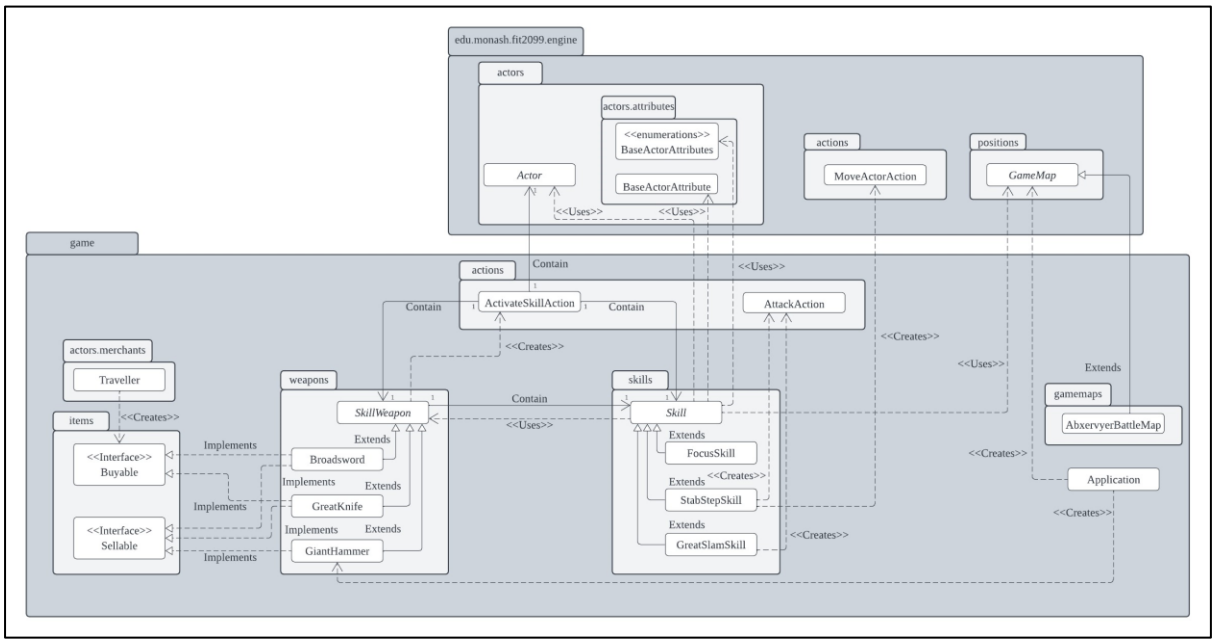
FIT2099 Assignment 2 Design Rationale

Requirement 4:



- The implementation of the skill feature has changed as compared to assignment 1. In this design, the effect of the skill is managed by the skill class instead of managing by the weapon that contains the particular skill. This follows the SRP principle as now the weapon class manages all the actions related to the weapon itself while the skill class manages the effect of the skill on the weapon that activates the skill.

- The new StabStepSkill and GreatSlamSkill are inherited from the skill class. The reason to implement it as a class instead of an interface is that all the skill will have some common attributes such as stamina cost or duration. Having an abstract Skill class can help to avoid code repetition (DRY).

- The ActivateSkillAction class now contains an additional attribute, skill. This additional skill attribute allows the player to choose what skill to be activated. This follows OCP principles as if a weapon with multiple different skills is added into the game, the weapon can simply create multiple ActivateSkillAction classes which contains different skill for the player to choose. In the design for assignment 1, it is assumed that the weapon only allows to have one skill. Hence, if multiple skills weapons are added in the future, the existing code has to be modified.

- The ActivateSkillAction class consists of two constructors with one having an Actor type target. The purpose of this constructor is to allow the target to be specified for skills such as Stab Step and Giant Slam which is used to attack the enemy directly.

- GreatKnife and GiantHammer are extended from the SkillWeapon class as they both have skill to be performed. Through inheritance, repetition of code can be avoided (DRY).

Pros:

- Every class managing their own responsibility (SRP) can help to make the code easier to maintain.
- Using inheritance avoids the repetition of code.

Cons:

- There are too many associations and dependencies between SkillWeapon class, Skill class and ActivateSkillAction class which may cause our classes to be tightly coupled.

Future extensions support:

- If new weapons with multiple skills are introduced, simply add the skills as an attribute in the weapon class and pass the skill to ActivateSkillAction class. The skill will manage the effect on the weapon itself.
- New skills introduced can be extended from the Skill abstract class to avoid code repetition.