Queensland University of Technology

CAB302 Software Development

**Report**

**Assignment 2: Inventory Management Application**

**Group 003**

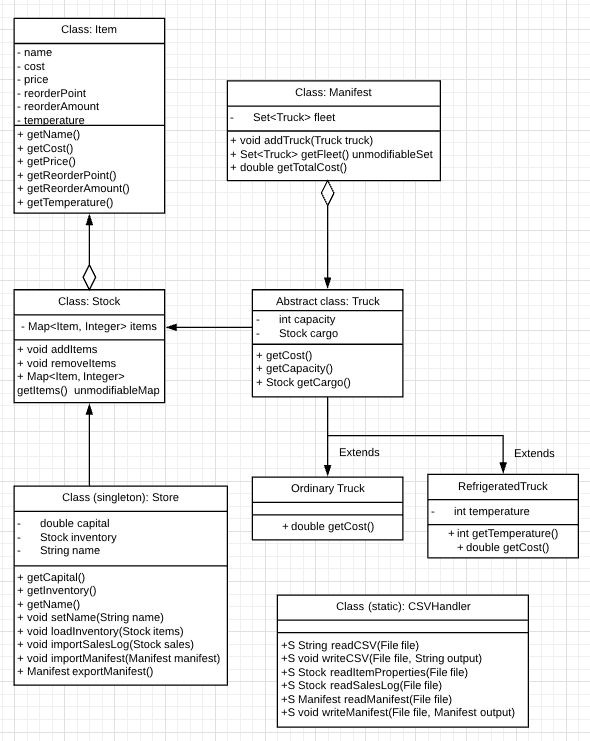
Alf Helge Jakobsen (n10142029, GitHub: alfhj)

Larysa McGookey (n9696563, GitHub: lara09)

Semester 1, 2018

Packages:

* stock: classes relating to items and collections of items
  + Item
  + Stock
  + StockException
* delivery: classes relating to delivery of goods
  + Truck
    - OrdinaryTruck
    - RefrigeratedTruck
  + Manifest
  + DeliveryException
* store: classes relating to the store itself
  + Store
  + StoreException
* csv:
  + CSVHandler
  + CSVFormatException



Class: Item

The class to store an item. It has 6 fields and 2 constructors, with and without temperature. Items can have a temperature from -20 to 10 C.

Class: Stock

A collection of Items. A HashMap is chosen to keep track of the included items and their amounts. The method "getStock" creates a copy (or unmodifiableMap) of the original HashMap, so that the values are protected.

Abstract class: Truck

An abstract class for trucks. The constructor (no access modifier) takes two arguments: capacity and cargo.

Class: OrdinaryTruck extends Truck

An unrefrigerated truck. The constructor takes a cargo and sets capacity to 1000. Costs between $750 and $1000

Class: RefrigeratedTruck extends Truck

A refrigerated truck. The constructor takes a cargo and sets capacity to 800. The truck's temperature is set to be equal to the lowest temperature of the items. Because items must have temperatures from -20 to 10 C, no further checking of the truck's temperature is necessary. Costs between around $1000 and $1700

Class: Manifest

A collection of trucks. Set is being used to avoid two identical trucks on the same manifest. "getTotalCost" calculates the total cost of the manifest by summing the cost of each truck together with the prices of all the items.

Class (singleton): Store

Singleton pattern, as there is only one store.

Approach to generating manifests: If needed items contains a refrigerated item, create a truck with temperature equal to the lowest temperature of the needed items. Fill the truck with the lowest temperature items. If it gets full, create a new truck depending on the temperature of the remainding items. Repeat.

GUI Test Report