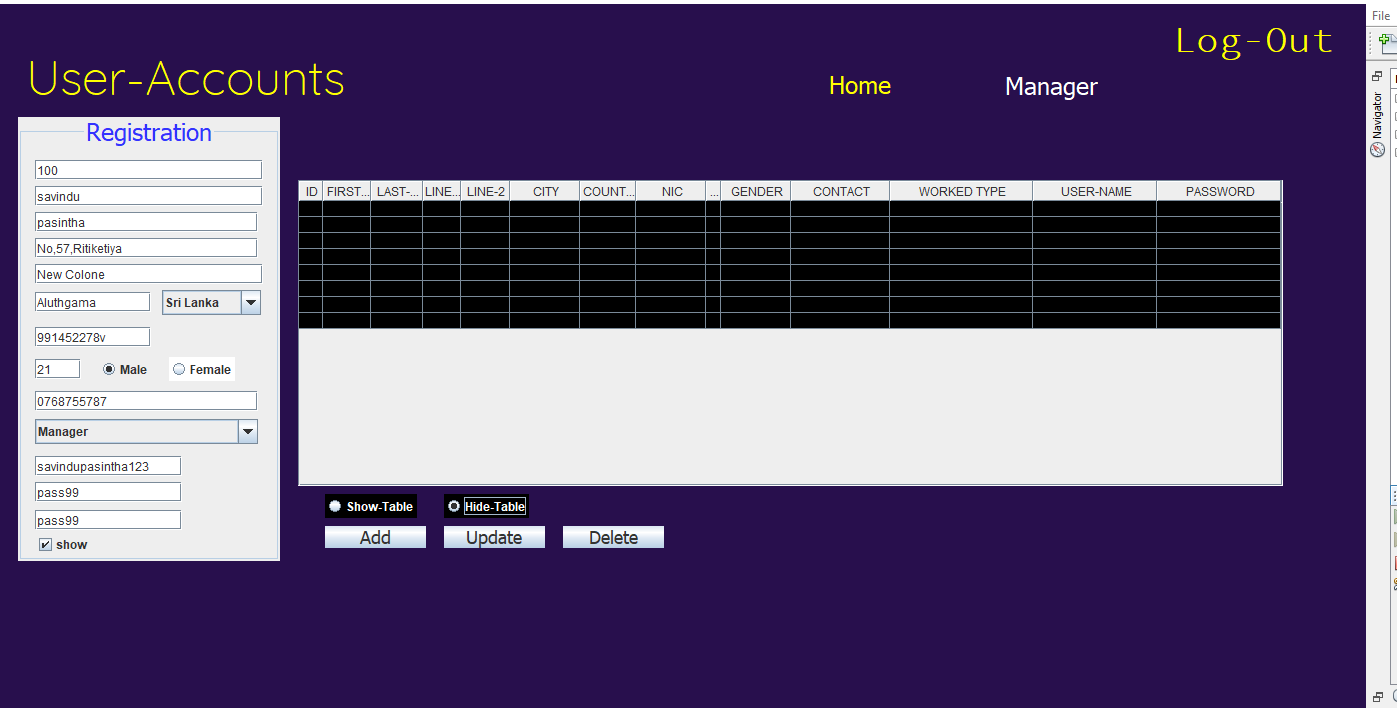
|  |  |
| --- | --- |
| **Test Case NO** | 01 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | User Account Registration to data add. |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. ID=“100” 2. First-Name= “savindu” 3. Last-Name= “pasingtha” 4. Address-Line 1= “No.57,Ritiketiya” 5. Address-Line 1= “New Colony” 6. City-Name= “Aluthgama” 7. Select country= “Sri Lanka” 8. NIC-Number= “991452273v” 9. Age= “21” 10. Contact-No= “0768755787” 11. Select Work type = “Manager” 12. USER-NAME= “savindupasintha123” 13. Password= “pass99” 14. Confirm password= “pass99” |
| **Expected Results** | Added success full message view and clear the all text fields’ values as an empty. Then “SHOW TABLE” click after view the “Registration data” |
| **Actual Results** | Expected result Successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |



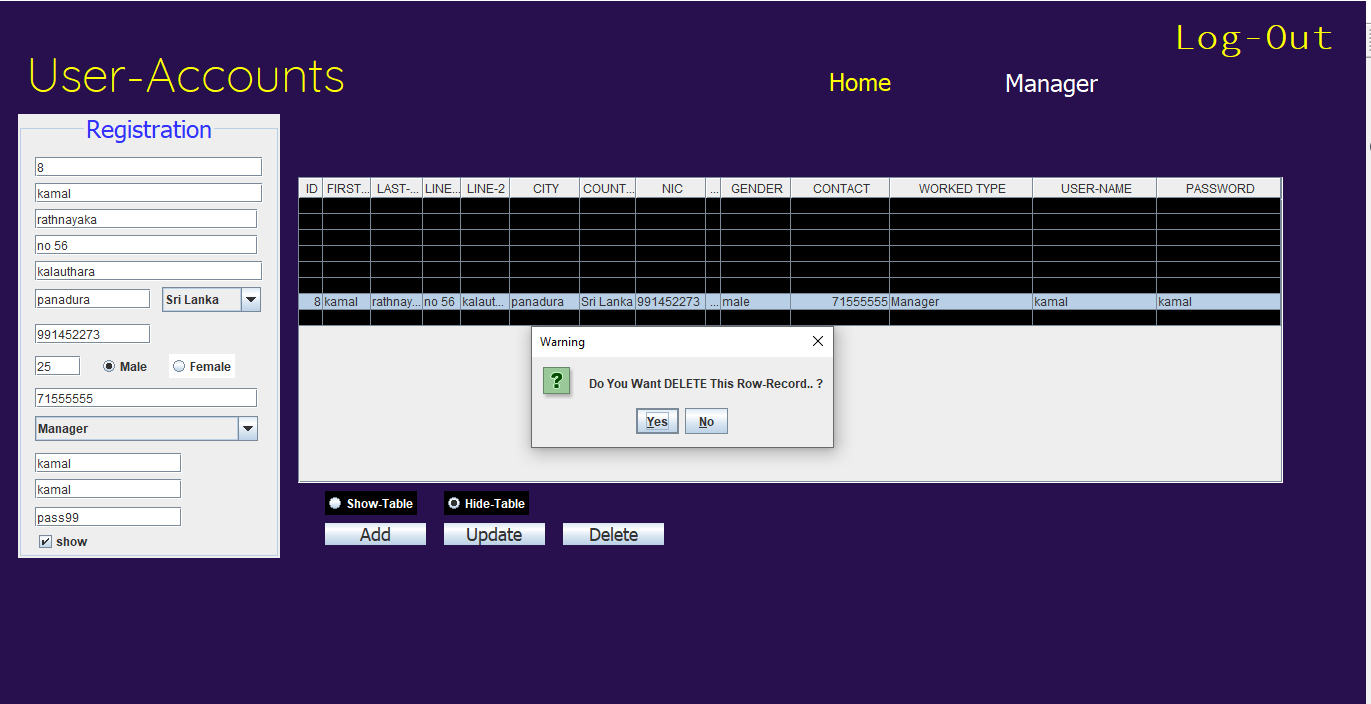


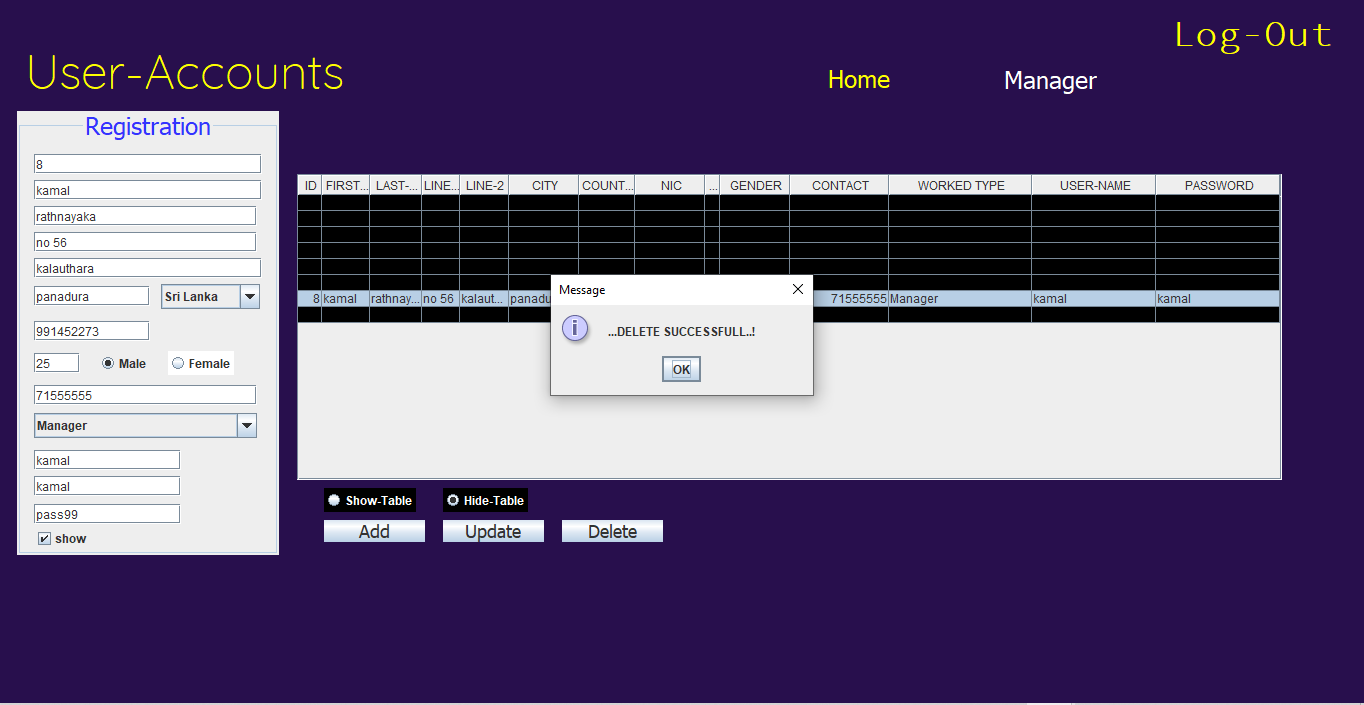


### User Registration Delete



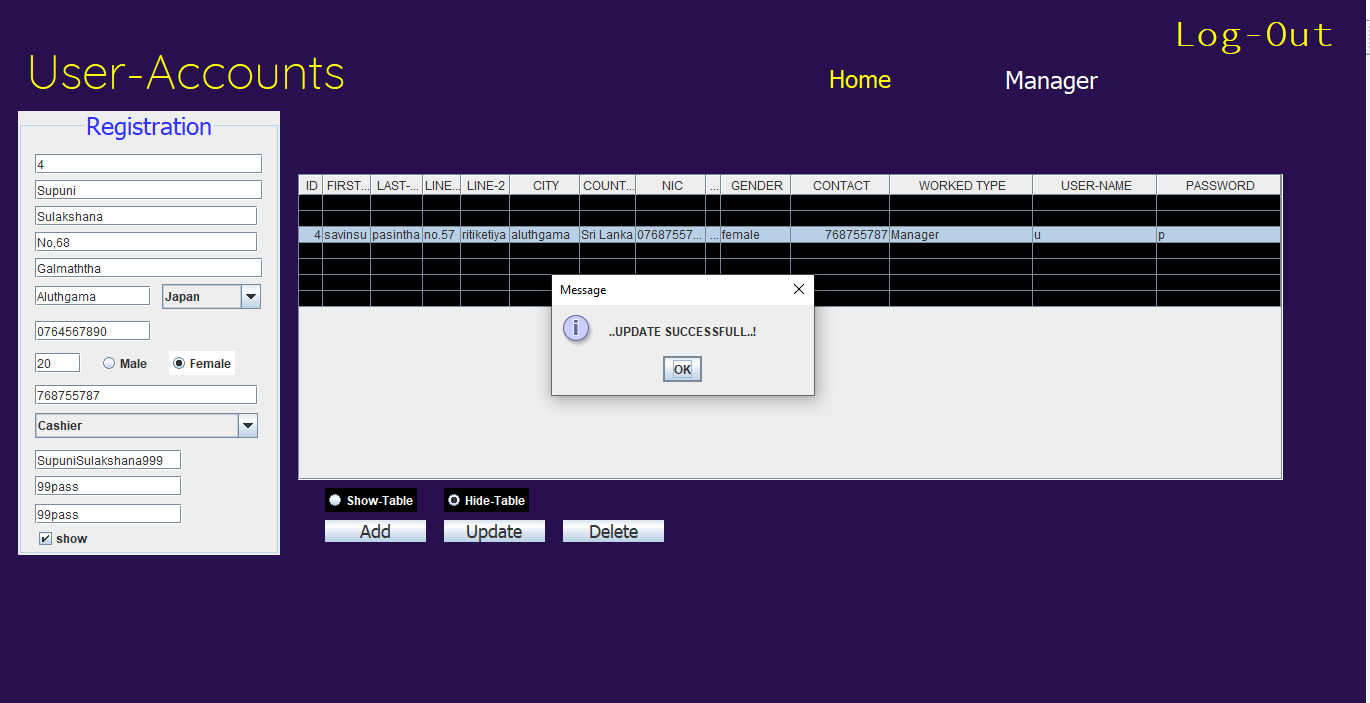
|  |  |
| --- | --- |
| **Test Case NO** | 01 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Registration data |
| **OBJECTIVE**  ID | 3 |
| **Test Data** | Click Table Row and Then Click Delete Button. |
| **Expected Results** | “Do you want to delete this row record?” message coming and after clicking ok then view “deleted successful” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

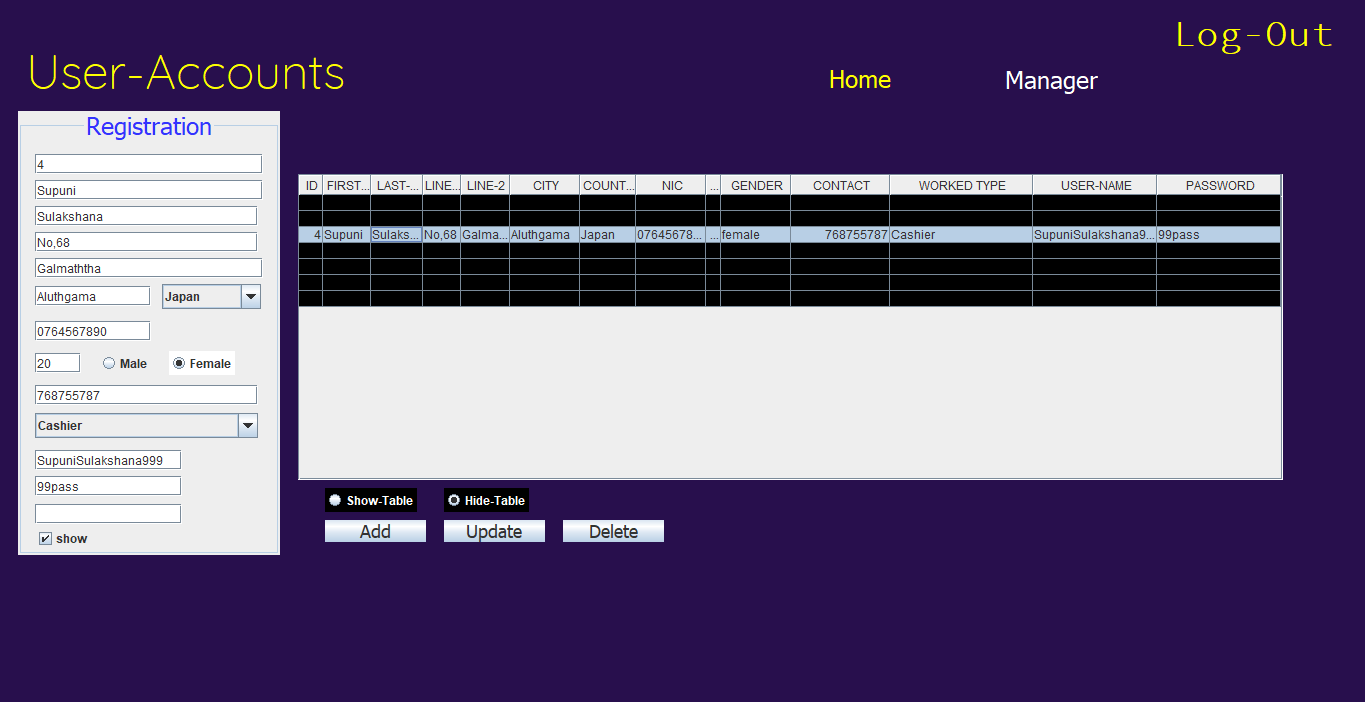




### User Registration Update

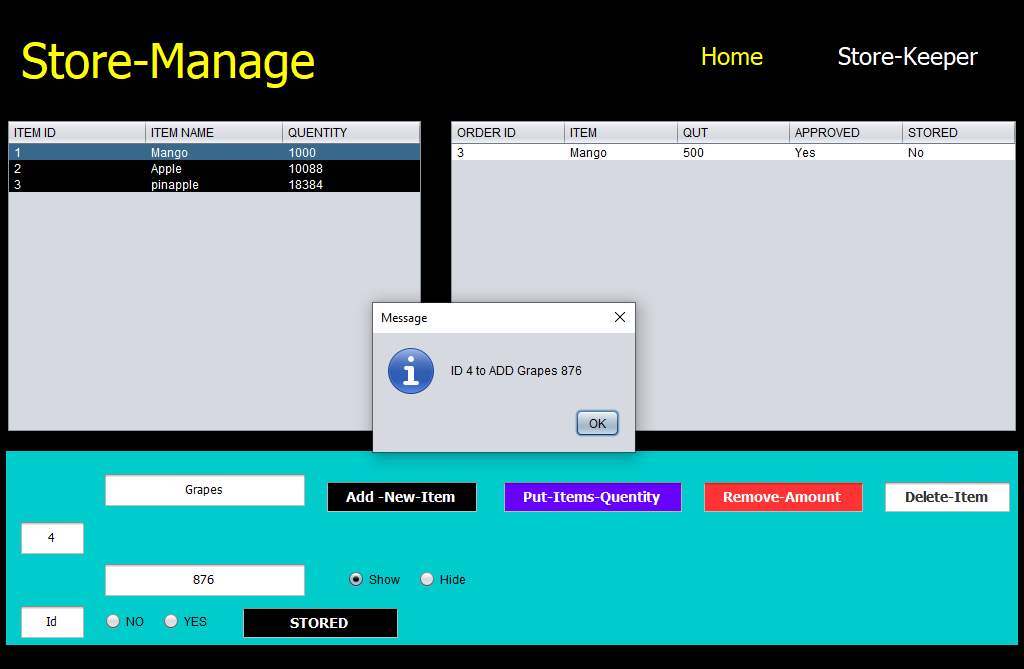
|  |  |
| --- | --- |
| **Test Case NO** | 01 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Update the User Registration data |
| **OBJECTIVE**  ID | 2 |
| **Test Data** | Click Table Row.  Enter Data below,   1. ID=“4” 2. First-Name= “Supuni” 3. Last-Name= “Sulakshana” 4. Address-Line 1= “No.68” 5. Address-Line 1= “Galmaththa” 6. City-Name= “Aluthgama” 7. Select country= “Japan” 8. NIC-Number= “0764567890” 9. Age= “20” 10. Contact-No= “0768755787” 11. Select Work type = “Cashier” 12. USER-NAME= “SupuniSulakshana999” 13. Password= “99pass” 14. Confirm password= “99pass”   Then Click Update Button |
| **Expected Results** | “Update successfully” message coming and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |





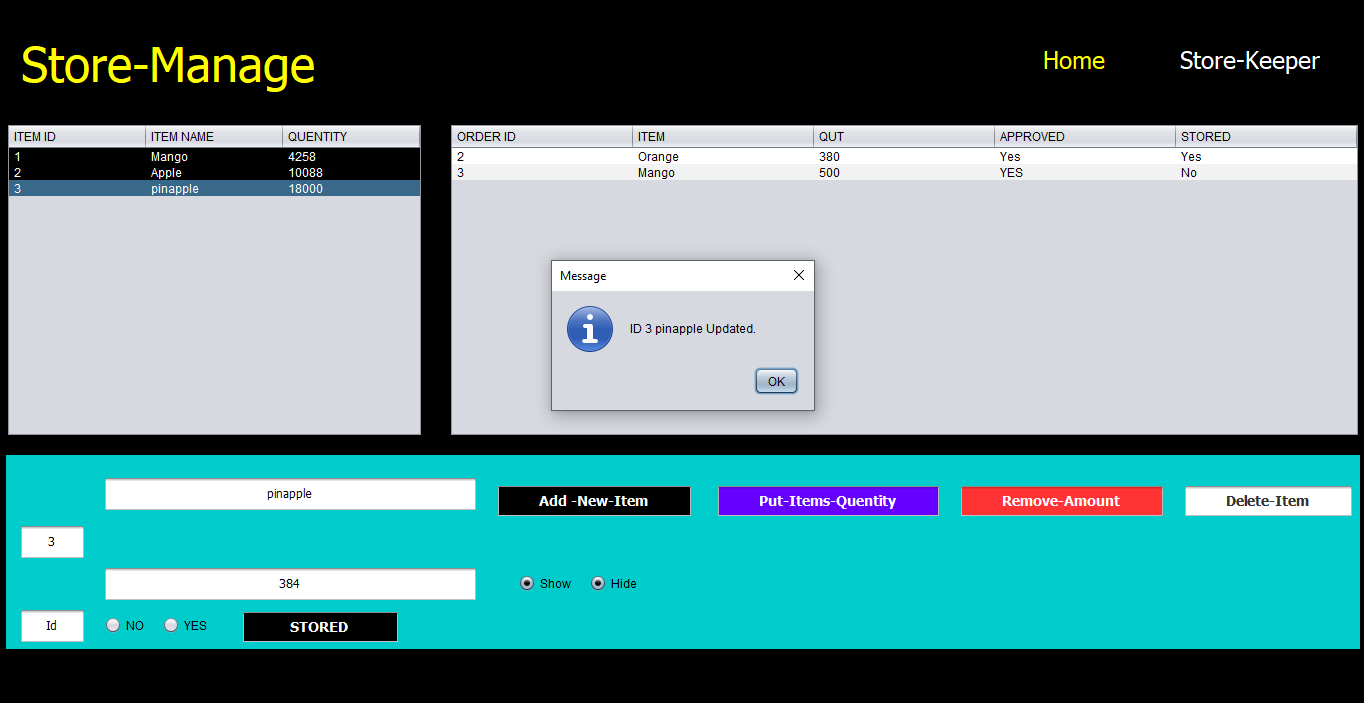
### Store-Item Entry

|  |  |
| --- | --- |
| **Test Case NO** | 06 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check ADD Items in the Store Manger |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | Enter Data below,   1. ID=“4” 2. Item-Name= “Grapes” 3. Quantity= “878”   Then Click ADD Button |
| **Expected Results** | “ID 4 to Add Grapes 876” message coming and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |



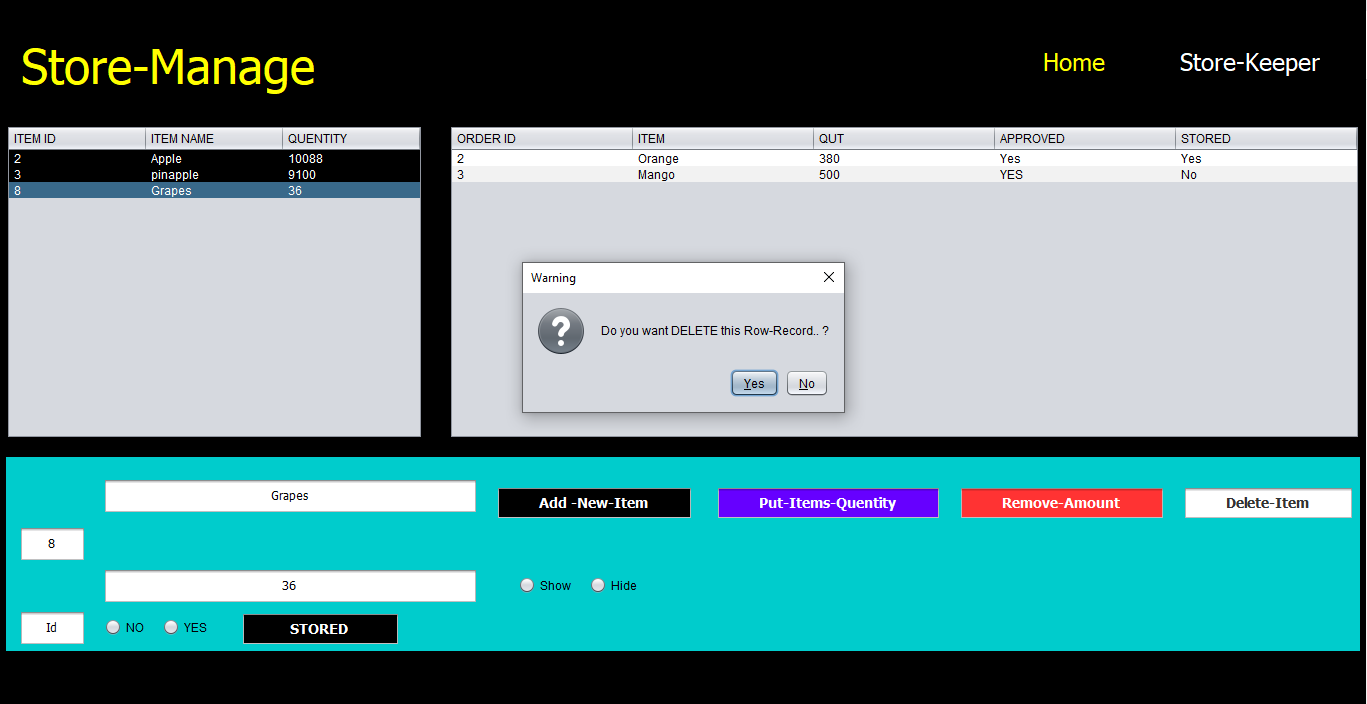
|  |  |
| --- | --- |
| **Test Case NO** | 06 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Update Items in the Store Manger |
| **OBJECTIVE**  ID | 2 |
| **Test Data** | Enter Data below,   1. ID=“3” 2. Item-Name= “Pineapple” 3. Quantity= “384”   Then Click Put-Item-Quantity Button |
| **Expected Results** | “ID 3 Pineapple Updated” message coming and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

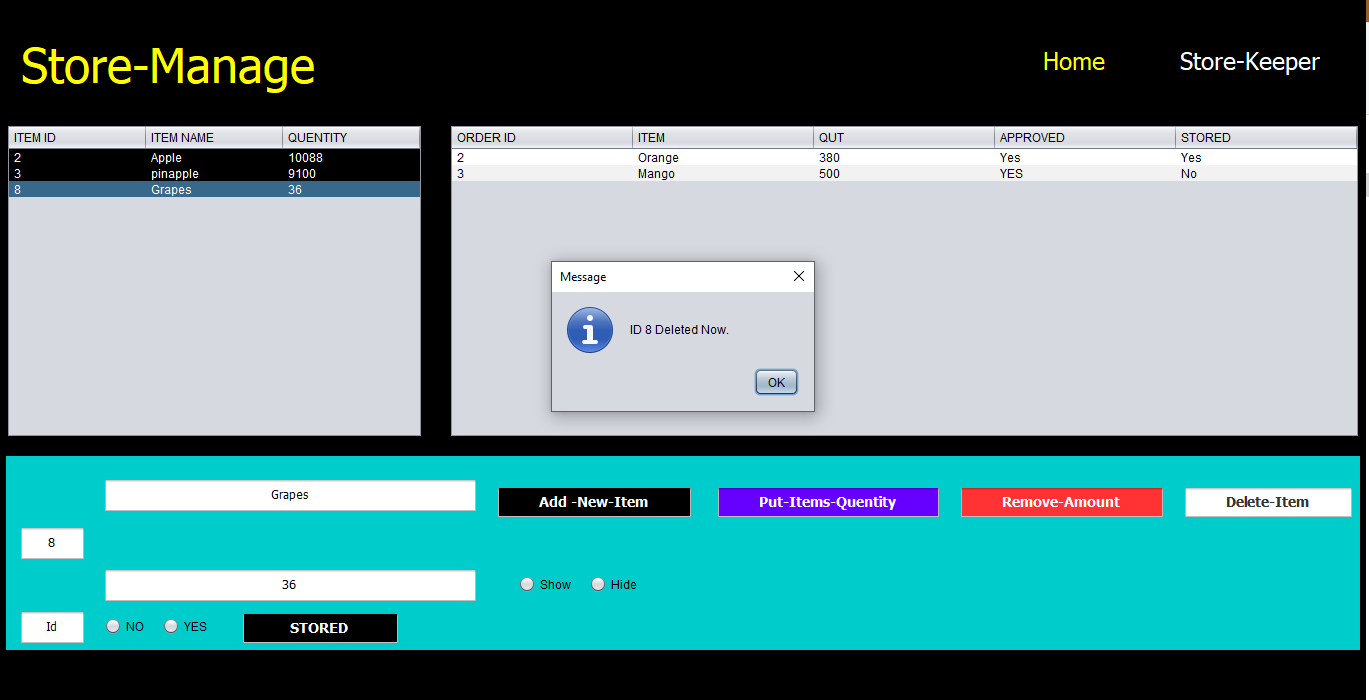
### Store-Item Update

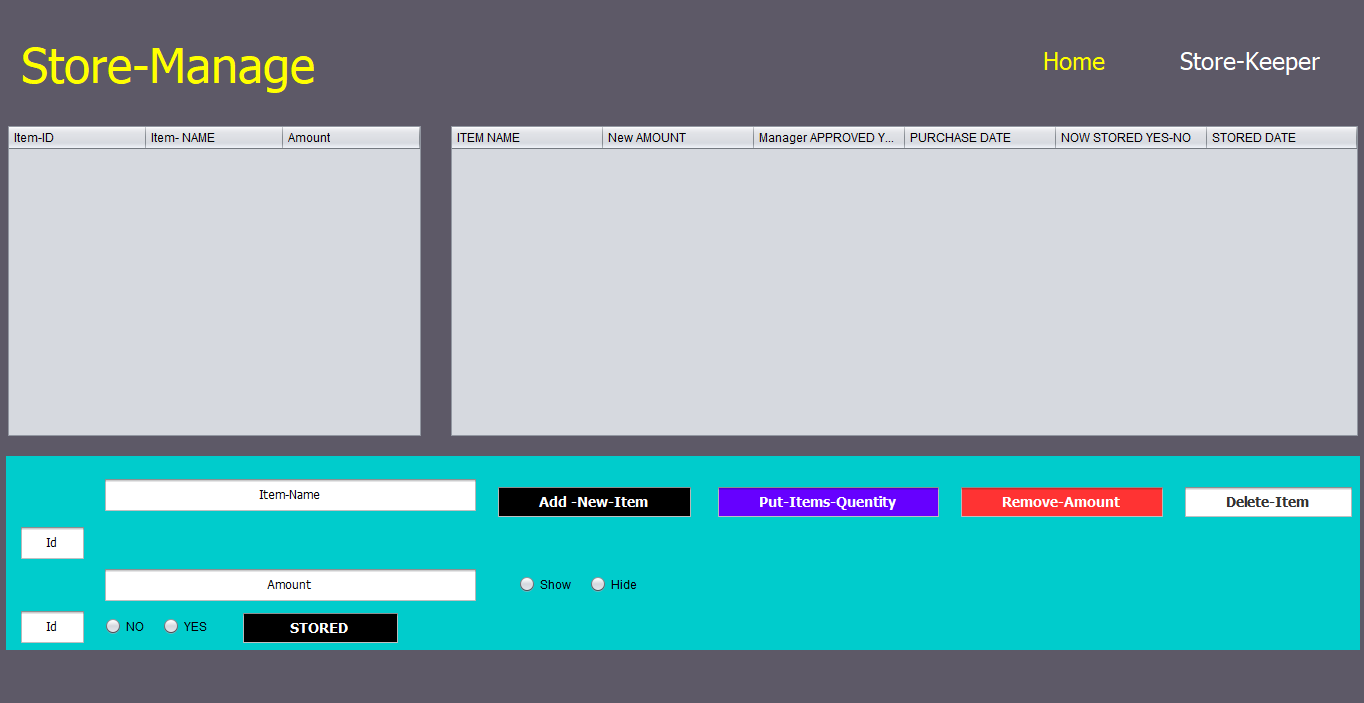


### Store-Item Delete

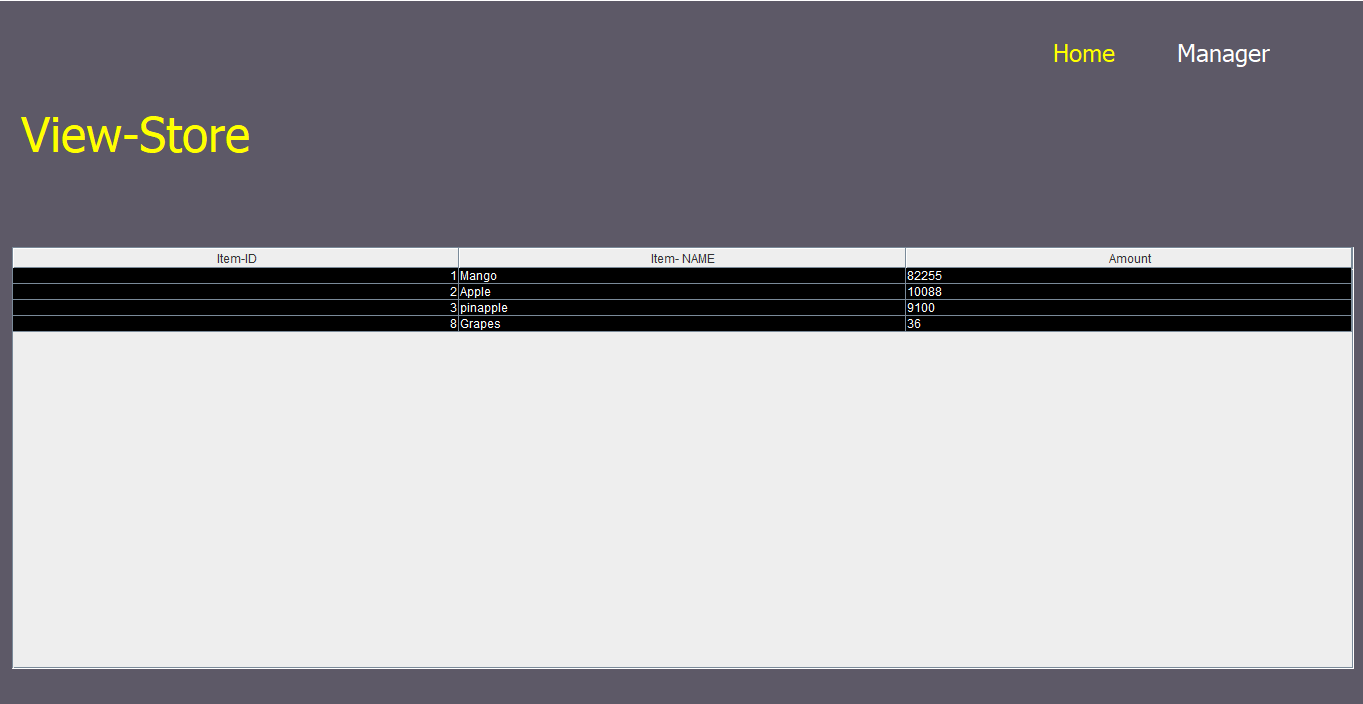
|  |  |
| --- | --- |
| **Test Case NO** | 06 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Store Item |
| **OBJECTIVE**  ID | 3 |
| **Test Data** | Click Table Row or ID = “8” enter Then Click Delete Button. |
| **Expected Results** | “Do you want to delete this row record?” message and view “deleted successful” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |





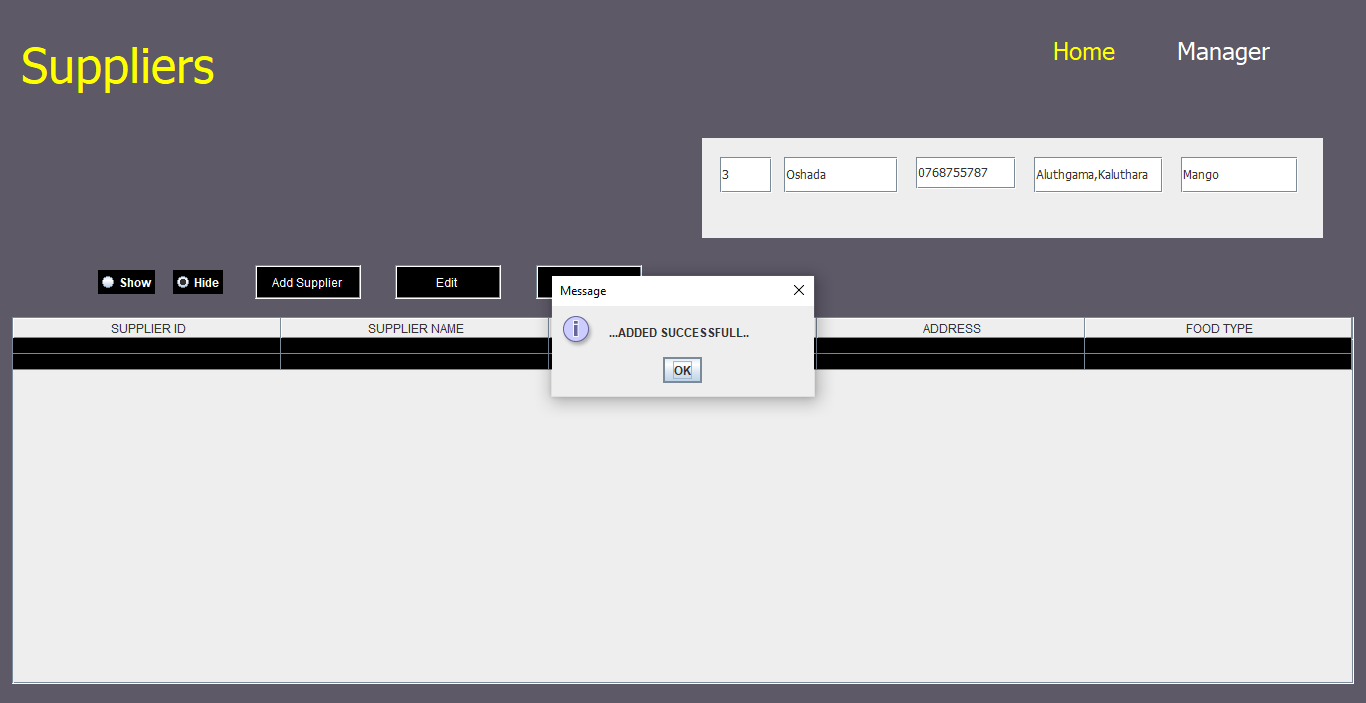


### Store view data



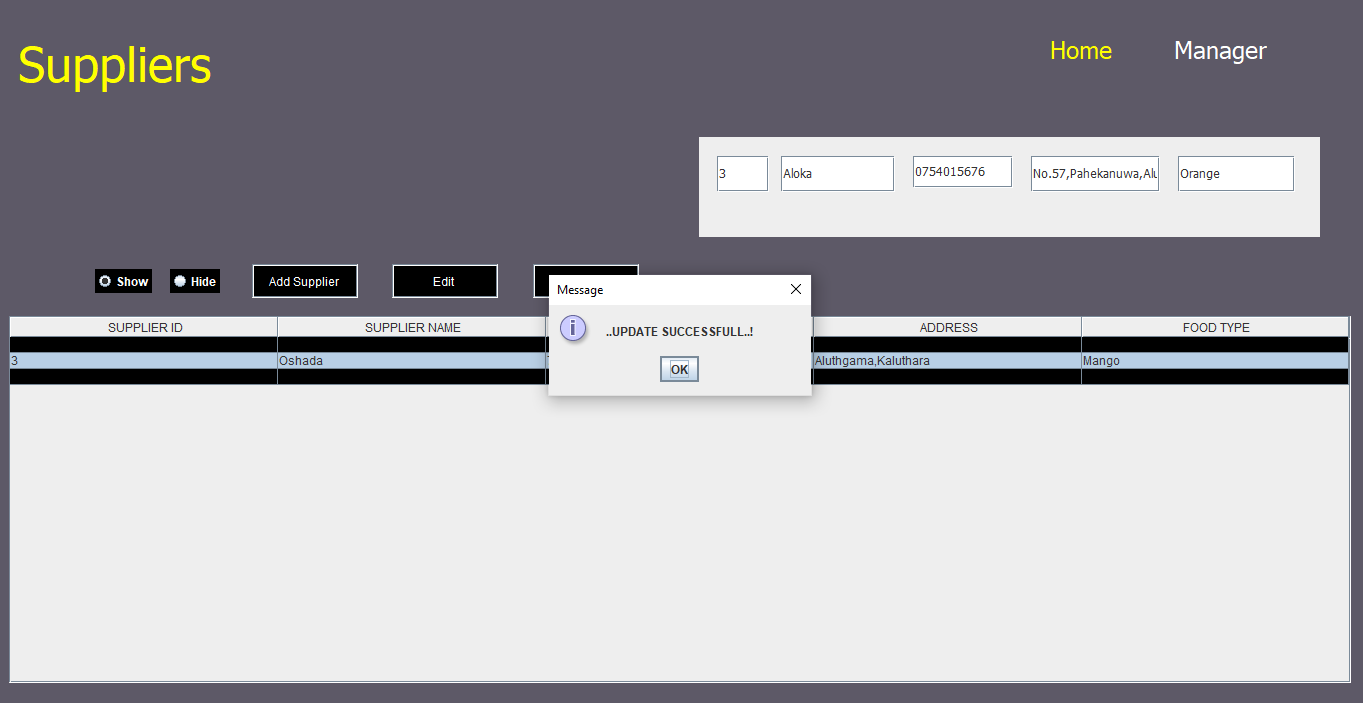
### Supplier Entry

|  |  |
| --- | --- |
| **Test Case NO** | 04 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Store Item |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. ID = “3” 2. Name = “Oshada” 3. Contact-No = “0768755787” 4. Address = “Aluthgama, Kalauthara” 5. Food-Type = “Mango”   Clicked added-supplier Button. |
| **Expected Results** | “Added Successfully” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |



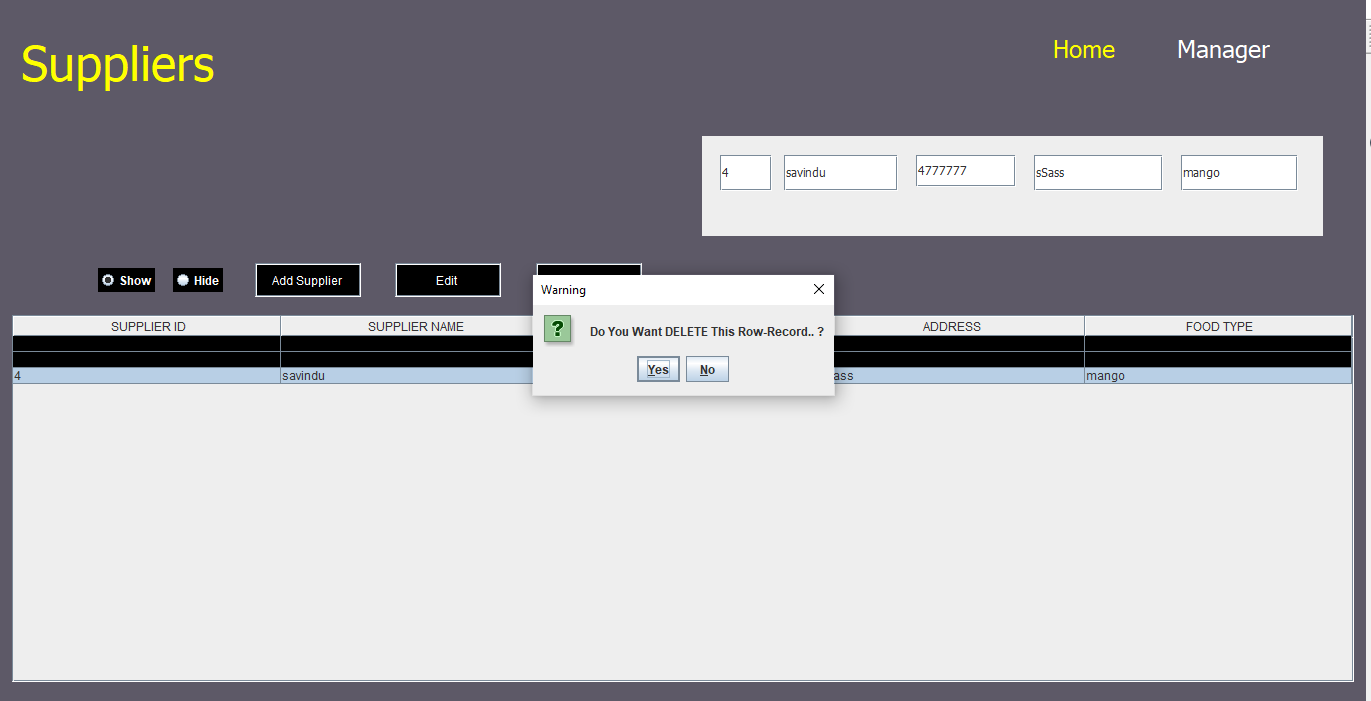
|  |  |
| --- | --- |
| **Test Case NO** | 04 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Store Item |
| **OBJECTIVE**  ID | 2 |
| **Test Data** | 1. ID = “3” 2. Name = “Aloka” 3. Contact-No = “0754015676” 4. Address = “No.57, Pahekanuwa,Aluthgama” 5. Food-Type = “Orange”   Clicked Edit button. |
| **Expected Results** | “Updated Successfully” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

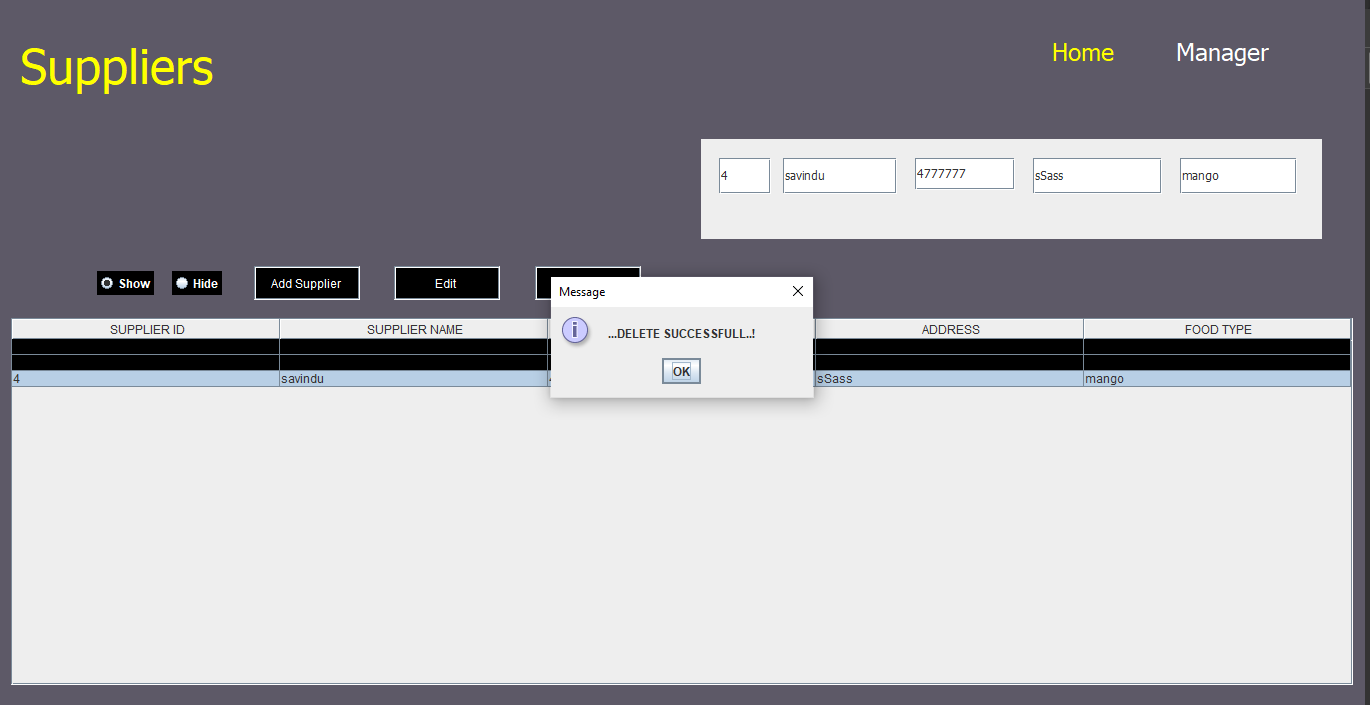
### Supplier Update

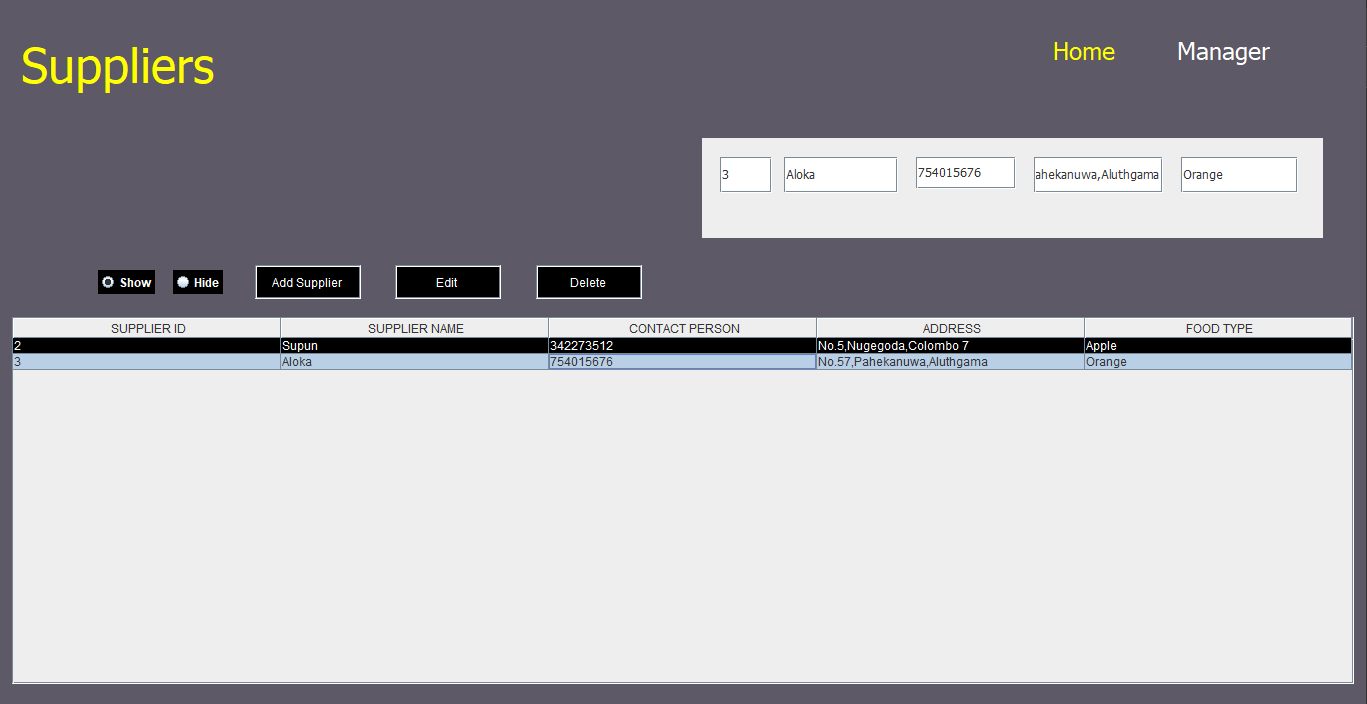


|  |  |
| --- | --- |
| **Test Case NO** | 04 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Store Item |
| **OBJECTIVE**  ID | 3 |
| **Test Data** | Click Table Row or ID = “4” enter Then Click Delete Button. |
| **Expected Results** | “Do you want to delete this row record?” message and view “deleted successful” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

### Supplier Delete

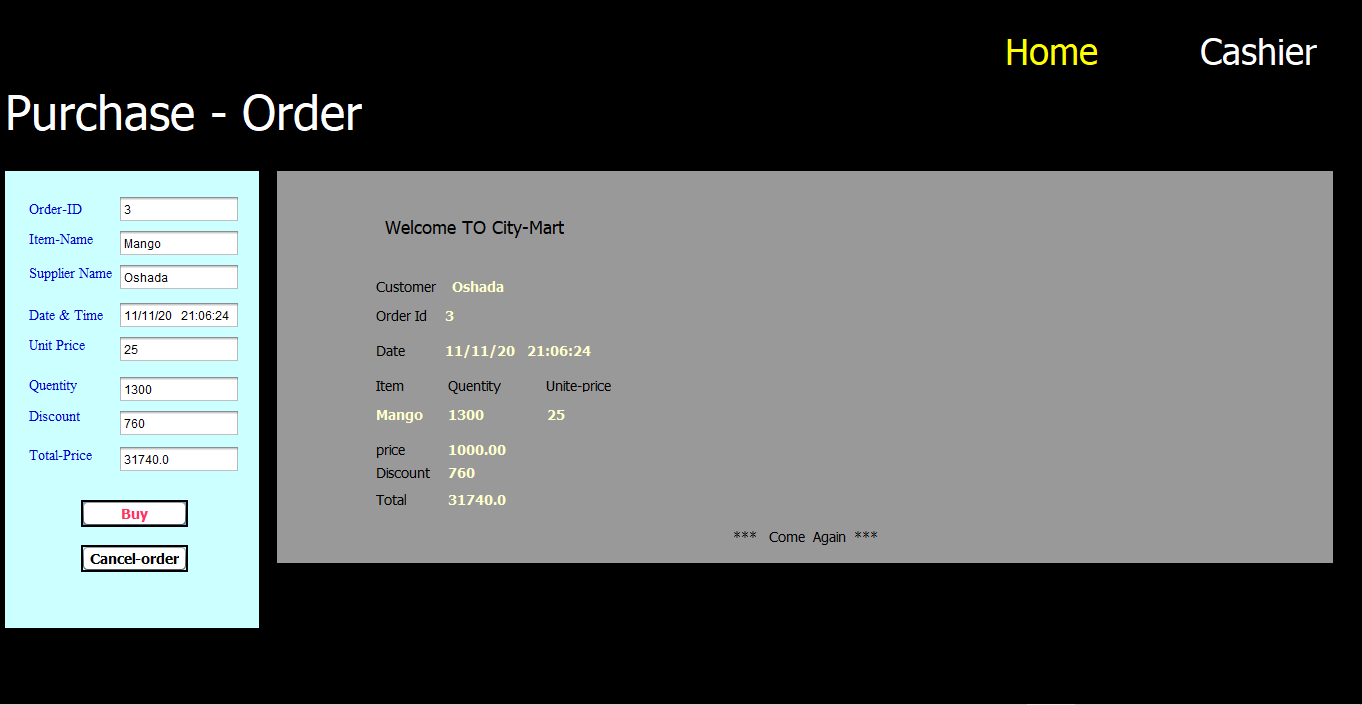






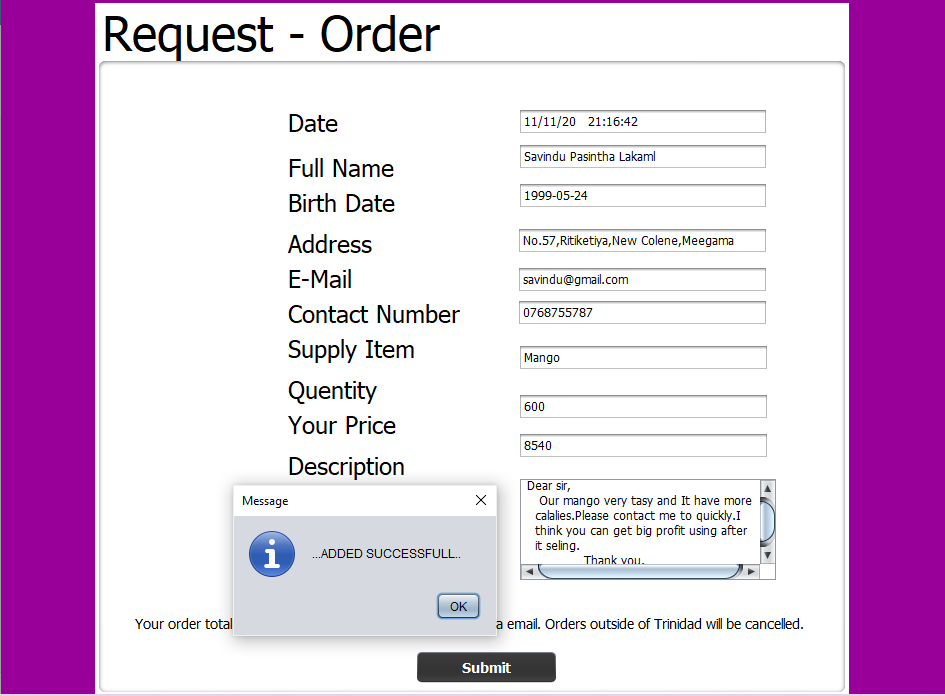
### Generate Purchase Order (PO)

|  |  |
| --- | --- |
| **Test Case NO** | 05 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Generate bill and Purchase order |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. Order-ID=“3” 2. Item-Name= “Mango” 3. Supplier-Name= “Oshada” 4. Date and Time = “” 5. Unit-Price = “25” 6. Quantity = “1300” 7. Discount = “760” 8. Total = “”   Clicked Buy Button. |
| **Expected Results** | Automatically Added “Total” and Bill print and clear the all text field’s values. |
| **Actual Results** | Expected result Successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

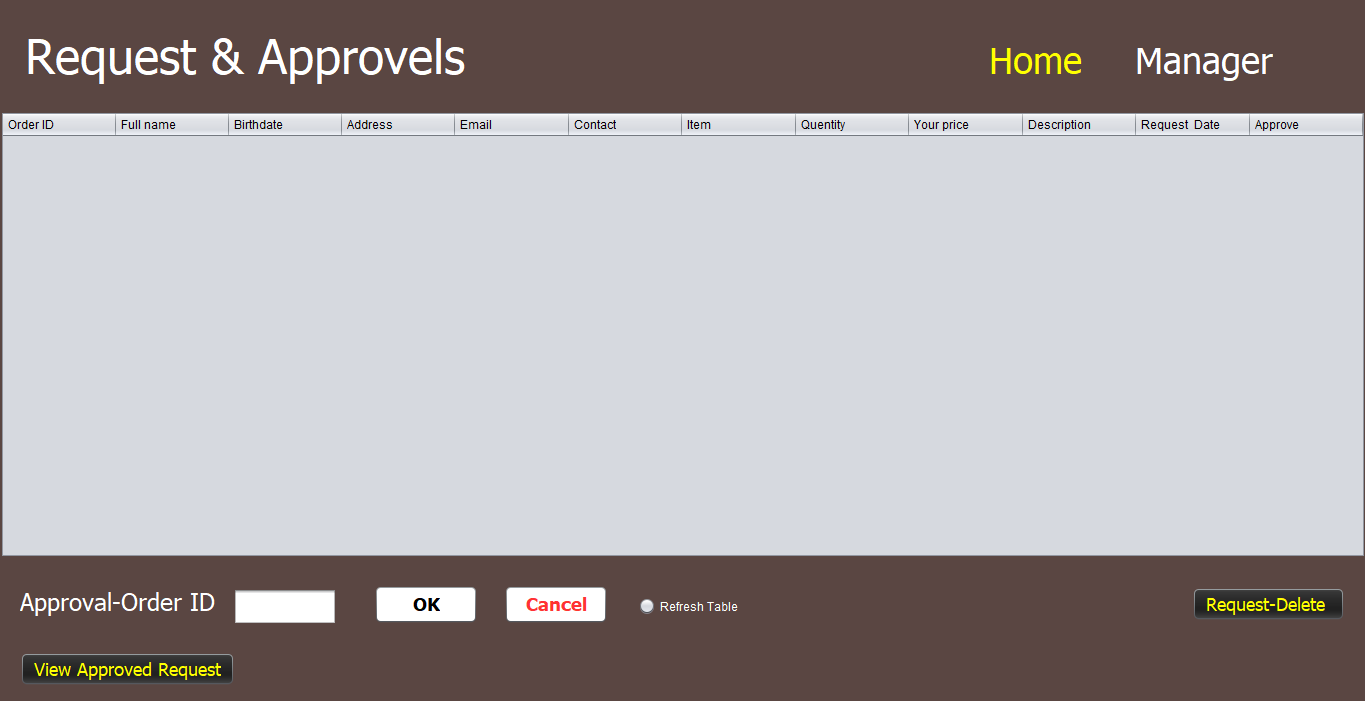


### Create a Purchase Requisition (PR)

|  |  |
| --- | --- |
| **Test Case NO** | 03 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | User Account Registration |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | 1. Date=“11/11/20 21:16:42” 2. Full Name= “Savindu Pasingtha Lakml” 3. Birth Date = “” 4. Address= “pasingtha” 5. E-Mail = “savindu@gmail.com” 6. Contact No = “0768755787” 7. Supply Item = “Mango” 8. Quantity = “600” 9. Your Price = “8540” 10. Description = “Dear sir, our mango is very tasty and it have more calories. Please contact me quickly and I think you can get big profit after selling. Thank you.”   Enter submit button. |
| **Expected Results** | Added success full message view and clear the all text fields’ values as an empty. |
| **Actual Results** | Expected result Successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

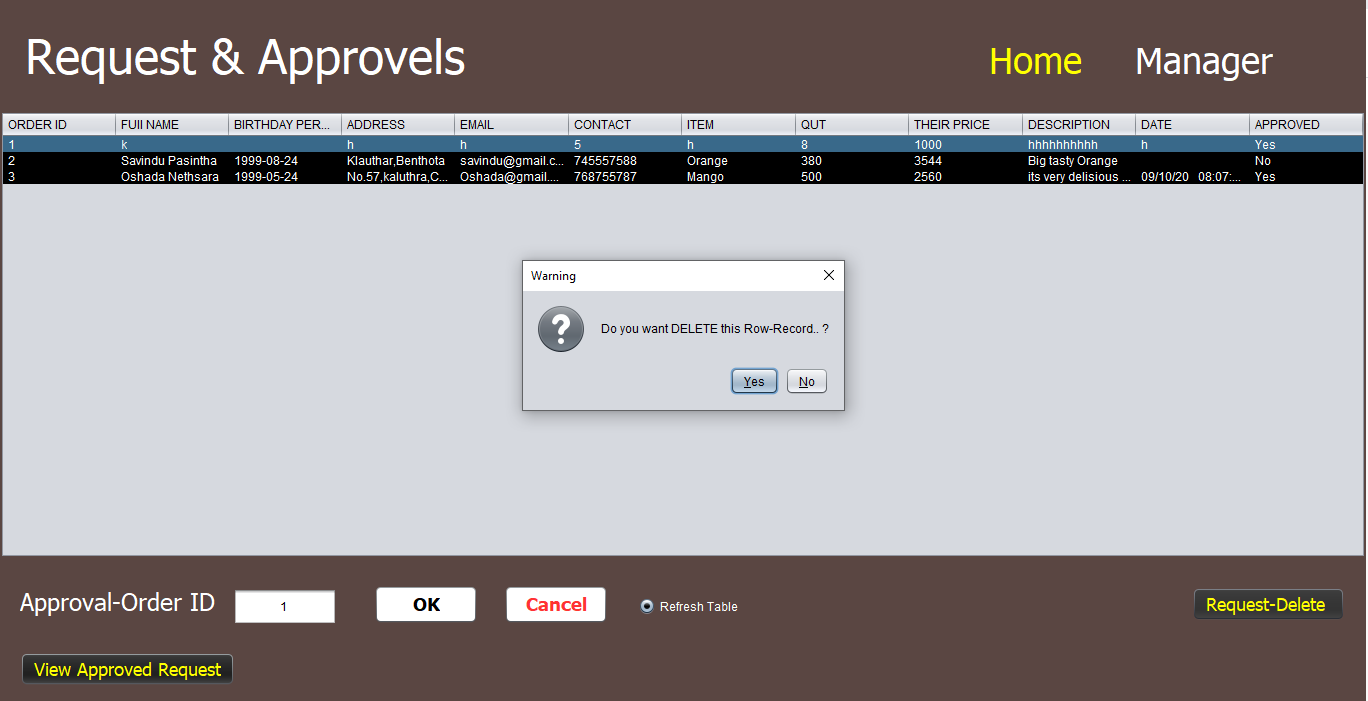


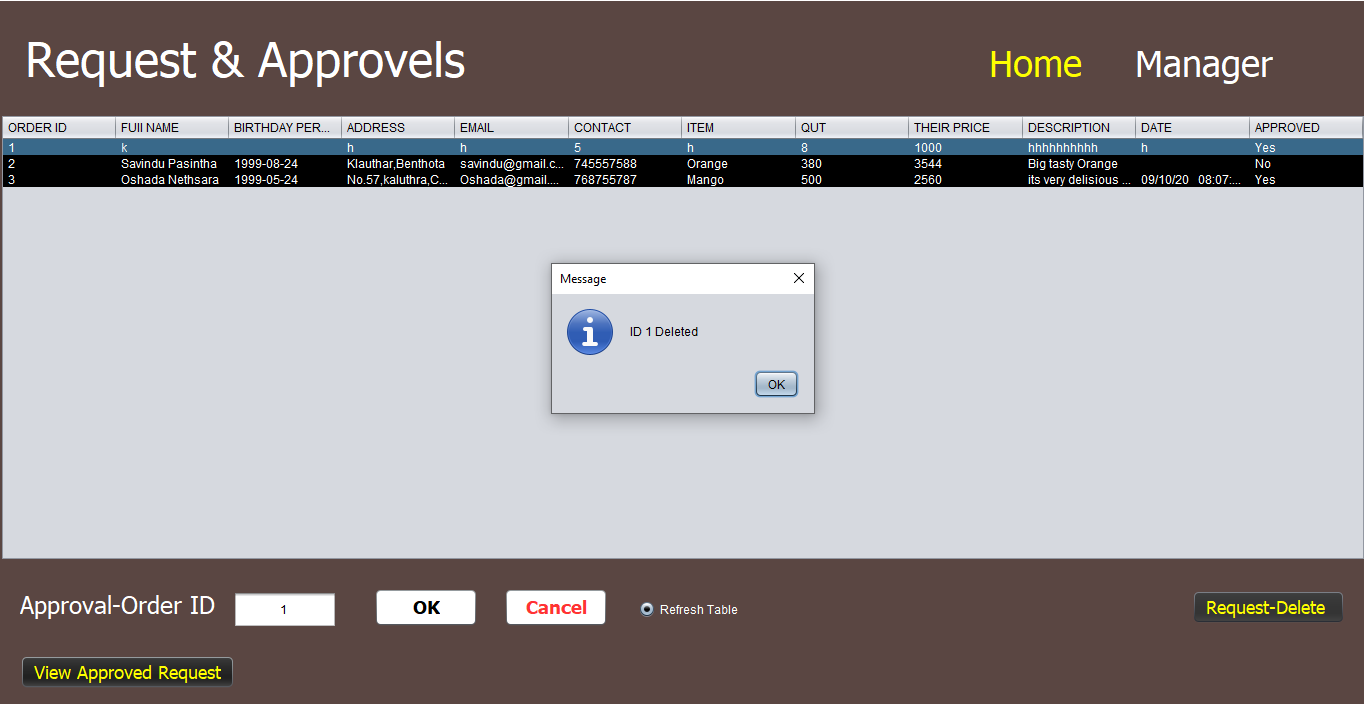
### Approve



### Request Delete

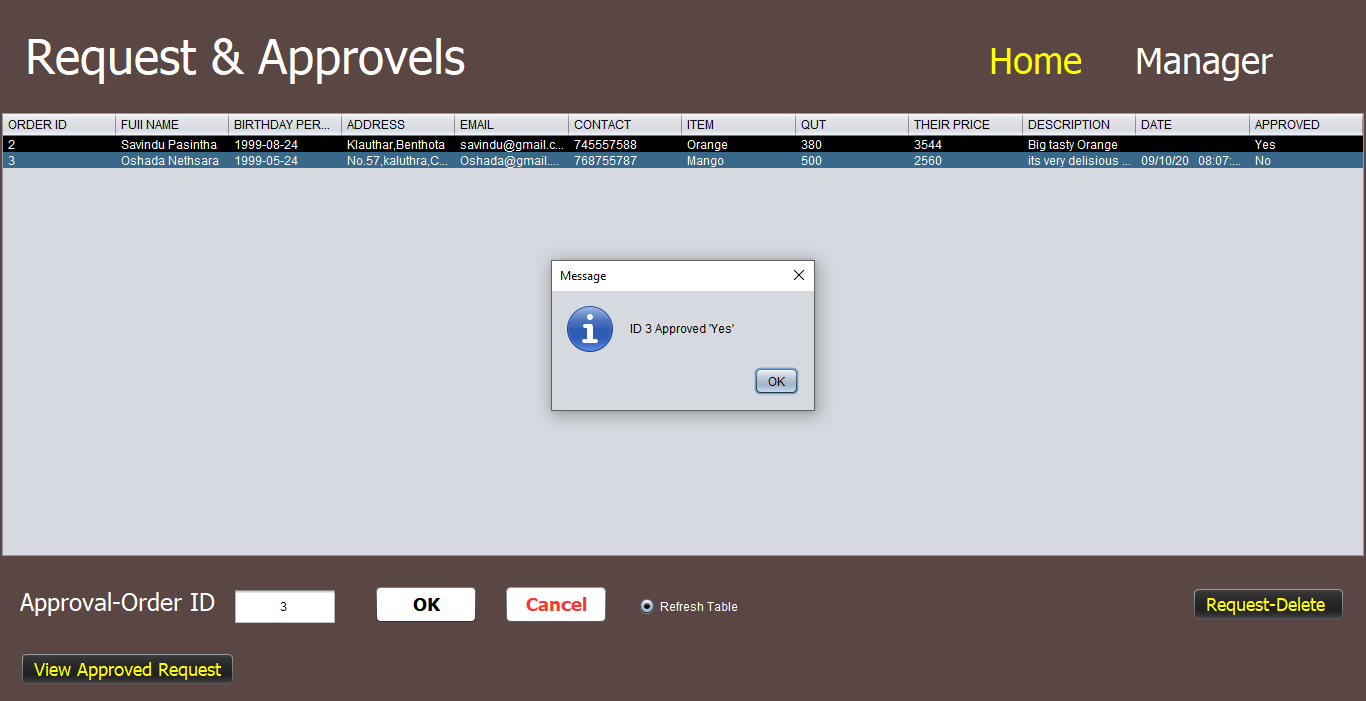
|  |  |
| --- | --- |
| **Test Case NO** | 02 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the Request |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | Click Table Row or ID = “1” enter Then Click Request-Delete Button. |
| **Expected Results** | “Do you want to delete this row record?” message and view “deleted successful” message and after table Refresh. |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |

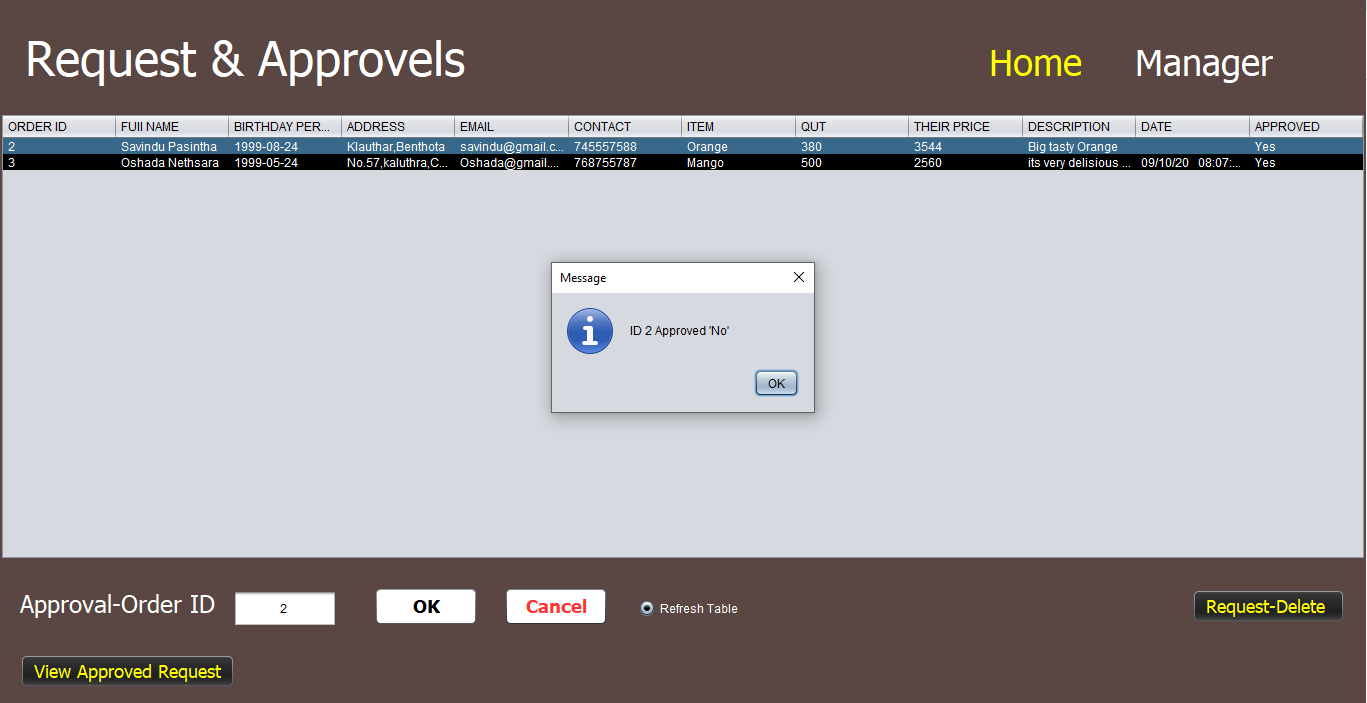


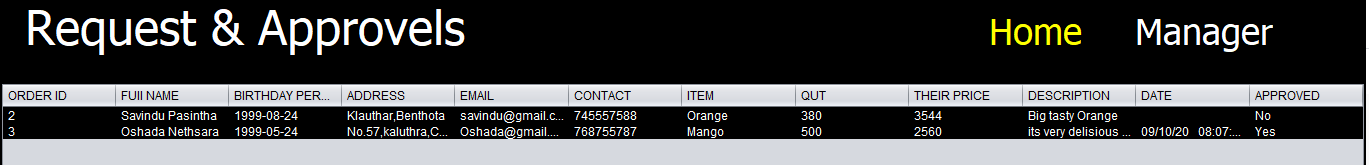


### Approve the Request

|  |  |
| --- | --- |
| **Test Case NO** | 04 |
| **TEST DATE** | 2020/11/11 |
| **Test OBJECTIVE** | Check Delete the User Store Item |
| **OBJECTIVE**  ID | 1 |
| **Test Data** | Click Table Row or ID = “3” enter Then Click “ok” and “cancel” Button. |
| **Expected Results** | When “ok” clicked updated as “YES”  When “cancel” clicked updated as “NO” |
| **Actual Results** | Expected result successfully. |
| **Pass/Fail** | Pass |
| **Conclusion** | Expected actual output. |







# CONCLUTION

This automation order management system to created “city mart”. We can identify main actors as manager, cashier, and store keeper in the system. This system allows difference functions as automation doing to this actors. Manager can view the sales, suppliers, customer requests, stock and he can delete requests and approve the requests. Not only has that he can creating and managing system user accounts and suppliers. Cashier allows generate orders and receive bill. Not only that I added new function as a show the approved request view page to when generating the order to sometime check request details. Stock keeper can manage stock and create purchase request to customers. This system have more facilities to user to easy. Tables show, hide and refresh functions allows some security of the system. It can hide what is the data in the Database. I used user like bright colors to decorate the system. Some labels and background clicked the mouse, I it to added change the background colors functions. This system using all the data automatically store to database in ‘MySQL’. I think this system very simple to handle any one. So “city mart” can increase their sales every day with handle the organic fresh foods. Any one enter the invalid data to system, then quickly popup the error messages. When create this system I used java programming language object oriented concepts with the “APACHE NETBEANS 12.0 IDE” to coding. I think this system can use any other order automation system, after little changes in the system. Not only that I think this system will be very useful to success their business.

# GANTT

# References

Anon., 2011. *Access Modifiers in Java.* [Online]   
Available at: https://www.javatpoint.com/access-modifiers  
[Accessed 1 November 2020].

Anon., 2011. *static keyword in java.* [Online]   
Available at: https://www.javatpoint.com/static-keyword-in-java  
[Accessed 1 November 2020].

Anon., 2020. [Online]   
Available at: https://www.javatpoint.com/method-overriding-in-java  
[Accessed 1 November 2020].

Anon., 2020. *abstract-class-in-java.* [Online]   
Available at: https://www.javatpoint.com/abstract-class-in-java  
[Accessed 1 November 2020].

Anon., 2020. *Java constructor.* [Online]   
Available at: https://www.javatpoint.com/java-constructor  
[Accessed 1 November 2020].

Anon., 2020. *Java Data Types.* [Online]   
Available at: https://www.w3schools.com/java/java\_data\_types.asp  
[Accessed 1 November 2020].

Anon., 2020. *Method Overloading in Java.* [Online]   
Available at: https://www.javatpoint.com/method-overloading-in-java  
[Accessed 1 November 2020].

Rouse, M., 2020. *object-oriented programming (OOP).* [Online]   
Available at: https://searchapparchitecture.techtarget.com/definition/object-oriented-programming-OOP  
[Accessed 1 November 2020].