Student Name: JESSICA SHORT
Student ID: 010500487
WESTERN GOVERNOR'S UNIVERSITY
Course: D287 – JAVA FRAMEWORKS
Assignment: ALN2 TASK 1: WEB-BASED SPRINT INVENTORY APPLICATION

PART B
Prompt: Create a README file that includes notes describing where in the code to find the changes you made for each of parts C to J. Each note should include the prompt, file name, line number, and change.

This document is the README file for Part B.

PART C
Prompt: Customize the HTML user interface for your customer's application. The user interface should include the shop name, the product names, and the names of the parts.
Note: Do not remove any elements that were included in the screen. You may add any additional elements you would like or any images, colors, and styles, although it is not required.

File Name: mainscreen.html
Line Number(s): 14, 19
Description of Changes: Customize the HTML user Interface for Green Country Dog Houses

File Name: mainscreen.html
Line Number(s): 21-55
Description of Changes: Later I went back and added the products and parts to mainscreen.html and moved the about button to the top of the page. (This was after I completed part E).

PART D
Prompt: Add an "About" page to the application to describe your chosen customer's company to web viewers and include navigation to and from the "About" page and the main screen.

File Name: about.html
Line Number(s): All
Description of Changes: Created about HTML page to give consumers more information about Green Country Dog Houses

File Name: AboutController.java
Line Number(s): All
Description of Changes: Created About Controller to map http://localhost:8080/about to the about.html template.

File Name: mainscreen.html
Line Number(s): 44-46 >> later moved to 22-25
Description of Changes: Added a navigation button that navigates to about.html.

File Name: about.html
Line Number(s): 90-92
Description of Changes: Added a navigation button that navigates back to mainscreen.html.

PART E
Prompt: Add a sample inventory appropriate for your chosen store to the application. You should have five parts and five products in your sample inventory and should not overwrite existing data in the database.

Note: Make sure the sample inventory is added only when both the part and product lists are empty. When adding the sample inventory appropriate for the store, the inventory is stored in a set so duplicate items cannot be added to your products. When duplicate items are added, make a "multi-pack" part.

File Name: BootStrapData.java
Line Number(s): 72-184
Description of Changes: Added inhouse parts, outsourced parts, and products and stored them in the repositories. Also, an if statement was added to check to see if the part and product lists are empty before adding the sample inventory.

PART F
Prompt: Add a “Buy Now” button to your product list. Your “Buy Now” button must meet each of the following parameters:

• The “Buy Now” button must be next to the buttons that update and delete products.
• The button should decrement the inventory of that product by one. It should not affect the inventory of any of the associated parts.
• Display a message that indicates the success or failure of a purchase.

File Name: mainscreen.html
Line Number(s): 120-121
Description of Changes: Added a "Buy Now" button to the products table. Mapped the button from /mainscreen to /buyProduct and the BuyProductController.

File Name: BuyProductController.java
Line Number(s): all
Description of Changes: Created the BuyNowProduct Controller. This controller check to see if the product is in stock and returns a success or an error message (2 new html templates). If the product is in stock, the inventory count is decreased by one.

File Name: buyNowFailure.html
Line Number(s): all
Description of Changes: If the selected product is out of stock this page is displayed to communicate a failure to purchase.

File Name: buyNowSuccess.html
Line Number(s): all
Description of Changes: If the selected product is in stock this page is displayed to communicate a successful purchase.

PART G
Prompt: Modify the parts to track maximum and minimum inventory by doing the following:

• Add additional fields to the part entity for maximum and minimum inventory.

• Modify the sample inventory to include the maximum and minimum fields.
• Add to the InhousePartForm and OutsourcedPartForm forms additional text inputs for the inventory so the user can set the maximum and minimum values.
• Rename the file the persistent storage is saved to.
• Modify the code to enforce that the inventory is between or at the minimum and maximum value.

File Name:Part.java
Line Number(s): 32-37
Description of Changes: Added fields minInv and maxInv. Used @Min and @Max to validate.

File Name:Part.java
Line Number(s): 53-60
Description of Changes: Added overloaded constructor with minInv and maxInv.

File Name:Part.java
Line Number(s): 94-103
Description of Changes: Added getters and setters setMinInv, getMinInv, setMaxInv, getMaxInv.

File Name:BootStrapData.java
Line Number(s): 81-82, 89-90, 97-98, 105-106, 113-114, 121-122, 129-130147-148, 156-157, 165-166, 174-175, 183-184
Description of Changes: Modified the sample inventory to include the maximum and minimum fields.

File Name:InhousePartForm.html
Line Number(s): 24-26
Description of Changes: Add to the InhousePartForm and OutsourcedPartForm forms additional text inputs for the inventory so the user can set the maximum and minimum values.

File Name:OutsourcedPartForm.html
Line Number(s): 25-27
Description of Changes: Add to the InhousePartForm and

OutsourcedPartForm forms additional text inputs for the inventory so the user can set the maximum and minimum values.

File Name:application.properties
Line Number(s): 6
Description of Changes: Renamed the file the persistent storage is saved to. Changed the data source file name to GCDogHousesDB.

File Name: Part.java
Line Number(s): 113-118
Description of Changes: Created invIsValid method. This method determines if inventory value is within minInv and maxInv limits.

File Name: AddInhousePartController.java
Line Number(s): 43-45
Description of Changes: Added an if statement within submitForm that utlizies invIsValid method to check if inventory is within limits. If invIsValid is false, and error message is generated.

File Name: AddOutsourcedPartController.java
Line Number(s): 44-55
Description of Changes: Added an if statement within submitForm that utlizies invIsValid method to check if inventory is within limits. If invIsValid is false, and error message is generated.

File Name: mainscreen.html
Line Number(s): 73-74, 83-84
Description of Changes: Added maximum and minimum inventory to parts table on mainscreen.html.

PART H
Prompt: Add validation for between or at the maximum and minimum fields. The validation must include the following:

• Display error messages for low inventory when adding and updating parts if the inventory is less than the minimum number of parts.
• Display error messages for low inventory when adding and updating products lowers the part inventory below the minimum.
• Display error messages when adding and updating parts if the inventory is greater than the maximum.

File Name: Part.java
Line Number: 120-132
Description of change: Added methods invAtMinimum and invAtMaximum.

File Name: EnufPartsValidator.java
Line Number: 36-37
Description of change: Modified the if statement to return false if changing the product inventory would drop associated parts inventory below minimum limit.

File Name: ValidEnufParts.java
Line Number: 20
Description of change: Edited error message to include the case that changing the product inventory would drop associated parts inventory below minimum limit.

PART I
Prompt: Add at least two unit tests for the maximum and minimum fields to the PartTest class in the test package.

File Name: PartTest.java
Line Number: 105-121
Description of change: Added unit tests maxLimitTest and minLimitTest. Both tests passed when run.

File Name: pom.xml
Line Number: 53-56
Description of change: Updated junit dependecy info.

PART J
Prompt: Remove the class files for any unused validators in order to clean your code.

File Name: DeletePartsValidator.java
Description of change: Deleted this class file. Using Code > Inspect Code in IntelliJ, I found and deleted the 1 unused validator.