Report on Blue Gravity Studios Interview task

-Task

Build a top down style RPG with shop, shopkeeper, buy/selling functionality and character customization (visually see the item being equipped on the character).

-Time limit to complete task

48 hours

-Development Process

The first thing to do was to create an environment. A simple asset containing a collection of sprites was downloaded from the Unity Asset Store.
Using the Unity sprite editor a sprite pallet was created. With this pallet a simple environment was created. The environment is divided in layers to facilitate
the creation process. The layers are also necessary for the easy creation of the colliders and to produce effects of showing and hiding certain parts of the environment when required.

With the environment ready, was time for the creation of the player character. Also using the sprites downloaded from the Asset Store, a simple Animator with a 2 dimensional Blend Tree was created. The blend tree takes 2 parameters (horizontal and vertical with a range of -1.0 to 1.0) to figure what sprite to use. This parameters are feeded by
the PlayerController following the player's input.

Once the player character was working, the CameraController was created to manage the movement of the camera. The camera will follow the character movement with a small delay, which is more pleasant to the eyes. The higher the smoothing factor in the CameraController the closer the camera will follow the character.

The door of the shop presented a small challenge. To prevent the player from "stepping" on the door sprite when entering the shop, an empty object marking the door position was used as reference so the ShopDoorController could tell when the player was entering and hide the door sprite. The final result is not pretty so, it was my intention
to create a separate door sprite, showing the door opened, but since the time was pressing, that was left for later. The same principle is used to hide/show the shop ceiling.

Next, the shopkeeper was created. Following the same processes and similar code used in the creation of the player character. The only difference is a crude automation
of a random movement, to keep the seller moving and make it more interesting. But, with that movement, a problem emerged. The random movement of the seller could led him to exit the store. To prevent that, a collider was placed on the empty object. The seller and the empty object layers were set to interact with each other, but not with the player character.

Following the development, it was time for an inventory system. I had previous experience using Json to manage different in-game items, but after a quick research, I found out about ScriptableObjects and decided to use those. With the first day of development coming to an end, I came to the conclusion that creating such a system from scratch would take too much time. So a tutorial on the subject was used as the basis for the inventory development (https://www.youtube.com/watch?v=w6_fetj9PIw). The most interesting part of the solution is a delegate function used with a callback. It makes the update of the UI much simpler. Every time there is an alteration on the inventory, the callback function is invoked. This function was previously registered with the actual function that will take care of all the updating process. An empty GameObject as a Singleton was created to manage all the inventory mechanisms.

Modifying the tutorial solution, I was able not only to create an inventory for the player and the shop, but to create a buy/sell system.

The last task was the creation of the "equip" mechanism. After a little research, a system that uses the names of the sprites to change the player appearance was created. Based on the item equipped, the system gets the current sprite being animated in the SpriteRenderer, obtain the sprite name and number (the sprites have to be named in a certain pattern for this to work), find the equivalent sprite of the equipped item in the sprites array (Organized by item. That is, each item has a duplicate of sprites required for all the animations), and replaces it in the SpriteRenderer. This solution is not elegant or scalable, but by the end of the second day, it was the best I could do.

-Personal Assessment

I had never built a game like this before. It was all very new for me, since I never worked on a RPG, not even for fun. But from the start I recognized that the inventory system was going to be the most difficult task.

With the proper time, I'm certain I would be able to come up with my own solution. But after I finished, I realized that I made the right choice, since even modifying a existing code to my specific needs took long enough.

Overall, I'm glad I was able to conclude the task in the required time, but 48 hours was not enough to come up with elegant, scalable solutions. I had to duplicate every single character sprite for each item, in addition to a "naked" character. There is certain a better solution for this, but not in 48 hours, not for me at least. If I had more time I would refactor the code to make it more simple, easier to read and maintainable. I would have created an open door sprite, or even an animation, to replace the closed door when entering the shop. I could have made a currency so the player could use coins to buy and sell items and finally an interface for the game.

Concluding, I evaluate this task as difficult in the time frame of 48 hours. The inventory system is complex, and manage the UI, with all the icons and buttons, although not complex, is time consuming, as it is the swapping of sprites in the player character.