## Manager View – Save Product

User saves a product to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “Manage Product” button | Display “Manage Product” Screen |  |
| Fill in the all of the product information and click the “Save Product” button | A message will be displayed showing the action was successful | The client sends the server the populated ProductModel as data in a MessageModel with the PUT\_PRODUCT code through Gson. The server checks to see if the product id matches a preexisting product id. If it does, then the old product is deleted. The product is then inserted into the database and the server sends back confirmation if the insertion was a success. |
| Click the “OK” button | Display the Home Screen |  |

## Manager View – Sales Summary

User saves a product to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “View Sales Summary” button | Display “Sales Summary” Screen | The client requests a full list of purchases from the server with the code, GET\_FULL\_PURCHASE\_LIST. The server sends back a PurchaseListModel and is displayed. |

## Cashier View – Save Customer

User saves a customer to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “Manage Customer” button | Display “Manage Customer” Screen |  |
| Fill in the all of the customer information and click the “Save Customer” button | A message will be displayed showing the action was successful | The client sends the server the populated CustomerModel as data in a MessageModel with the PUT\_CUSTOMER code through Gson. The server checks to see if the customer id matches a preexisting customer id. If it does, then the old customer is deleted. The customer is then inserted into the database and the server sends back confirmation if the insertion was a success. |
| Click the “OK” button | Display the Home Screen |  |

## Cashier View – Save Purchase

User saves a purchase to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “Manage Purchase” button | Display “Manage Purchase” Screen |  |
| Fill in the “Product ID” field | The “Product Name” and “Product Price” labels update | The client sends the server the product id and the server sends back the corresponding product from the database. If the product does not exist, the label will be filled accordingly: |
| Fill in the “Customer ID” field | The “Customer Name” label updates | The client sends the server the customer id and the server sends back the corresponding customer from the database. If the product does not exist, the label will be filled accordingly: |
| Fill in the “Quantity” field | The “Cost”, “Tax”, and “Total Cost” labels update    If there are not enough items in stock, a message displays:    If the quantity is invalid: |  |
| Fill in the “Purchase ID” field and click the “Save Purchase” button | Display the Home Screen | The client sends the server the populated PurchaseModel as data in a MessageModel with the PUT\_PURCHASE code through Gson. The server checks to see if the purchase id matches a preexisting purchase id. If it does, then the old purchase is deleted. The purchase is then inserted into the database and the server sends back confirmation if the insertion was a success. |

## Customer View – Add Purchase

User saves a purchase to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “Manage Purchase” button | Display “Manage Purchase” Screen |  |
| Fill in the “Product ID” field | The “Product Name” and “Product Price” labels update | The client sends the server the product id and the server sends back the corresponding product from the database. If the product does not exist, the label will be filled accordingly: |
| Fill in the “Customer ID” field | The “Customer Name” label updates | The client sends the server the customer id and the server sends back the corresponding customer from the database. If the product does not exist, the label will be filled accordingly: |
| Fill in the “Quantity” field | The “Cost”, “Tax”, and “Total Cost” labels update    If there are not enough items in stock, a message displays:    If the quantity is invalid: |  |
| Fill in the “Purchase ID” field and click the “Save Purchase” button | Display the Home Screen | The client sends the server the populated PurchaseModel as data in a MessageModel with the PUT\_PURCHASE code through Gson. The server checks to see if the purchase id matches a preexisting purchase id. If it does, then the old purchase is deleted. The purchase is then inserted into the database and the server sends back confirmation if the insertion was a success. |

## Customer View – Purchase History

User saves a product to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “View Sales Summary” button | Display “Sales Summary” Screen | The client requests a full list of purchases from the server with the code, GET\_PURCHASE\_LIST. The server sends back a PurchaseListModel and is displayed. |

## Customer View – Search Product

User saves a product to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “View Sales Summary” button | Display “Product Search” Screen |  |
| Fill in the information and click the “Search” button. | Display the found product information. | The client requests a full list of purchases from the server with the code, GET\_PRODUCT\_LIST. The server sends back a ProductListModel and is displayed. |

## User – Update Information

User saves a product to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Fill out the “New Password” and “New Full Name” fields and click the “Update” button. | Display “Updated Successfully” screen. | The client sends the server the UserModel with the PUT\_USER code. The user is put in the table with the new updated password or name. The server sends back a confirmation code. |

## Admin View – Manage User

User saves a customer to the database.

|  |  |  |
| --- | --- | --- |
| **Client** | **System** | **Server** |
| Click the “Manage User” button | Display “Manage User” Screen |  |
| Fill in the all of the user information and click the “Save User” button | A message will be displayed showing the action was successful | The client sends the server the populated UserModel as data in a MessageModel with the PUT\_USER code through Gson. The server checks to see if the username matches a preexisting username. If it does, then the old user is deleted. The user is then inserted into the database and the server sends back confirmation if the insertion was a success. |