This requirement is about using the Pokémon’s affection for the player in order to determine if the Pokémon can/cannot be captured and if it will follow the player around (behaviours of the actor). The requirement also determines if the Pokémon’s affection increases, or decreases based on fruits given to them and their element. This affection point system is managed by the Affection Manager class which would store the Pokémon with their respective affection points. Furthermore, the status Enum is also utilised in this design as this makes it easy to know which Pokémon are catchable and which are not and can be changed based off affection points. This Enum ensures that code doesn’t use an excessive number of literals and follows the open-closed principle by keeping the code much more concise and making it, so we won’t have to code a various range of conditional for checks. We also create a ConsumePokeFruit action class that acts as a subclass for Action and is used to program what is done if the poke fruit is consumed by Pokémon. Instead of consuming the poke fruit directly in the Items subclass, we regard the intended usage of the engine and follow the single responsibility principle in this scenario by reflecting that most of the functionalities are executed after the poke fruit is given to the Pokémon.

The given Sequence Diagram outlines the process of a ConsumePokeFruit by a Pokémon to increase affection: