Enhacement Two: Algorithms and Data Structures

By Ivy Pokorny

Artifact Description:

What is it? When was it created?

For my Algorithms and Data Structures enhancement project, I selected **Stock't**, an application developed in CS 360: Mobile Architecture and Programming, aimed at streamlining inventory management for flower shops. This app addresses critical user needs, including low stock alerts, easy inventory tracking, and efficient communication of inventory levels through SMS notifications. Users can enter and modify inventory information directly on their phones, adjusting counts as products are sold or lost. Additionally, the app notifies users when any product falls below a stock threshold of 50 units, ensuring they remain informed about inventory status. This project not only highlights my technical skills but also reflects my commitment to creating user-centric solutions that enhance operational efficiency. This project includes multiple screens. There is a Login screen and a Signup screen for entering into the User Database. The Home screen displays item analytics from Enhancement 1 of this class and it allows the Graph screen to be entered as well. The Items screen allows users to Delete, or adjust Quantity on the inventory. It also allows the user to Add Items on a separate screen.

Justification for Inclusion

Why did you select this item? What specific components of the artifact showcase your skills and abilities in algorithms and data structure? How was the artifact improved?

I selected this artifact because it effectively demonstrates my skills in algorithms and data structures, particularly through the implementation of an undo/redo functionality using stack data structures. Previous functionality in this app included the Lists of Items in the inventory, that is then modified in and updated from the SQLite database. This new Undo/Redo functionality not only enhances user experience by allowing for corrective actions but also showcases my understanding of efficient data management within the application. The enhancements I made included adding item IDs to the undo/redo stack, ensuring that each operation is accurately tracked and can be reverted appropriately.

The features that have been added are easier to explain with the following figures. Firstly, the Inventory screen now contains an added “Inv History” button which takes you to the Inventory History. To begin with the History screen is filled with the Undo stack containing the first few items added to the item database for testing. These additions can be seen below in Figure 1.

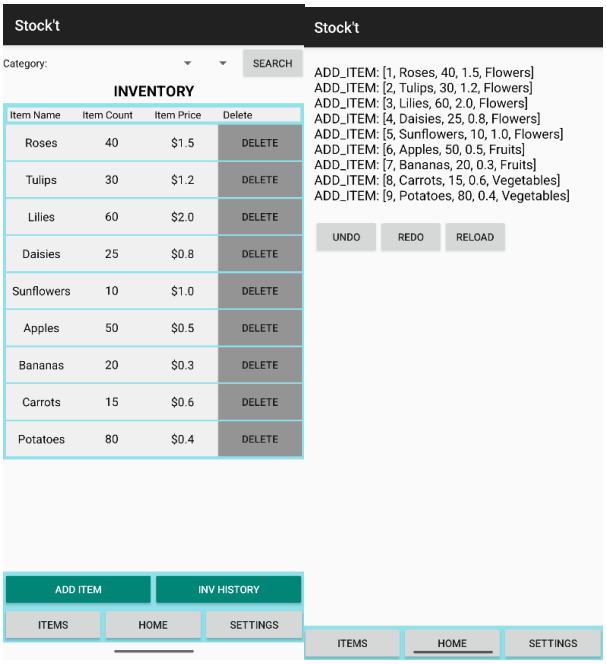

Figure 1: Inventory and History Screens

Within the History screen the three buttons “Undo,” “Redo,” and “Reload” contain the main features of the Undo/Redo Stack. “Reload” is simply used for updating the current page with the Undo and Redo stack. The “Undo” button, obviously, undoes the Action directly above it. Similarly, the “Redo” button redoes the Action directly below it. For UX, the “top” of the Undo stack is at the bottom of the undo list and the “top” of the Redo stack is at the top of the redo list. Thus figure 2 shows undoing 4 actions from the Undo stack.

Figure 2: Basic undoing

Actions supported by the undo and redo stack are all the adjustments to the database that are currently possible. These actions are:

Adding An Item: Undoing deletes the item, Redoing recreates the item in the database

Deleting An Item: Undoing recreates the item in the database, Redoing deletes the item

Adjusting A Quantity: Undoing adjusts the quantity back to the previous value, Redoing adjusts it back

The results of these actions are shown in figure 3. Figure 3 shows the database with Potatoes, 20 Sunflowers, and no Bananas. Then it displays the History before and after undoing the addition of Potatoes, the adjustment of the Sunflowers, and the deletion of the Bananas. Then the resulting Inventory again

Figure 3: Undoing Basic 3 Actions

In addition to this, I reversed the redoList in the HistoryActivity to ensure that both the undo and redo stacks grow and shrink correctly. This modification was crucial for maintaining the integrity of the stack operations and demonstrated my ability to address potential logical flaws in the design. And a final minor modification was making sure the Reset Database button in the settings menu also reset the Undo/Redo stack.

Course Outcomes and Updates

Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

Through these enhancements, I met several course outcomes, particularly in designing, developing, and evaluating computing solutions using algorithmic principles. The use of stacks for managing undo and redo operations illustrates my ability to apply data structures effectively. The modifications made were not part of the original plan discussed in Module One. Reflecting on the feedback from Module One and Module Two I consulted the Enhancement Tips announcement and decided that, given the updating, creating, and editing nature of the inventory app, the Undo/Redo using stacks was the perfect enhancement for this category and inclusion in my portfolio.

Looking ahead, my outcome-coverage plans include a final “Data Structures Improvement” for the Data Structures Category, where I aim to implement a database specifically for tracking inventory changes over time. This will not only provide a historical log of changes but also lay the groundwork for future data visualization options. The database will

included the item ID, name, quantity adjustment, date & time, and the reason for the adjustment. This will allow **Stock't** users to see how much of their inventory is being lost for different reasons, as well as analyze the trends and spikes in the flower shop selling seasons.

Reflection on the Enhancement Process

What did you learn as you were creating it and improving it? What challenges did you face?

The process of enhancing and modifying this artifact has been both enlightening and challenging. I learned the importance of maintaining clear and efficient data handling within an application, especially when implementing features that rely heavily on user interactions, such as undo and redo. This experience reinforced my understanding of stack operations and their practical applications in software design.

One of the key challenges I faced was ensuring that the undo and redo functionalities worked seamlessly without causing inconsistencies in the application state. I needed to adjust previously written functions to ensure that the itemIDs were returned so that the IDs could then be used to recreate the same Item in the Item database in the case where an adjustment need to be made to an item that was recreated in the process of undoing the deletion. This alos, involved Overloading a method so that items could be added to the databse using those specific IDs. Updating the item quantity also needed to be improved in order to check the previous Quantity value so that both the previous and new Quantities could be logged for proper undo/redoing. Addressing these challenges really reinforced the importance of thorough testing in software development.

Additionally, actions taken by the user need to be logged in the stack, but adjustments to the inventory via the History screen needed to stay out of the stack. Otherwise, undoing an item

creation would add an action to the undo stack that would indicate the item was deleted. User and Undo/Redo actions needed to stay separate.

Overall, this milestone has strengthened my skills in programming solutions to problems using algorithms and data structures. It has also demonstrated my ability to thoroughly test code, which is essential for user-facing environments in the field of computer science.