Matthew Hunt

CPT 237 Final Project

October 17th, 2019

HuntFinalProgram

As previously discussed in the e-mail, this program is a real-world program that could be utilized by the quality department at Mercedes-Benz Vans in Ladson, SC. There are over 100 studs that go on each van – some automatically added by a robot, some manually added by an employee. Both processes can result in missing, broken, or off location studs that require rework/repair further down the assembly line. I made a program to help track the type of defect, when it happened, which stud, and what that stud is used for. There are also a few other features not mentioned.

Unfortunately I did not get to make changes discussed in my e-mail question. I should have asked much earlier. I didn’t notice the second part in the instructions until I e-mailed you. I’m capable of doing it, but I have work at 2pm and don’t have enough time to add it and then test the program for any crashes/errors. I do believe the way I have it now is pretty clean, but obviously I’m not too experienced to really know. I thought that only reading it in when the program starts, rather than every time a tableview is shown, would be more efficient and faster. I need to learn a lot more about the hardware/memory/ram used by programs like this. I feel pretty good on the syntax/logic side of everything, but not too knowledgeable when it comes to the specs/hardware side. Anyway – here is a breakdown of the program:

**Main Window / Plant Window**

**Main Window – Plant window that has 4 options**

**1) Find Stud**

**2) Show Defects**

**3) Show Studs**

**4) Done**

**Find Stud**

When find stud is clicked, a new window will popup that will ask for a stud ID. You can search or hit done to close the window.

**A)** If the stud is found the window will expand to include all the details of the stud object. It will also include a TableView table to show all findings of this stud. You can then “Add Findings” or “Edit Containment”. Containment is the actions we are doing in house to fix the defect.

When add findings is clicked, another window pops up with the stud Id, 3 radio buttons, a date calendar, and a text field for units affected. User MUST select a date and enter a positive number for units affected or the user will get an error alert. If input is all correct and “Add” button is clicked, the defect will be added to the defect array list and written to the defect csv file. Error alert given if there is an error adding to the file. Cancel simply shuts the window.

The table and values in Find Stud window are not updated immediately with an observable. I hope to implement this when I do not have an immediate deadline. Click search again with the same stud ID and the values will update.

When “Edit Containment” is clicked, a simple window will show up to type in a new containment. If submit is clicked, the containment for that stud will change. Also does not update immediately. Would also implement that and have that written to the file as well when I have more time.

**B)** If the stud is **NOT** found, a new button, “Add Stud” will become visible. When this button is clicked, a new window will pop up similar to Add Findings, that will allow a user to create a new stud. Clicking add stud will also close Find stud window to prevent users from adding the same stud multiple times. Stud will be created using the process selected by user and the name added. New stud object is added to stud array list and added to csv stud file.

**Show Findings**

Simply a tableview that shows a table of all defect findings, sorted by date using a comparator. When a new defect is added, the new defect will be included if window is closed and reopened. Would like to implement observable down the road for this as well.

**Show Defects**

Essentially the same as show findings, except that all studs are shown, rather than all defects. New studs are added and ordered by studID. Also need to implement observable.

**Done**

Closes the window!

Really looking forward to any feedback on this program – good or bad. First time creating a program of my own idea from scratch, and I’m pretty content with the end result. I also definitely see where new features and improvements can be added.