**PasoEats Design Explanation**

**Encapsulation**:

* All fields are private. Variables are only accessible through getter and setter methods.

**Inheritance**:

* Customer, Driver, and Admin extend User to reuse its class attributes (UUID id, String username, String name).

**Polymorphism**:

* For our future CLI we will use different fields from the User class for each user type.

**Abstraction**:

* User class is abstract and has abstract methods for each user type.
* Orders exposes place/acceptNext/markStatus/get while hiding its Map and ArrayDeque

**Data Structures:**

* byId: Map<UUID, Order> - to store orders by their ID and have O(1) lookup time.
* intake: ArrayDeque<UUID> - to store orders in the order they were placed and have O(1) insertion and removal time. FIFO (First In, First Out).
* List<OrderItem> (inside Order) - to store the items in the order.
* UUID (User, Order, Driver, Customer, Admin) - to identify each object uniquely and control access to them in a secure way.

**Overall Complexity:**

**Time Complexity:** O(1) for all operations. (*Subject to change if the design evolves.*)

**Space Complexity:** O(n) for the map and queue.

**Class Diagram**

