My idea for this project was a game where the player could customize their character through a shop interface and an inventory interface, which would show all their wearable objects, so I split the project code into four main systems:

Gameplay: This system holds every functionality related to the player; player data management, input, movement, animation, and shop/inventory navigation.

Interactable: This system deals with every object that can be interacted with. For now, there is only one interactable, the Shop, that manages shop stock functionality. There is an interactable detector attached to the player that checks every collision with interactables.

Item: Composed of Item Data, a base abstract scriptable object that holds information that every item has. For each type of item, there should be a class the derives from Item Data. For now, there is only the WearableItem, which holds data of items that can be equipped.

From the base item asset, instances of these items can be created with the Item Instance class to represent them in the game world, which are used in shop stock and inventory management.

The player animations are handled by a single Animator Controller, which sends animation events at every animation frame to the Player Animator class. The player character has 8 wearable slots: Hair, Head, Chest, Legs, Feet, Hands, Ring, Neck and Back. Whenever a wearable is equipped, its respective slot is activated with a new renderer layer according to the wearable layer type, which is then updated by the Player Animator at every animation frame.

UI: Holds code that communicates directly with the Gameplay system and manages UI elements.

I think my performance was good given the time I had available and the results, and I think I could finish it even faster if it wasn’t for the breaks I took during the task, but I would get tired faster too, so I took the first approach.