README

(Programmer Interview-NATAN CĂBĂU)

Shop Simulator is a game where a Shopping system is implemented. It is created from multiple subsystems, among which the following are included:

- **ShopItem, which is a class assigned to each item in the shop and implements the IShopItem interface. This interface extends the methods Buy() and DisplayOffer(). It contains the sprite, price, and name of the item. With the help of this class, we retrieve the information and send it to the ItemShop.**
- **ItemShop is the class that stores information about purchased items. It possesses the functions Equip() and Sell(), which are attached to the UI buttons of the inventory. It is responsible for equipping the player with items from the inventory and for selling items.**
- **InventoryManager is used to store the body parts of the player that have clothes on them. It also has a function that receives information from the inventory item and equips the player, replacing the sprite with the one from the inventory.**

- **In GameManager, the main character, their budget, the general offer, and the budget's texture are stored. By implementing a singleton, it becomes easy to access the GameManager and to set and retrieve information from it.**

The project files are divided into departments: Art, Programming (Mechanics), Audio, and the scenes. Then, the files within each department are distributed by features, such as Player, Shop, and GameManager.

As programming principles, I've applied some SOLID principles like OCP and SRP. Additionally, to maintain clean code, I've used DRY. For readability and functionality, I've aimed to be as concise as possible, and I've also commented on code sections.

Additionally, for the implementation of the shopkeeper, I utilized NavMesh along with a simple NPC script (ShopKeeper) to enable patrolling.

I would like to mention that the NavMesh package wasn't written by me; it was obtained from GitHub. However, the ShopKeeper script is my own creation.

This task wasn't difficult for me at all, and I'm pleased with how I managed it, especially given my prior experience with similar tasks. My senior experience is evident here, as I strived to create code that is both optimized and readable, in my opinion. However, I also believe that there's always room for improvement, and we never truly become too skilled – we can only strive for perfection.