The game system is divided by: Player, UI systems, Shops and a Wardrobe.

The Player character has two scripts:

1. PlayerStats – Hold all information and stats the player may have. In this case, money and the equipped hat animations are stored.

In EquipHat() and UnequipHat(), this component uses an AnimatorOverrideController to replace the AnimationClips of the EquippedHat object with the animations provided by the ItemDatabase object, who stores all information about items;

2. PlayerController – Controls the behavior of the player character.

This component the movement and aspects of the animations of the player. I opted to use raycasting over Unity's built in systems for better control and aimed to emulate Zelda's movement.

In Update():

- A raycast will check for collision objects and either let the player walk in the direction or put it in the object's boundaries.
- The player's input is checked and sent to the AnimationController. The states synchronizes the hat animations to match theirs.
- If the interaction button is pressed (space), another ray is cast in front of the player to check for interactables.

Any interactable will make use of the InteractableBehaviour interface and have an Interact() method.

The shops have two scripts:

1. ShopInventory – Receives item inputs from the inspector and creates a list of items based on the information, then stores it in a List.

2. ShopBehaviour – Handles dialogue and opening/closing shop.

Update(): The shops have a Boxcollider2D as an interact check. If the player leaves the area, it will act as if closing the shop (will unequip previewing hats).

Interact(): Calls the animation to show/hide the UI and generate a dialogue from a random selection of provided options.

DialogueLoop(): After opening the shop, it will keep adding characters to the dialogue box to animate it according to the speed chosen.

The wardrobe is functionally similar.

The Shop and Wardrobe UI generate a list of prefab buttons according to the list of items provided and handle the items interactions.

ShopUI:

PreviewItem(): Equips the hat animation and stores the original hat to be re-equipped later.

UnequipPreview(): Equips the original hat.

BuyItem(): Adds the item to the player Wardrobe and removes from its Inventory. If the item is being previewed, it equips it.

SellItem(): Adds the item to the Shop inventory and removes from the Wardrobe. Unequips the hat if equipped.

WardrobeUI:

PreviewItem(): Same function as the shop's.

EquipItem(): Equips selected hat, stores the item's index in the Inventory and updates the object's equipped status.