My code review partner and I had a lot of discussions on the pros and cons for creating an interface for every class. My code doesn't create an interface for every class, but his does. To better illustrate our different design decisions, I will give an example. For example, I created an Item class, he also created this class, but he also had an IItem interface and having that Item class implement this interface. I argue that it is not necessary to have this interface in this case, because you are essentially creating another abstraction layer for the sake of abstraction. But he argued that this interface follows dependency inversion principle, as the class is now dependent on interface abstraction. This is the case if you are going to create different Item Class such as Desk, Shoes, etc. then it is better to have these classes implementing IItem interface, but in my case, I do not have these many classes, I am just creating objects based on my Item interface, then all of those item objects are Item. I also argued that interface should be favored when you want to define shared behaviors.

In the future, I will continue sticking with single responsibility principle as I could see the benefit of creating a class and defining attributes and behaviors only unique to this class, and DIP and Open-Closed principle which allows me to add functionalities without modifying the code.