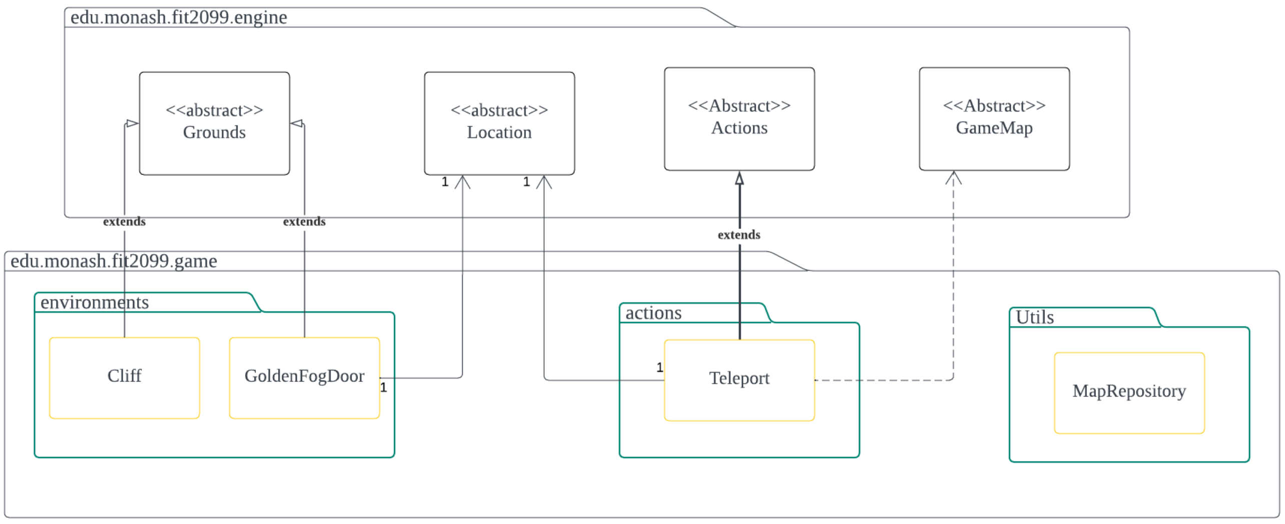
REQ1: Travelling between Maps

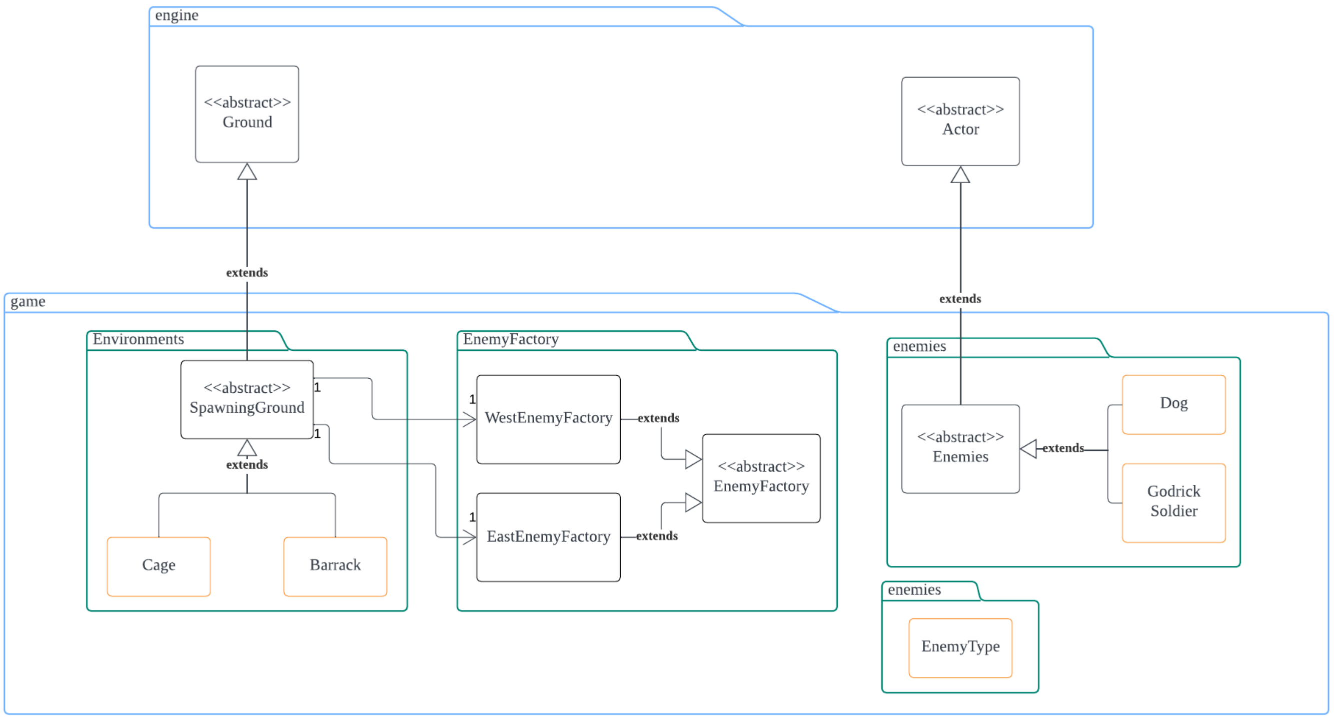


Cliff and GoldenFogDoor are new Grounds that were introduced in this requirement. They are not responsible for spawning any Enemy Characters, therefore, we decided to extend it from the Ground abstract class. This will comply with the Open Closed Principle which allows our code to be open for extension but closed for modification.

The Teleport class is extended from Action abstract class which moves an actor to a specific location when called. It is specifically used by GoldenFogDoor which is a teleporting door. When the player comes to the surrounding of the door , an allowable action will be prompted for the player to select.

Finally, the Map Repository class is simply just a class to store the Array of Lists maps which does not need any extensions nor implementations.

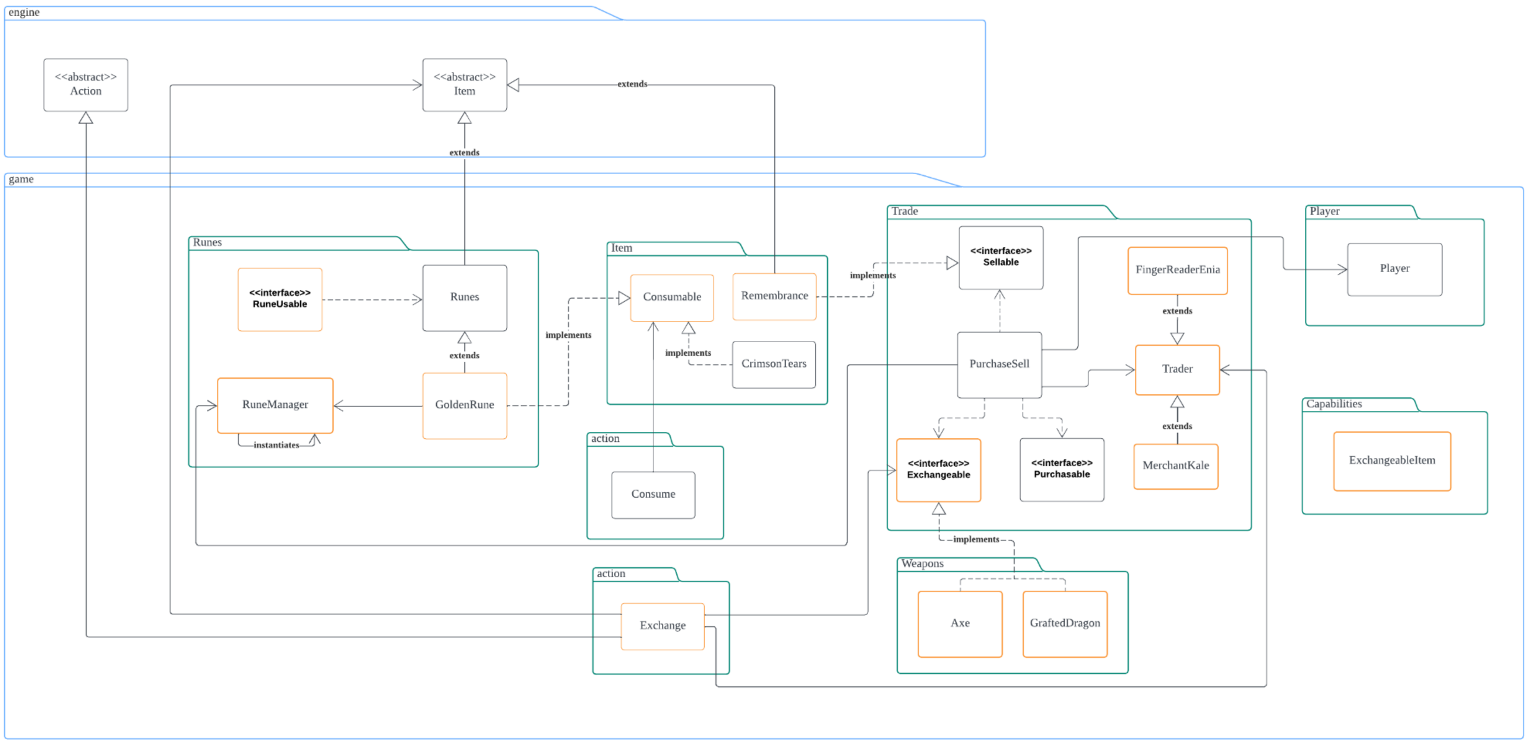
REQ2: Inhabitant of the Stormveil Castle



The new spawning grounds required in this requirement are the Cage class that is used to spawn Dogs at each turn while the Barrack class is used to spawn Godrick Soldiers at each turn. They are all extended from SpawningGround to adhere (DRY - Don't repeat Yourself Principle)

The Dog and GodrickSoldier class are both extended from the Enemies abstract class for the same reason as well. A new capability under EnemyType was introduced. It has been classified as Castle type so that the dogs will not attack the GodrickSoldier and vice versa since they are enemies of the same type.

REQ3: Godrick the Grafted



In this Assignment3, more classes are introduced to fulfil new requirements and interfaces to align with the SOLID principles.

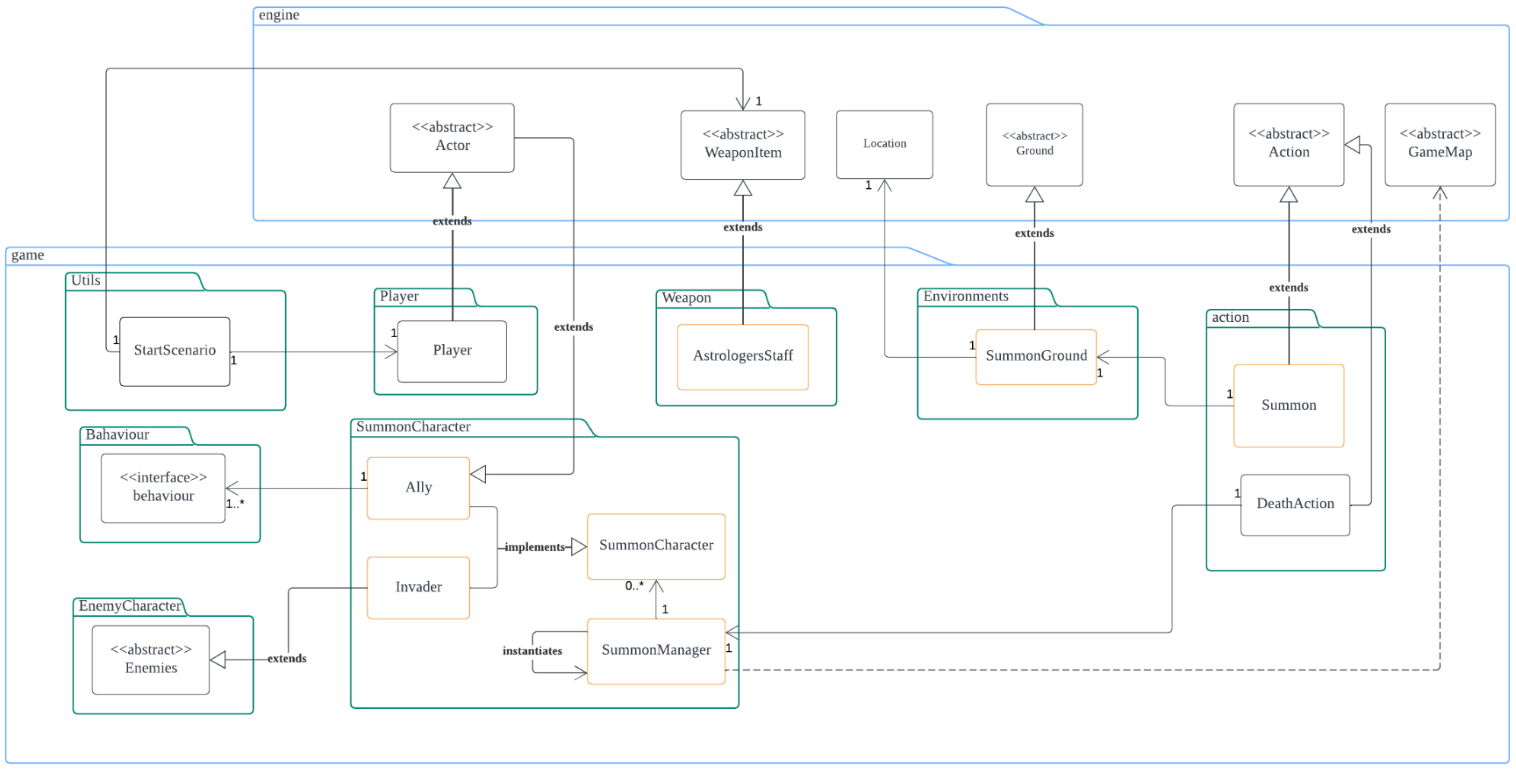
For the GoldenRune, which is a special type of Rune which has all the characteristics of the Rune, so the Rune class is extended to create the GoldenRune class with its unique implementation in alignment with the DRP (Don't Repeat Yourself Principle). The same reasons are applied to the Remembrance which extends from Item class; PickGoldenRune that extends from PickUpAction; FingerReaderAnia class extending from Trader; Axe and GraftedDragon extending from WeaponItem; Lastly, Exchange class extending from Action class.

Apart from those new classes, the new Interface Exchangeable Interface is introduced to allow all the Items or Weapons that are supposed to be Exchangeable to be Exchanged with the FingerReaderAnia by using Remembrance. At the same time, it would provide the abstract implementations that would only be needed by the Exchangeable items in future. In addition, the Remembrance also can be sold to the FingerReaderAnia for 20000 Rune as it implements the Sellable Interface which would adhere with the ISP - Interface Segregation Principle.

In addition, the RuneUsable interface is created to facilitate the actors that can use the Rune to implement all the methods provided by the RuneManager which by the meaning of its name is responsible to handle all the Rune-related processing. As for such design, all Rune-related methods could be transferred to the RuneManager while complying with the SRP - Single Responsibility Principle.

Nevertheless, we have also implemented a Consumable interface and implemented it on CrimsonTears and GoldenRunes. Inside the Consumable interface, we have a method called consumed() which every consumable item will have different ways to consume. From the previous design we did not implemented this, so it is inconvenient for us to add more consumable item in the future. This approach has achieved the open-closed principle which we can open for extension but closed for modification.

REQ4: A Guest from Another Realm



AstrologersStaff extends WeaponItem. AstrologersStaff is a new weapon which the player has as their starting weapon if they choose the scenario of Astrologer.

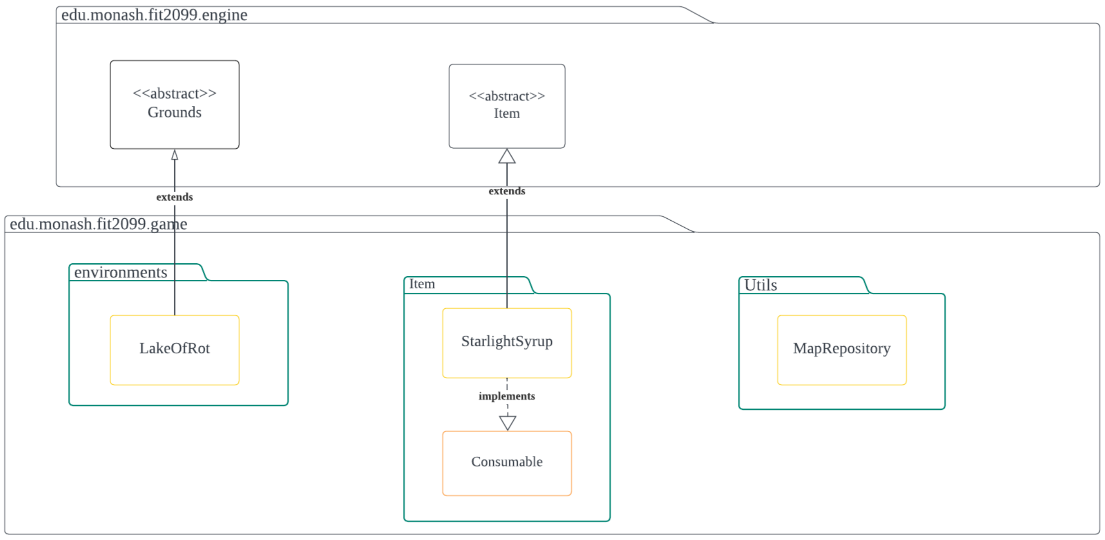
SummonGround extends Ground. Summon ground is a ground that allows the enemy to summon an ally or an invader.

A Summon Character interface is created and it is implemented by Ally and Invader.By creating this interface, it is easier to let the SummonManager manage these characters. For example, summon characters will be removed from the map when the player dies, so in the DeathAction class we can just use the method SummongManager to remove these summon characters. This implementation has achieved the Dependency Inversion Principle.

Ally is a class that extends Actor. The reason that Ally is associated with Behaviour is because it is a type of NPC thus it is more suitable for it to have a hashmap list to store the behaviour with priority order and each behaviour can return an action. By doing so, Ally can determine what actions to do by looping through the behaviour list and find the action that has more priority.

Since invaders can attack any type of actor except the actor of the same type (invader), and also it will drop runes to the player when it is defeated by the player. Thus, we extend Invader from Enemies because they have common attributes and methods, it seems sensible to abstract these identities in order to prevent repeats (DRY).

REQ5: Creative requirement



Description of our design :
The original route to Boss Room starts from the end of Limgrave, travels through the teleporting door to Stormveil Castle, walks till the other end of the map and travels through a door again to the Boss Room. The journey is considered quite far and challenging, so we've decided to create a new map that transports players to the Boss Room through a shorter distance. However, the trade offs are such that the player has to walk through the Lake Of Rot which will damage the player's hitpoints by 35 hp every turn. We have also added some enemies that will spawn through Puddle of water in the lake area, Graveyard that spawns a HeavySkeletalSwordsman on the ground. The mechanisms are such that the HeavySkeletalSwordsman cannot enter the lake. As for crabs that were spawned, they are allowed to move to the ground as well such that if a player passed by a crab, it might follow the player to the ground.

We have also added a consumable Item named Starlight Syrup. It is an item that the player can choose to consume directly or pick up into their inventory. The effects are such that it heals the player gradually with 20 hit points each turn. This item's effect can only last for 5 rounds but it would be pretty useful for the player while travelling through the Lake of Rots.

UML Design
The Lake of Rot is not responsible for spawning any Enemy Characters. Therefore, the relationship would be extended from the ground abstract class as an extension of it, hence complying to the Open Closed Principle .

As for the StarlightSyrup, it is extended from the Item abstract class and also implements Consumable. By using these responsibilities from distinct classes (Item class and Consumable interface), the StarlightSyrup class adheres to the Single Responsibility Principle. Each class and interface has a clear and specific responsibility that can be used by the StarlightSyrup.