**Memo**

To: Professor Pisano

From: Daniellia Sumigar, Tristen Liu, Juan Vecino, Ramy Attie

Team: 14

Date: 04/11/2024

Subject: Final Testing Report

1. **Setup Summary**

In order to prepare for the Final Testing, several boxes needed to be purchased from a shipping supplier and taped in order to provide testing boxes and containers. These boxes included two 8x8x8” boxes, five 12x9x6” boxes, and one 20x20x20” container. After the boxes were taped and their dimensions were checked with a tape measure, the application was built on the testing device (iPhone 14 Pro) and a preliminary check of the working features was conducted. After every feature was checked and ensured to be working, the testing device screen was mirrored onto the MacOS display and the setup for testing was over.

**2.0 Experimental Results**

**2.1 Scanning and Manual Input Mode**

First, the Container Preset collection screen in Manual Input mode was displayed and tested. We demonstrated the selection of various containers and showed the different companies that we had set in the application (FedEx, UPS, UHaul). Then, the inches-to-centimeters setting was tested in order to demonstrate the change of display measurements when the user switched from inches to centimeters and vice versa. Finally, the Scanning function was demonstrated on multiple boxes, specifically the 8x8x8” and the 12x9x6” boxes in order to confirm the accuracy of the scan. After the scanning function was successfully tested and the dimensions were confirmed, a previous session’s data was loaded in order to both save time and show the session history function of the application. A previous session was successfully loaded, and the dimensions from the previous tests were overwritten by the session’s data. This session’s data consisted of the testing we did in the setup process of the same boxes and container.

* 1. **Python Backend/Cloud Server**

One of the main project functionalities was to implement automatic packing schematic generation. To measure the performance of the python backend hosted on the cloud server for automatic schematic generation, the time to send and receive data was recorded. When the user finishes inputting dimensions of both the container and objects, the data is parsed into JSON format and sent into the python backend API hosted on the Heroku cloud server via POST HTTP request. The return data used to construct the packing schematic was returned in 0.01 seconds for automatic schematic generation.

* 1. **Packing Schematic**

After inputting dimensions of both the container and objects, the user is directed into the packing schematic screen. The default view displays the wireframe outline of the container from a top-down perspective. This schematic can be rotated and zoomed in and out by the user. When pressing the “Begin Packing Tutorial” button, the user views the individual placement of the objects in the order that they should be packed. Each object is color-coded with the corresponding name displayed on the screen as it is placed within the container. The schematic also includes back and forward arrows to move through the sequence of placed objects. Additionally, on the top of the screen, the names of all unfitted objects and the final space utilization are shown.

**3.0 Discussion & Conclusion**

Based on the final testing results, the scanning method we had implemented was efficient to the nearest quarter inch, when scanned from a distance of at least one foot. It was shown that if the user attempts to move too close to the scanning surface, the scanning points begin to drift, making the scan inaccurate. When scanning the dimensions of a box, it took less than 20s to determine the dimensions of a box in its entirety, including the Length, Width and Height. In addition, the Manual Input Mode and Session History loading was demonstrated to be working and easy to use.

The testing results also demonstrated the completion of the automatic schematic generation functionality of the project. The packed box coordinates were successfully queried from the Heroku cloud server, and a schematic was generated using the coordinates. Although the packing output was suboptimal in terms of fitting all of the items into the container, the boxes that were indicated to fit in the container were able to be placed in their respective locations as specified by the schematic.

One observation made during the final testing was that there were different ways to fit the items in the container, in addition to the arrangement determined by the packing schematic. This suggests that our current implementation of the packing algorithm arrives at one locally optimal solution, but depending on the user requirements may be suboptimal overall. One key difference from the other possible packing arrangements was that our current solution offered from the schematic maximized stability, where each placed object was supported under all 4 bottom vertices. As a result, although all the boxes were not fitted into the container in our demonstration, the boxes that were able to fit were made to fit with the most stability possible. On the other hand, some of the alternative packaging solutions that were possible would not have considered stability as a heuristic. Overall, this means that we have room for customization within our packing algorithm, allowing our user to decide what kind of solution they are looking for, whether that be maximum space utilization, boxes fit, or stability.

Another area of improvement for our packing schematic is to include a way to distinguish between each side of the container. At times, it was difficult to figure out where to place the objects as suggested from the tutorial; Especially for boxes with equal dimensions on all sides like our testing container. In order to eliminate any possibility of confusion, we are working on making the schematic as intuitive to the user as possible. This could include marking each side of the container in an unobtrusive way, clearly indicating to the user which side is top, bottom and the other four sides.

Finally, multiple suggestions were made to improve upon our application, such as automatically selecting the minimum box size for the chosen boxes. In regards to this, some progress has already been made towards modifying the API in order to detect when the user wants the API to determine the container size instead of being user selected. Additionally, we have also implemented the ability for the user to add multiple containers, which the API will then optimally distribute the boxes between the containers.