**Criterion C**

Table of Contents

[Libraries Imported 2](#_Toc434439941)

[Login 2](#_Toc434439942)

[sqlConnection 2](#_Toc434439943)

[inventoryMenu 2](#_Toc434439944)

[employeeMenu 3](#_Toc434439945)

[SQL Commands 3](#_Toc434439946)

[Classes 5](#_Toc434439947)

[Login 5](#_Toc434439948)

[Node 6](#_Toc434439949)

[addClient 7](#_Toc434439950)

[LinkedList 8](#_Toc434439951)

[GUI Code 9](#_Toc434439952)

[Login 9](#_Toc434439953)

[inventoryMenu 10](#_Toc434439954)

[addClient 11](#_Toc434439955)

[Unit Testing 11](#_Toc434439956)

# Libraries Imported

## Login

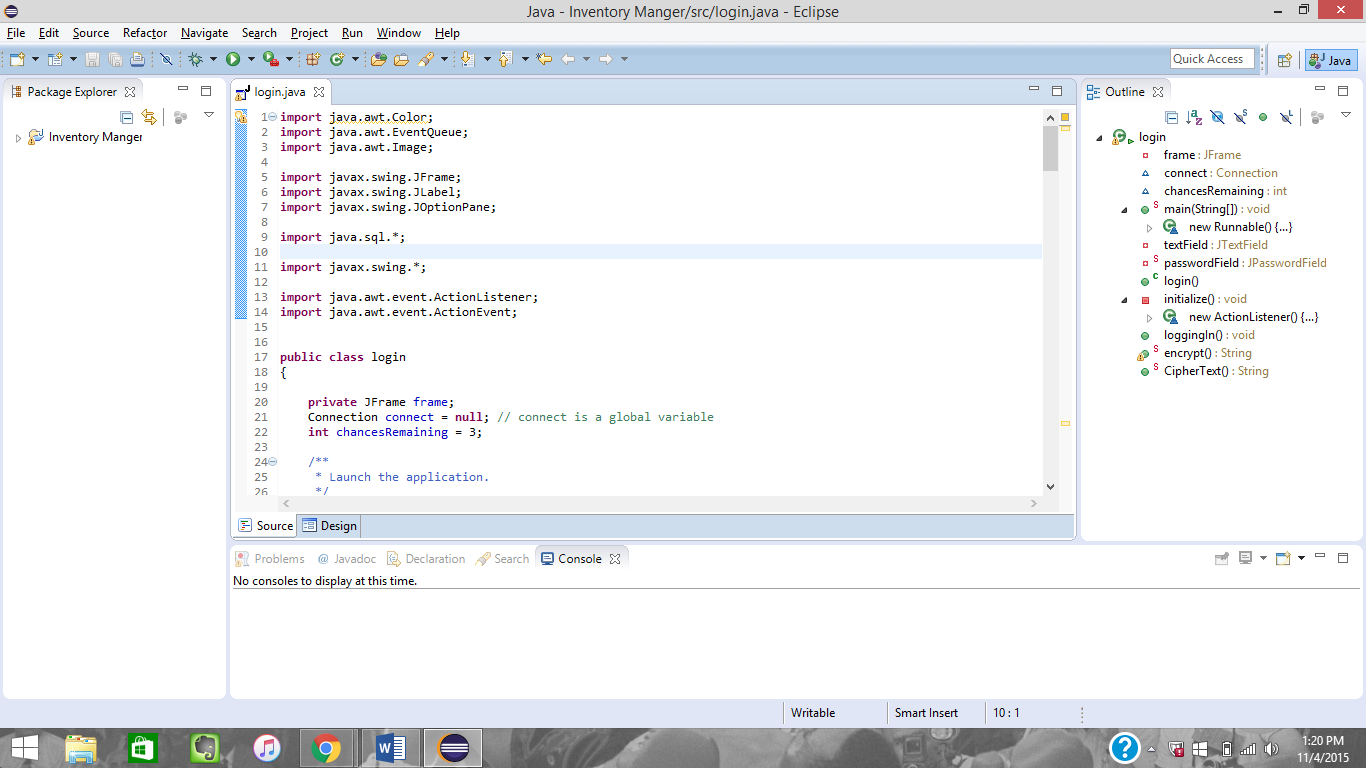


Figure - Libraries imported for the login class

The java.awt libraries imported were used to import images and set the color of the type that was in the JLabels. The swing libraries were used to create the frame and JLabel along with the JOptionPane is used for error messages. I imported the java.sql library to help establish the connection with the sql database.

## sqlConnection

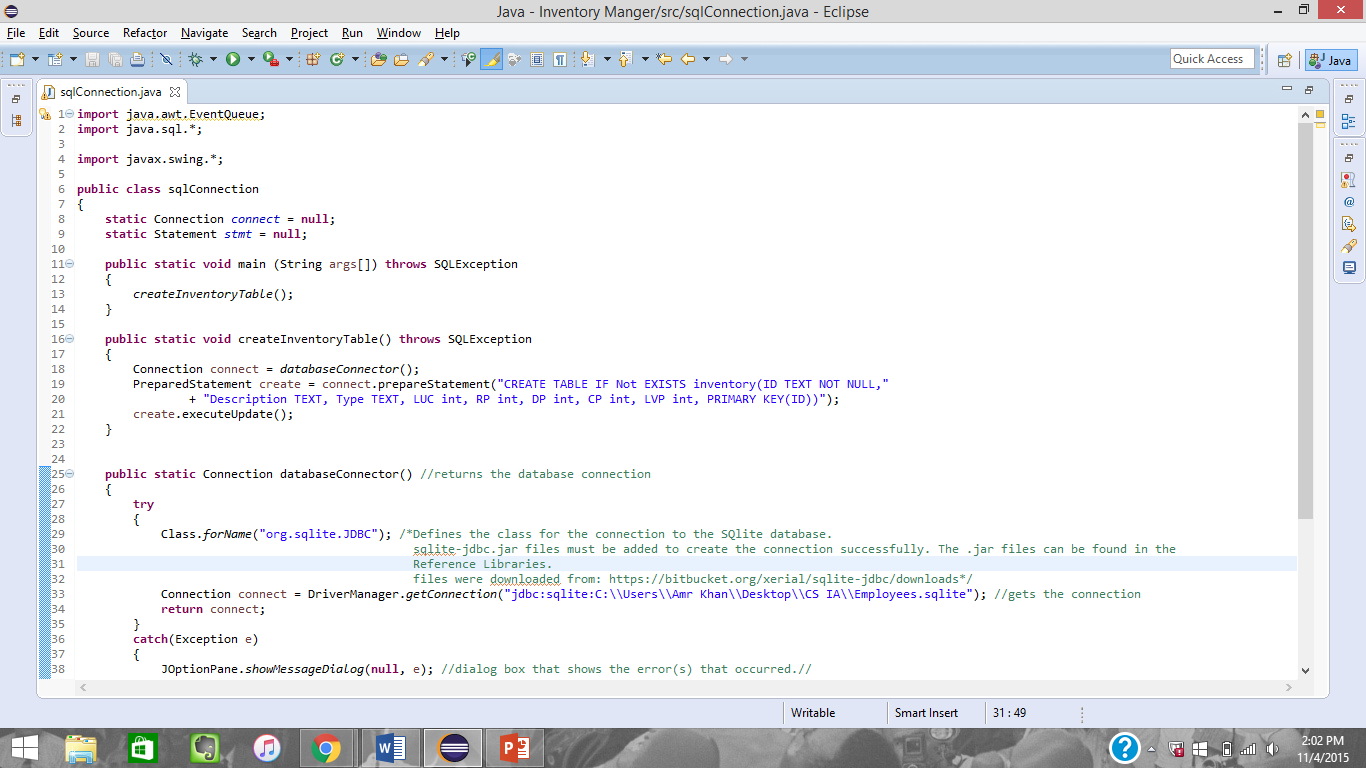


Figure - Libraries imported for the sqlConnection database

The sql connection is needed to establish the connection with the SQLite database manager.

## inventoryMenu

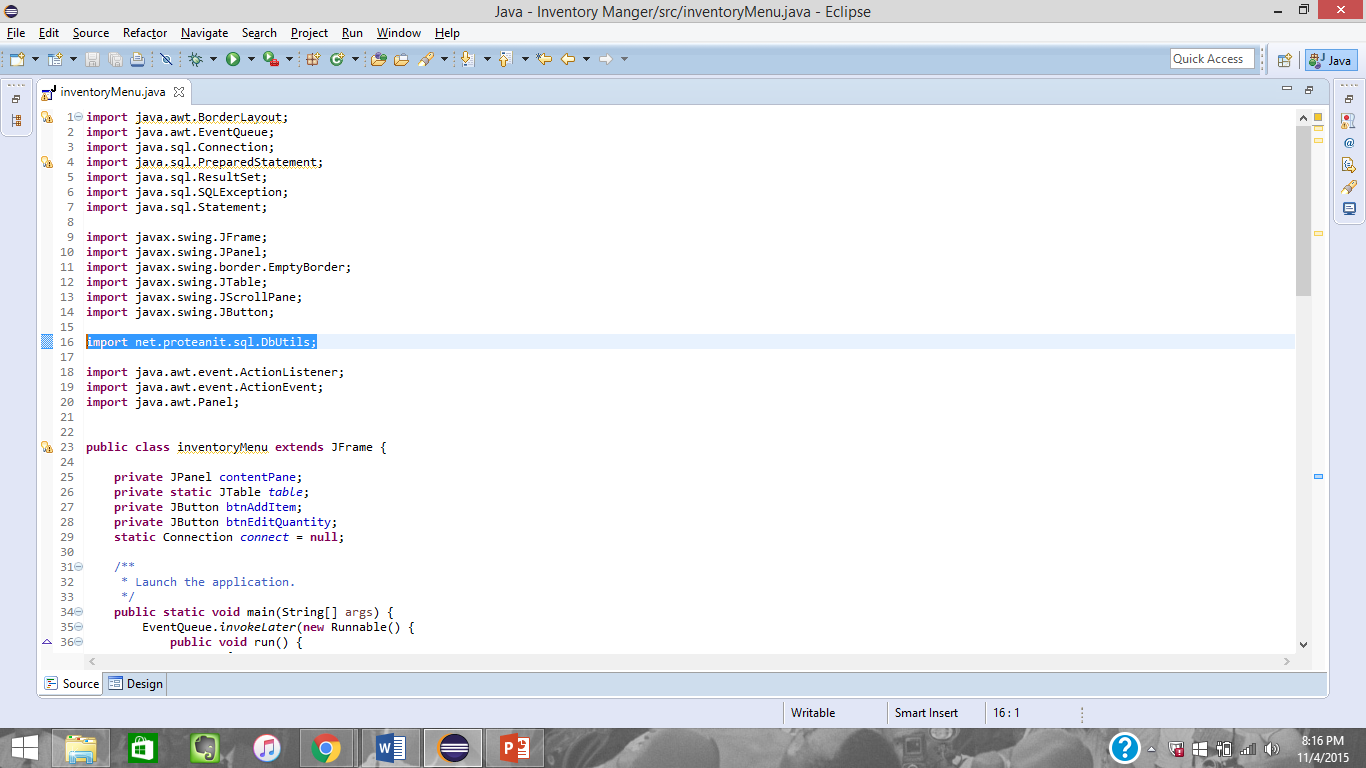


Figure - Libraries imported to create the inventory menu screen

The library that has been highlighted is used to format the table and automatically fill in the table from the SQL database.

## employeeMenu

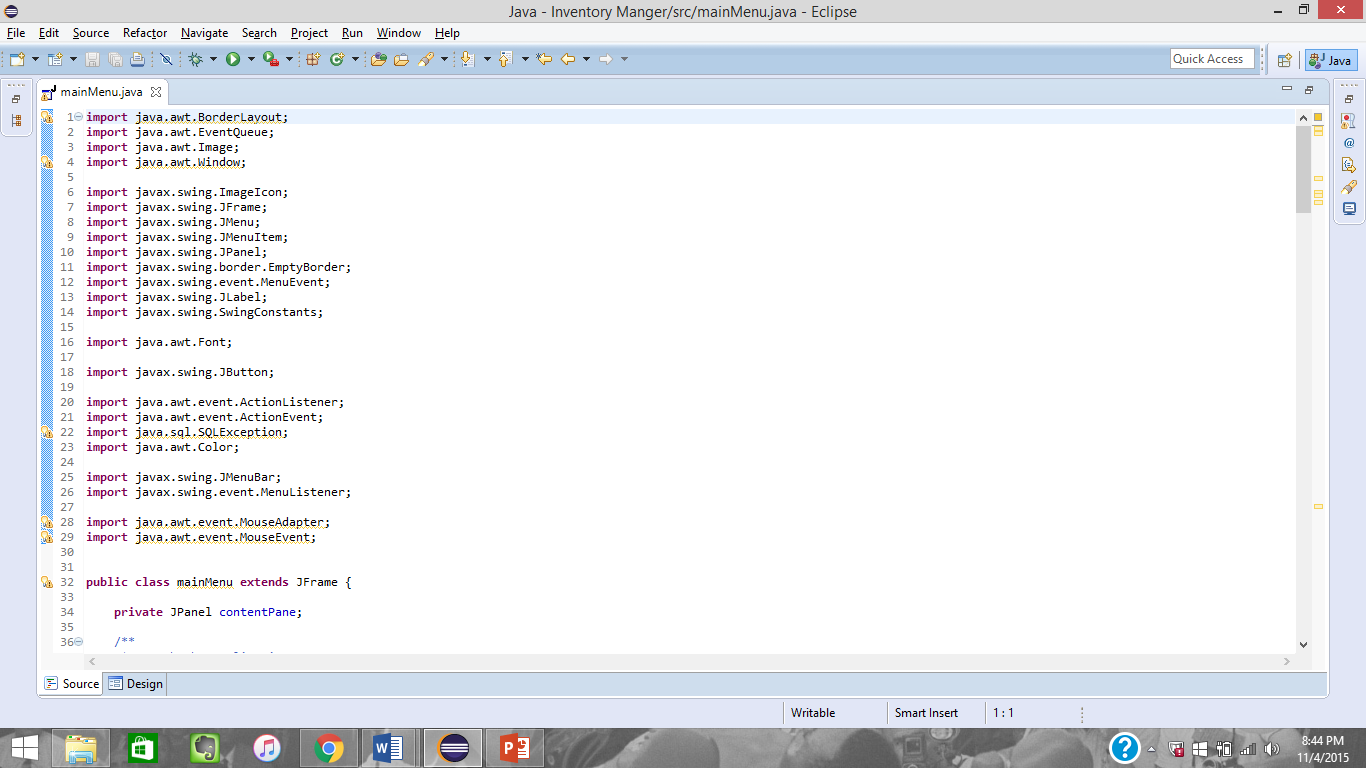


Figure - Libraries imported for the employee screen

# SQL Commands

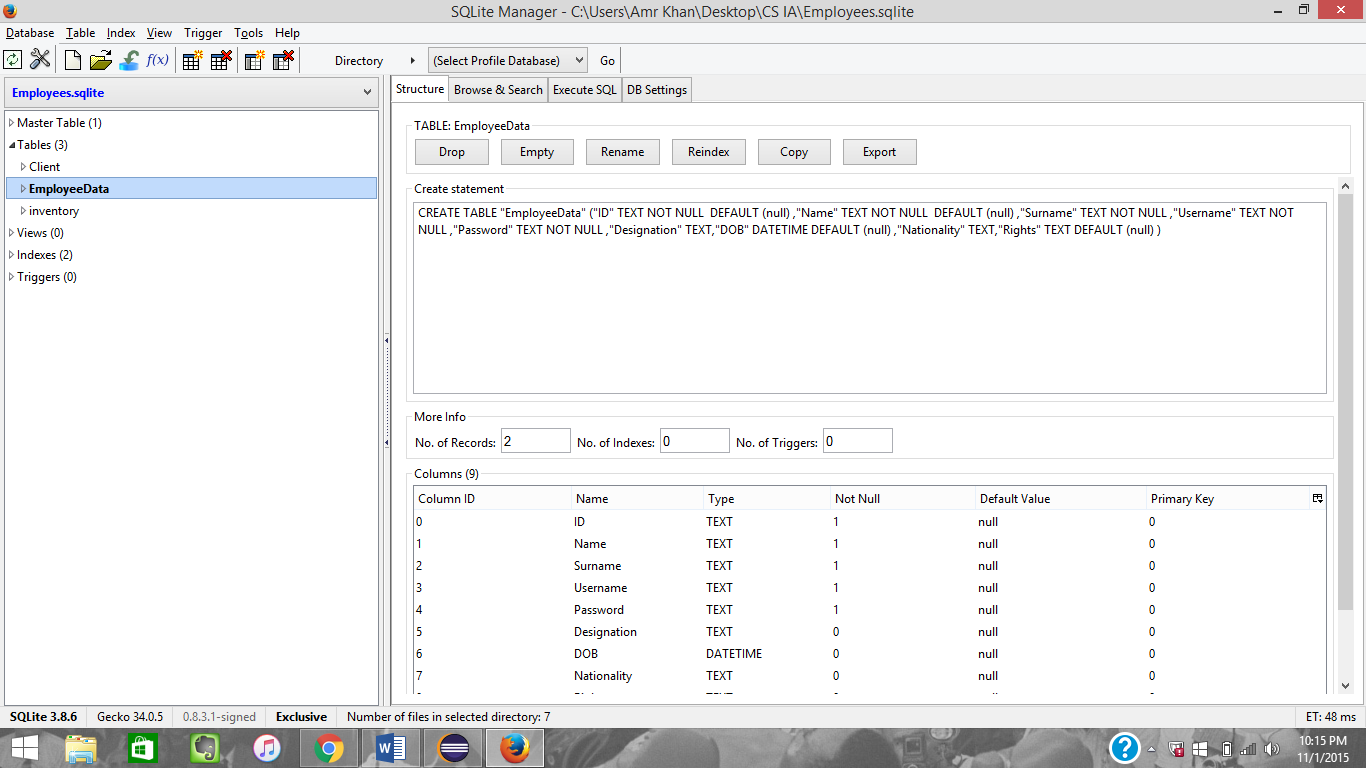


Figure - The create table statement for the employee database. Screenshot taken from SQLite Manager.

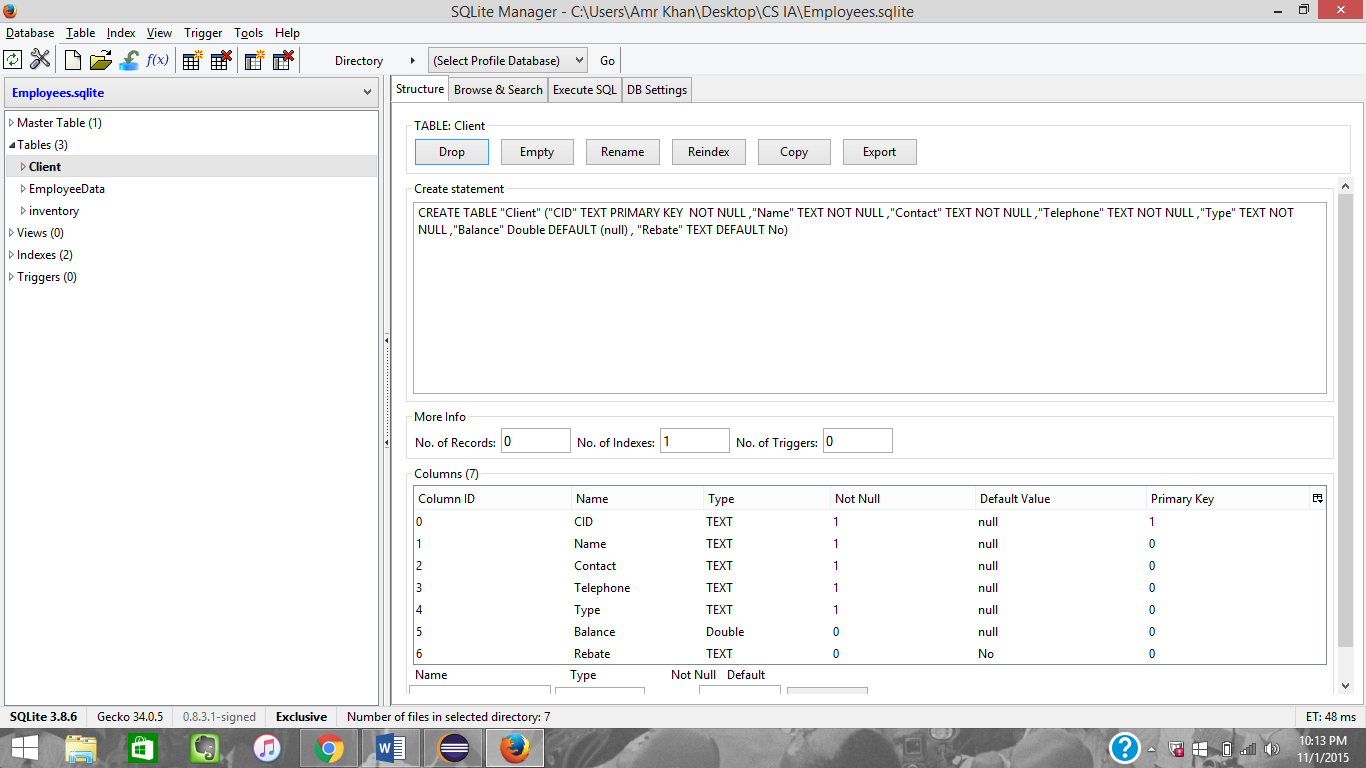


Figure - The create table statement for the Client database. Screenshot taken from SQLite Manager.

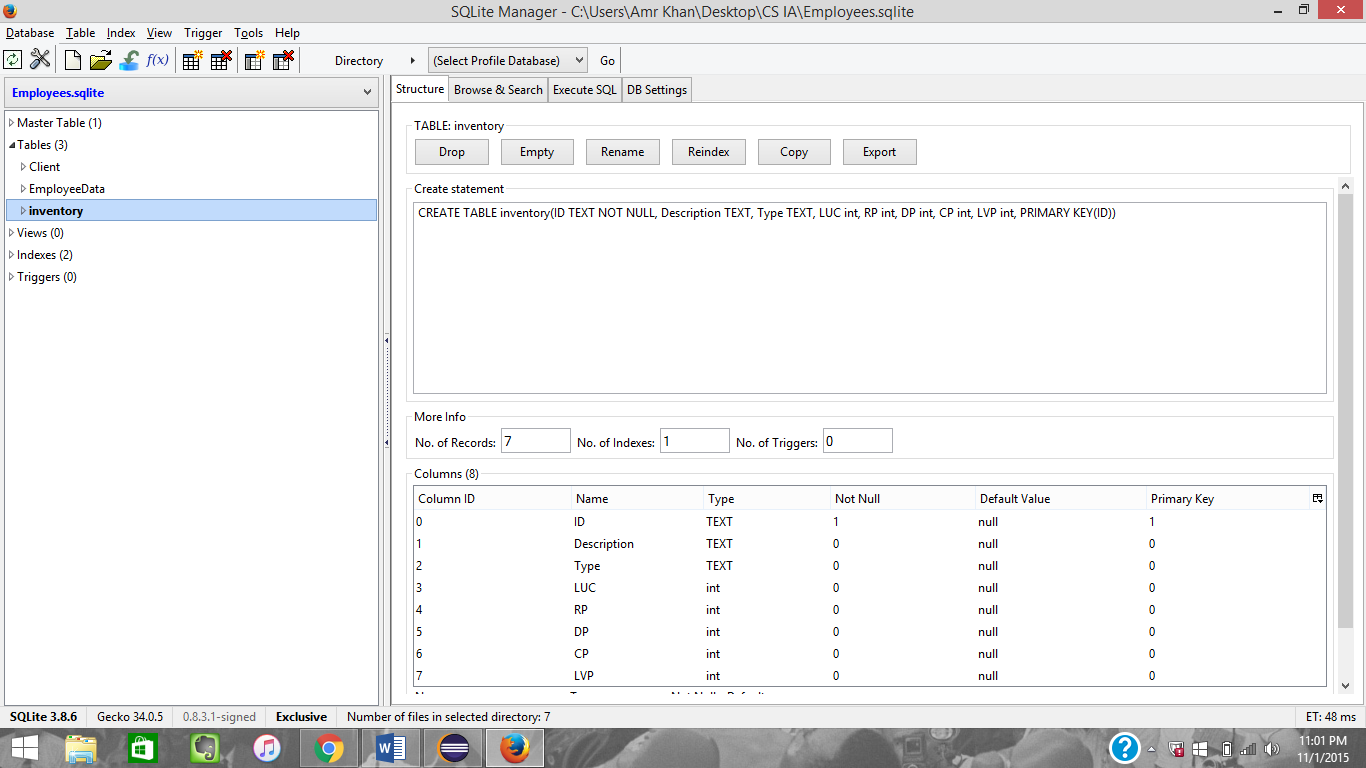


Figure - The create table statement for the inventory database. Screenshot taken from SQLite Manager.

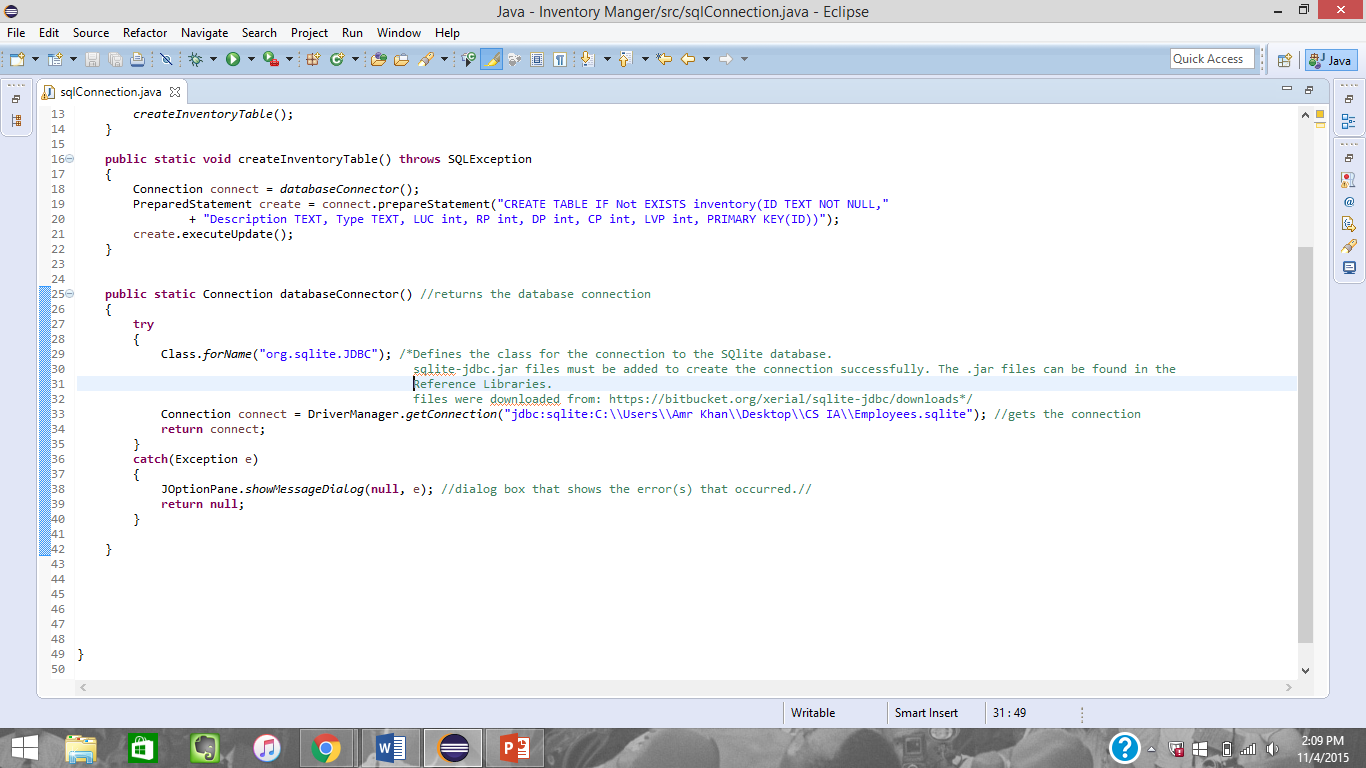


Figure – Method for creating the SQL connection with the database

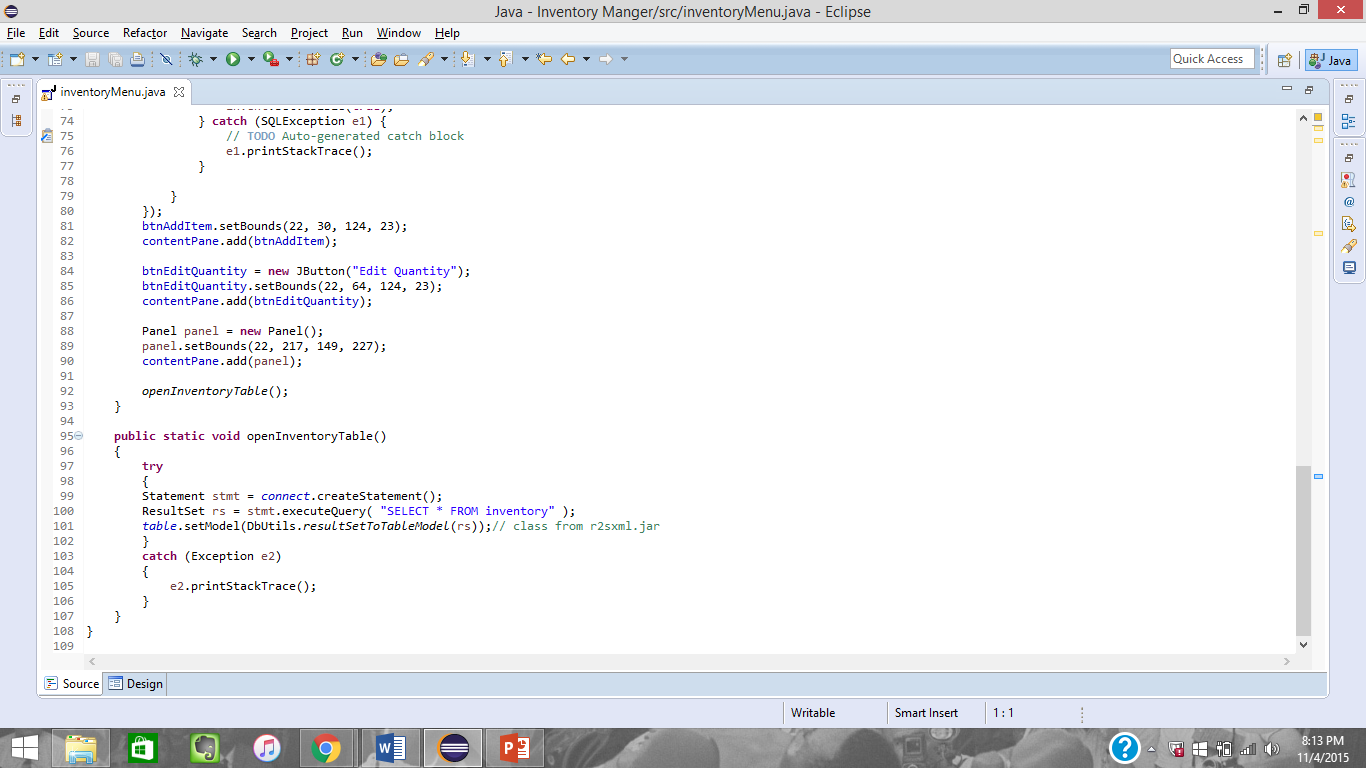


Figure - This method is used to automatically open the inventory database and create a table to hold that information. This methods is also called to refresh the database once modification to data take place.

# Classes

## Login

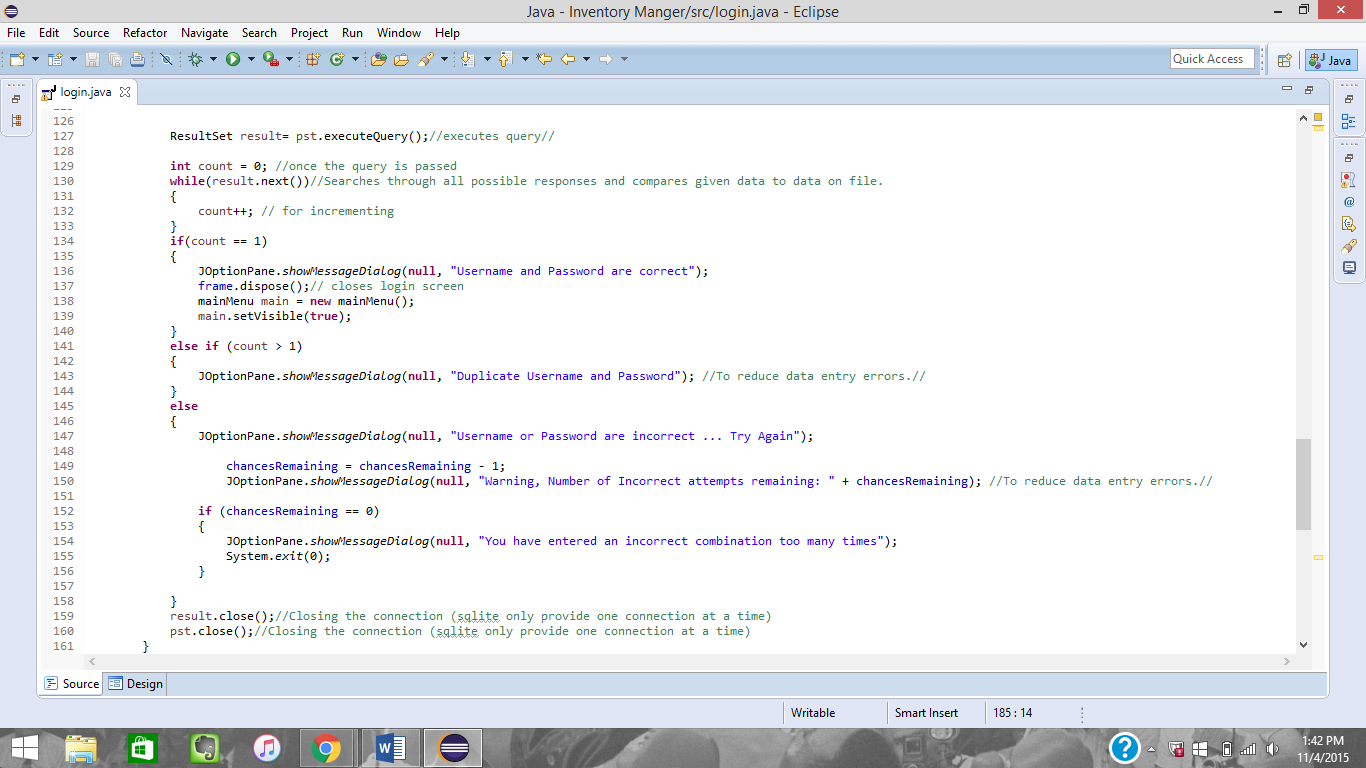


Figure - Code for checking if the login details are correct

The while loop checks through the entire SQL employeeData database. The count value is used to check if the details exist of it there are duplicate entries. In each case a separate but generic message is generated. If the incorrect details are entered 3 times the program automatically closes.

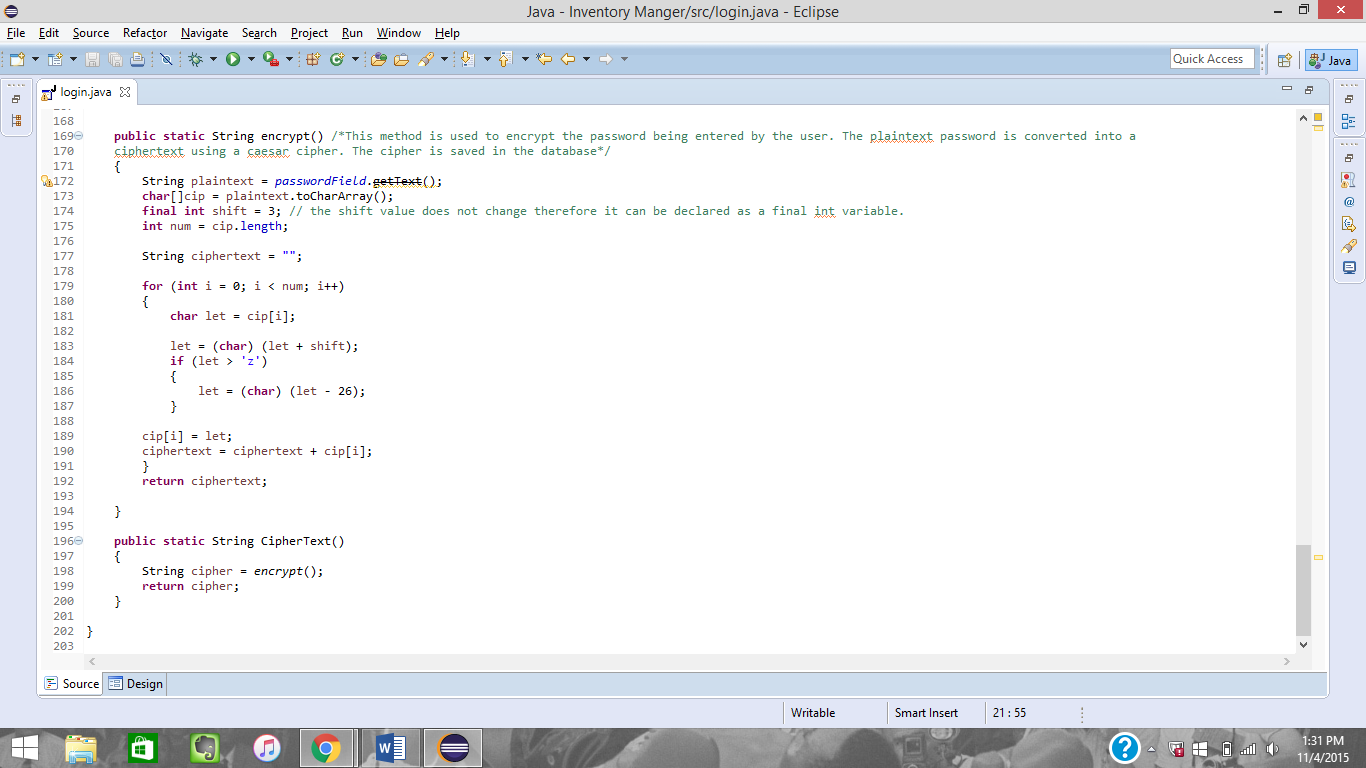


Figure - Caesar Cipher to store random strings of characters as the password

This methods is a simple encryption technique that has been used for many decades. It is a basic Caesar cipher to render the password inputted into unintelligible text. Once the password is converted to the ciphertext it is checked with the ciphertext on file. This is to make sure that the employeeData file does not have the person’s actual password stored. The password is converted to a character array. Final int is used to change the number of characters the letter would shift by since the shift value does not change. Each individual element of the array is shifting in a for loop, here the length of the loop in not declared but rather is set when the password is read so that the array does not take up any unnecessary space in the memory also a null pointer error does not occur when if the password is less than the array length. In the loop the shift value is added to the character and if the value is greater than the ASCII value of ‘z’ then 26 is subtracted to make sure that the stored character is a letter.

## Node

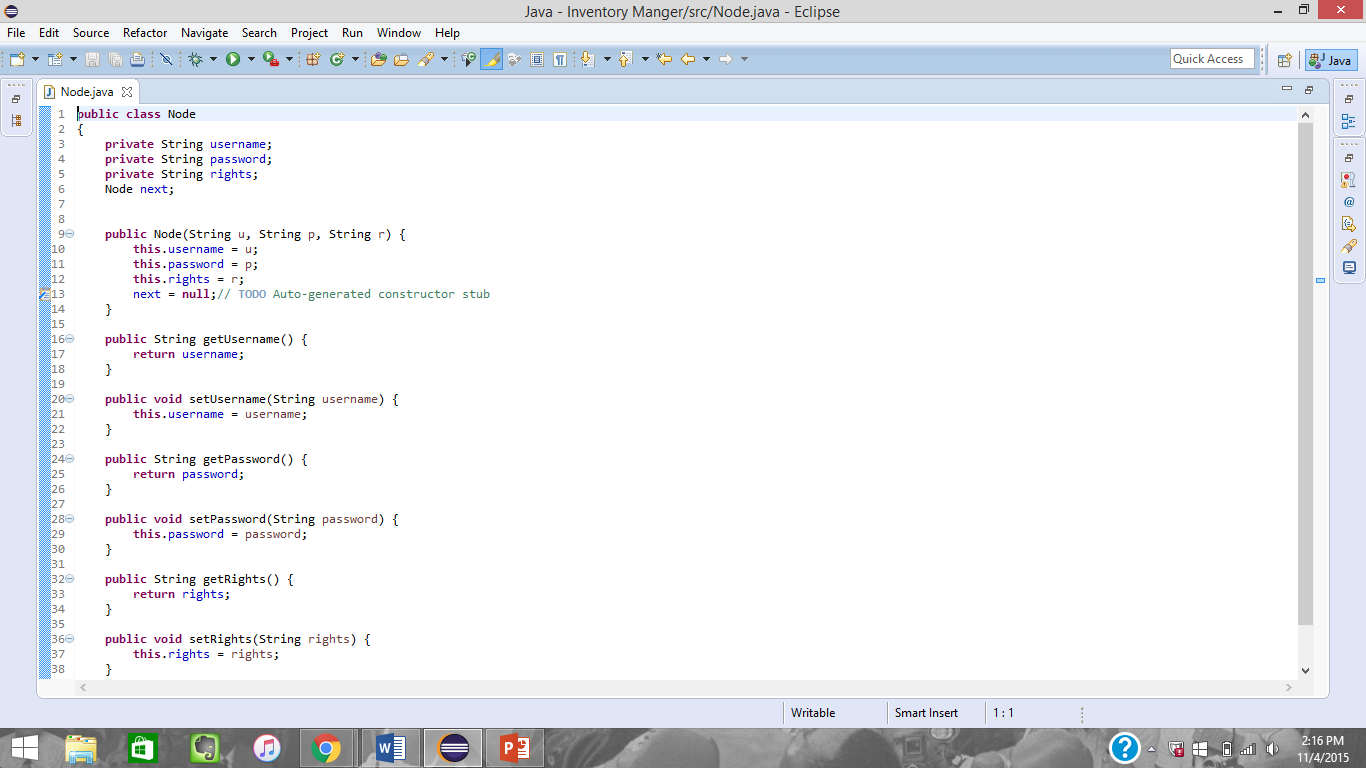


Figure - Code for Node class

The node class is used to store the employee details and is used to store to create the audit log details that are in the audit log class. Also is used to have access rights.

## addClient

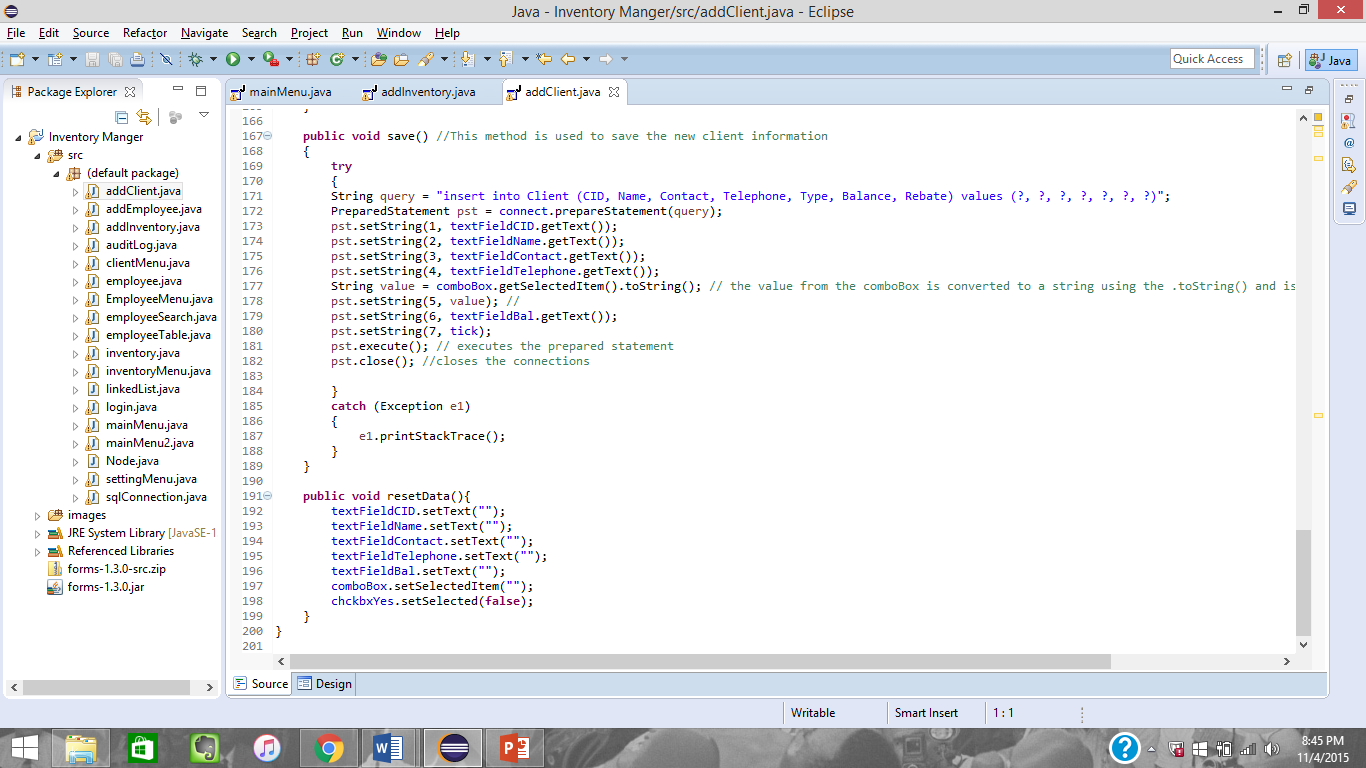
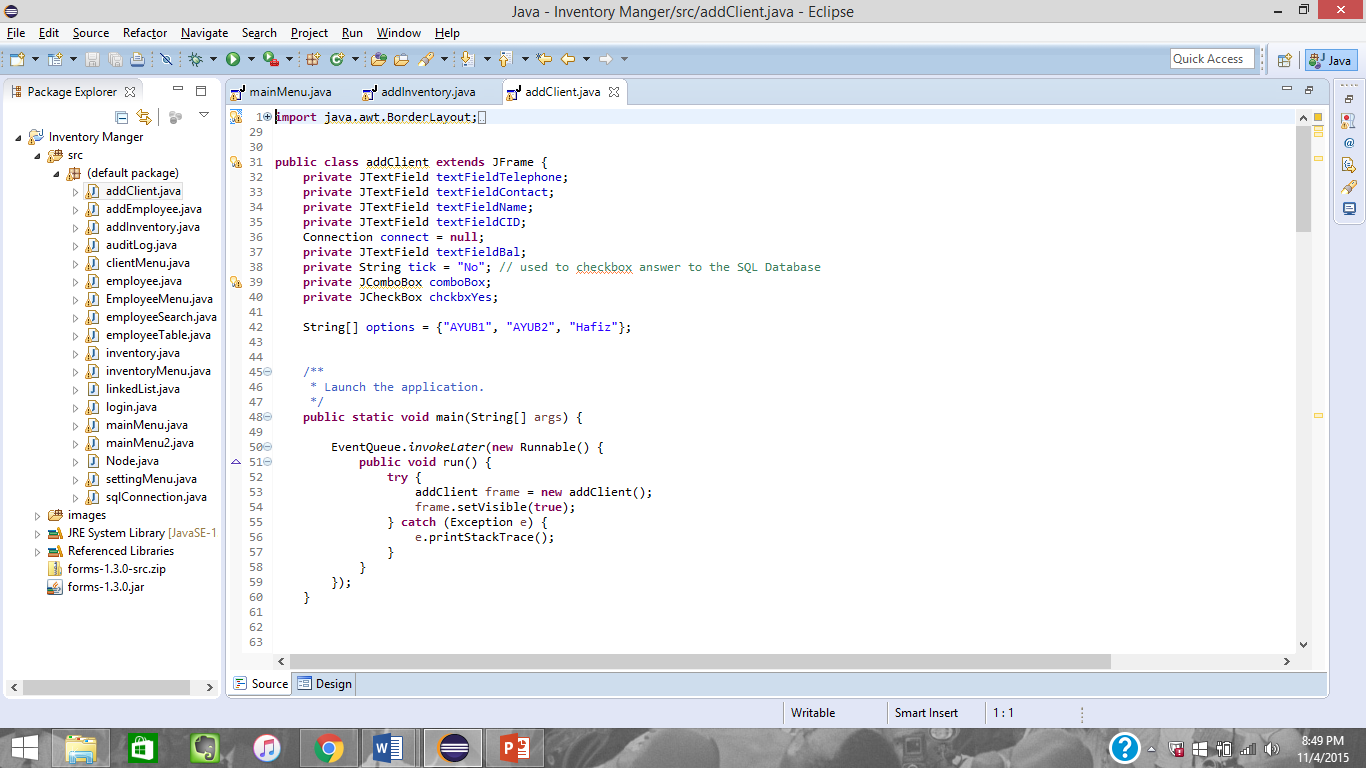


Figure - Save and reset methods in the client class

The save() methods create a query that will be used to add the new entry into the SQL database. The preparedstatement establishes the connection with the SQL database. Each of the values that needed to be added in query are obtained from the GUI components that they were entered into. The toString() method is used to get the data from the combobox. The resetdata() method simply clears all the information in the GUI elements so that additional clients can be added.



Different codes are given to the type of client. These values are stored in the combox box to make it easier to search for them. The values were given by the client. An array was used because there are a set number of options and in the correspondence with the client, he mention that these options do not change.

## LinkedList

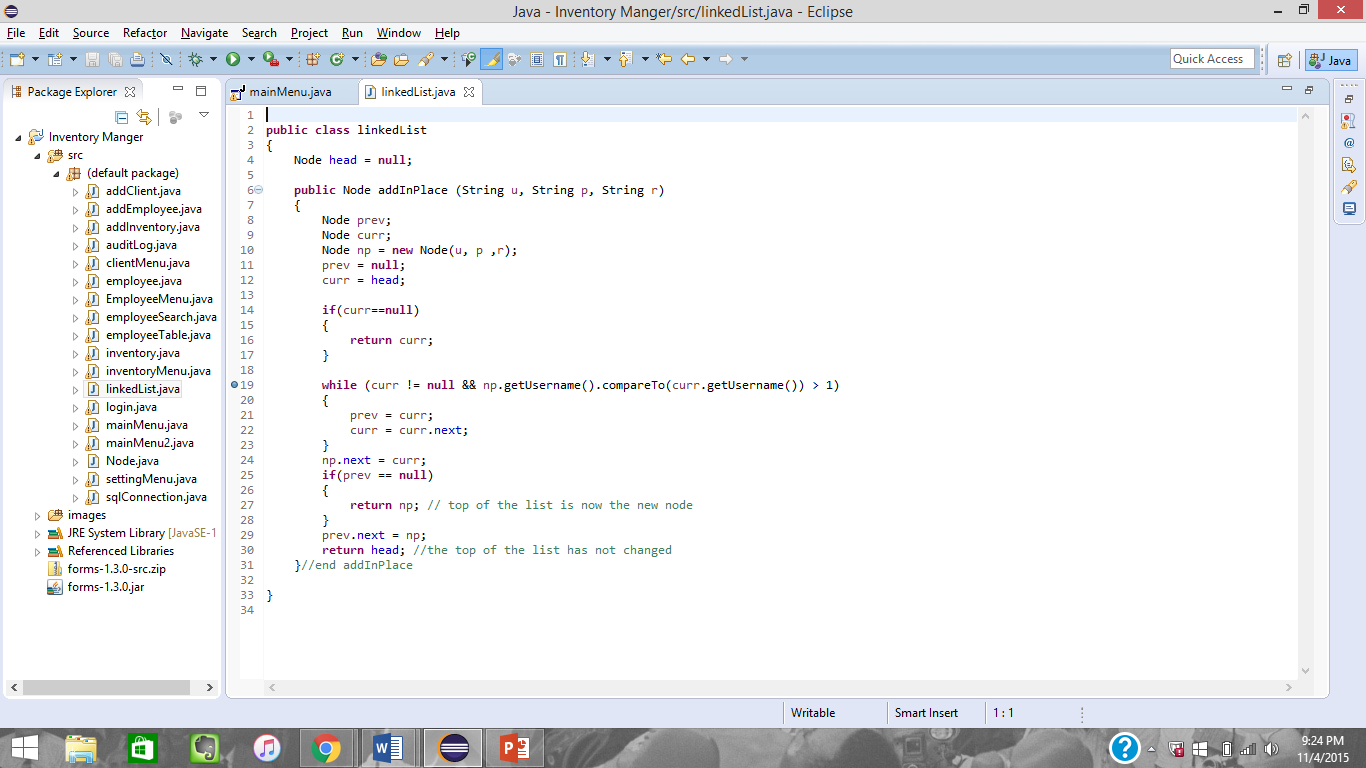


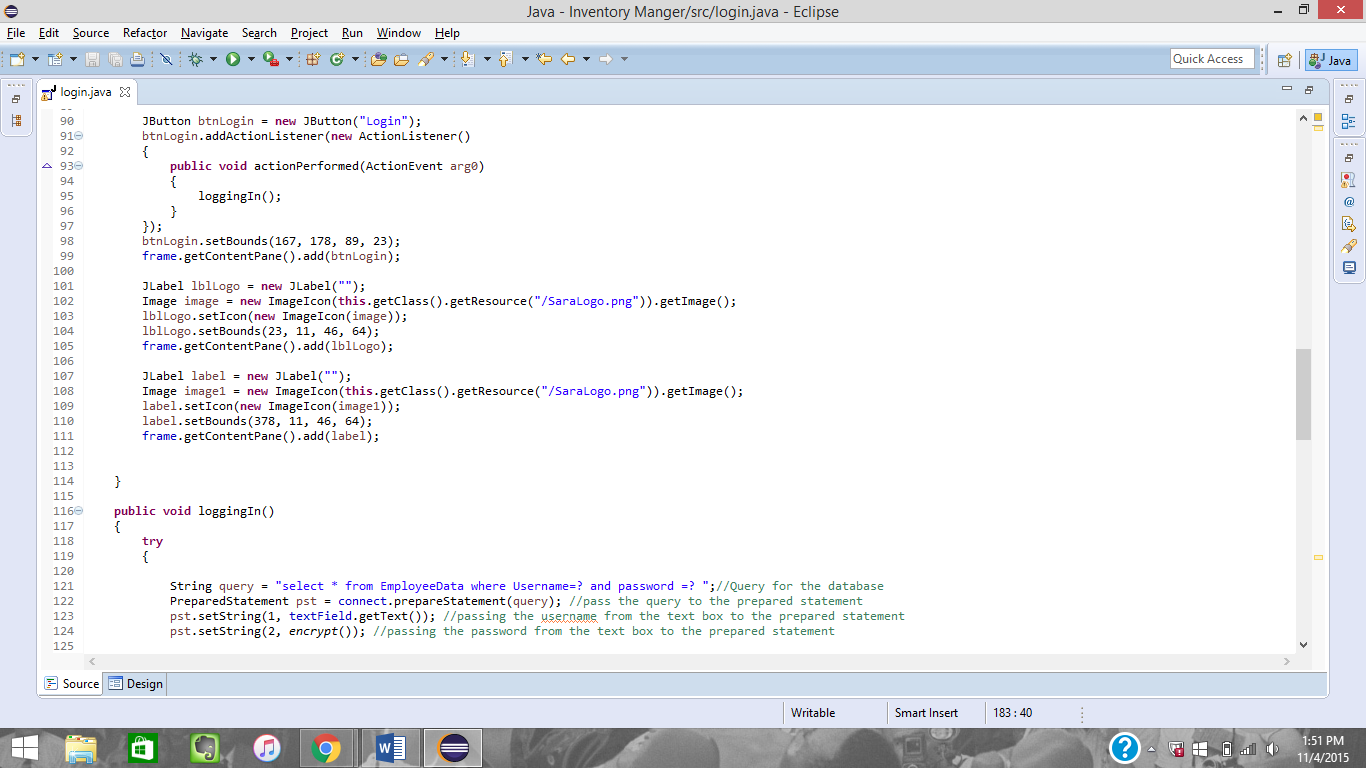
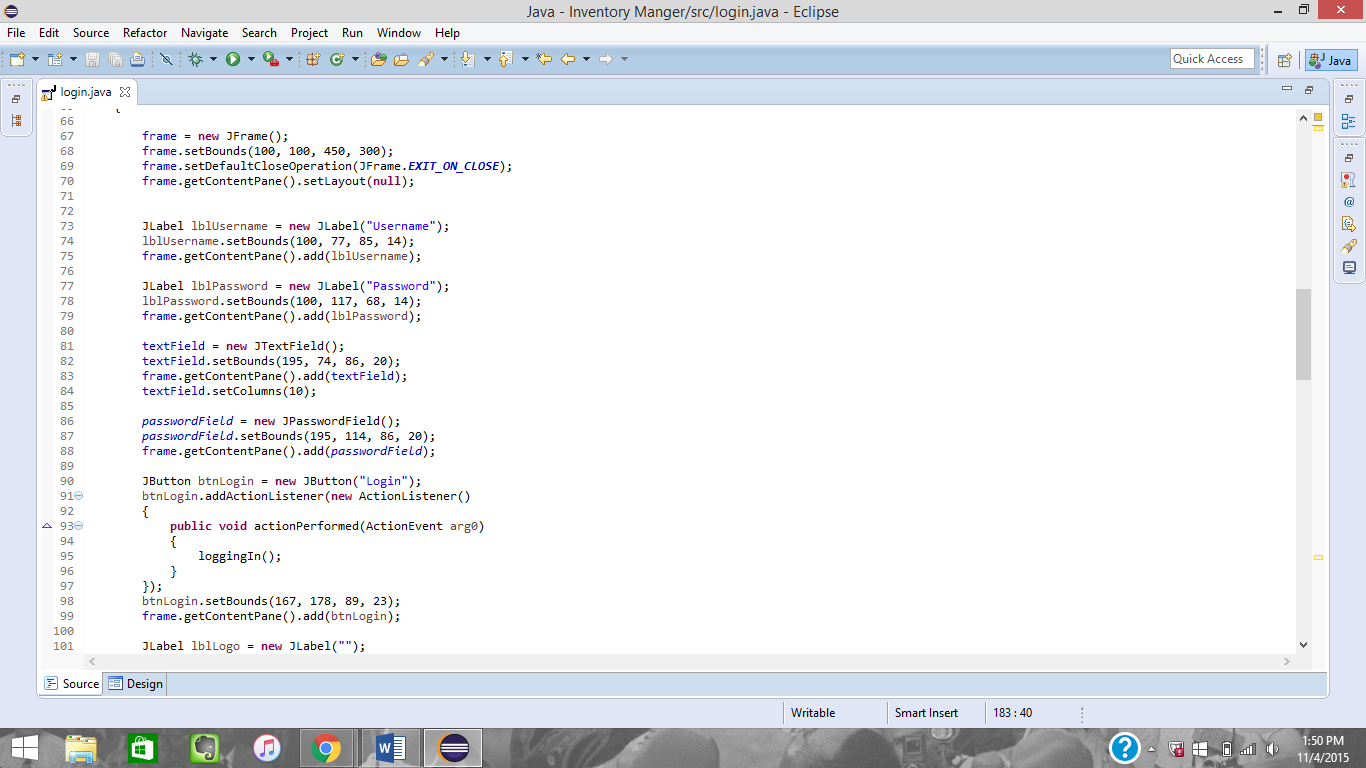
Figure - Code for the linked list class

This is the code that is used to create the linkedlist that will store the action preformed. The each node is add in alphabetical order and it compared using the username. This is a singly linked list since the action will go through the list once only.

# GUI Code

## Login

Figure - GUI component that are to create the login screen



## inventoryMenu

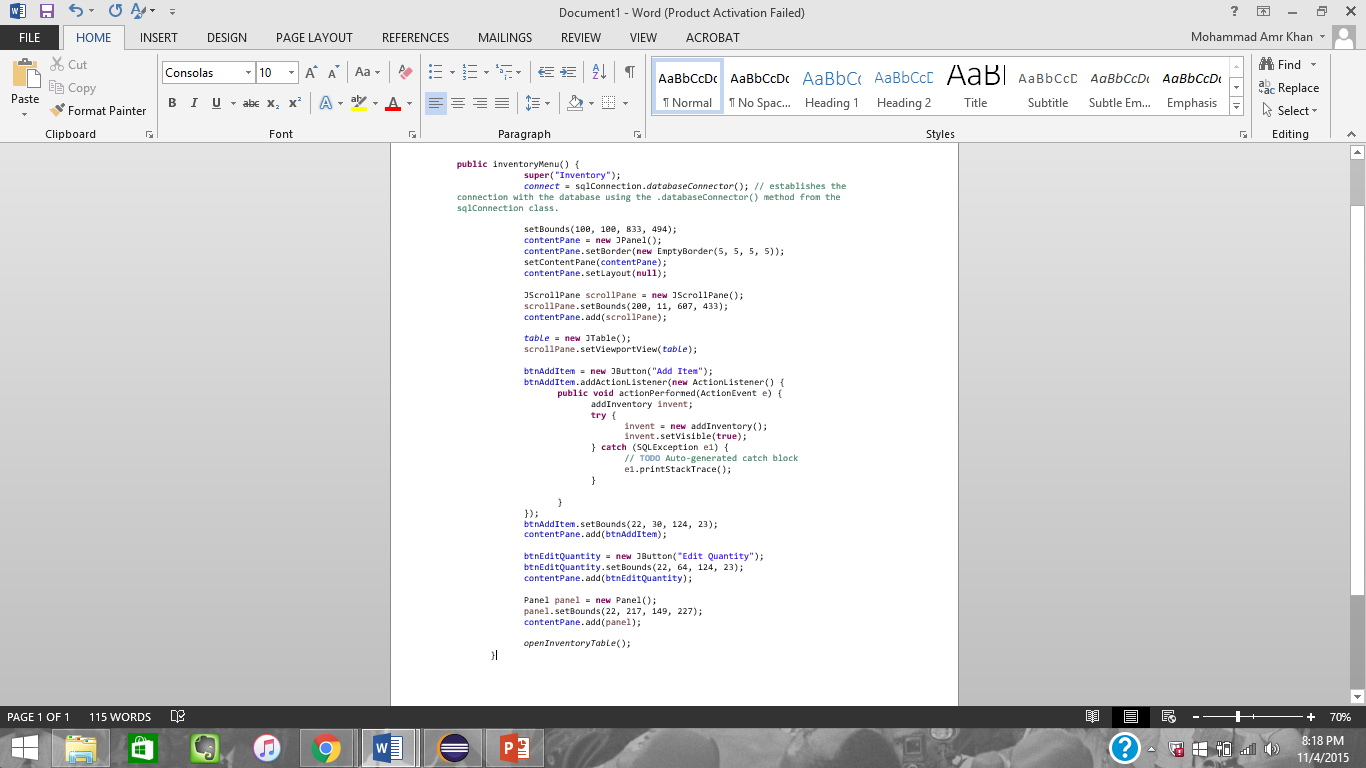
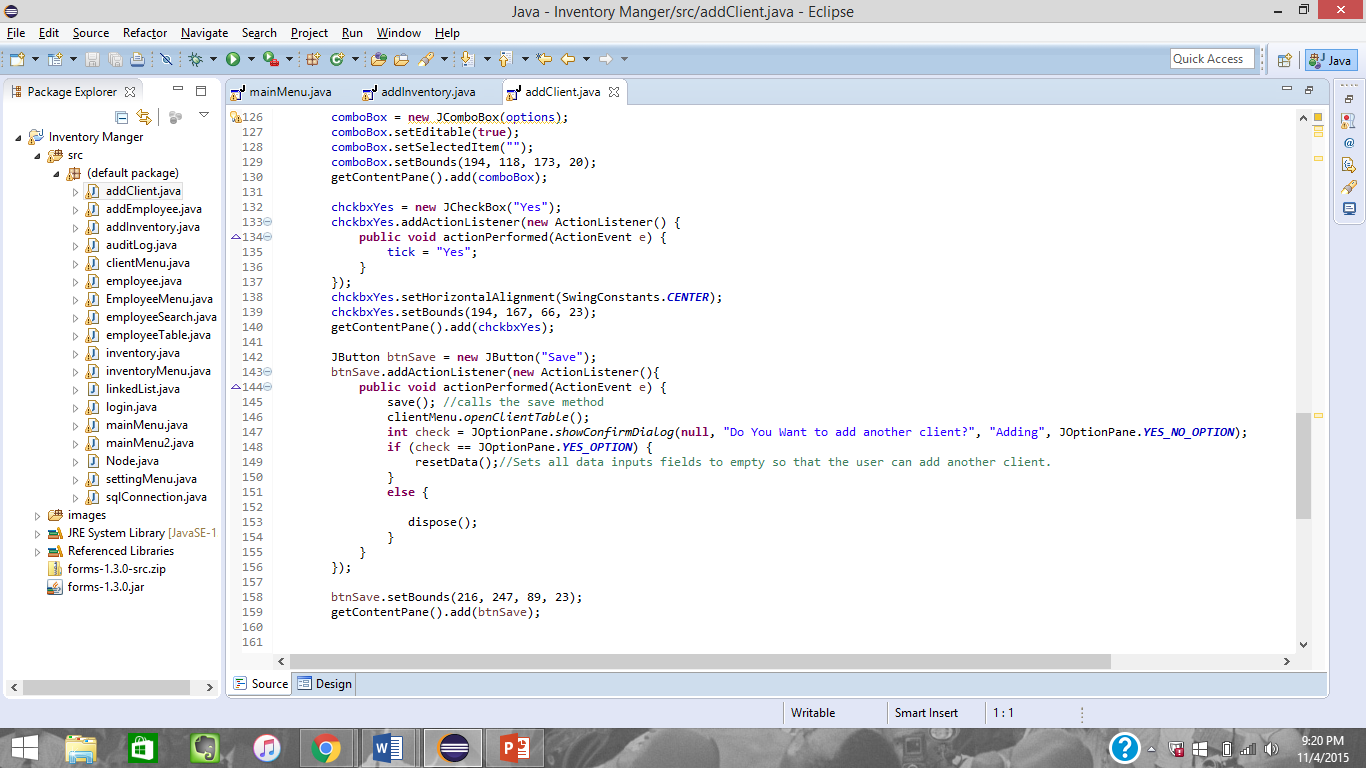


Figure - GUI components being declared in the inventory menu class

There are actionlisteners activated for all the buttons so that once an action is performed the user is taken to another screen to continue the process.

## addClient



The options are added to the combo box in the first line of the code. The save button uses method that are outlined in the classes section and uses JOptionPane to generate prompts. I choose to include JOptionPanes because the user will immediately focus on the prompt as it contain pertinent information.

# Unit Testing

[Errors were occurring in unit testing code, when fixed will be added to criterion C]