**Product Lifecycle**

**Requirements**

Fellows are required to have completed the following Outcomes: 1 - 8

**Problem Description**

A product lifecycle where goods go from one stage to the other in real time. E.g A product goes from the **Producer** to the **Retailer and** to the **Consumer** and the **Recycler** takes the waste from the **Consumer** and recycle for the **Producer** to use in the next production linein real time. The retailer waits for the producer to produce the goods when they ran out of stock. The consumer waits for the retailer to buy from the producer and the **Recycler** waits on the Consumer to recycle. Think of it as a Mineral or Bear bottling product. When any of the above-listed stages run out of resources to execute, It waits for the supplier before it can continue.

All actions are listed printed to the console and log into a file in real time. A file containing initial actions for each stage will be provided. Your output file should be added to git before final submission.

**How will I complete this project?**

1. Use ES6 consistently throughout the project
2. You can use Node JS for this task only as an application server, Use design patterns as much as possible. Avoid using inbuilt Node features and do not require or import any Node Lib accept fs for writing and reading from files.
3. Organise the folders for your module (application), to house both your code base and the tests
4. Write tests to cover all the methods to be written, before development begins (TDD)
5. Test acceptance is 97% coverage.
6. From the Product-Input file, The producer should produce all items without a stage listed beside.
7. All impunity should be removed and not processed. E.g // and random spaces should not be processed too.
8. Performance of your solution is supper important and should never hang or crash.
9. The output file should state where the product is coming from and where it’s going without impunity
10. The input file should never be touched.
11. All products are to be produced by the same producer.

**Steps to evaluate the Product Life Cycle checkpoint**

1. Prototype Methods
   1. At Minimum
      1. All tests must test all methods, including the conditionals that they may contain (Arguments are customized in the tests, so all conditions in the method get tested)
      2. Repeated activities are put into functions and referenced from there
   2. Exceed Specifications
      1. The codebase is efficient, not redundant loops or method calls that could deter performance.
      2. All Repeated activities are put into functions and referenced from there
2. Test Coverage
   1. At Minimum
      1. All the prototyped methods are to be covered fully with tests
      2. The tests should cover the methods as well as the conditions/procedures that the methods employed
      3. All the comparators used for the tests must be the optimum or most efficient comparator for the scenario