**Task Management System**

Linked lists are fundamental data structures in computer science, and there are several types, each with its own characteristics:

### 1. Singly Linked List

**Structure**: Each element (node) contains two parts:

* + **Data**: The value or information stored in the node.
  + **Next**: A reference (or pointer) to the next node in the sequence.

**Traversal**: we can only traverse the list in one direction, from the head to the end of the list.

**Operations**:

* + **Insertion**: Can be performed at the beginning, end, or a specific position.
  + **Deletion**: Similar to insertion, it can be done at the beginning, end, or a specific position.
  + **Search**: To find an element, we must traverse from the head until the element is found or the end is reached.

**Advantages**:

* + Simple and uses less memory compared to doubly linked lists.
  + Easier to implement.

**Disadvantages**:

* + Cannot traverse backwards.
  + More complex to reverse the list or perform some operations that are straightforward in a doubly linked list.

### 2. Doubly Linked List

**Structure**: Each element (node) contains three parts:

* + **Data**: The value or information stored in the node.
  + **Next**: A reference to the next node in the sequence.
  + **Prev**: A reference to the previous node in the sequence.

**Traversal**: we can traverse the list in both directions, from the head to the end and from the end to the head.

**Operations**:

* + **Insertion**: Can be performed at the beginning, end, or a specific position, and the previous and next pointers are adjusted accordingly.
  + **Deletion**: Similar to insertion, but requires updating both the previous and next pointers of the adjacent nodes.
  + **Search**: Can be done in either direction, but still requires traversing through the nodes.

**Advantages**:

* + More flexible than singly linked lists due to bidirectional traversal.
  + Easier to implement operations like reversing the list or inserting and deleting nodes from both ends.

**Disadvantages**:

* + Uses more memory than singly linked lists due to the extra pointer (Prev) in each node.
  + Slightly more complex to implement and manage due to the extra pointers.