Milestone Three

Russell Reckhow

Southern New Hampshire University

CS499: Computer Science Capstone

Professor Joseph Martinez

Narrative

The artifact I used was my Weight Tracker app from CS360. It was created around June of this year. For this enhancement, I added a merge sort so users can sort the weight list by date. They can switch between newest to oldest or oldest to newest.

I added this because it shows I can improve existing software by adding a new feature. It also shows both frontend and backend work. On the front end, I added a sort menu with a check option so the user can see which one is active. I also saved the choice so when a user logs out and logs back in, the option stays the same. On the back end, the entries get sorted by date with the merge sort and date conversion. This makes the list easier to read and gives the user more control.

This enhancement met the outcomes I wanted so far. It shows I can design and use solutions by picking and implementing an algorithm that works. It also shows I can use tools and techniques like Android Studio, Java time, PopupMenu, MenuProvider, and merge sort. I showed communication by writing about it in my ePortfolio and GitHub and talking about it in my code review video. The only outcome I still need is security, which will be covered when I add my notebook database later.

I learned how to integrate an algorithm into my project and connect it to my app. I also learned how to update the list and save the users sort order. There were a few challenges. The first was figuring out how to add the drop-down menu next to the sort button. That was solved with an anchor view. The second was when both options showed checked for a moment when closing the menu, and I fixed it by removing a line of code. The last issue was with the dates. They were saved as “m/d/yyyy” text, and I couldn’t compare them directly. I fixed that by

converting them into LocalDate so the merge sort would work. Solving these problems helped me get better at troubleshooting and finding different ways to fix things.

Merge sort implementation:

```java
/**
 * Merge sort order over dates.
 *
 * @param arr the list of weight entries to sort
 * @return a new list sorted by date
 */
3 usages
private List<WeightEntry> mergeSort(List<WeightEntry> arr) {
    if (arr == null || arr.size() <= 1) {
        return arr == null ? new ArrayList<>() : new ArrayList<>(arr);
    }
    int mid = arr.size() / 2;
    List<WeightEntry> left  = mergeSort(new ArrayList<>(arr.subList(0, mid)));
    List<WeightEntry> right = mergeSort(new ArrayList<>(arr.subList(mid, arr.size())));
    return merge(left, right);
}
```

```java
/**
 * Merges two sorted lists into one.
 *
 * @param left the left side list
 * @param right the right side list
 * @return a list in order
 */
1 usage
private List<WeightEntry> merge(List<WeightEntry> left, List<WeightEntry> right) {
    ArrayList<WeightEntry> result = new ArrayList<>( initialCapacity: left.size() + right.size());
    int i = 0, j = 0;

    while (i < left.size() && j < right.size()) {
        LocalDate dateLeft  = convertDate(left.get(i).getDate());
        LocalDate dateRight = convertDate(right.get(j).getDate());

        // Keep order when dates match
        if (!dateLeft.isAfter(dateRight)) {
            result.add(left.get(i++));
        } else {
            result.add(right.get(j++));
        }
    }
    // Add leftovers
    while (i < left.size()) result.add(left.get(i++));
    while (j < right.size()) result.add(right.get(j++));
    return result;
}
```

Sort menu in the app: