Matthew Muller

**Project One Option 1: Inventory App**

**Goals**

The overarching goal of this app will be to provide a virtual inventory system that allows users to easily track the items in a warehouse. The app will allow for seamless addition and removal of items from the inventory system, as well as the ability to manage the number of each item, with an alert being sent if an item reaches 0. Users will also have the ability to view a screen with all items in the warehouse listed in a grid format. This inventory data will be stored in one table with the app also including a table to store user data. This user data will consist of login-password pairs that are stored when a user creates an account. These accounts will be created from the login page, which will also provide access to return users.

**Users**

The three different types of users on the application will be differentiated by the functionality that they are provided access to. These users are designed following the assumptions of a regular hierarchy of employees, with access growing as the employee’s station does. The lowest level of access will be granted to users who only need to manage increasing and decreasing specific items’ inventory. This would likely be used by lower level employees. The next user type will also have access to this functionality, as well as the ability to add and remove items from the inventory system. This user will likely be used by staff in managerial type rolls. Finally, the third user type will have full access to the inventory system’s functionality, as well as the ability to manage all user accounts. This user will be reserved for the business owner themself or the person that they give full control over the inventory.

**UI Design**

This app will require a few different screens in order to implement all of its intended functionality. To begin, there will need to be a login screen with two EditText widgets, one being plain text and one using a password type. There will also be two buttons on this page, one to login and one to create a new user. The screen that displays all items in the inventory system will be created using a GridLayout that will essentially appear as a table with columns for each attribute of the items. The alert for when an item reaches zero will likely be a TextView widget that becomes visible when an item’s quantity equals zero and will disappear after the user clicks a button to dismiss it. There will also be a screen that the user uses to add/remove/update items. This screen will have an EditText widget with a number type, as well as buttons for adding and removing items. Clicking the add button will prompt the user to enter values that will be paired with the key of each attribute of the item. Navigating between these screens will be done via a row of buttons that will be displayed at the bottom of the screen at all times (after login). This design will comply with Android Design and Quality Guidelines by providing a user experience that is easy to understand and navigate while running safely and smoothly.

**Code Design**

The main functionality of the app will be provided by the connection of the inventory and user tables to the UI. On the login screen, the two EditText widgets will accept text inputs and either search the user table for this key-value pair or add it to the table if a user is being created. The screen that displays all of the inventory in a grid format will be achieved by simply looping through the inventory table and displaying each piece of information in the corresponding TextView. The pop-up for an item reaching zero will be created by changing the value of a boolean that will be used to toggle the visibility of the TextView. Edits to the inventory will result in changes being made to the inventory table. For example, if the user adds an item or updates its quantity, the set of key-value pairs will be added to the table or the corresponding item’s quantity value will be updated, respectively. The menu display will be connected to code that will display the corresponding screen when a button is clicked.