REQ 2 Design Rationale

The diagram represents the second requirement/part of an object-oriented system for a text-based “rogue-like” game inspired by Elden Ring.
We have two design goals for this requirement: implementing runes and implementing the trader and trading actions.

We intend to implement a different mechanism to pick up runes compared to other items. For example, when picking up runes, the runes' value would just add on to an existing runes object's value inside the player's inventory, rather than just adding new rune objects to the player's inventory everytime they pick up runes. Hence why we decided to create a new PickUpRuneAction. Moreover, we extended PickUpRuneAction from PickUpAction rather than PickUpItemAction because we want to avoid modifying the parent's methods completely (OCP). The same applies for DropRuneAction.

We introduced seperate classes for purchasing and selling each individual weapon rather than a single class for purchasing weapons (e.g. PurchaseAction) and another class for selling weapons (e.g. SellAction). We did this to avoid having the PurchaseAction and SellAction classes from having too many responsibilities/god class (SRP). Instead, we have made them abstract classes where each weapon purchase/sell action extends from them to avoid repitition between the action classes (DRY).

We then also package all the classes and interfaces into their respective package for organization, modularity, and encapsulation.