Performance Comparison and Analysis of Chosen Data Structures and Algorithms:

Data Structure Used:

Data Structure	Purpose	Implementation Type
Binary Search Tree (BST)	Storing and managing Products	Dynamic, hierarchical
Singly Linked List	Managing Suppliers and Stock	Linear, pointer-based

Time Complexity Comparison

Operation	BST (Product)	Linked List (Supplier/Stock)
Insertion	Average: O(log n) Worst: O(n)	O(1) (at head)
Deletion	Average: O(log n) Worst: O(n)	O(n)
Search	Average: O(log n) Worst: O(n)	O(n)
Traversal (Display)	O(n)	O(n)

BST time complexity assumes a balanced tree. If unbalanced, worst-case becomes O(n).

Space Complexity

Data Structure	Space Usage
BST	O(n) for n nodes
Linked List	O(n), but uses more memory due to pointers

Sorting Algorithms

Implemented:

Bubble Sort (used for Products and Suppliers)

Merge Sort (used for Stocks)

Sorting Time Complexity

Algorithm	Best	Average	Worst	Stable?	In-place?
Bubble Sort	O(n)	$O(n^2)$	$O(n^2)$	Yes	Yes
Merge Sort	O(n log n)	O(n log n)	O(n log n)	Yes	No (uses extra space)

Bubble Sort is simple, but inefficient for large data.

Merge Sort is fast and reliable, but uses extra space.

Searching Algorithms

Implemented:

Linear Search (used for both)

Binary Search (used for sorted arrays)

Searching Time Complexity

Algorithm	**Best**	**Average**	**Worst**	**Requires Sorted Data?**
Linear Search	O(1)	O(n)	O(n)	No
Binary Search	O(1)	O(log n)	O(log n)	Yes

Linear Search is easy and works on unsorted data.

Binary Search is faster, but only works if the array is sorted.

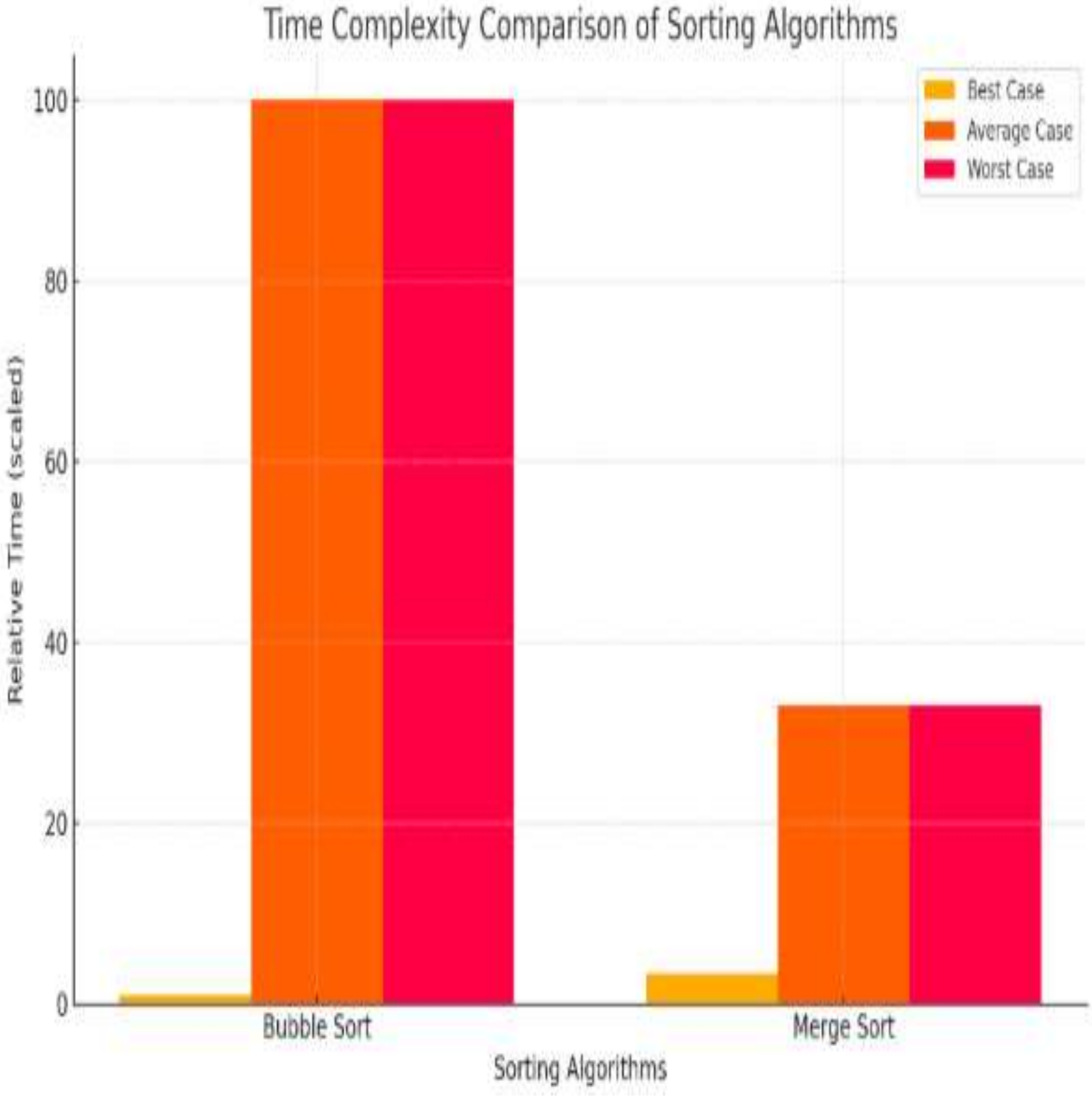
Justification & Suitability

Component	**Data Structure / Algorithm**	**Reason**
Products	BST	Efficient insert/search with unique IDs
Suppliers	Linked List	Simple structure, fewer entries
Stocks	Linked List	Allows easy quantity updates
Sort Products	Bubble Sort	Simpler to implement, acceptable for small data
Sort Stocks	Merge Sort	Handles large stock records efficiently
Search Products	Binary + Linear	Fast search on sorted list; fallback to linear
Search Suppliers	Binary + Linear	Same logic as Products

Bar graph comparing the **time complexity of Bubble Sort and Merge Sort** in best, average, and worst cases. As shown:

Bubble Sort grows very quickly in time (inefficient for large data).

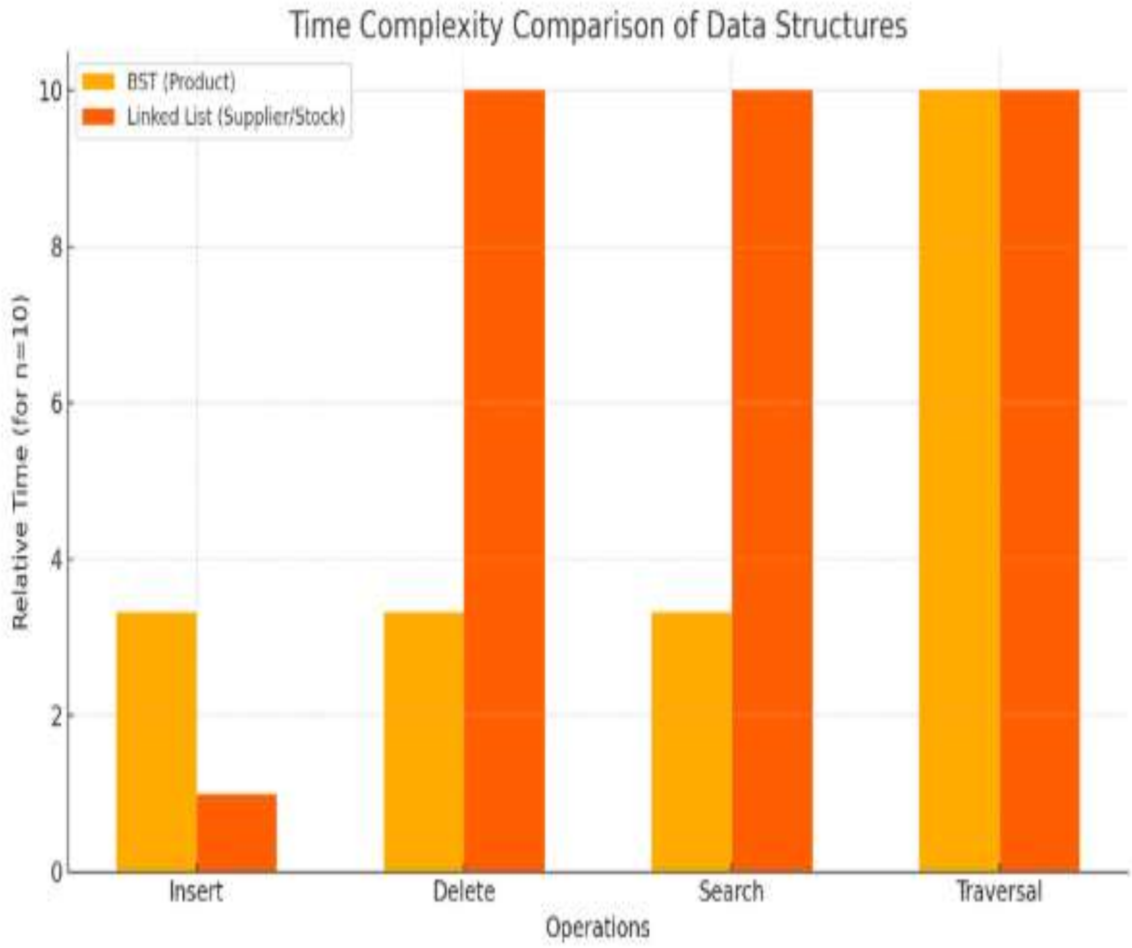
Merge Sort is consistently faster and more scalable, especially for larger datasets.



Bar chart comparing the **Binary Search Tree (BST)** and **Linked List** in terms of common operations:

BST is efficient (logarithmic) for insert/search/delete in average cases.

Linked List is fastest only for insertion at the head, but slow (linear) for other operations.


Time Complexity Comparison of Data Structures
BST (Product)
Linked List (Supplier/Stock)
Relative Time (for n=10)
Insert
Delete
Search
Traversal
Operations