Software Engineering Final Project

Inventory Tracking Service

Michael Prescott

04/26/18

Abstract

This document will highlight the background story, requirements, analysis, design, implementation, and testing of Inventory Tracking Service. The background story will show the real-world application of this program and how it can be used. In the requirements section, it will be noted how the program meets the needs of the users. In the analysis, design, and implementation portions of this paper, the foundation for how this program was written will be documented. Lastly, the means of testing will be noted by how the program can hold up against certain situations.

Background Story

Every retail business needs a way of tracking their inventory. In many cases, programs that serve this purpose are confusing and clunky. Inventory Tracking Service is designed in a way that provides a simple interface and simple controls. This software would best suit a small business with a need for a lightweight way to store and manage their inventory. The only thing the customer would need is just an internet connection. No training would be required to operate a program, like this. The simple controls provide an easy and streamlined way for no prior knowledge of computers to be needed. compute

Requirements

This program was built around a handful of requirements. First, the program needed to be a lightweight solution, since too many commercial inventory tracking systems are quite clunky and outdated. Next, this program needed to provide an easy-to-use solution to other software, of the same nature. The program was deliberately designed in way that made it to were anyone would be able to use it. The program also needed a way to easily store data in a remote online database, for quick access. Also, security was needed to prevent, for instance, injections into the database, which can be deadly for a business. Next, the program needed to be able to add products, remove products, and be able to lookup the full inventory. All of these, combined, make up the full functional and non-functional requirements for this system.

Analysis

This system makes it certain that all requirements are met. To fulfill the lightweight requirement, the program makes use of simple Java libraries, minimizing the amount of custom operations needed. With the program using large and easy-to-read buttons, those with poor eyesight or limited knowledge of computers will have an easy time using the system. This program also makes use of the Java Database Connectivity interface to quickly and easily connect to the database containing the entire inventory. To meet the security needs, this program makes use of SQL prepared statements to prevent any malicious user from using SQL injections to manipulate the database. With the program having specialized add, lookup, and remove methods, it fulfills the requirements of a user changing data in the database.

Design

The system is built on very simple and powerful design patterns. In this system, the visitor, singleton, controller, factory, and observer design patterns are used. The visitor pattern is utilized in way that allows certain methods to work on any subtype, by using double dispatch. The singleton pattern is used multiple times. For instance, the class containing all work on the database is used as a singleton since there is no need for an actual instance of the class, let alone many sitting in the program. The controller method is used to abstract the main method to a single line and operate the window being displayed. The factory method is used with the visitor

pattern to provide a way to pass around any subtype of the window being displayed. Lastly, the observer method is used with the controller to, from a distance, work with user input to provide functionality to the buttons.

Implementation

This system is implemented in a simple and easy to follow way. When the initial window is opened, the controller is sitting and waiting for the user to click a button. When one of the buttons is clicked, the controller will determine what it was a perform the correct action. In the case of adding an item, another window will be opened an three fields will be displayed asking for the price, name, and serial number of the product. After those are entered and confirming if the user wants to submit the product, a connection is made to the database Inventory and a SQL statement is built and submitted. When the user clicks the lookup icon, a connection is made to the database and a query is built to return the full table, displaying it to the screen. Lastly, when the remove button is clicked, the user will be prompted to enter a serial number corresponding to a certain product in the database. A statement is built off the user input and the item is removed, if it exists.

Testing

Testing for this system is very straightforward. First it was tested whether continually opening and closing windows would cause a crash. After numerous tests, no crashes occurred. Also, constant queries and updates were performed to see if the database contained all that is should have, at that point in time and no problems ever arose. Invalid data was also entered into the system to see if malformed data would be put into the database and the system still only accepted properly formatted input.

Conclusion

Inventory Tracking Service was designed with simplicity in mind. It was made to be an easy-to-use tracking service for businesses looking to quickly store products in their store. This system contained the requirements of being lightweight, easy to use, storing data in an online database, preventing breaches, adding data, removing data, and being able to look up the full inventory. It makes use of interfaces, such as, JDBC to easily store

the business information in an online database. Building off numerous design patterns, the system utilizes the visitor, singleton, controller, factory, and observer design patterns. The implementation of the program follows a simple pattern in which a controller waits for input and decides what to do from there, abstracting the main function to a single line. This program was also testing by inserting malformed data and constantly pressing buttons, but no problems ever ensued.