**REFACTORING DOCUMENT**

.**Refactoring tasks** (also added into big and detailed stories)

1. Code Smell: Duplication
   * Originally, in Owner.java and Pharmacist.java and Patient.java, there were many duplicate methods (with tiny changes for the Patient implementation) such as *searchMedicineByName*, *searchMedicineByType*, and *searchPatientByName*.
   * We applied refactoring to fix the duplication by deleting the duplicate copies of the method and moving one remaining copy into a class that makes sense (for ex. *ListOfUsers* or *User*). For the *searchMedicineByName* and *searchMedicineByType* methods, this required to add an additional parameter that indicated which user invoked the method (since admins and patients have different access to the inventory).
   * Now, there is no duplicated method found in multiple classes since only one copy of each of these methods remain.
2. Code Smell: Large Class (x2)
   * Inventory.java
     + Originally, Inventory.java was quite a large class since it contained methods for searching, modifying and displaying the medicine inventory.
     + We applied refactoring to fix this large class code smell by identifying that the ‘display’ of the inventory can be separated from this class. So, we created a new class called SortInventory.java and moved the 3 long display/sorting methods in there and invoked these methods in Inventory.java.
     + Now, Inventory.java has significantly decreased in size as it has gotten a helper class to help it out.
   * DisplayInitialScreen.java
     + Originally, there was a single class that was very large, contained a lot of variables and also contained several methods that were too long.
     + We applied refactoring by separated each of those large methods into their own classes to correspond with which type of user is logging in and which screen they should see.
     + Now, the original DisplayInitialScreen.java invokes the correct new class accordingly, greatly shortening the amount of code in this single class.
3. Code Smell: Large Method
   * Originally, in DisplayDescription.java and DisplayReport.java, there was a single method that did everything in both these classes.
   * We applied refactoring and the single methods has now been divided into several parts and the functionalities are spread out into different methods.
   * Now, the original method calls these new methods respectively. Therefore, the method that was originally too long is divided into several sub methods based on their responsibility.