Final

ThreadCount

Developed for **Nicholas duchon**

CMSC 495

Jennifer Prizeman, ian Chrisman, Justin Jennings, edward caro, Shawn thompson, colin crowley

2016

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision #** | **Revision Date** | **Document Name** | **Changes** |
| 0.1 | October 31, 2016 | Project Plan | Initial Project Plan Created |
| 0.2 | November 2, 2016 | [Project](mailto:jennifer.prizeman@gmail.com) Plan | Edits made |
| 0.3 | [November 5, 2016](mailto:jennifer.prizeman@gmail.com) | [Project](mailto:jennifer.prizeman@gmail.com) Plan | Edits made |
| 0.4 | [November 6, 2016](mailto:jennifer.prizeman@gmail.com) | [Project](mailto:jennifer.prizeman@gmail.com) Plan | Edits made / Project Submitted |
| 0.5 | November 10, 2016 | User Guide & Test Plan | Initial User Guide & Test Plan Created |
| 0.6 | November 11, 2016 | User Guide & Test Plan | Edits made |
| 0.7 | November 13, 2016 | User Guide & Test Plan | Post-peer review Edits/ User Guide & Test Plan Submitted |
| 0.8 | November 15, 2016 | Project Design | Initial Project Design Created |
| 0.9 | November 16, 2016 | Project Design | Edits made |
| 1.0 | November 19, 2016 | Project Design | Edits made |
| 1.1 | November 20, 2016 | Project Design | User Guide updated  Edits made / Project Submitted |
| 1.2 | November 21, 2016 | Phase I | Initial Phase I documentation created |
| 1.3 | November 23, 2016 | Phase I | Edits made |
| 1.4 | November 25, 2016 | Phase I | Edits made/Phase I posted for Peer Review 2 |
| 1.5 | November 27, 2016 | Phase I | Phase I updated  Edits made/Phase I submitted |
| 1.6 | December 1, 2016 | Phase II | Initial Phase II documentation created |
| 1.7 | December 2, 2016 | Phase II | Edits Made |
| 1.8 | December 3, 2016 | Phase II | Edits Made |
| 1.9 | December 4, 2016 | Phase II | Edits made/Phase II submitted |
| **Revision #** | **Revision Date** | **Document Name** | **Changes** |
| 2.0 | December 7, 2016 | Phase III | Initial Phase III documentation created |
| 2.1 | December 9, 2016 | Phase III | Edits Made |
| 2.2 | December 10, 2016 | Phase III | Edits Made |
| 2.3 | December 11, 2016 | Phase III | Edits made/Phase III submitted |
| 2.4 | December 14, 2016 | Final | Initial Final documentation created |
| 2.5 | December 15, 2016 | Final | Edits Made |
| 2.6 | December 16, 2016 | Final | Edits Made |
| 2.7 | December 17, 2016 | Final | Edits Made |
| 3.0 | December 18, 2016 | Final | Post-peer review Edits/ Final Submitted |

Table of Contents

[1. Requirements Specification 8](#_Toc469823466)

[1.1 Purpose, Scope, and Objectives 8](#_Toc469823467)

[1.2 Project Background 8](#_Toc469823468)

[1.3 User Profile 8](#_Toc469823469)

[1.4 Assumptions and Constraints 9](#_Toc469823470)

[1.5 Project Deliverables 9](#_Toc469823471)

[1.6 Project Outline 9](#_Toc469823472)

[1.6.1 UML Diagram 10](#_Toc469823473)

[1.6.2 User Scenarios 11](#_Toc469823474)

[2. System Specification 12](#_Toc469823475)

[2.1 Hardware 12](#_Toc469823476)

[2.2 Software 12](#_Toc469823477)

[3. Project Plan 13](#_Toc469823478)

[3.1 Key Milestones 13](#_Toc469823479)

[3.2 Key Roles 14](#_Toc469823480)

[3.3 Upcoming Tasks and Schedule 14](#_Toc469823481)

[4. User Guide 15](#_Toc469823482)

[4.1 Running the Software 15](#_Toc469823483)

[4.2 Configuration Settings Window 15](#_Toc469823484)

[4.3 Main Screen 16](#_Toc469823485)

[4.4 Customer 17](#_Toc469823486)

[4.5 Catalog 18](#_Toc469823487)

[4.6 Sales 19](#_Toc469823488)

[4.7 Inventory 20](#_Toc469823489)

[4.8 Reports 21](#_Toc469823490)

[4.9 Developer’s Notes 21](#_Toc469823491)

[4.9.1 ThreadCount database schema 21](#_Toc469823492)

[5. Test Plan 22](#_Toc469823493)

[5.1 Test Team 22](#_Toc469823494)

[5.1.1 Lead Tester Responsibilities (Expanded) 22](#_Toc469823495)

[5.2 Test Environment 22](#_Toc469823496)

[5.3 Test Acceptance Criteria 22](#_Toc469823497)

[5.4 Test Deliverables 23](#_Toc469823498)

[5.4.1 Phase I Testing 23](#_Toc469823499)

[5.4.2 Phase I Testing Report 23](#_Toc469823500)

[5.4.3 Phase II Testing 24](#_Toc469823501)

[5.4.4 Phase II Testing Report 24](#_Toc469823502)

[5.4.5 Phase III Testing 24](#_Toc469823503)

[5.4.6 Phase III Testing Report 25](#_Toc469823504)

[5.4.7 Phase IV Testing 25](#_Toc469823505)

[5.4.8 Phase IV Testing Report 26](#_Toc469823506)

[5.4.9 Final Testing Report 26](#_Toc469823507)

[5.5 GUI Test Plan 26](#_Toc469823508)

[5.5.1 Scope 26](#_Toc469823509)

[5.5.2 GUI Test Cases 27](#_Toc469823510)

[5.6 Database Test Plan 28](#_Toc469823511)

[5.6.1 Scope 28](#_Toc469823512)

[5.6.2 Database Test Types 28](#_Toc469823513)

[5.6.3 Database Test Approach 28](#_Toc469823514)

[5.6.4 Database Test Activities and Schedule 29](#_Toc469823515)

[5.6.5 Risks 29](#_Toc469823516)

[5.6.6 Database Test Cases 29](#_Toc469823517)

[6. Design 34](#_Toc469823518)

[6.1 Overall Approach 34](#_Toc469823519)

[6.1.1 Workflow 34](#_Toc469823520)

[6.2 Program Structure 35](#_Toc469823521)

[6.2.1 UML Diagram 35](#_Toc469823522)

[6.2.2 Classes, Methods, Fields, & Interfaces 36](#_Toc469823523)

[6.2.3 Class Design 36](#_Toc469823524)

[6.3 User Interface 36](#_Toc469823525)

[6.3.1 Customer tab flow chart 36](#_Toc469823526)

[6.3.2 Catalog tab flow chart 37](#_Toc469823527)

[6.3.3 Sales tab flow chart 37](#_Toc469823528)

[6.3.4 Reports tab flow chart 37](#_Toc469823529)

[6.3.5 Inventory tab flow chart 37](#_Toc469823530)

[6.4 GUI 38](#_Toc469823531)

[6.4.1 GUI Workflow 38](#_Toc469823532)

[6.4.2 GUI Design 38](#_Toc469823533)

[6.4.3 GUI Components 38](#_Toc469823534)

[6.4.4 GUI Tabs 39](#_Toc469823535)

[6.5 Database 40](#_Toc469823536)

[6.5.1 Sample SQL Schemas 40](#_Toc469823537)

[6.5.2 SQL Queries 41](#_Toc469823538)

[6.5.3 Connectivity 41](#_Toc469823539)

[6.5.4 Database Components 41](#_Toc469823540)

[6.5.5 File Import/Export Mechanism 41](#_Toc469823541)

[6.6 Design Phases 41](#_Toc469823542)

[6.6.1 Phase 1 (Due Nov 27th) 41](#_Toc469823543)

[6.6.2 Phase 2 (Due Dec 4th) 41](#_Toc469823544)

[6.6.3 Phase 3 (Due Dec 11th) 42](#_Toc469823545)

[6.6.4 Phase 4 / Final (Due Dec 18th) 42](#_Toc469823546)

[7. Phase I 43](#_Toc469823547)

[7.1 Phase I Milestones 43](#_Toc469823548)

[7.2 Roles and Responsibilities 43](#_Toc469823549)

[7.3 Phase I Recap 43](#_Toc469823550)

[7.3.1 User Interface 44](#_Toc469823551)

[7.3.2 How to Run Source Code in IDE 44](#_Toc469823552)

[7.3.3 Upcoming Functionality 44](#_Toc469823553)

[7.4 Problems Encountered 44](#_Toc469823554)

[7.5 Reconsiderations 44](#_Toc469823555)

[7.6 Schedule 44](#_Toc469823556)

[7.7 Document Changes 44](#_Toc469823557)

[8. Phase II 46](#_Toc469823558)

[8.1 Phase II Milestones 46](#_Toc469823559)

[8.2 Roles and Responsibilities 46](#_Toc469823560)

[8.3 Phase II Recap 46](#_Toc469823561)

[8.3.1 User Interface 46](#_Toc469823562)

[8.3.2 Upcoming Functionality 47](#_Toc469823563)

[8.4 Problems Encountered 47](#_Toc469823564)

[8.5 Reconsiderations 47](#_Toc469823565)

[8.6 Schedule 47](#_Toc469823566)

[8.7 Document Changes 47](#_Toc469823567)

[9. Phase III 48](#_Toc469823568)

[9.1 Phase III Milestones 48](#_Toc469823569)

[9.2 Roles and Responsibilities 48](#_Toc469823570)

[9.3 Phase III Recap 48](#_Toc469823571)

[9.3.1 User Interface 48](#_Toc469823572)

[9.3.2 Upcoming Functionality 48](#_Toc469823573)

[9.4 Problems Encountered 49](#_Toc469823574)

[9.5 Reconsiderations 49](#_Toc469823575)

[9.6 Schedule 49](#_Toc469823576)

[9.7 Document Changes 49](#_Toc469823577)

[10. Final Phase 50](#_Toc469823578)

[10.1 Final Phase Milestones 50](#_Toc469823579)

[10.2 Roles and Responsibilities 50](#_Toc469823580)

[10.3 Final Recap 50](#_Toc469823581)

[10.4 Problems Encountered 51](#_Toc469823582)

[10.5 Reconsiderations 51](#_Toc469823583)

[10.6 Document Changes 51](#_Toc469823584)

[10.7 Conclusion 51](#_Toc469823585)

[10.7.1 Lessons Learned 51](#_Toc469823586)

[10.7.2 Design Strengths 51](#_Toc469823587)

[10.7.3 Limitations 51](#_Toc469823588)

[10.7.4 Suggestions for Future Improvement 52](#_Toc469823589)

# 1. Requirements Specification

This section will list, in detail, the project objectives, product background, user profile, assumptions, constraints, deliverables, and product outline.

## Purpose, Scope, and Objectives

The objective of the ThreadCount program is to provide software to increase the efficiency of personnel, particularly those who manage small clothing boutiques. By providing a software application to manage inventory, sales, and customer information, the program will save time and effort for the user. Inventory and balance will be adjusted in the database each time the user enters a transaction. The user interface will enable the application user to display various reports about sales, customer, and inventory data. These reports will give the salesperson pertinent information, such as their current inventory and shipment information.

## Project Background

The direct sales model is making a comeback. With social media being widely available, the sales and marketing opportunities to sell products to friends and acquaintances has exploded. This has made it easier now than ever to start and run a home business. This is a particularly attractive idea for stay at home parents seeking to supplement family income and have fun at the same time. However, these homes are rarely equipped with business class IT infrastructure or software to help run the business. While inexpensive software exists for generic tasks such as accounting and taxes, these new small business owners lack the ability to properly track inventory[[1]](#footnote-1), and create important reports that help run the business. ThreadCount will create a simple, inexpensive solution to help these new businesses flourish. Our program is a prototype for a sole proprietor type of user, but it will be designed to be scalable for larger enterprises.

## User Profile

Sheri McCoy, the CEO of AVON products, explained “[Our Founder David H. McConnell] understood that women were natural salespeople who could easily relate to other women and passionately market beauty products.”[[2]](#footnote-2) Likewise, and based on our research, the largest demographic for creating new direct sales businesses are women aged 22-40. They are typically suburban Generation X or Millennials. Additionally, our target users are from affluent households with enough disposable income to purchase sufficient inventory to launch the business. However, we see the lure of opening a business decline the wealthier that family is. As a result, the average income for the target audience has incomes roughly in the range of $50k - $150k per year. These are important considerations when considering several aspects of the software solution. First, the application’s look and feel should appeal to young women. Second, the complexity should be very low, as this would drive up the cost of the software and become ultimately unaffordable to the target group. Ideally, the software will cost around $10 per month. Lastly, this segment is not typically formally trained with IT products, so the software should be very easy to use and navigate.

## Assumptions and Constraints

For the development of the software, it is assumed that an information technology (IT) deployment exists in order for the software to be installed. Additionally, the development of the software is designed around portability. By focusing on the use of Java language and minimizing the user interface requirements, this software can be portable to almost all operating system platforms.

It is assumed the end users will not be familiar with information security best practices and will use the software to store private customer information and business sales information. The development team will evaluate these functional requirements to establish appropriate system security constraints in order to ensure confidentiality, integrity and availability of the information.

## Project Deliverables

Project deliverables for the software package are focused on the following:

1. Software package consisting of:
   1. Main GUI
   2. Item Creator
   3. Supplier Creator
   4. Shipment Creator
   5. Inventory Display
   6. Customer Display
   7. Report Builder
2. User Guide and manual

The software that will be delivered is a Java application that can be run via an executable .jar file. This software will also contain an example configuration file for database connectivity, with configuration options for MySQL.

Instructions for running the ThreadCount program are found in the User Guide. Considering our target user, our goal is to deliver a final software product that is user friendly. The software will be made available for download electronically for customers wishing to receive digital media. The User Guide will be delivered via electronic download as well, in the form of a Microsoft Word or PDF document.

Successful delivery of these features is dependent on testing acceptance after the software has passed key milestones where the software is versioned (alpha, beta, version 1.0).

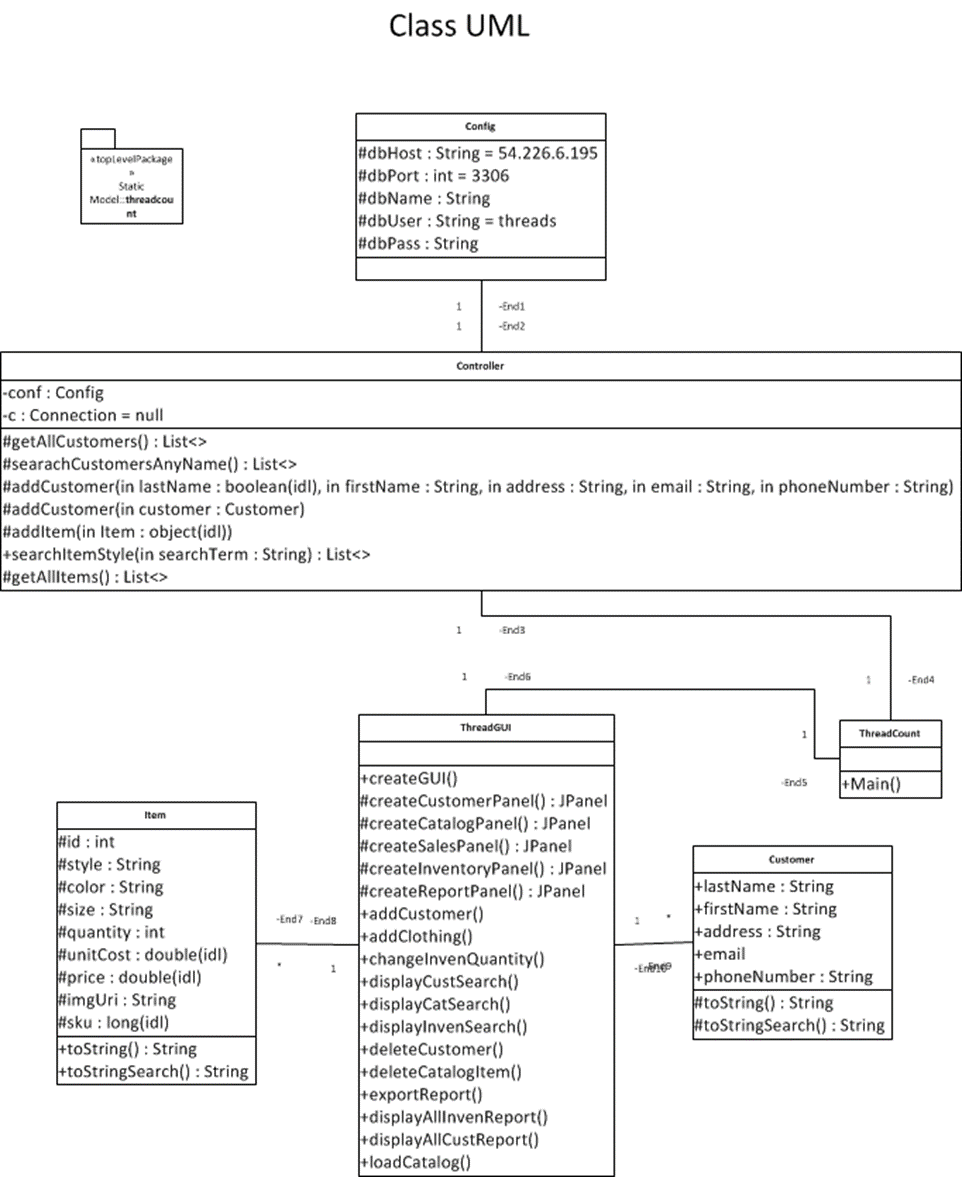
## Project Outline

The basic outline of the ThreadCount application will be as follows:

* GUI Code – Will be generic, calling methods in other classes and passing data.
* Two creator classes; Inventory and Sales
* Basic setup of database(s):
  + Inventory Table columns: include SKU, Style, Size, Wholesale Price, Retail Price, Quantity
  + Sales Table columns: Table key/transaction number, Date, Customer Name, Item purchased, Sales Balance, Address, Email

### 1.6.1 UML Diagram

The UML diagram below depicts the relationships and dependencies among classes for the project’s source code.



***Figure 1. UML diagram***

### 1.6.2 User Scenarios

The software will support 5 major user scenarios. It will allow the user to obtain and manipulate information regarding sales, customers, inventory, catalog, and various reports.

#### 1.6.2.1 Customers

The application will maintain a list of customers and a history for each customer. From the Customer screen, the user will be able to search for a customer by name, address, email, etc. Further, the user will need to get past sales history about customers. For instance, the user will want to be able to identify customers that buy frequently.

#### 1.6.2.2 Catalog

A complete list of items that are possible to be sold through the business will be maintained by the application. These items may or may not be in stock. The user will need to be able to search and find items by searching characteristics of the products.

#### 1.6.2.3 Sales

The user will be able to assemble a list of items for a customer’s purchase. Items can be selected from the pre-existing catalog and dropped into a ‘shopping cart’. Once the item is in the cart, quantities and pricing can be adjusted if it needs to be different than the default.

#### 1.6.2.4 Reports

A report screen will offer the user a set of commonly used reports. These will be preconfigured for the user. Reports such as sales history, customer history, and current inventory will be available.

#### 1.6.2.5 Inventory

The program will keep a list of products that are currently in inventory. From the inventory screen the user will be able to tell what products that they can currently offer to customers. They will need to be able to search by characteristics like clothing type (shirt, pants), size, color, etc. so that they can find suitable offerings for a particular customer’s needs.

# 2. System Specification

This section will list the required hardware and software that will be used to complete the ThreadCount application.

## 2.1 Hardware

* Hardware consisting of:
  + Windows PC
  + MacBook Air using Mac OS

## 2.2 Software

* Software consisting of:
  + Language – Java SE Development Kit (JDK) 8 and associated libraries
  + Database Implementations – MySQL
  + Integrated Development Environment (IDE) – NetBeans 8.1
  + Source Code Repository – GitHub
  + Documentation – Microsoft Word and Adobe Acrobat

# 3. Project Plan

The project plan provides an outline of key milestones, key roles, upcoming tasks, schedule, test plan, and error handling.

## 3.1 Key Milestones

Milestones will be structured so that they coincide with the weekly required documents due.

| **Weekly Document Requirements** | **Due Date** | **Assigned to** |
| --- | --- | --- |
| Project Plan | November 6, 2016 | Edward/Jennifer |
| Test Plan (Front End) | November 13, 2016 | Edward |
| Test Plan (Back End) | November 13, 2016 | Colin |
| User Guide | November 13, 2016 | Shawn |
| Design | November 20, 2016 | Edward/Jennifer (group input) |
| Phase I Source Documentation | November 27, 2016 | Group |
| Phase II Source Documentation | December 4, 2016 | Group |
| Phase III Source Documentation | December 11, 2016 | Group |
| Final Documentation | December 18, 2016 | Edward/Jennifer |

***Figure 2. Document Milestones***

| **Project Milestones** | **Due Date** | **Assigned to** | **Complete Y / N** |
| --- | --- | --- | --- |
| Create User Profile | November 4, 2016 | Shawn | Y |
| Create Product Background Info | November 4, 2016 | Shawn | Y |
| Create UML Diagram | November 5, 2016 | Ian | Y |
| Create GUI Sketch | November 6, 2016 | Shawn | Y |
| Create database with test data | November 13, 2016 | Justin | Y |
| Display inventory from test database | November 15, 2016 | Ian | Y |
| Display different report types from database | November 15, 2016 | Ian/Justin | Y |
| Enter information into database | November 17, 2016 | Ian | Y |
| Modify information in database | November 25, 2016 | Justin | Y |
| Display inventory from real database | December 1, 2016 | Justin | Y |
| Display different types of reports from real database | December 1, 2016 | Ian | Y |
| Backup /encrypt database information | December 8, 2016 | Colin | Y |
| Log in credentials | December 11, 2016 | Ian/Justin | Y |
| Error handling | December 18, 2016 | Ian | Y |

***Figure 3. Project Milestones***

## 3.2 Key Roles

There are six members on this project: Jennifer Prizeman, Ian Chrisman, Justin Jennings, Edward Caro, Shawn Thompson, and Colin Crowley. Each member of the team is assigned a specific lead role along with assisting others on various aspects during the development cycle of the project. Each team member's specific lead role is given in *Figure 4*, below, along with additional duties which are italicized.

| **Roles** | **Name** | **Contact Info** |
| --- | --- | --- |
| Project Manager / *Documentation* | Jennifer Prizeman | jennifer.prizeman@gmail.com |
| Lead Front End Coding | Ian Chrisman | ichrisman@student.umuc.edu |
| Lead Back End Coding | Justin Jennings | justin.robert.jennings@gmail.com |
| Documentation / *Front End Testing* | Edward Caro | edward.m.caro@gmail.com |
| User Guide / *Front End Coding* | Shawn Thompson | stt9000@gmail.com |
| Security / *Back End Coding* / *Back End Testing* | Colin Crowley | colin.t.crowley@gmail.com |

***Figure 4. Team Roles***

## 3.3 Upcoming Tasks and Schedule

Listed below are the upcoming tasks for the week.

* December 12th – 17th: Modify Phase III Source Code, Begin Phase IV Source Documentation
* December 16th: Post Phase IV Source for Peer Review 3
* December 18th: Complete Phase IV Source Documentation and Submit Final Project to UMUC Assignments Folder

# 4. User Guide

This software will assist home-based business in tracking their clothing inventory, customers, and record sales. It will provide insights into the business that will aid in decision making such as when and how much inventory should be purchased.

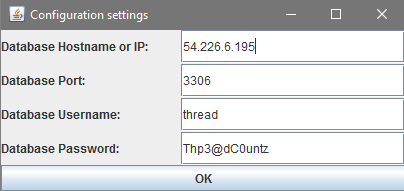
The software has an easy to use interface and can be run on standard desktop or laptop computers. An internet connection is required to use the software. All data is kept in the cloud so there is no need to worry about data loss if something happens to the computer.

## 4.1 Running the Software

* Using GitHub link – go to the following website:
* <https://github.com/ianchrisman/SynergySoft/blob/master/ThreadCount/store/ThreadCount.jar>
* Click “Download”, then double-click the downloaded .jar file. This will open the configuration settings window. From there, see the instructions below, in Section 4.2.

## 4.2 Configuration Settings Window

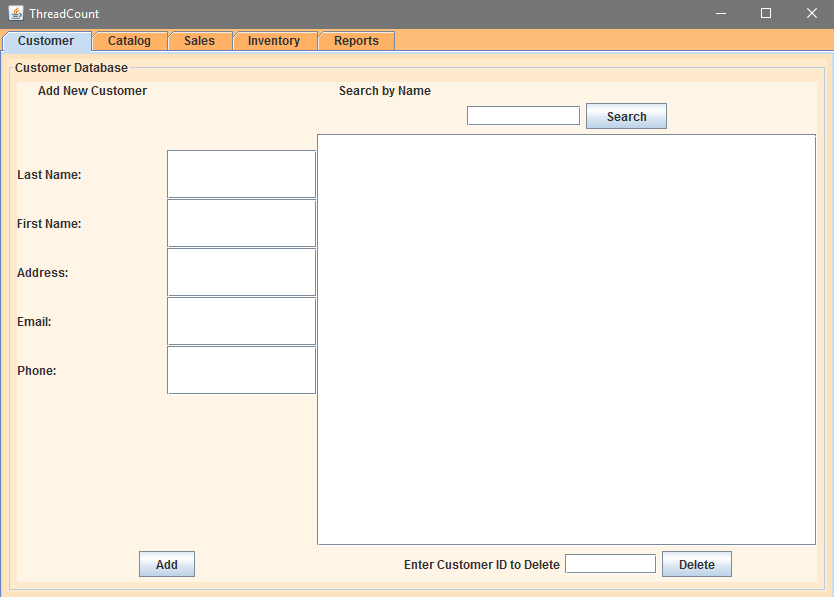
When the Configuration settings window opens, the required credentials should be populated. If they are not, use the following:



***Figure 5. Configuration Settings Window***

\*\*The Database Username will have threadDELETEME populated in the username field. This is so that the program does not auto login. Simply remove DELETEME from threadDELETEME.

## 4.3 Main Screen

******

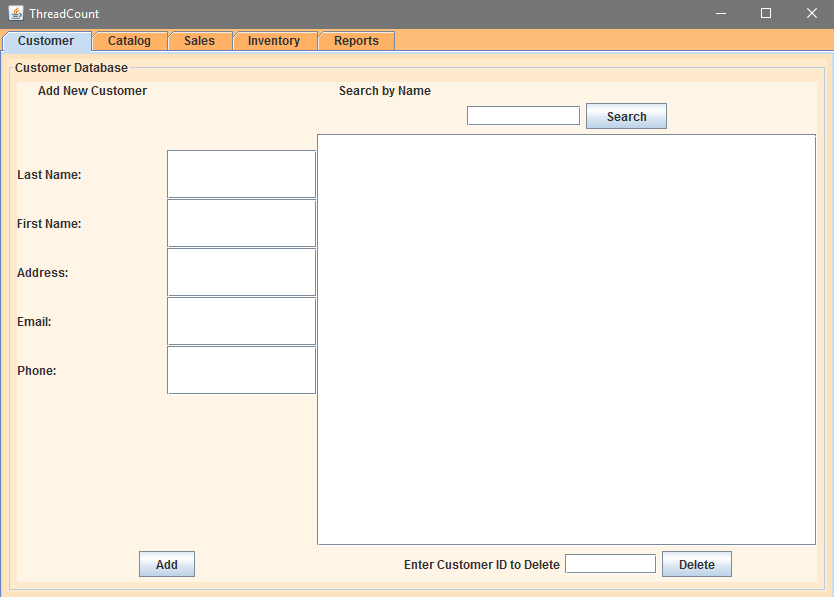
***Figure 6. ThreadCount Home screen***

* Customer – The Customer tab provides information about your prospective clients. Here, you will be able to find and store information about your customer, like names, addresses, emails, etc. You can also add and delete customers.
* Catalog – The Catalog tab provides information about any of the products that your business sells. This tab will provide information like type, size, color, etc., of the clothing items that you offer for sale. You can add, delete, or load bulk items from a file to your portfolio.
* Sales – Use this tab to enter new sales.
* Inventory – This tab allows you to see what you currently have in your inventory. You can search to find items that meet your customers’ needs.
* Reports – Several reports are at your fingertips. Use this tab to find out how much in sales your business did and other useful information that helps you grow your business.

Choose one of the tabs that are displayed on the top of the screen. This will take you to a new screen where you can find and edit items related to that category. Go ahead and click the Customer tab!

## 4.4 Customer

The Customer screen provides information about your prospective clients. Here, you will be able to find and store information about your customer, like names, addresses, emails, etc. You can also add, delete, and search for customers.

***Figure 7. ThreadCount Customer screen***

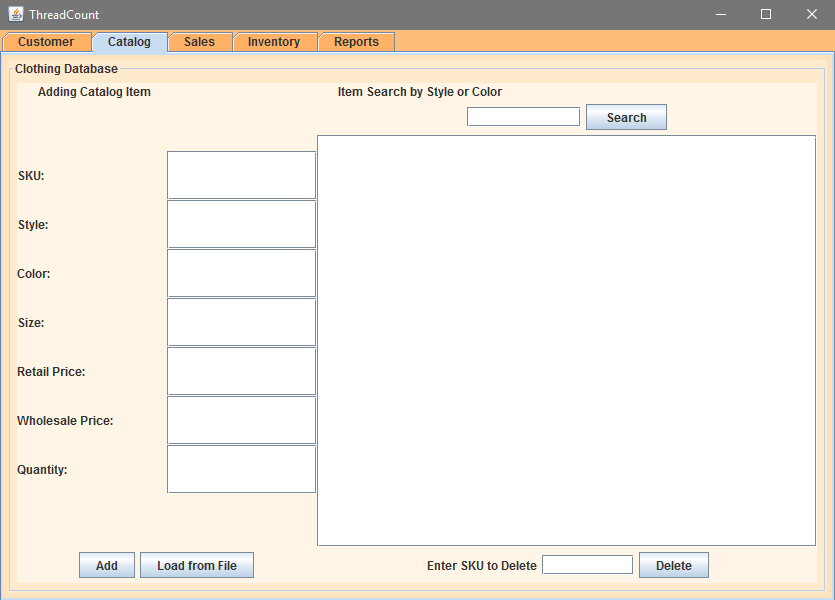
There are 3 main tasks that can be performed in the Customer screen:

* Search – Enter any part of a name and any entries that match the search will be displayed in the results area.
* Add new customer – Enter new customer information to add a new customer to your database!

Go ahead and add a new customer. Don’t worry, if it is a fictitious for now, you can delete it later. After adding the customer, try to search for it using the search tool.

## 4.5 Catalog

The Catalog tab provides information about any of the products that your business sells. This tab will provide information like type, size, color, etc., of the clothing items that you offer for sale. This is not intended for customers’ use. Here you can add, delete, or load bulk items from a file to your portfolio.

***Figure 8. ThreadCount Catalog screen***

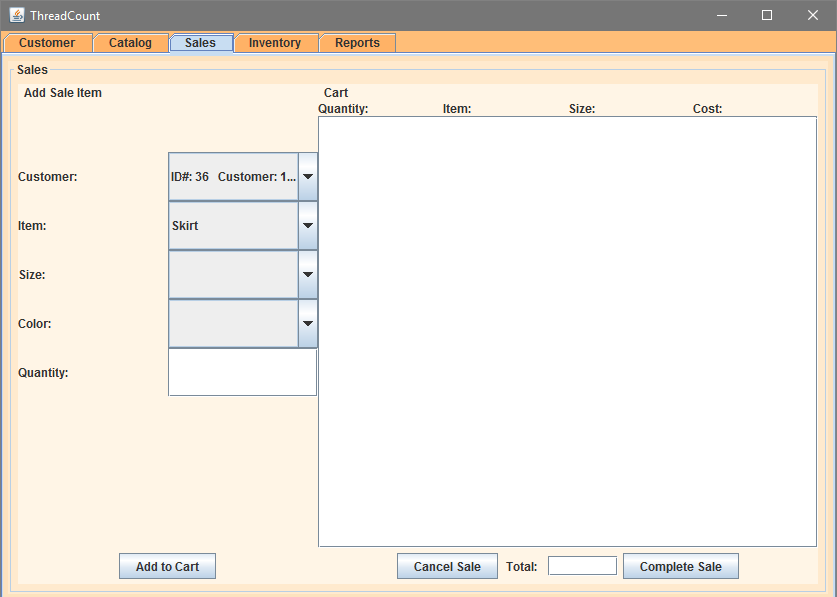
There are 3 main tasks that can be performed in the Catalog screen:

* Search – Enter any part of a name or type of clothing, and any matching items will appear in the Results area.
* Delete – After searching and finding the item that you want to delete, enter the SKU number of the item into the Enter SKU to Delete field and click the Delete button.
* Add new item – Enter new item information to add a new product to your sales offerings!
* To add a list of items, click the Load from File Button, which specifies acceptable file formats.

Go ahead and add an item to your product offering. Don’t worry if its fictitious for now, you can delete it later. After adding the item, try searching and deleting it!

## 4.6 Sales

The Sales screen is used to enter new sales.

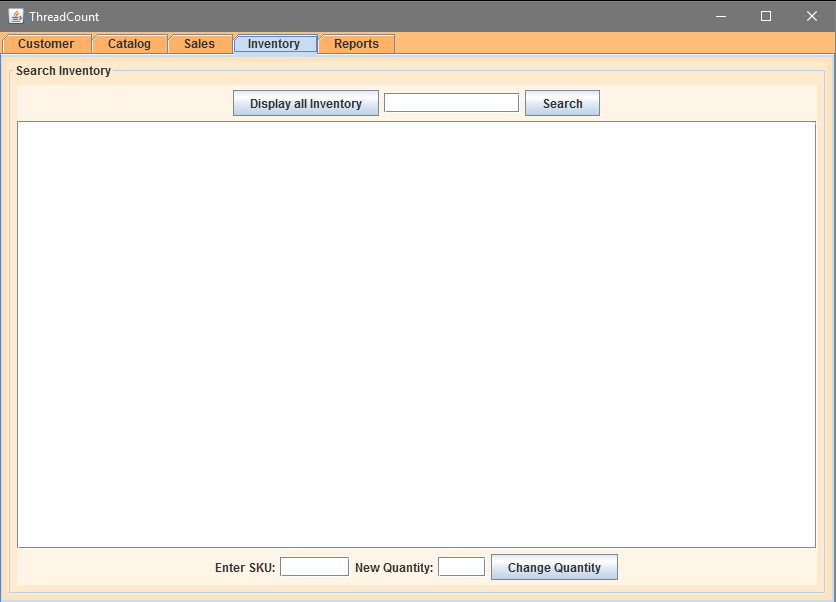
***Figure 9. ThreadCount Sale screen***

The Sales screen allows you to create a Cart and execute a sale to a customer:

* Customer is a drop-down menu allowing you to choose the correct customer that you are making the sale to
* Item is a drop-down menu that shows the items are currently in your inventory.
* Size is a drop-down menu that shows the sizes available for the item previously selected
* Color is a drop-down menu that shows the colors available for the item previously selected
* After filling out the customer, item, size, price and quantity, click the Add to Cart button to add the listing to the Cart.
* After the Cart is correct, click on Complete Sale to finalize the transaction.

## 4.7 Inventory

The Inventory screen allows you to see what you currently have in your inventory. You can search to find items that meet your customers’ needs. You can also click the Display All Inventory button to see a complete list.

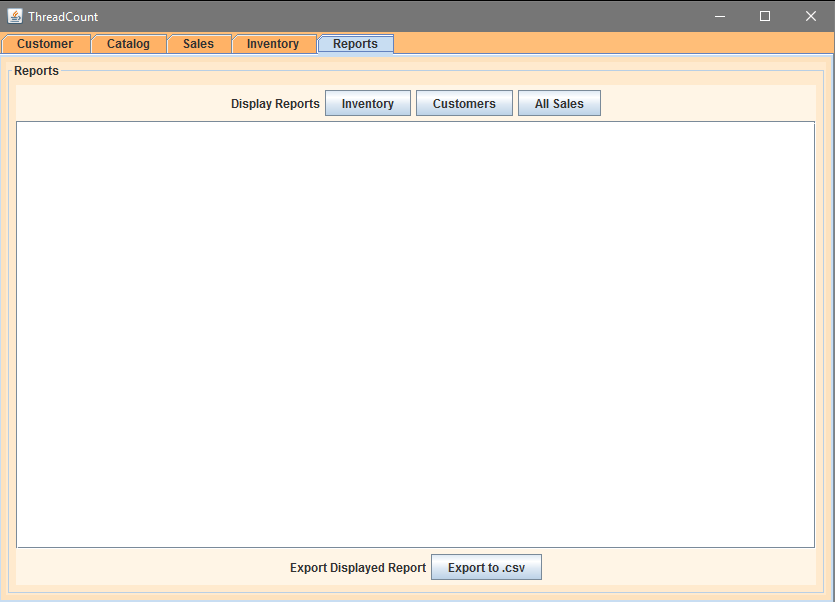
*** Figure 10. ThreadCount Inventory screen***

There are 3 main tasks that can be performed in the Inventory screen:

* Search – Enter any part of a name or type of clothing, and any matching items will appear in the Results area.
* Search All – To see all the items within your inventory, simply click the Display all Inventory button
* Modify – After searching and finding the item that you want, enter the SKU number of the item into the Enter SKU field, next enter the new quantity, and finally click the Change Quantity button.

## 4.8 Reports

Use the Reports screen to find out how much your business did in sales, and other useful information that helps you grow your business.

***Figure 11. ThreadCount Reports screen***

The Reports screen provides information and key insights about your business:

* Inventory – This report will give you a complete listing of your current inventory.
* Customers –This report outputs a complete listing of all your customers in alphabetical order.
* All Sales – Outputs a sales report with all the sales to date statistics.

## 4.9 Developer’s Notes

To create tables, first obtain threadcounts.sql by going to the following GitHub repository:

https://github.com/ianchrisman/SynergySoft/blob/master/threadcounts.sql

### 4.9.1 ThreadCount database schema

If your database does not already contain the requisite tables, you can import this schema like so:

prompt> mysql -u <username> -p threadcounts < threadcounts.sql

# 5. Test Plan

This test plan will illustrate the testing plan, approach, and overall framework that will drive the testing of ThreadCount. The Test Plan includes:

* Test Team
* Test Environment
* Test Acceptance Criteria
* Test Deliverables
* Graphical User Interface (GUI) Test Plan and Test Cases
* Database Test Plan and Test Types

## 5.1 Test Team

The table below shows the members of the Testing Team. Coding Leads will collaborate with their respective Testing Leads in order to get the project fully functional.

| **Team Member** | **Lead Area** | **Test Responsibilities** |
| --- | --- | --- |
| Ian Chrisman | Front End Coding | Create and manage front end coding of project |
| Justin Jennings | Back End Coding | Create and manage back end coding of project |
| Edward Caro | GUI Testing | Test front end of project |
| Colin Crowley | Database Testing | Test back end of project |

***Figure 12. Testing Responsibilities***

### 5.1.1 Lead Tester Responsibilities (Expanded)

Creates the test plan, creates and reviews test data and queries, analyzes and complies test results for final deliverable, conducts test queries, documents individual test results

## 5.2 Test Environment

In order to fully test the application, the following will be required for the testing environment. Each team member will take part in testing to some degree. This is done to test a variety of operating systems and other variables that might not appear in just one system.

| **Requirements** | **Client Side** |
| --- | --- |
| Hardware | Intel/AMD processor 1GHz+ |
| Operating System | Windows 7, 10, Mac OS X |
| Software | JDK 8.0, NetBeans IDE |

***Figure 13. Test Environment***

## 5.3 Test Acceptance Criteria

Testing will use Pass/Fail criteria for all test cases. A test case will only be considered passed if the result matched the expected result. Otherwise it will be considered failed.

## 5.4 Test Deliverables

The following deliverables will be updated and added to throughout the lifecycle of the project.

| **Deliverable** | **Description** | **Due Date** |
| --- | --- | --- |
| Test Plan | Document that provides an overall guidance of testing efforts throughout the project’s development cycle | November 13, 2016 |
| Phase I Test Report | Testing to ensure coding of Phase 1 is working correctly | November 18, 2016 |
| Phase II Test Report | Continued testing of Phase 1 along with the testing of Phase 2 coding | December 1, 2016 |
| Phase III Test Report | Testing to ensure all coding in the project is working as designed | December 8, 2016 |
| Phase IV/Final Test Report | Final testing to ensure all implemented functionality is working properly | December 17, 2016 |

***Figure 14. Testing Deliverables***

### 5.4.1 Phase I Testing

| **Phase I** | | | |
| --- | --- | --- | --- |
| **Task** | **Description** | **Assigned to** | **Due Date** |
| GUI Testing | * Test the five tabs * Test the functionality of the five tab buttons thoroughly | Edward | November 18, 2016 |
| Database Testing | * Input tests * Retrieval tests * Report tests | Colin | November 18, 2016 |
| Execute Testing Plan | Run tests and document any problems | Colin/Edward | November 20, 2016 |
| Phase I Testing Report | Compile a report of the documented problems | Colin/Edward | November 25, 2016 |
| Review of Report | Review the report and make any necessary changes to the code | Group | November 26, 2016 |

***Figure 15. Phase II Testing***

### 5.4.2 Phase I Testing Report

| **Phase I** | |
| --- | --- |
| **Bugs Detected** | **Resolved/Unresolved** |
| Application locks up after not being used for a period of time | Un-resolved |
| No confirmation on screen that customer has been added/deleted | Resolved |
| Information returned from search shows up as one line | Resolved |

***Figure 16. Phase II Testing Report***

### 5.4.3 Phase II Testing

| **Phase II** | | | |
| --- | --- | --- | --- |
| **Task** | **Description** | **Assigned to** | **Due Date** |
| GUI Testing | * Test that the Executable Jar File works in Windows * Test Configuration Window * Test Updated Catalog, Sales, and Reports tabs | Edward | December 1, 2016 |
| Database Testing | * Input tests * Retrieval tests * Report tests | Colin | December 1, 2016 |
| Execute Testing Plan | Run tests and document any problems | Colin/Edward | December 3, 2016 |
| Phase II Testing Report | Compile a report of the documented problems | Colin/Edward | December 4, 2016 |
| Review of Report | Review the report and make any necessary changes to the code | Group | December 5, 2016 |

***Figure 17. Phase II Testing***

### 5.4.4 Phase II Testing Report

| **Phase II** | |
| --- | --- |
| **Bugs Detected** | **Resolved/Unresolved** |
| Unable to open and load JAR file on Linux Machine | Resolved |
| Unable to load Jar file while not in the ThreadCount folder | Resolved |
| Unable to delete customer using customer ID | Resolved |

***Figure 18. Phase II Testing Report***

### 5.4.5 Phase III Testing

| **Phase III** | | | |
| --- | --- | --- | --- |
| **Task** | **Description** | **Assigned to** | **Due Date** |
| GUI Testing | * Test that the Configuration Settings window closes correctly * Test Load from File in Catalog Tab * Test Export to .csv in Reports Tab | Edward | December 10, 2016 |
| Database Testing | * Input tests * Retrieval tests * Report tests | Colin | December 10, 2016 |
| Execute Testing Plan | Run tests and document any problems | Colin/Edward | December 10, 2016 |
| Phase II Testing Report | Compile a report of the documented problems | Colin/Edward | December 11, 2016 |
| Review of Report | Review the report and make any necessary changes to the code | Group | December 12, 2016 |

***Figure 19. Phase III Testing***

### 5.4.6 Phase III Testing Report

| **Phase III** | |
| --- | --- |
| **Bugs Detected** | **Resolved/Unresolved** |
| ThreadCount GUI is incorrectly named ThreadCounts | Resolved |
| No verification that a .csv file has been loaded correctly | Resolved |
| Search Results: Returned results from Customer & Catalog tabs display information starting at the bottom of window | Un-resolved |

***Figure 20. Phase III Testing Report***

### 5.4.7 Phase IV Testing

| **Phase IV** | | | |
| --- | --- | --- | --- |
| **Task** | **Description** | **Assigned to** | **Due Date** |
| GUI Testing | * Test Error Handling on Customer & Catalog tabs * Test Sales & Inventory tab updates | Edward | December 15, 2016 |
| Database Testing | * Input tests * Retrieval tests * Report tests | Colin | December 15, 2016 |
| Execute Testing Plan | Run tests and document any problems | Colin/Edward | December 15, 2016 |
| Phase II Testing Report | Compile a report of the documented problems | Colin/Edward | December 16, 2016 |
| Review of Report | Review the report and make any necessary changes to the code | Group | December 17, 2016 |

***Figure 21. Phase IV Testing***

### 5.4.8 Phase IV Testing Report

| **Phase IV** | |
| --- | --- |
| **Bugs Detected** | **Resolved/Unresolved** |
| Sales tab: There is a mismatch in what is available in inventory and what is shown on drop-down menus, particularly the Color drop-down | Un-resolved |
| Input validations for all text fields | Un-resolved |

***Figure 22. Phase IV Testing Report***

### 5.4.9 Final Testing Report

| **Final** | |
| --- | --- |
| **Bugs Detected** | **Resolved/Unresolved** |
| Application locks up after not being used for a period of time | Un-resolved |
| No confirmation on screen that customer has been added/deleted | Resolved |
| Information returned from search shows up as one line | Resolved |
| Unable to open and load JAR file on Linux Machine | Resolved |
| Unable to load Jar file while not in the ThreadCount folder | Resolved |
| Unable to delete customer using customer ID | Resolved |
| ThreadCount GUI is incorrectly named ThreadCounts | Resolved |
| No verification that a .csv file has been loaded correctly | Resolved |
| Search Results: Returned results from Customer & Catalog tabs display information starting at the bottom of window | Un-resolved |
| Sales tab: There is a mismatch in what is available in inventory and what is shown on drop-down menus, particularly the Color drop-down | Un-resolved |
| Input validations for all text fields | Un-resolved |

***Figure 23. Final Testing Report***

## 5.5 GUI Test Plan

The objective of the GUI test plan is to assure that the ThreadCount software application is working correctly according to the User’s guide in section 4. This section is subject to change. Additional functionality may be added to the project or functions may be removed as needed. There is a back end portion for this application, but it is out of scope for the GUI test and accounted for in the Database test plan.

### 5.5.1 Scope

The purpose of this testing is to make sure the GUI is working and interfacing correctly with other modules. This testing is carried out by clicking the various tab buttons to ensure the output is as expected.

### 5.5.2 GUI Test Cases

Below are the current test cases for the GUI portion of the project. The list will change as the project progresses.

| **GUI Test Cases** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Test #** | **Test Objective** | **Test Steps** | **Expected Result** | **P/F** | **Comments** |
| 1.1 | Start Application in Windows | Double Click on the ThreadCount JAR | The ThreadCount Configuration Settings Window will open | P |  |
| 1.2 | Start Application in IOS | Double Click on the ThreadCount JAR | The ThreadCount Configuration Settings Window will open | P |  |
| 1.3 | Start Application in Linux | Double Click on the ThreadCount JAR | The ThreadCount Configuration Settings Window will open | P |  |
| 2.1 | Confirm that Configuration Settings work | Enter Configuration settings and click OK | The ThreadCount application loads on system | P |  |
| 2.2 | Confirm that the Configuration Window closes correctly | On the Configuration settings window, click the X close button | The Configuration Settings Window will close | P |  |
| 3.1 | Confirm that the Customer tab opens | Click the Customer tab | A new screen will open where customer information can be entered | P |  |
| 4.1 | Confirm that the Catalog tab opens | Click the Catalog tab | A new screen will open where Catalog information can be entered | P |  |
| 4.2 | Test Load from File in Catalog Tab | In the Catalog tab, Click Load from File. Navigate to the files location and click Open | The .csv files contents will be loaded into the database | P |  |
| 5.1 | Confirm that the Sales tab opens | Click the Sales tab | A new screen will open where Sales information can be entered | P |  |
| 5.2 | Confirm that within the Sales Tab, Item & Size drop-down list works | On the Sales Tab, click the drop-down list next to Item or Size | List of available Items or Sizes will be displayed in the drop-down list | P |  |
| 6.1 | Confirm that the Inventory tab opens | Click the Inventory tab | A new screen will open where Inventory information can viewed | P |  |
| 7.1 | Confirm that the Reports tab opens | Click the Reports tab | A new screen will open where Reports can be viewed | P |  |
| 7.2 | Test Export to .csv in Reports Tab | In the Reports tab, select the type of Report you want to export than click Export to .csv. Name, then Navigate to where you would like to save the file and click Save | The report will be compiled and saved to the location you selected | P |  |

***Figure 24. GUI Test Cases***

## 5.6 Database Test Plan

This is the test plan for the back end Database (“DB”) of the ThreadCount inventory management software application. ThreadCount is used to enable analysis, storage and retrieval of customer and inventory information for a small-to-mid sized clothing retailer. There is a front end GUI interface for this application, but it is out of scope for the DB test and accounted for in the GUI test plan.

### 5.6.1 Scope

The following information is critical to the performance of the inventory management system database:

* The type fidelity of DB information
* That the DB schema accurately captures the required inventory and customer information

### 5.6.2 Database Test Types

* Input tests
  + Input tests will use MySQL commands to determine if customer and inventory data is correctly formatted and saved in the DB according to the design requirements
* Retrieval tests
  + Retrieval tests will use MySQL queries to determine if customer and inventory data is:
    - Retrieved and not altered during queries
    - Correctly pulls the needed data
* Report tests
  + Report tests will use MySQL queries to determine if customer and inventory data is:
    - Displayed in the correct format
    - Displays the correct DB fields and attributes
    - Completes in a reasonable time frame

### 5.6.3 Database Test Approach

* Customer Information tests
  + SQL command to add Last Name, First Name, Address, Email, Phone info
  + SQL command to get Last Name, First Name, Address, Email, Phone info
* Catalog Information tests
  + SQL command to add Type, Color, Size, Brand, Collection info
  + SQL command to get Type, Color, Size, Brand, Collection info
* Sales Information tests
  + SQL command to add Customer, Item, Cost, Quantity info
* Reports Information tests
  + SQL command to generate inventory report
  + SQL command to generate customer report
* Inventory Information tests
  + SQL command to generate inventory report

### 5.6.4 Database Test Activities and Schedule

* Analyze and review database requirements (required input)
* Prioritize risks and identify most likely scenarios (required input)
* Create test queries based on scenarios
* Create sample test data
* Finalize test queries
* Finalize sample test data
* Conduct test approach for all five test scenarios
* Compile and analyze test results
* Prepare final deliverables

### 5.6.5 Risks

The following items and events present risk to successful test implementation and completion:

* Insufficient test DB size. Ex., small sample DB vs. one of similar size and complexity to real DB
  + Mitigated by random data generation
* Insufficient number of test inputs to validate schema
  + Conduct analysis to identify most likely input scenarios

### 5.6.6 Database Test Cases

| **Database Test Cases** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Test #** | **Test Objective** | **Test Steps** | **Expected Result** | **P/F** | **Comments** |
| 1.1 | Confirm commands for Customer tab will function | SQL command to add Last Name, | Command creates new entry in DB under last name | P |  |
| 1.2 | Confirm commands for Customer tab will function | SQL command to add First Name, | Command creates new entry in DB under first name | P |  |
| 1.3 | Confirm commands for Customer tab will function | SQL command to add Address | Command creates new entry in DB under address | P |  |
| 1.4 | Confirm commands for Customer tab will function | SQL command to add Email | Command creates new entry in DB under email | P |  |
| 1.5 | Confirm commands for Customer tab will function | SQL command to add Phone | Command creates new entry in DB under phone | P |  |
| 1.6 | Confirm commands for Customer tab will function | SQL command to search DB for customer info | Command successfully searches across last name, first name, address, email, phone fields and returns relevant results | P |  |
| 1.7 | Confirm commands for Customer tab will function | SQL command to delete customer info | Command successfully deletes customer row info based on customer id | P |  |
| 2.1 | Confirm commands for Catalog tab will function | SQL command to add SKU | Command creates new entry in DB under SKU | P |  |
| 2.2 | Confirm commands for Catalog tab will function | SQL command to add Style | Command creates new entry in DB under Style | P |  |
| 2.3 | Confirm commands for Catalog tab will function | SQL command to add Color | Command creates new entry in DB under color | P |  |
| 2.4 | Confirm commands for Catalog tab will function | SQL command to add Size | Command creates new entry in DB under size | P |  |
| 2.5 | Confirm commands for Catalog tab will function | SQL command to add Retail Price | Command creates new entry in DB under retail price | P |  |
| 2.6 | Confirm commands for Catalog tab will function | SQL command to add Wholesale Price | Command creates new entry in DB under wholesale price | P |  |
| 2.7 | Confirm commands for Catalog tab will function | SQL command to add Quantity | Command creates new entry in DB under quantity | P |  |
| 2.8 | Confirm commands for Catalog tab will function | SQL command to search across SKU, style, color, size, retail price, wholesale price, quantity | Command returns valid results from fields of new entry in DB under SKU, style, color, size, retail price, wholesale price, quantity | P |  |
| 2.9 | Confirm commands for Catalog tab will function | SQL command to delete row of data based on SKU | Command deletes a row of data from the DB based on the inputted SKU | P |  |
| 3.1 | Confirm commands for Sales tab will function | SQL command to add customer ID | Command creates entry under customer ID | P |  |
| 3.2 | Confirm commands for Sales tab will function | SQL command to add item | Command creates entry under item | P |  |
| 3.3 | Confirm commands for Sales tab will function | SQL command to add size | Command creates entry under size | P |  |
| 3.4 | Confirm commands for Sales tab will function | SQL command to add cost | Command creates entry under cost | F | Cost removed from options to change |
| 3.5 | Confirm commands for Sales tab will function | SQL command to add quantity | Command creates entry under quantity | P |  |
| 3.6 | Confirm commands for Sales tab will function | SQL command to complete sale and enter sale record to DB | Command creates entry under sale record | P |  |
| 4.1 | Confirm commands for Inventory tab will function | SQL command to change quantity based on SKU | Command successfully edits quantity entry tied to an inputted SKU | P |  |
| 4.2 | Confirm commands for Inventory tab will function | SQL command to search and return inventory results based on input of SKU | Command successfully returns relevant field values based on SKU entry | F | Search only returns based on Item Style |
| 5.1 | Confirm commands for Reports tab will function | SQL command to query inventory report | Command successfully returns properly formatted inventory report | P |  |
| 5.2 | Confirm commands for Reports tab will function | SQL command to query customer report | Command successfully returns properly formatted customer report | P |  |
| 5.3 | Confirm commands for Reports tab will function | SQL command to query sales/month report | Command successfully returns properly formatted sales.month report | F | This functionality has been removed |
| 5.4 | Confirm commands for Reports tab will function | SQL command to query all sales report | Command successfully returns properly formatted all sales report | P |  |

***Figure 25. Database Test Cases***

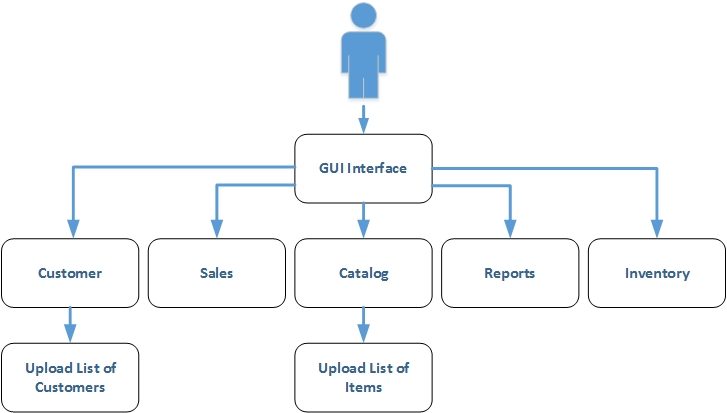
# 6. Design

The ThreadCount software application utilizes a GUI interface to allow the user to manage inventory, sales, and customer information for a small clothing boutique. The user interface will enable the application user to display various reports about sales, customer, and inventory data.

This section will provide specifications for the system architecture in a modular format along with detailing the class and structure design of the application. Diagrams and descriptions are provided to illustrate the program’s functionality.

## 6.1 Overall Approach

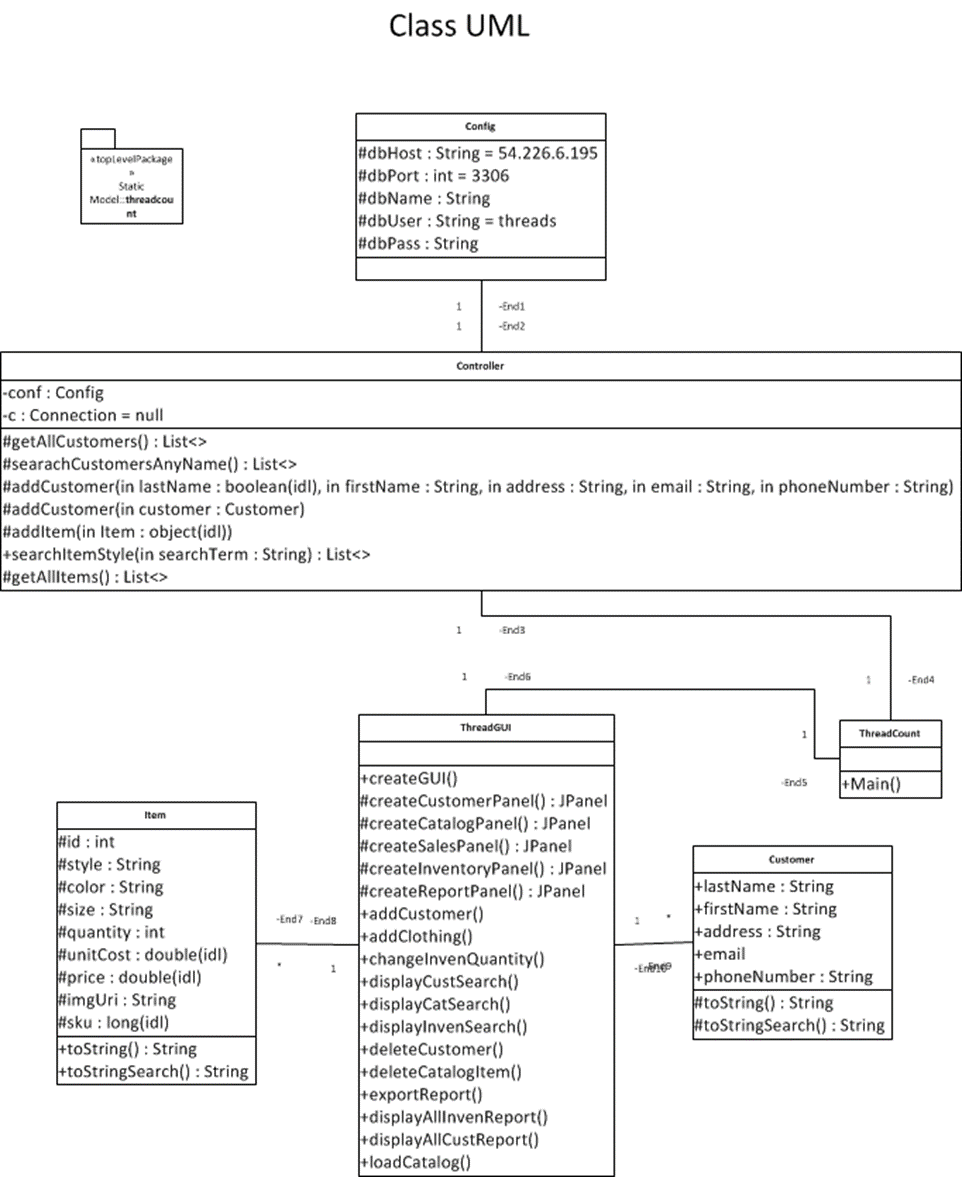
### 6.1.1 Workflow



***Figure 26. Workflow diagram***

## 6.2 Program Structure

### 6.2.1 UML Diagram

***Figure 27. UML diagram***

### 6.2.2 Classes, Methods, Fields, & Interfaces

* Class ThreadCount
* Class ThreadGUI
* Class Item
* Class Customer
* Class ReportBuilder
* Class Controller
* Class Config
* Class Cart

### 6.2.3 Class Design

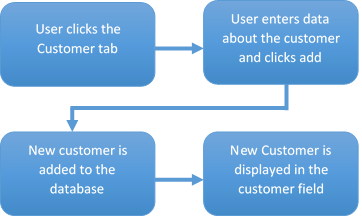
| **Class** | **Description** |
| --- | --- |
| ThreadCount | This class is the starting point for the application. It instantiates the GUI. |
| ThreadGUI | This class defines the GUI for the application, and sets up ActionListeners, etc. |
| Item | This class contains the data structure for a catalog item, as well as related methods. |
| Customer | This class contains the data structure for Customer information, as well as related methods. |
| ReportBuilder | This class will be used to generate various reports on data points of interest to the user. |
| Controller | This class does all the SQL work, object instantiation, etc., and exposes useful methods to the GUI class. |
| Config | This class hold the information required to connect to the database. If invalid credentials are supplied, it will pop up a window which will allow the user to supply the correct information. |
| Cart | This class holds the contents of the shopping cart. |

***Figure 28. Class Design table***

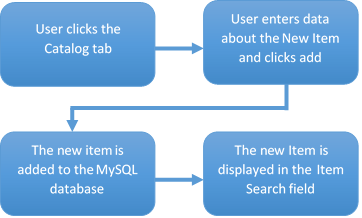
## 6.3 User Interface

The user interface will be a single window that dynamically changes depending upon which tab is selected.

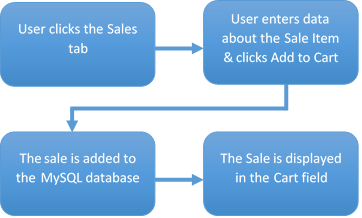
### 6.3.1 Customer tab flow chart



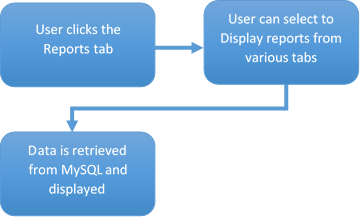
### 6.3.2 Catalog tab flow chart



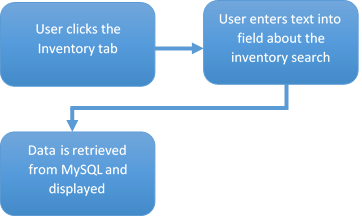
### 6.3.3 Sales tab flow chart



### 6.3.4 Reports tab flow chart

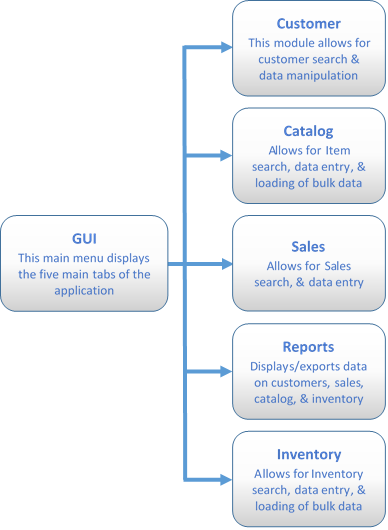


### 6.3.5 Inventory tab flow chart



## 6.4 GUI

### 6.4.1 GUI Workflow



### 6.4.2 GUI Design

The design of the GUI provides visual and conceptual elements for the user in the form of Java Buttons, Text Fields, and Labels. The GUI is designed to take value inputs from the user to manipulate data within the MySQL database.

### 6.4.3 GUI Components

| **Component** | **Description** |
| --- | --- |
| JFrame | Creates a top-level window with a border and title, in which other GUI components are placed. |
| JTabbedPane | Allows several panels to share the same space. The ThreadCount GUI will have a tabbed pane that includes the Customer, Catalog, Sales, Reports, and Inventory tabs. |
| JPanel | Multiple JPanels will display GUI components like JLabels, JTextfields, and JButtons. JPanels include customerPanel, catalogPanel, salesPanel, reportPanel, and inventoryPanel. |
| JComponent | Adds components to subpanels, such as centerSalesComponent and rightCartComponent. |
| JLabel | Displays unselectable text labels that identify corresponding fields and buttons. Labels used in the ThreadCount GUI include Last Name:, First Name:, Address:, Style:, Color:, Size:, Cost:, as well as Display Reports, Export Reports, etc. |
| JTextfield | JTextfields accompany the labels, and allow users to edit a line of text in order to input information. The textfields are used for entering customer and clothing item information, as well as search functions. They include firstNameText, emailText, searchCustomerText, styleText, colorNameText, searchClothingText, etc. |
| JButton | JButtons have labels and can be clicked in order to generate an event. JButtons include addCustomerButton, customerSearchButton, addSaleButton, inventoryButton, salesReportButton, displayAllButton, loadButton, etc. |
| JScrollPane | Scroll panes create a scrollable view of a component, and the ThreadCount GUI has shoppingCart, reportPane, and inventoryPane scroll panes. |
| JComboBox | The combobox allows the user to choose one of several options. Our GUI uses comboboxes for selecting size and item. |

***Figure 29. GUI Components***

### 6.4.4 GUI Tabs

ThreadCount uses the following tabs to allow the user to manipulate customer, catalog, sales, reports, and inventory data.

| **Component** | **Description** |
| --- | --- |
| Customer | The Customer tab provides information about prospective clients. The left side of the panel allows the user to add a new customer. The right side of the panel allows the user to search for a customer. |
| Catalog | The Catalog tab provides information about the products that the user’s business sells. The left side of the panel allows the user to add a new item. The right side of the panel allows the user to search for an item. The bottom section allows the user to load bulk items from a .csv file along with the ability to delete an item using the SKU number. |
| Sales | The Sales tab is used to enter new sales. The left side of the panel allows the user to add a new sale to cart. The right side of the panel allows the user to see all the sales in the cart. The bottom section allows the user to complete or cancel a sale. |
| Inventory | The Inventory tab allows the user to see what items are currently in the inventory. The top half of the panel allows you to search for an item by Item Style and the ability to display all items within the inventory. The bottom half of the panel allows the user to enter a new quantity for an item using the SKU number. |
| Reports | The Reports tab is used to find out how much in sales your business did, and other useful information. The top half of the panel allows the user to display reports for inventory, customers and all sales. The bottom half of the panel allows the user to export reports to a file. |

***Figure 30. GUI Tabs***

## 6.5 Database

The ThreadCount application will use a MySQL as the database to store its data.

### 6.5.1 Sample SQL Schemas

Below are the schemas for the ‘catalog’, ‘customer’, and ‘sales’ tables.

Catalog:

+-----------+---------------+------+-----+-------------------+-----------------------------+

| Field | Type       | Null | Key | Default        | Extra                    |

+-----------+---------------+------+-----+-------------------+-----------------------------+

| id     | int(11)    | NO   | PRI | NULL           | auto\_increment           |

| style | varchar(30)   | NO   | | NULL           |                          |

| color | varchar(15)   | NO   | | NULL           |                          |

| size   | varchar(15)   | NO   | | NULL           |                          |

| quantity  | int(11)    | NO   | | NULL           |                          |

| unit\_cost | decimal(10,2) | NO   | | NULL           |                          |

| price | decimal(10,2) | NO   | | NULL           |                          |

| img\_uri   | varchar(50)   | YES  | | NULL           |                          |

| sku    | bigint(20) | NO   | | NULL           |                          |

| created   | timestamp | NO   | | CURRENT\_TIMESTAMP | on update CURRENT\_TIMESTAMP |

+-----------+---------------+------+-----+-------------------+-----------------------------+

Customer:

+--------------+--------------+------+-----+-------------------+-----------------------------+

| Field     | Type      | Null | Key | Default        | Extra                    |

+--------------+--------------+------+-----+-------------------+-----------------------------+

| id        | int(11)   | NO   | PRI | NULL           | auto\_increment           |

| last\_name | varchar(30)  | NO   | | NULL           |                          |

| first\_name   | varchar(30)  | NO   | | NULL           |                          |

| address   | varchar(100) | NO   | | NULL           |                          |

| email     | varchar(30)  | NO   | | NULL           |                          |

| phone\_number | varchar(15)  | NO   | | NULL           |                          |

| created   | timestamp | NO   | | CURRENT\_TIMESTAMP | on update CURRENT\_TIMESTAMP |

+--------------+--------------+------+-----+-------------------+-----------------------------+

Sales:

+-------------+---------------+------+-----+-------------------+-----------------------------+

| Field    | Type       | Null | Key | Default        | Extra                    |

+-------------+---------------+------+-----+-------------------+-----------------------------+

| id       | int(11)    | NO   | PRI | NULL           | auto\_increment           |

| customer\_id | int(11)    | NO   | | NULL           |                          |

| item\_id | int(11)    | NO   | | NULL           |                          |

| quantity | int(11)    | NO   | | NULL           |                          |

| sale\_price  | decimal(10,2) | YES  | | NULL           |                          |

| created | timestamp | NO   | | CURRENT\_TIMESTAMP | on update CURRENT\_TIMESTAMP |

+-------------+---------------+------+-----+-------------------+-----------------------------+

### 6.5.2 SQL Queries

Below are some examples of the kinds of SQL queries that will be run on these tables in order to get meaningful information to the user.

Search for a customer by first or last name:

PreparedStatement ps = c.prepareStatement("SELECT id, last\_name, first\_name, address, email, phone\_number FROM customer where last\_name like ? or first\_name like ?");

ps.setString(1,  "%" + searchTerm + "%");

   ps.setString(2,  "%" + searchTerm + "%");

Add an item to the catalog:

PreparedStatement ps = c.prepareStatement("INSERT INTO catalog (style, color, size, quantity, unit\_cost, price, sku) VALUES (?, ?, ?, ?, ?, ?, ?)");

   ps.setString(1, i.style);

   ps.setString(2, i.color);

   ps.setString(3, i.size);

   ps.setInt(4, i.quantity);

   ps.setDouble(5, i.unitCost);

   ps.setDouble(6, i.price);

   ps.setLong(7, i.sku);

### 6.5.3 Connectivity

There is a Config class, which handles the database connection information (database host, database name, username and password. The requirement is a MySQL database. The program connects to it using JDBC and J/Connector. The J/Connector driver is required to be in the classpath.

### 6.5.4 Database Components

The database for this application consists of a MySQL database to which the application connects, and stores its data. This database consists of three tables- ‘customer’, ‘catalog’, and ‘sales’.

### 6.5.5 File Import/Export Mechanism

The user will be able to import, in CSV format, a file specifying new items or quantities of items to be added to inventory. The ability to save generated reports to file is also provided.

## 6.6 Design Phases

### 6.6.1 Phase 1 (Due Nov 27th)

**Expectations:**

* Basic GUI functionality, with five tabs working correctly
* Sample Database created
* GUI and MySQL databases connected successfully
* Sample queries of database can be performed

### 6.6.2 Phase 2 (Due Dec 4th)

**Expectations:**

* Several queries of sample database can be performed, with expected results
* GUI allows user to retrieve and display a variety of data from database

### 6.6.3 Phase 3 (Due Dec 11th)

**Expectations:**

* Database with realistic amount of inventory, sales, and customer data created
* Several queries of database are performed successfully
* GUI allows user to retrieve and display data from database
* GUI allows user to modify database, including adding and deleting information

### 6.6.4 Phase 4 / Final (Due Dec 18th)

**Expectations:**

* Final Project and Documentation posted for grading, including:
  + Overview - Summary of individual contributions
  + Project Plan
  + Requirements Specification
  + System Specification
  + Users Guide
  + Test Plan and Results
  + Development History
  + Conclusions / Lessons Learned

# 7. Phase I

## 7.1 Phase I Milestones

| **Phase I Project Milestones** | **Due Date** | **Assigned to** | **Complete Y / N** |
| --- | --- | --- | --- |
| Generate mock GUI for discussion as part of User Guide | November 4, 2016 | Shawn | Y |
| Create basic ThreadCount GUI | November 10, 2016 | Ian | Y |
| Insert user scenarios | November 20, 2016 | Shawn | Y |
| Sample data created | November 27, 2016 | Justin | Y |
| Phase 1 Documentation and posting for Peer Review 2 | November 27, 2016 | Jennifer/Edward | Y |
| Phase 1 Testing Chart | November 27, 2016 | Colin | Y |
| SQL queries of sample database performed successfully | November 27, 2016 | Justin/Ian | Y |
| Update UML diagram | December 4, 2016 | Ian | N |
| Connect MySQL database and GUI | November 27, 2016 | Shawn/Justin | Y |
| Create How to Run instructions | November 27, 2016 | Ian | Y |
| Basic GUI functionality with tabs working correctly | November 27, 2016 | Ian | Y |
| Update source code, including Item.java, and adding delete methods to controller class | November 27, 2016 | Ian/Justin | Y |

***Figure 31. Phase I Milestones***

## 7.2 Roles and Responsibilities

* Ian Chrisman: Took the lead coding role for all major duties for this phase. Designed the UI and underlying structures and circulated test code for input. Justin Jennings also contributed source code during this phase.
* Colin Crowley, Jennifer Prizeman, Shawn Thompson, Edward Caro, and Justin Jennings: Contributed documentation sections/review as well as application validation/testing.

## 7.3 Phase I Recap

This week, Edward and Jennifer are responsible for the Phase I Source documentation. Ian has continued to improve the GUI’s functionality so that all tabs work properly, and has also added instructions on how to run the current version of the project. Justin has created the MySQL database, and Shawn has coordinated with Ian and Justin to ensure the connectivity between the GUI and database. Colin is creating a testing plan to determine whether the Phase 1 milestones have been met in terms of functionality

### 7.3.1 User Interface

* Customer adds and Searches are functioning properly
* Catalog adds and Searches are functioning properly
* Reports tab displays the inventory and customer reports successfully

### 7.3.2 How to Run Source Code in IDE

1. Place the .java files, in a package called threadcount.
2. Compile and run in your IDE of choice. We used NetBeans and Eclipse while writing this code, so it can be run in both.
3. It is recommended you download the latest versions so you have the most recent libraries available.
4. For Netbeans: Once you’ve created a package and have the source code imported you will need to add the the MySQL JDBC Library. While in Netbeans, click on the Projects tab, (this is usually on the left). Under the ThreadCount project right click on Libraries folder and click “add Library” Select MYSQL JDBC Driver. You should be able to compile and run the source code now.

### 7.3.3 Upcoming Functionality

* GUI allows user to modify real database, including adding and deleting information
* User can display inventory and different types of reports from real database
* GUI allows user to import data from an excel or .csv file
* All relevant queries of database can be performed successfully
* User can retrieve and display all relevant data from database
* Error handling
* Backup and encrypt database information
* Ability to download an executable JAR file to run ThreadCount program

## 7.4 Problems Encountered

No major problems have been encountered at this time. Bugs that have been detected are annotated in section 5.4.2 Phase I Testing Report.

## 7.5 Reconsiderations

As part of this section, we will consider the following feedback from the Peer Review 2.

* Popup message when the user enters invalid information
* Display search result information on separate lines with category labels in bold to make it easier to read

Also, Colin suggested during Phase I discussions that we change the three export buttons on the reports page to a single export button that saves off the displayed report, so Ian will implement this.

## 7.6 Schedule

We are currently on schedule to meet our deadlines.

## 7.7 Document Changes

* Section 4: updated with new screenshots
* Section 4.1: updated text to coincide with screenshots
* Section 4.5 & 4.6: rearranged to follow order of working GUI
* Section 5: Added section 5.4.2 Phase I Testing Report
* Section 5.5.2: Updated with the results from testing outcomes
* Section 5.6: Added Database Test Cases chart
* Section 6.4.3: Corrected figure numbers
* Section 6.4.4: Corrected wording to match the rest of the document
* Section 6.5: Corrected section numbering
* Section 6.6: Added section 6.6.4 Phase 4 / Final

# 8. Phase II

## 8.1 Phase II Milestones

| **Phase II Project Milestones** | **Due Date** | **Assigned to** | **Complete Y / N** |
| --- | --- | --- | --- |
| Complete Phase II Testing Chart | December 4, 2016 | Edward | Y |
| Add instructions for downloading and running program to User Guide | December 4, 2016 | Edward/Jennifer | Y |
| Create an executable JAR file to run ThreadCount program | December 4, 2016 | Ian | Y |
| Update UML diagram | December 4, 2016 | Shawn | Y |
| Add sole proprietor explanation to Project Background | December 4, 2016 | Jennifer | Y |
| Functionality of Delete methods | December 4, 2016 | Justin | Y |
| Exporting report into .csv file | December 4, 2016 | Ian/Justin | Y |
| Functionality of Sales tab | December 4, 2016 | Ian/Justin | Y |
| Phase II Source Documentation | December 4, 2016 | Edward/Jennifer | Y |
| Run .jar file successfully | December 4, 2016 | Group | Y |
| Size and Color combo-boxes, and ‘order-by’ sorting functionality | December 4, 2016 | Justin | Y |
| Pop-up window appears if database credentials do not work | December 4, 2016 | Justin | Y |

***Figure 32. Phase II Milestones***

## 8.2 Roles and Responsibilities

* Ian Chrisman updated source code for this phase, created an executable .jar file, and circulated it for input. Justin Jennings also updated source code during this phase, and created a Configuration Settings Window for testing.
* Colin Crowley, Jennifer Prizeman, Shawn Thompson, Edward Caro, and Justin Jennings: Contributed documentation sections/review as well as application validation/testing.

## 8.3 Phase II Recap

This week we continued to improve the program, evaluated what areas were a priority for the next two weeks, and updated items in prior documentation. A big step was creating an executable .jar file, which makes ThreadCount easy to download and run for our target user. Another was updating our UML diagram to reflect our current project. At this point, the vast majority of the functionality we set out to do has been successfully implemented.

### 8.3.1 User Interface

* Delete methods are functioning properly
* Size and Color combo boxes added
* Sales Tab functions properly

### 8.3.2 Upcoming Functionality

* ThreadCount program allows user to import data from file
* Error handling
* Backup and encryption of database information
* Log-in / credentials fully functional
* Shopping cart functionality

## 8.4 Problems Encountered

Bugs that have been detected are annotated in section 5.4.4 Phase II Testing Report.

## 8.5 Reconsiderations

This week we discussed how inventory will be added to the database. Initially we thought of hard-coding dropdown boxes for style, size, and color, then adding a description field that would be typed by the user. We wanted to limit the amount of user error that could occur while adding items. Our concern was that a typo could prevent the item from appearing in a later search for it. We wanted to keep the style dropdown from becoming too lengthy, so initially we decided that style would be general -- for example, “jeans”, “sweaters”, etc. After some discussion, however, we realized that the description would end up being key to finding specific pieces of inventory, so the style dropdown was not adding anything useful. Ultimately we decided that style should not be separate from description, and should be combined into one “style” field. On the other hand, size and color fields are more fixed, so hard-coding dropdown boxes for those would be manageable. The concern regarding lost items is alleviated by having a search which allows a user to search for all items of a certain size or color, and even do a blank search that returns all items in inventory.

## 8.6 Schedule

We are currently on schedule to meet our deadlines.

## 8.7 Document Changes

* Section 1.2: Updated Project Background
* Section 1.5: Updated Project Deliverables
* Section 1.6.1: Updated UML diagram
* Section 3.1: Updated Key Milestones
* Section 3.3: Updated Tasks and Schedule
* Section 4.1: Added instructions on downloading and executing .jar file to User Guide
* Section 4.2: Added instructions for Configuration Settings window
* Section 4.5: Updated Graphic and clarified target user
* Section 4.6: Updated Graphic
* Section 4.8: Updated Graphic
* Section 5.4.3: Added section
* Section 5.4.4: Added section
* Section 5.5.2: Updated GUI Test Cases
* Section 5.6: Updated Table to match formatting
* Section 6.2.1: Updated UML diagram
* Section 7.3.2: Amended section description

# 9. Phase III

## 9.1 Phase III Milestones

| **Phase III Project Milestones** | **Due Date** | **Assigned to** | **Complete Y / N** |
| --- | --- | --- | --- |
| Complete Phase III Testing Chart | December 10, 2016 | Edward/Colin | Y |
| Program allows user to import data from .csv file | December 11, 2016 | Ian | Y |
| Log-in credentials for database fully functional | December 11, 2016 | Justin | Y |
| Improve Sales tab and cart functionality | December 11, 2016 | Justin | Y |
| Program runs successfully on Linux machine | December 11, 2016 | Shawn | Y |
| Program allows .csv export for Inventory and Customer Reports | December 11, 2016 | Ian | Y |
| Sample .csv data file with inventory created to perform testing | December 11, 2016 | Colin | Y |
| Updated .jar file created | December 11, 2016 | Ian | Y |

***Figure 33. Phase III Milestones***

## 9.2 Roles and Responsibilities

* Ian Chrisman updated source code for this phase, including adding export and import functionality, shopping cart functionality, and error handling. Justin Jennings also updated source code, including shopping cart functionality and adding the option of creating tables in database.
* Colin Crowley researched security, SSL, and encryption issues for the database. He also created a sample inventory containing entry mistakes for testing purposes once error handling is implemented.
* Jennifer Prizeman, Shawn Thompson, and Edward Caro: Contributed documentation sections/review as well as application validation/testing.

## 9.3 Phase III Recap

This week we continued to improve the ThreadCount program and discussed the functionality elements that were in progress.

### 9.3.1 User Interface

* Configuration Settings window closes correctly now
* Importing and Exporting .csv files is working properly
* Program can be run on Windows, Mac, and Linux machines

### 9.3.2 Upcoming Functionality

* Add confirmation to let user know the import from file task has been completed successfully
* Export function to be available for reports in remaining tabs
* Error handling, including pop-ups that warn user of incorrect types, bad CSV input data, etc.
* Full functionality of Shopping cart in Sales Tab
* Encryption and/or backup of database information
* Create customer dropdown list in Sales Tab
* Add option of creating tables in database

## 9.4 Problems Encountered

Bugs that have been detected are annotated in section 5.4.6 Phase III Testing Report.

## 9.5 Reconsiderations

This week we discussed changing the customer text field within the Sales tab to a dropdown box. One concern was the customer list length. On the other hand, entering the customer ID number is not very intuitive for users. For this version of ThreadCount, we decided to use a dropdown for the customer name in the Sales tab. Since our target user is a sole proprietor, it is unlikely that the list would get so long that a simple scroll feature could not handle it. For a future version, we would like to implement a dynamic search feature, which would auto-populate as you type the customer’s name.

We also considered security and encryption strategies for information in the database, including whether adding SSL is practical for this version, or whether it should be a future feature. Since it would add complications at a late stage of the project, we’ve decided to leave open the possibility for implementing this feature in a future ThreadCount version.

## 9.6 Schedule

We are currently on schedule to meet our deadlines.

## 9.7 Document Changes

* Section 3.3: Updated section
* Section 4.1: Added GitHub instructions for running .jar file
* Section 5.4.5: Added section
* Section 5.4.6: Added section
* Section 5.5.2: Added new test cases, revised old ones
* Section 5.6: Rearranged sections to coincide with Section 5.5
* Section 5.6.6: Provided P/F results of testing
* Section 6.2.2: Amended Class list to agree with Section 6.2.3
* Section 6.2.3: Added Config and Cart classes, along with descriptions, to Class Design table

# 10. Final Phase

## 10.1 Final Phase Milestones

| **Final/Phase IV Project Milestones** | **Due Date** | **Assigned to** | **Complete Y / N** |
| --- | --- | --- | --- |
| Complete Final Phase Testing Chart | December 17, 2016 | Edward/Colin | Y |
| Full functionality of Shopping cart in Sales Module | December 17, 2016 | Justin | Y |
| Remove unnecessary unused space within GUI | December 17, 2016 | Ian | Y |
| Errors in imported sample .csv file handled properly | December 17, 2016 | Ian | Y |
| Error handling, including pop-ups that warn user of incorrect types, bad CSV input data, etc. | December 17, 2016 | Ian | Y |
| Add confirmation to let user know the import from file task has been completed successfully | December 17, 2016 | Ian | Y |
| Create Customer dropdown list in Sales tab | December 17, 2016 | Justin | Y |
| Encryption and/or backup of database information (AWS) | December 18, 2016 | Justin/Colin | Y |
| Masked password in log-in screen | December 18, 2016 | Ian | Y |
| Updated .jar file created | December 18, 2016 | Ian | Y |
| PDF Users Guide created | December 18, 2016 | Edward | Y |

***Figure 34. Final / Phase IV Milestones***

## 10.2 Roles and Responsibilities

* Ian Chrisman and Justin Jennings updated source code to address any outstanding issues.
* Jennifer Prizeman, Shawn Thompson, and Edward Caro, and Colin Crowley: Contributed documentation sections and documentation review, as well as application validation and testing.

## 10.3 Final Recap

This week we were able to complete all the basic functionality for the ThreadCount program. We added masked password field in place of a visible one. The ‘Clear Cart’ functionality that was added to the Sales module is working properly. The program can now properly display all sales, and the export .csv report function within the Sales module is working properly.

To make the program more visually appealing, we adjusted the windows to get rid of some unused space. We also modified the layout so that when the user makes the window smaller, the module tabs at the top (Customer, Catalog, etc.) will rearrange so that they can still be seen.

## 10.4 Problems Encountered

We did not have enough time to address all of the window resizing issues or implement JTable in the report windows.

## 10.5 Reconsiderations

This week we discussed our “must-haves” for the ThreadCount program, since the semester was ending. In determining our final changes, we considered Professor Duchon’s comments and the feedback we received from our classmates.

## 10.6 Document Changes

* Section 3.1: Updated project milestones
* Section 3.4: Removed, no longer relevant
* Section 4: Revised language
* Section 4.1: Reordered bulleted list
* Section 4.2: Updated text
* Section 4.3: Updated with user-friendly language and updated graphic
* Section 4.4: Updated with user-friendly language and updated graphic
* Section 4.5: Updated with user-friendly language and updated graphic
* Section 4.6: Updated with user-friendly language and updated graphic
* Section 4.7: Updated with user-friendly language and updated graphic
* Section 4.8: Updated with user-friendly language and updated graphic
* Section 4.9: Added this new Developer’s Notes section
* Section 5.4: Updated test deliverables
* Section 5.4.7: Added section
* Section 5.4.8: Added section
* Section 5.4.9: Added section
* Section 5.6.3: Revised test list
* Section 5.6.6: Updated results from testing latest version of software
* Section 6.4.4: Amended Reports description

## 10.7 Conclusion

### 10.7.1 Lessons Learned

A major lesson was the benefit of planning our project before we began coding. This helped us thoroughly consider our target user and create a set of relevant goals for the program.

We have also learned the value of dividing the workload among group members. It was helpful for each member to have concrete tasks and responsibilities each week, and for us to establish a schedule of pre-deadlines that carried through each week.

### 10.7.2 Design Strengths

Importing and Exporting .csv files is a great feature of this software. It provides the convenience of adding data in bulk and creating usable reports. Also, the main navigation through the program, via the horizontal tabs at the top, is very intuitive.

### 10.7.3 Limitations

In this version of ThreadCount, if a user wants to modify a customer’s information, they will need to delete the customer and re-add the correct customer information.

### 10.7.4 Suggestions for Future Improvement

Customer Module: For a future version, we would like to implement a dynamic search feature, which would auto-populate as the user types a customer’s name. Further, we would break down customer addresses into several fields; street, town/city, state, zip, and perhaps country. We would also add a button that would allow the user to update or modify customer information more easily.

Inventory Module: In a future version, we would implement JTable in order to present inventory information in a table. For now, we have added a Developer’s Notes section to the User Guide.

All Modules: In a future version, input validations would be added to all text fields to prevent unreliable data input into the database.

Other: A future version could have a 'reset password' prompt after the user first logs in with the default password, perhaps with secret questions that allow future password retrieval.

1. Hughes, B. (2015, Dec 30). *How to Improve Inventory Tracking: 5 Solutions for Small Businesses*. Retrieved from https://smallbiztrends.com/2015/12/improve-inventory-tracking-systems-top-5-solutions-for-small-businesses.html. See also, Lovering, J. (2016, Oct 16). *Small businesses consider Facebook critical to business success, survey says*. Mississauga News. Retrieved from http://www.mississauga.com/news-story/6913475-small-businesses-consider-facebook-critical-to-business-success-survey-says/ [↑](#footnote-ref-1)
2. Emmert, J.M. (2014, May 29). *The Big History of Direct Selling*. Retrieved from http://directsellingnews.com/index.php/view/the\_big\_history\_of\_direct\_selling#.WB545\_krLb0 [↑](#footnote-ref-2)