Development Thought process on the Functional Clothes Shop Project:

To define the goals of the project, I listed the following items as the most important qualities of the project:

Top-down view (like Stardew Valley), Player animation, Object / character interaction, Character movement, Items with icons and prices, Item store, Equipping outfits (visible in the game world).

While no art assets were created, I developed all the necessary code files. On the art side, I looked for the free assets such as "Mighty Heroes (Rogue)" and tiles to establish a world framework, to showcase a proper character navigation. Utilizing a UI pack and a low-poly font to match UI and world.

I used Unity Tiles for the environment, creating wall collisions, and showcasing the character movement. There is a script for reading values from the New Input System, checking colliders via RayCast2D, and arranging the movement allowed at that moment.

Regarding UI, I utilized the low-poly package and the sprites from character clothing to create buttons and UI representation of the items. However, managing the character UI sprites, comprising 15 sprites each time, was time consuming and could be handled with a data structure for the sprites renderer consuming. Initially intending to treat each sprite as an individual item, but the changes in clothes would not be very visible.

Because of the lack of formatting to the sprites for sprite renderers, I duplicated the code three times within the project - for item descriptions, UI character merging, and sprite rendering over the character armature. Unifying this code would make the project more maintainable, albeit at the cost of additional implementation time for operations and monads.

On the positive side, I effectively reused logical components like the interaction finder - a 2D prefab applicable to any game object. It facilitated passing a name (for UI information) and an event to be invoked, implemented initially as a button for player interaction with the shopkeeper. This functionality could be extended to direct object clicking or collision mechanics, such as combat, or to trigger animations.