README and Documentation

Abstract:

This document explains the various components of ProducePOS in great detail. Before using the software, we suggest you read this document to become accustomed to everything.

License:

Documentation:

1. Controller.java contains the event handlers which handle events occurring in the GUIs.

Fields – various input and output textfields, buttons and decimal formatting
handleLoginExit() - quits the application when invoked
handleLogin() - gets username and password from the input textfields, passes these values to SystemLogic.login(). If the username and password are found, the user is allowed access to the system. Else, the user is alerted with an error message.

handleScan() - gets product code from the input textfield, and passes this value to SystemLogic.scan(). If the code is found in the database, then the item name and price are retrieved from the SystemLogic fields and displayed. Else, an error message is displayed. Also invokes SystemLogic.calcSubTotal().

voidAll() - invokes SystemLogic.voidAll(), clears all textfields and disables the Void All and Finish & Pay buttons. Reason being, having buttons related to a sale enabled doesn't make sense when no transaction is in progress.
handleMainScreenExit() - closes the main screen
handleSaleScreen() - transition from the main screen to the sale screen.
handleToMain() - transition from the sale screen to the main screen
handleFinish() - invokes SystemLogic.pay(), gets values and outputs them on the textfields

quitError() - handles closure of the error form.
QuitBadCode() - handles closure of the bad PLU form
quitBadLogin() - handles closure of the bad login form
launchLookup() - transition from MainScreen to PLU lookup
handleLookupExit() - handles closure of the PLU lookup window
handleQuery() - Gets input query from the appropriate textfield, and passes this to SystemLogic.lookup(). If the query is found, item price and information is displayed in the output textfield. Else, the user is notified that their query was not found in the system.

2. SystemLogic.java contains the subroutines needed to connect the frontend to the backend.
Fields - hold monetary amounts and product names. We have some getter methods here.

getLookname() - returns the product name for the PLULookup window
getLookPrice() - returns the price of the item for the PLULookup window
getTotal() - returns the total amount owed
getSubTotal() - returns the subtotal.
getName() - returns the item name for sale mode

getItemPrice() - returns the item price for sale mode

login() - this method accepts a username and password and returns a boolean value if they are found in the Employee database. A connection is established with the database, and a SQL query is created which selects everything from the table within the database. A linear search algorithm cross checks each username and password in the database, with the ones passed to the method. If the username and password are found, the connection and query are closed, values are set and true is returned. Else, the connection and query are closed and false is returned.

scan() - this method accepts a PLU and returns a boolean value if they are found in the Product database. A connection is established with the database, and a SQL query is created which selects everything from the table within the database. A linear search algorithm cross checks each PLU in the database, with the one passed to the method. If the code is found, the connection and query are closed, values are set and true is returned. Else, the connection and query are closed and false is returned.

calcTotal() - sums the tax and subtotal, invokes calcTax() to calculate the tax

calcTax() - returns the value of the tax

calcSubTotal() - calculates the value of the sub total

pay() - calculates the change due, and handles any special cases involving amount tendered appropriately

lookup() - this method accepts a PLU and returns a boolean value if they are found in the Product database. A connection is established with the database, and a SQL query is created which selects everything from the table within the database. A linear search algorithm cross checks each PLU in the database, with the one passed to the method. If the code is found, the connection and query are closed, values are set and true is returned. Else, the connection and query are closed and false is returned. This subroutine works with the PLULookup window, and its associated fields.

3. Main.java will create a stage for login and provide entry into the program.
4. StartBadCode.java sets the stage for badlogin. The method loads the fxml file and creates a stage and shows it. And then it also includes the method which hides the stage.

5. StartError.java creates the stage for displaying the error window and includes the method which hides the stage.
6. StartLookup.java create stage for looking up PLU of items and includes the method which hides the stage.

7.StartMain.java creates the stage for Main screen for POS and includes the method which hides the stage.
8. StartScale.java will create a stage for sale screen and includes the method which hides the stage.

FXML:
BadLogin.fxml – markup for the bad login error message
ErrorWindow.fxml – markup for the general error message
InvalidCode.fxml – markup for the bad code error message
Login.fxml – markup for the login screen
PLULookup.fxml – markup for the PLULookup
MainScreen.fxml – markup for the main screen of the system
SaleScreen.fxml – markup for the POS screen

Instructions:
When the program is first launched, the landing screen is a login dialog. To login, there are six valid usernames and passwords contained in the EmployeeBase.db file. Enter your credentials and press login. The main screen should then launch (assuming valid credentials), giving you the option of selecting PLULookup or Sale modes. The PLULookup provides a quick way of seeing whether or not a product is in the system (handy for managers). Simply type in your query in the appropriate textfield, and press lookup. Item information and pricing will be displayed. If the query returns a miss, you will be notified. In the Sale window, there are four buttons; Scan, Void All, Finish & Pay and Quit. Scan will retrieve pricing, product information and calculate the subtotal. Void All is used to start a new sale. Finish & Pay initiates the payment process. To start a new sale, ensure that Void All and Finish & Pay are disabled. If this is the case, begin by entering in a PLU of an item. Press the scan button. The subtotal and product information will be displayed. Keep scanning items until everything has been entered. Next, press Finish & Pay. Note that scanning is not allowed at this point, as the scan button becomes disabled upon clicking. Tax and total will be calculated. You will be prompted to enter a value for amount tendered. Enter in a value, and press Finish & Pay once more. Change will be calculated and the sale will be over. Simply press Void All to start a new sale at any time. To Quit, press quit in whatever window you are in to go back to the main screen. Press quit in the main screen to return back to the login. Press quit in the login screen to exit the application.

When you run the Jar file, please ensure the databases are in the same directory as the Jar file. When opening in the IDE, you may need to add the SQLite - JDBC as a library. You will find it in the src directory as a Jar. Please ensure that the database files are stored in ProducePOS directory before running in the IDE.