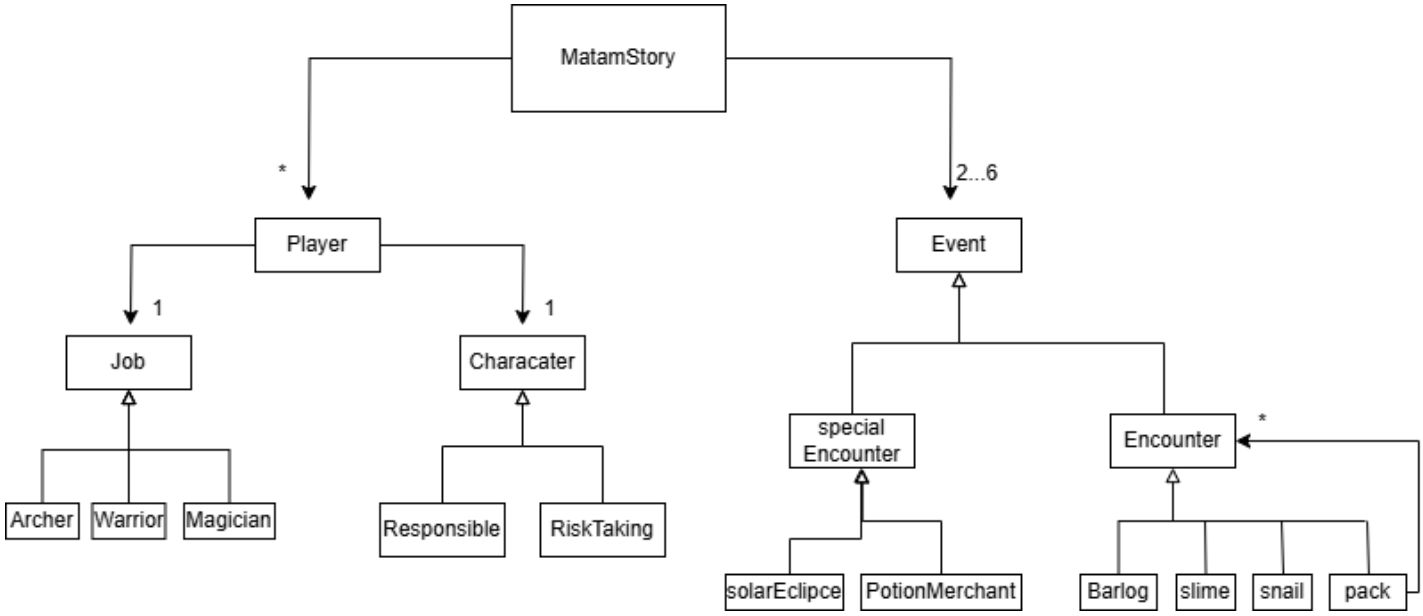

<u>חלק היבש:</u>

2- we used 2 Patterns:

The **Strategy pattern** : We can see this design pattern in the classes: Job and Character classes, where they define different behavioral strategies for the Player, such as Responsible or RiskTaking for Character and Warrior and Archer and Magician for Job. Each of these represents a separate strategy, encapsulating unique decision-making processes and behaviors.

The **Composite design pattern:** The Composite pattern is applied in the Encounter class, where the Pack contains a collection (container) of Encounter objects.

3- If we want to add a new job called **Rogue**, we need to create a class named **Rogue** that inherits from **Job**. Additionally, we must modify the function in **Encounter** (**<u>APPLY FUNCTION</u>**) by adding a condition to check if the player's job is **Rogue**. If so, and the encountered monster has at least **twice the combat power** of the player, the Rogue will **evade the battle** without engaging in combat, displaying an appropriate message.

4- Yes, we can implement this by adding **Divine Inspiration** as a subclass of the **Special Encounter** class, overriding its relevant functions. However, we also need to introduce new functions in the **Player** class, such as setNewJob() to allow changing the player's job dynamically and .randomJobFunction() to assign a random job when the event occurs