Test Plan

The first set of tests will pertain to our web scraper. We need to ensure that we are getting the correct results before displaying them. One such test is to make sure that the model number of a particular part matches the model number on a different site. Along with this, we need to make sure that the data is being put in the right spots as well as all the data being sent. We will also need to test how long our script takes to run as there are a lot of webpages being run and being able to optimize this will help greatly.

The second set of tests will pertain to our website. We need to ensure that the right type of parts are being displayed on the right site. Along with this, we need to check that the right prices are being displayed. We also need to check that the hottest deals shown on the main page are actually the hottest deals.

Test Cases

1.1. SCRAPE-01
1.2. Check that the price is being farmed from the correct element.
1.3. The script will take in a string of the element where the price should be located. Our script will then farm this element and check that it includes a dollar sign. Lastly, the script will output it to be entered into the database.
1.4. String containing the CSS selector of element where price should be.
1.5. Price of current part
1.6. Normal
1.7. Whitebox
1.8. Functional
1.9. Unit

Results: Success. With this we ensured that we always had the price instead of a different element in the page.

2.1. SCRAPE-02
2.2. Check if there are multiple pages of a type of part on Microcenter.
2.3. The script should make sure to farm all parts not just the ones on the first page.
2.4. Link to the first page of parts
2.5. Array of links to additional pages of parts if they exist
2.6. Normal
2.7. Whitebox
2.8. Functional
2.9. Unit

Results: Success. When there were multiple pages the script would then go through the other pages of parts to farm them as well.

3.1. SCRAPE-03
3.2. Check that only in stock items are chosen.

3.3. We only want the user to be able to see a part if it is in stock and able to be purchased. This also helps as we noticed that out-of-stock parts sometimes have different page layouts that can cause issues with our script.
3.4. String containing the CSS selector of element where out-of-stock would be.
3.5. Link to the part to continue farming data for it.
3.6. Normal
3.7. Whitebox
3.8. Functional
3.9. Unit

Results: Success. Our script now only farms data for in stock parts and this also alleviated past issues we had due to the layouts of out-of-stock parts being different.

4.1. SCRAPE-04
4.2. Check that the part number matches on both Microcenter and Newegg.
4.3. We need to make sure we are getting the price for the exact same part when querying on Newegg. The easiest way to do this is to search Newegg for a part number and then check the top results for a matching part number.
4.4. String containing the current part number.
4.5. True if matching part number
4.6. Normal
4.7. Whitebox
4.8. Functional
4.9. Unit

Results: Partial Success: For most cases this was a success, however, we did notice that in some cases the model number was just one character different even though they were the same part which meant we did not farm the item.

5.1. SCRAPE-05
5.2. Check that the part data is farmed in a timely fashion
5.3. While this script was only planned to be ran once or twice a day, if we wanted to do it more than this, it would need to be able to finish fast enough.
5.4. Scraping Script.
5.5. Data to MongoDB database
5.6. Normal
5.7. Blackbox
5.8. Performance
5.9. Integration

Results: Partial Success: The script finishes fast enough to be ran every half hour or hour; however, it still takes a large amount of time. This could potentially be reduced by using multiple threads.

6.1. MONGO-01
6.2. Check that the data is formatted correctly.
6.3. To ensure that the data is displayed correctly we need to make sure that the data in the database is formatted correctly for the website.

6.4. String for finding a part
6.5. The database entry for the part
6.6. Normal
6.7. Whitebox
6.8. Functional
6.9. Unit

Results: Success: When querying the database for different parts we found that everything was formatted correctly.

7.1. WEB-01
7.2. Check that the best deals are actually the best deals.
7.3. When displaying the best deals, which are parts where the difference between the Microcenter and Newegg price is greatest, we need to make sure that the displayed parts actually have the greatest price difference.
7.4. Navigate to home page.
7.5. Top 5 best deals
7.6. Normal
7.7. Blackbox
7.8. Functional
7.9. Unit

Results: Success: We found when testing this on different days when the prices had change, there would be new best deals.

8.1. WEB-02
8.2. Check that when selecting between AMD and NVIDIA graphics, the correct manufacturer is displayed.
8.3. We need to make sure that the parts that correspond to NVIDIA or AMD are displayed on the correct pages. We don't want the user to search for NVIDIA parts and then find AMD parts.
8.4. Select AMD or NVIDIA page
8.5. Designated page based on selection
8.6. Normal
8.7. Blackbox
8.8. Functional
8.9. Unit

Results: Success: When navigating to the specified page we were met by only the manufacturer we had selected.

9.1. WEB-03
9.2. Check that links to Microcenter and Newegg work.
9.3. We need to make sure that when a user decides to go one of the websites where a part is sold, the correct link is given.
9.4. Selecting link for one of the sites.
9.5. Site displayed where part can be purchased
9.6. Normal

9.7. Blackbox
9.8. Functional
9.9. Unit

Results: Success: We found that the correct link was always given, and we could verify that the name, brand, and other details displayed on our site matched the site where the part could be purchased.

10.1. WEB-04
10.2. Check that the prices on our site match the prices of store
10.3. We need to ensure that the prices being displayed on our site are correct, so the user knows the correct price before going to purchase it.
10.4. Selecting link for one of the sites.
10.5. Site displayed where part can be purchased
10.6. Normal
10.7. Blackbox
10.8. Functional
10.9. Unit

Results: Success: Like the last test we used the selected link to verify that the price on our site matched the seller site.

Test Case Matrix

Test Case ID	Normal/Ab.	Black/Whitebox	Funct/Perform	Unit/Integr.
SCRAPE-01	Normal	Whitebox	Function	Unit
SCRAPE-02	Normal	Whitebox	Function	Unit
SCRAPE-03	Normal	Whitebox	Function	Unit
SCRAPE-04	Normal	Whitebox	Function	Unit
SCRAPE-05	Normal	Blackbox	Performance	Integration
MONGO-01	Normal	Whitebox	Function	Unit
WEB-01	Normal	Blackbox	Function	Unit
WEB-02	Normal	Blackbox	Function	Unit
WEB-03	Normal	Blackbox	Function	Unit
WEB-04	Normal	Blackbox	Function	Unit