**Warehouse Management System Project Summary**

**Introduction**

The Warehouse Management System (WMS) is designed to efficiently manage the storage and retrieval of products in a warehouse. The system employs a set of Java classes to represent products, totes (containers for products), and the overall warehouse. The data structure is designed to facilitate product loading, order fulfillment, tote merging, and warehouse status reporting.

**Data Structure**

**Product Class**

The **Product** class represents a product with a unique identifier (UPC). This class is a simple container with a single attribute, **upc**, and a getter method. Products are stored as instances of this class.

**Tote Class**

The **Tote** class represents a container that can hold multiple products. It contains a list of products and methods to check if the tote is full, empty, or to add products. The **Tote** class is used to organize and group products efficiently

**Warehouse Class**

The **Warehouse** class represents the entire warehouse and implements the **IWarehouse** interface. It contains a list of totes and methods for loading products, fulfilling orders, merging totes, and displaying warehouse details.

**IWarehouse Interface**

The **IWarehouse** interface defines the contract for warehouse operations. It includes methods for loading products, fulfilling orders, merging totes, and displaying warehouse details.

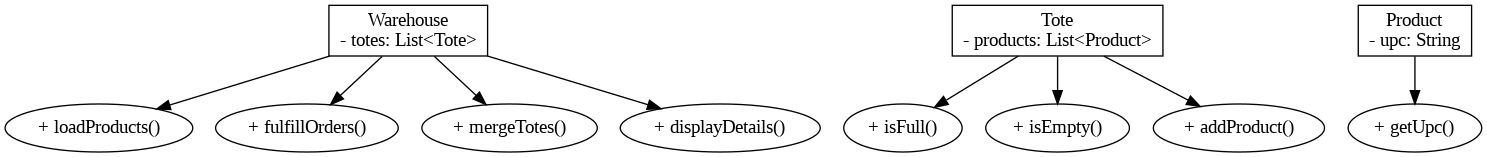
**Memory Storage**

Products are stored as instances of the **Product** class, and totes are collections of products stored in instances of the **Tote** class. The **Warehouse** class maintains a list of totes.

**Rationale**

1. **Product Class**: The **Product** class provides a clean and simple representation of a product. Each product is uniquely identified by its UPC, making it easy to search and retrieve.
2. **Tote Class**: The **Tote** class allows the efficient organization of products. It provides methods to add products, check if the tote is full, and determine if it is empty.
3. **Warehouse Class**: The **Warehouse** class orchestrates warehouse operations. It maintains a list of totes and provides methods for loading products, fulfilling orders, merging totes, and displaying warehouse details.
4. **IWarehouse Interface**: The interface abstracts the operations that a warehouse should support, allowing for flexibility in implementation.

**UML Diagram**



**Conclusion**

The chosen data structure provides a modular and extensible design for a Warehouse Management System. It allows for efficient storage, retrieval, and management of products within a warehouse. The classes and interface offer a clear separation of concerns, making the system easy to understand, maintain, and extend.