

Assignment 4

AI-TE KUO

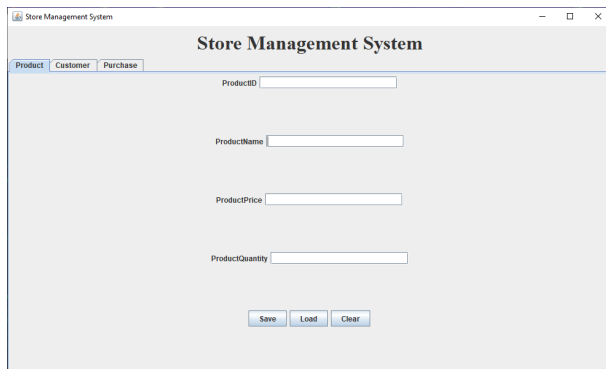
Oct 25, 2019

1. Rewrite two common use cases for each user story. Sketch the screens the system should display in each use case.
2. Design the screens (UI windows and widgets) the system should display in each use case.

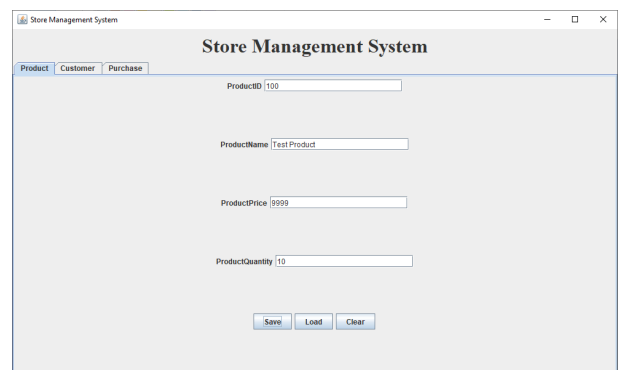
(a) Product

- i. Common case - When a new product is arrived, the system shall be able to provide a functionality that the user can add a new product to the system.

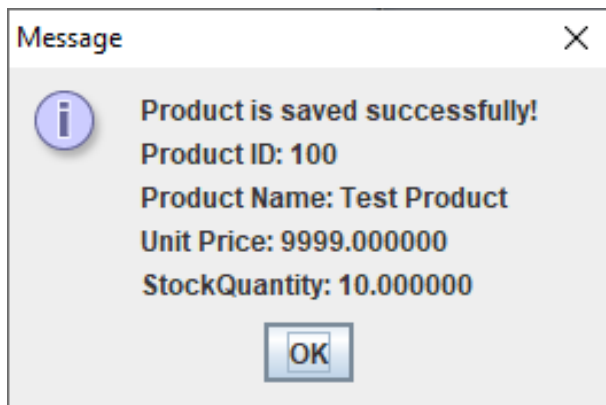
1. The user clicks on the product tab and the system shall display the add product page.



2. The user inputs all the product details.



3. The user clicks on "Save" button and the system shall pop out the message showing the product has been successfully saved.



- ii. Common case - It's approaching holidays and the manager would like to make some certain items on sale at 20% off, the system shall provide a functionality that the manager can change the price of items.

1. The user clicks on the product tab and the system shall display the add product page.

Store Management System

Product Customer Purchase

ProductID

ProductName

ProductPrice

ProductQuantity

Save Load Clear

2. The user types the product id.

Store Management System

Product Customer Purchase

ProductID 100

ProductName

ProductPrice

ProductQuantity

Save Load Clear

3. The user clicks on "Load" button and the system shall load all the corresponding item details with the given id.

Store Management System

Product Customer Purchase

ProductID 100

ProductName Test Product

ProductPrice 9999.0

ProductQuantity 10.0

Save Load Clear

4. The user types the product details to be changed.

Store Management System

Product Customer Purchase

ProductID 100

ProductName Test Product

ProductPrice 1000.0

ProductQuantity 10.0

Save Load Clear

5. The user clicks on "Save" button and the system shall pop out the message showing the product has been successfully edited.

Message

Product is edited successfully!

Product ID: 100

Product Name: Test Product

Unit Price: 1000.000000

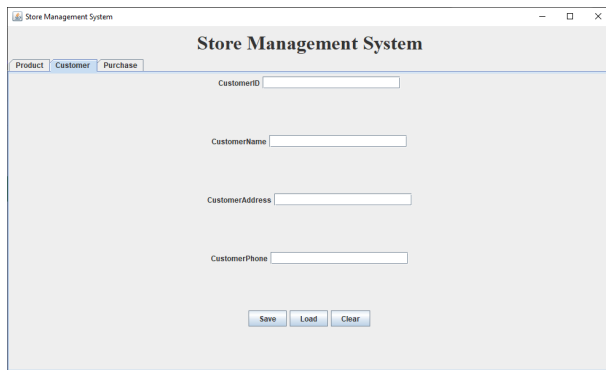
StockQuantity: 10.000000

OK

(b) Customer

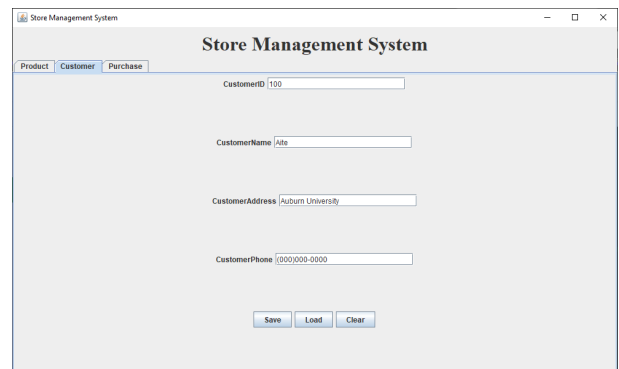
- i. Common case - When the customer wants to register a membership to get benefits at the customer service, the system shall provide a functionality that the employee can help the customer with their membership registration.

1. The user clicks on the customer tab and the system shall display the customer product page.



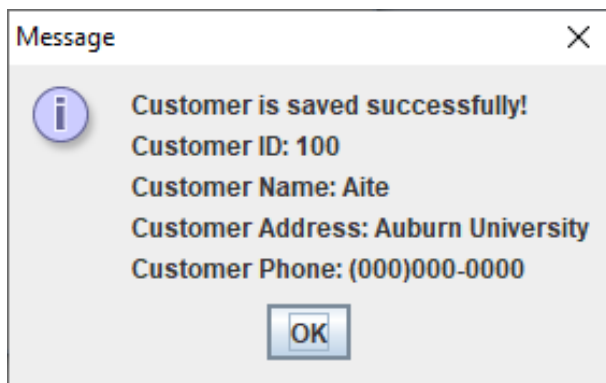
The screenshot shows a web application window titled "Store Management System". It has three tabs: "Product", "Customer", and "Purchase". The "Customer" tab is selected. The form contains four input fields: "CustomerID" (with the value "100"), "CustomerName" (with the value "Aite"), "CustomerAddress" (with the value "Auburn University"), and "CustomerPhone" (with the value "(000)000-0000"). At the bottom, there are three buttons: "Save", "Load", and "Clear".

2. The user inputs all the customer details.



This screenshot is identical to the previous one, showing the "Store Management System" window with the "Customer" tab selected. The input fields are filled with the same data: "CustomerID" is "100", "CustomerName" is "Aite", "CustomerAddress" is "Auburn University", and "CustomerPhone" is "(000)000-0000". The "Save", "Load", and "Clear" buttons are at the bottom.

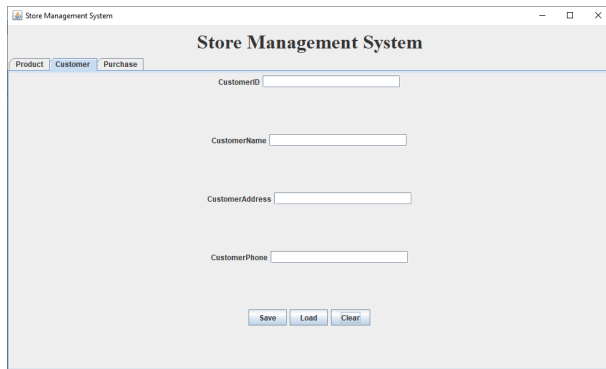
3. The user clicks on "Save" button and the system shall pop out the message showing the customer membership has been successfully registered.



The screenshot shows a "Message" dialog box with a close button (X) in the top right corner. It contains an information icon (i) and the following text: "Customer is saved successfully!", "Customer ID: 100", "Customer Name: Aite", "Customer Address: Auburn University", and "Customer Phone: (000)000-0000". At the bottom, there is an "OK" button.

- ii. Common case - When the customer recently changed his/her address and phone, the system shall provide a functionality that the customer can change their profile data after the registration.

1. The user clicks on the customer tab and the system shall display the add customer page.



Store Management System

Product Customer Purchase

CustomerID

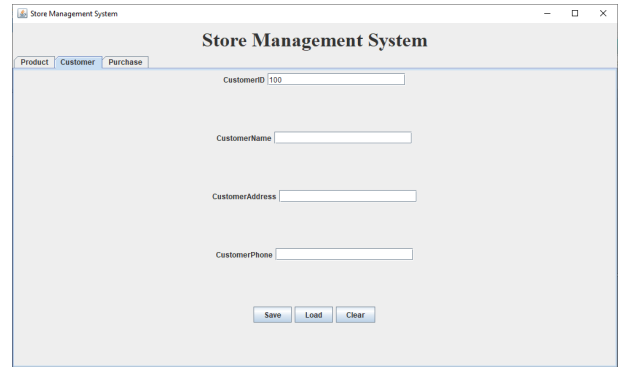
CustomerName

CustomerAddress

CustomerPhone

Save Load Clear

2. The user types the customer id.



Store Management System

Product Customer Purchase

CustomerID

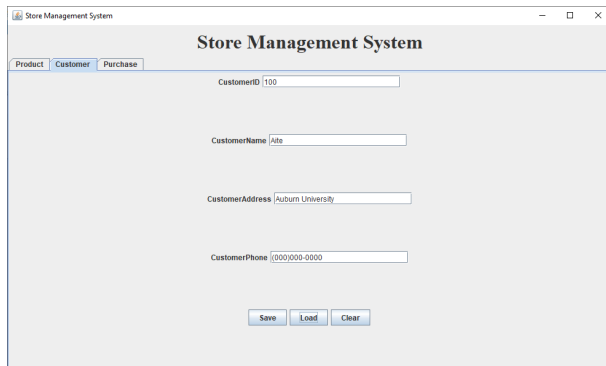
CustomerName

CustomerAddress

CustomerPhone

Save Load Clear

3. The user clicks on "Load" button and the system shall load all the corresponding customer details with the given id.



Store Management System

Product Customer Purchase

CustomerID

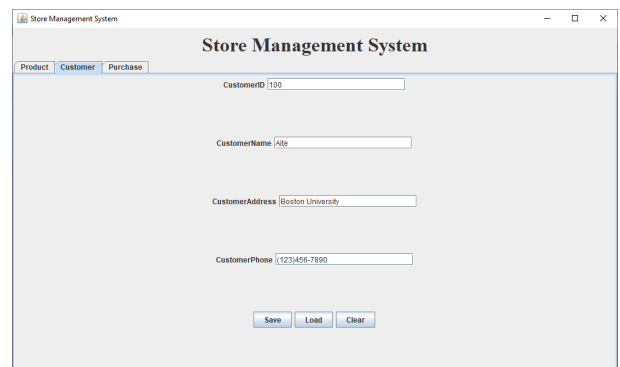
CustomerName

CustomerAddress

CustomerPhone

Save Load Clear

4. The user types the customer details to be changed.



Store Management System

Product Customer Purchase

CustomerID

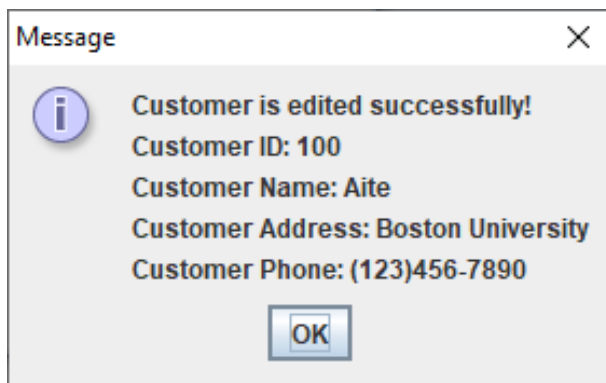
CustomerName

CustomerAddress


CustomerPhone

Save Load Clear

5. The user clicks on "Save" button and the system shall pop out the message showing the product has been successfully edited.



Message


 **Customer is edited successfully!**

Customer ID: 100

Customer Name: Aite

Customer Address: Boston University

Customer Phone: (123)456-7890



(c) Purchase

- i. Common case - When the user wants to add a purchase in the system, the system shall provide a functionality that the user can record the purchase and transaction history into the database.

1. The user clicks on the product tab and the system shall display the add product page.

Store Management System

Product Customer Purchase

Purchase Date: Fri Oct 25 20:50:55 CDT 2019

Purchase ID:

Customer ID:

Customer Name:

Product ID:

Product Name:

Product Price:

Purchase Quantity:

Cost:

Tax:

Total Cost:

Save Load Clear

2. When the user types the customer id, the system shall automatically display the corresponding customer name.

Store Management System

Product Customer Purchase

Purchase Date: Fri Oct 25 20:50:55 CDT 2019

Purchase ID:

Customer ID: 100

Customer Name: John

Product ID:

Product Name:

Product Price:

Purchase Quantity:

Cost:

Tax:

Total Cost:

Save Load Clear

3. When the user types the product id, the system shall automatically display the corresponding product name.

Store Management System

Product Customer Purchase

Purchase Date: Fri Oct 25 20:50:55 CDT 2019

Purchase ID: 100

Customer ID: 100

Customer Name: John

Product ID: 100

Product Name: Test Product

Product Price: 1000.0

Purchase Quantity:

Cost:

Tax:

Total Cost:

Save Load Clear

4. When the user types the purchase quantity, the system shall automatically show the cost, tax and total cost.

Store Management System

Product Customer Purchase

Purchase Date: Fri Oct 25 20:50:55 CDT 2019

Purchase ID: 100

Customer ID: 100

Customer Name: John

Product ID: 100

Product Name: Test Product

Product Price: 1000.0

Purchase Quantity: 1

Cost: 1000.0

Tax: 90.0

Total Cost: 1090.0

Save Load Clear

5. The user clicks on "Save" button and the system shall pop out the message showing the purchase is saved.

Message

Purchase is saved successfully!

Purchase ID: 100

Customer ID: 100

Product ID: 100

Cost: 1000.000000

Tax: 90.000000

TotalCost: 1090.000000

Price: 1000.000000

Quantity: 1.000000

Date: Fri Oct 25 20:50:55 CDT 2019

OK

- ii. Common case - When the user types in the wrong quantity in the system, the system shall provide a functionality that the user can edit afterwards.

1. The user clicks on the product tab and the system shall display the add purchase page.

The screenshot shows the 'Store Management System' window with the 'Purchase' tab selected. The form contains the following fields: Purchase Date (Fri Oct 25 21:13:27 CDT 2019), Purchase ID (100), Customer ID, Customer Name, Product ID, Product Name, Product Price, Purchase Quantity, Cost, Tax, and Total Cost. At the bottom right, there are 'Save', 'Load', and 'Clear' buttons.

2. The user types the purchase id.

The screenshot shows the 'Store Management System' window with the 'Purchase' tab selected. The 'Purchase ID' field now contains the value '100'. The other fields are empty. At the bottom right, there are 'Save', 'Load', and 'Clear' buttons.

3. The user clicks on "Load" button and the system shall load all the corresponding purchase details with the given id.

The screenshot shows the 'Store Management System' window with the 'Purchase' tab selected. The form is populated with the following data: Purchase Date (Fri Oct 25 20:50:55 CDT 2019), Purchase ID (100), Customer ID (100), Customer Name (N/A), Product ID (100), Product Name (Test Product), Product Price (100), Purchase Quantity (1.0), Cost (100.0), Tax (90.0), and Total Cost (1090.0). At the bottom right, there are 'Save', 'Load', and 'Clear' buttons.

4. The user types the purchase details to be changed.

The screenshot shows the 'Store Management System' window with the 'Purchase' tab selected. The form is populated with the following data: Purchase Date (Fri Oct 25 20:50:55 CDT 2019), Purchase ID (100), Customer ID (100), Customer Name (N/A), Product ID (100), Product Name (Test Product), Product Price (100), Purchase Quantity (2), Cost (100.0), Tax (90.0), and Total Cost (1090.0). At the bottom right, there are 'Save', 'Load', and 'Clear' buttons.

5. The user clicks on "Save" button and the system shall pop out the message showing the purchase has been successfully edited.

The screenshot shows a 'Message' dialog box with a close button (X) in the top right corner. It contains an information icon (i) and the following text: 'Purchase is edited successfully!', 'Purchase ID: 100', 'Customer ID: 100', 'Product ID: 100', 'Cost: 2000.000000', 'Tax: 180.000000', 'TotalCost: 2180.000000', 'Price: 100.000000', 'Quantity: 2.000000', and 'Date: Fri Oct 25 20:50:55 CDT 2019'. At the bottom, there is an 'OK' button.

3. Describe the protocol for two sides: client and server.

Answer:

The client encodes the request code and data in a JSON format and sends the JSON text to the server. After the server receives the data, the server starts to process the request. It first decodes a JSON string, obtains the request code, delegates the request to the corresponding method according to its code, and finally gets the returned result. Afterwards, the server sends back the answer to the client including an error code (status code) and the response data. The client then is able to handle the response according to its error code and data.

Video Demo: https://www.youtube.com/watch?v=RdtJq7_CYNw