*Greyden Scott & Alex Holm*

*[Company Name] | [Company Address]*

Assignment 2: Inventory Management Application

CAB302 Software Development

Table of Contents

[Technical Description 2](#_Toc514661207)

[Item 2](#_Toc514661208)

[Stock 2](#_Toc514661209)

[Truck (Abstract) 2](#_Toc514661210)

[RefrigeratedTruck 2](#_Toc514661211)

[OrdinaryTruck 2](#_Toc514661212)

[Manifest 2](#_Toc514661213)

[Store 2](#_Toc514661214)

[UML Diagram 3](#_Toc514661215)

[GUI Test Report 4](#_Toc514661216)

# Technical Description

## Item

The item class instantiates an object type of item which consists of several key properties:

Name, Cost, Price, Reorder Point, Reorder Amount, Storage Temp and Quantity.

## Stock

The stock class represents a collection of items, this class is utilised for representing different types of inventory: store inventory, cargo to be loaded onto trucks, cold items, non-refrigerated items, inventory to be reordered. The stock class is able to sort the collection of items by temperature.

## Truck (Abstract)

Truck is an abstract class for the two truck types, RefrigeratedTruck and OrdinaryTruck. It has the follow abstract functions: getCapacity, getCost, getCargo. In addition, the Truck class also has a function to return the inventory on the truck as a list and the inventory on the truck as a Stock type.

## RefrigeratedTruck

The RefrigeratedTruck class extends the Abstract class Truck, It overrides getCost, getCapacity and getCargo with its own specific code related to the Rerigerated Truck type. In addition, the Refrigerated truck has 2 addition functions, getTemperature which returns the temperature the truck operates at and setTemperature, which sets the trucks temperature based on the coldest item in its inventory.

## OrdinaryTruck

The OrdinaryTruck class extends the Abstract class Truck, It overrides getCost, getCapacity and getCargo with its own specific code related to the Ordinary Truck type.

## Manifest

The manifest class creates a collection of trucks on instantiation, all with cargo ready to be exported to a manifest file. The manifest class has a function to return the collection of trucks as a List.

## Store

The store class uses a singleton pattern, ensuring that the only once instance exists. It does this by check if it already exists, if so it returns the existing Store, otherwise it creates a new one.

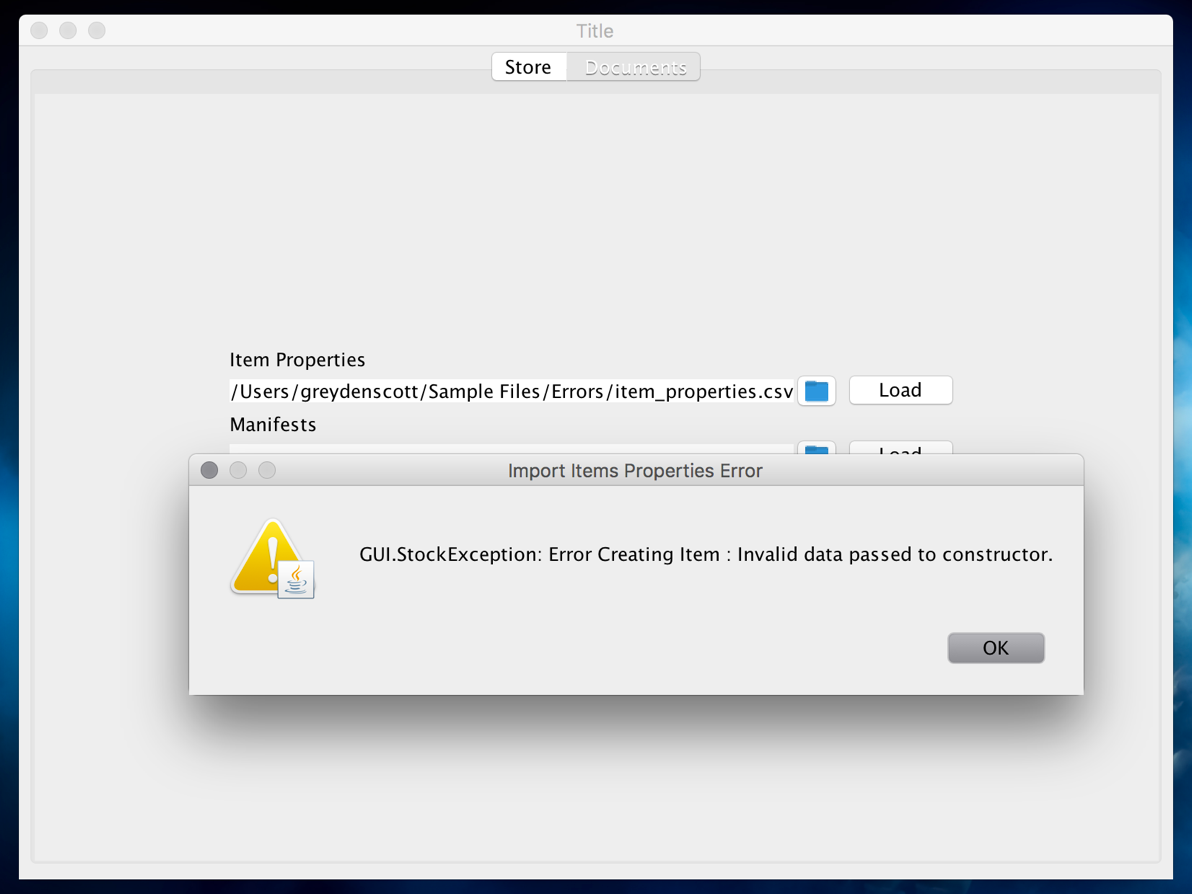
When the item properties file is loaded, many Item polymorphic type objects are instantiated based on the data supplied, these objects are stored in a Stock type object, which is used for also storing collections of items split into categories based on individual attributes. When Manifests are generated, different Truck type objects are instantiated and associated with Stock objects based on the item objects within the Stock objects.

## UML Diagram

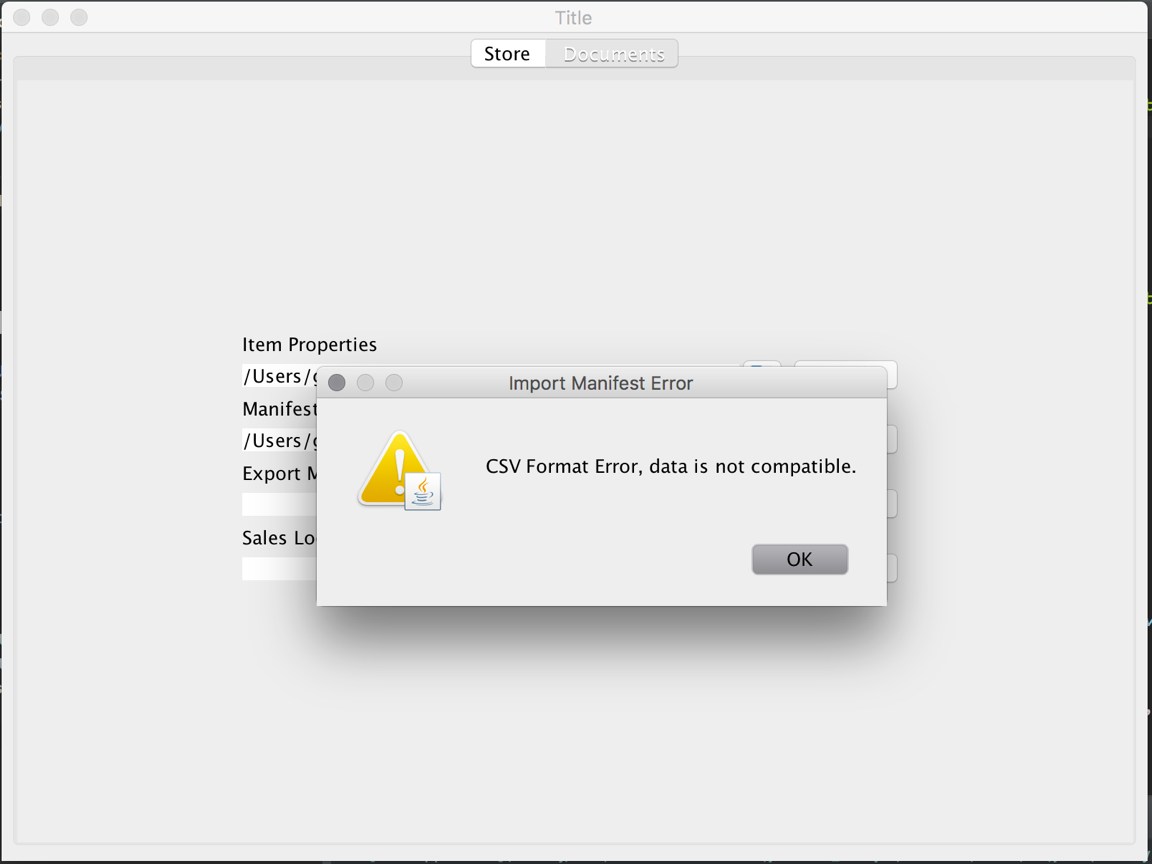


# GUI Test Report

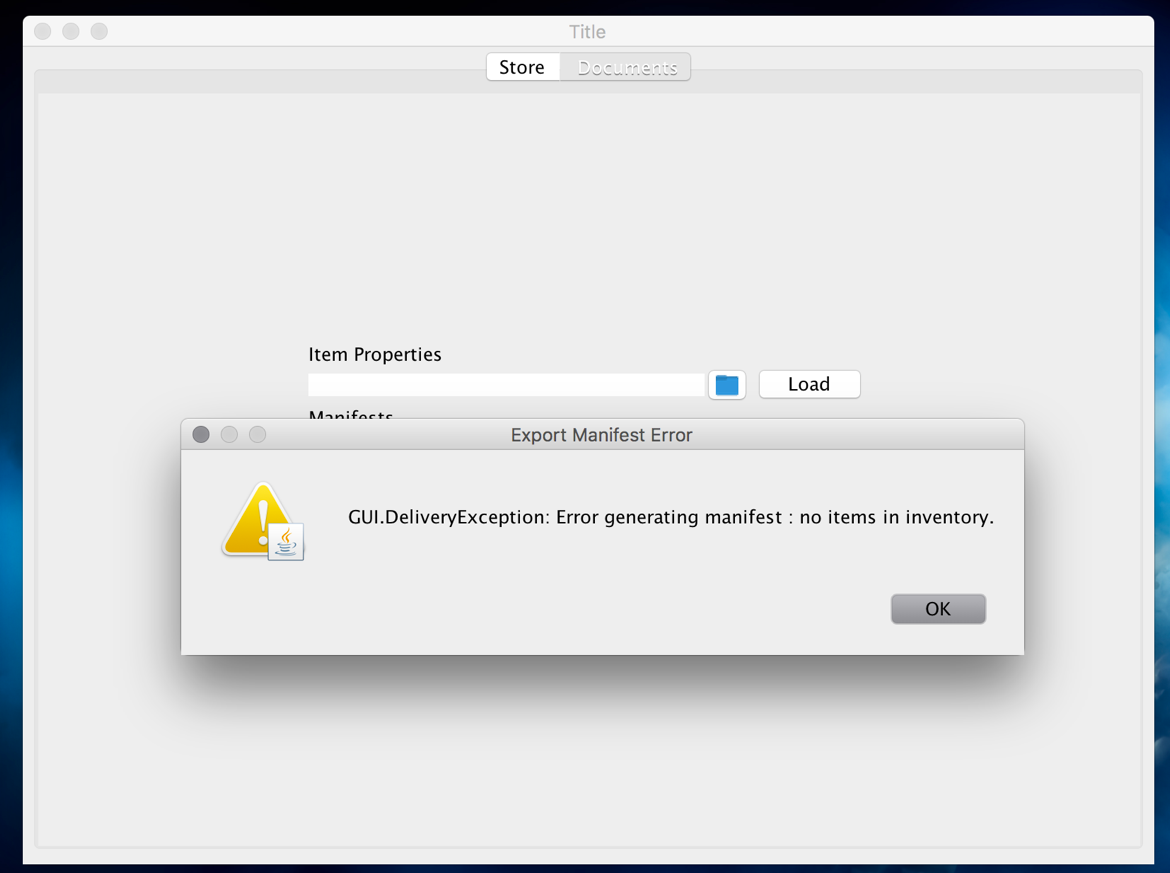
## Item Properties Import Error



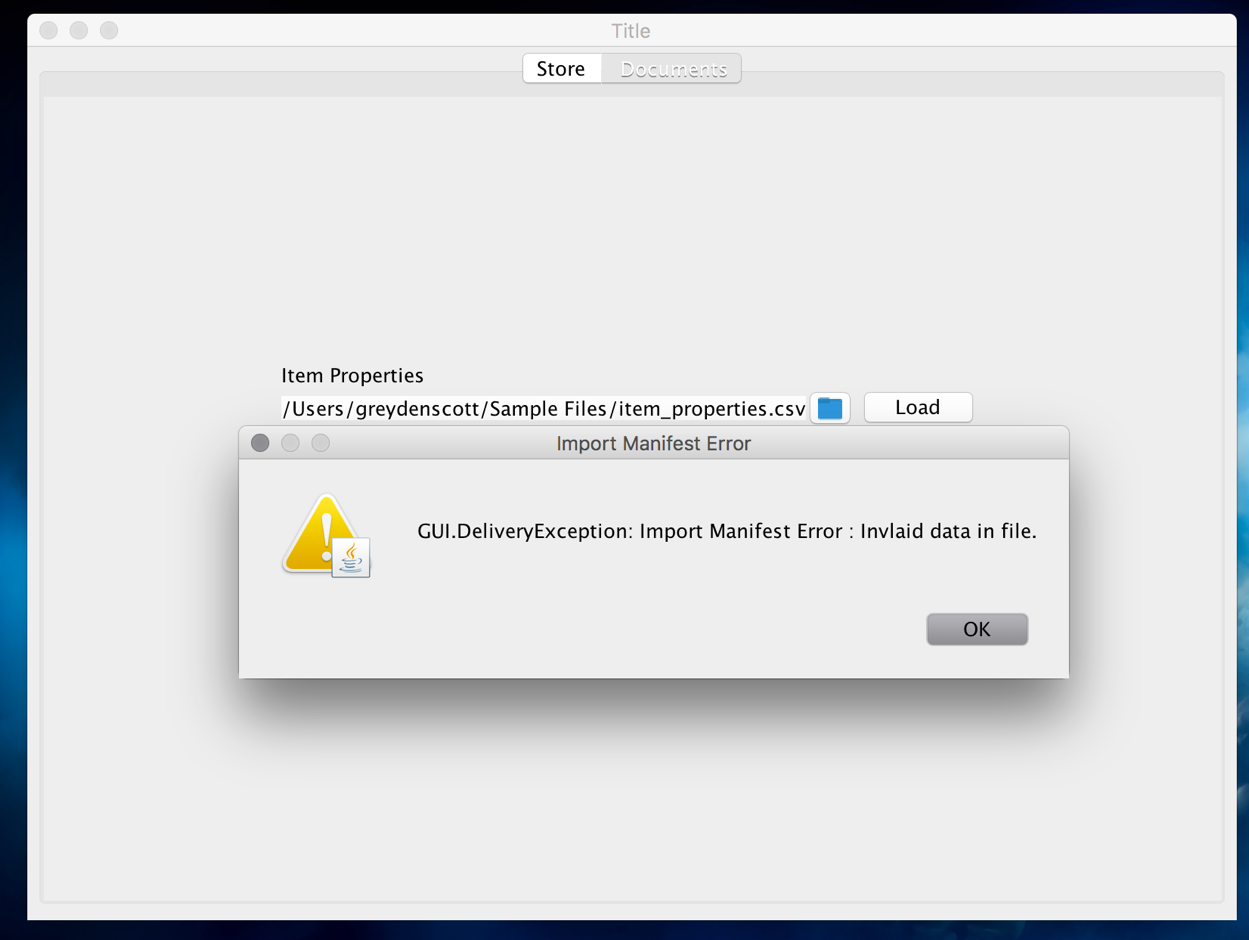
## Import Manifest Error



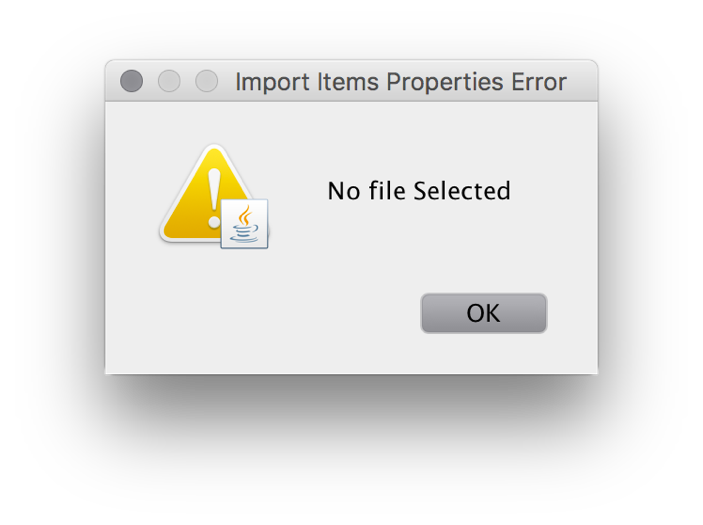
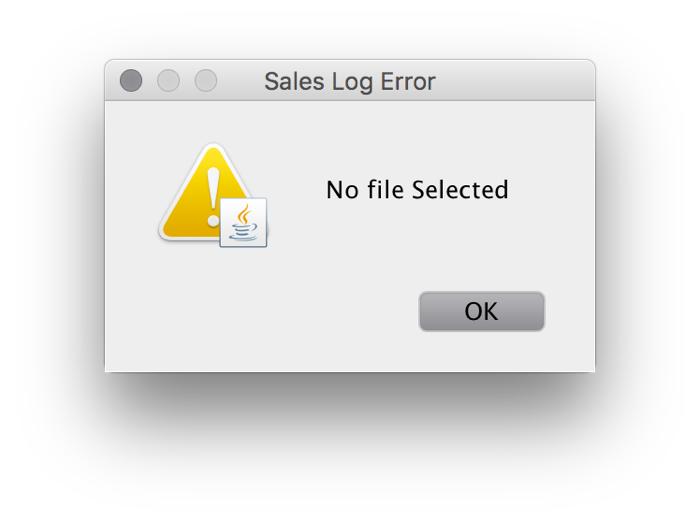
## Import Manifest Error 2



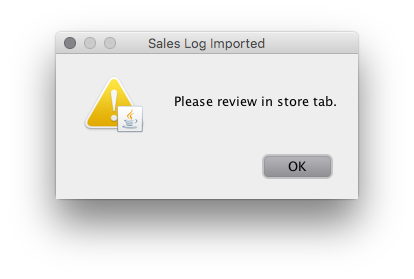
## Import Sales Log Error

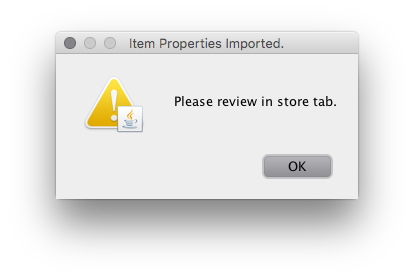
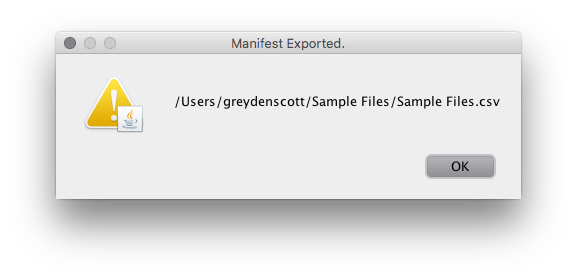
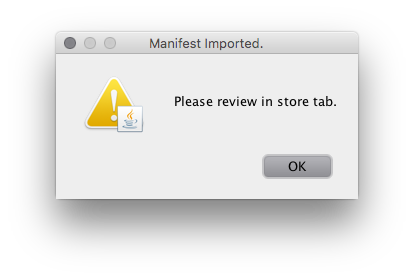


## No File Selected Error



## Importing Export Successful





## Functioning Store Tab

