**E-COMMERCE PLATFORM**

**Big O Notation**

* **Definition**: Big O notation is a way to describe how the performance of an algorithm changes as the size of the input increases. It helps in figuring out the efficiency of an algorithm, especially as the dataset grows larger.
* **Purpose**: It helps us in understanding how much scalable a code is by giving as the worst time complexity as well as allows us to compare different algorithms and select the one that will perform best as the amount of data increases.

**Best, Average, and Worst-Case Scenarios for Search Operations**

* **Best Case**: The scenario is when the element to be searched is found the immediately. For example, in a linear search, this occurs when the target is the first element. For binary search, it occurs when the target is the middle element.
* **Average Case**: This the case when the target element could exist anywhere in the dataset and generally depends on the size of dataset.
* **Worst Case**: This the case when the target is found after many iterations, either when the element is found at the last or is not even present.

**Time Complexity Comparison**

* **Linear Search**:
  + **Best Case**: O(1) - The element is the first in the array.
  + **Average Case**: O(n) - The element is somewhere in the middle.
  + **Worst Case**: O(n) - The element is the last or not present.
* **Binary Search**:
  + **Best Case**: O(1) - The element is the middle one.
  + **Average Case**: O(log n) - The array is halved each time.
  + **Worst Case**: O(log n) - The element requires halving the array until it is found or the search space is empty.

**Suitability for Platform**

* **Linear Search**: Suitable for small datasets where the overhead of sorting is not justified, or for unsorted datasets where binary search cannot be applied.
* **Binary Search**: More suitable for large datasets where the data is sorted. The O(log n) complexity provides much better performance as the size of the dataset grows.