Company background: You work at an ecommerce website nocollar.com, which sells graphic T-shirts that are supplied by multiple independent sellers. The website makes deals with different sellers, like “Graphy Shirts” and “Tommy’s Fashion House" (TFH), so they can showcase and sell their T-shirts on nocollar.com

Challenge: T-shirts sellers offer a wide range of pricing that results in extreme diverse prices for what seems to be the same product. Your customers don’t see the vendors’ information when buying T-shirts. So the customer don’t know the difference between Graphy’s and Tommy’s products. This makes it confusing for the customer to understand the price differences between T-shirts. Some sellers will drop their prices in order to get more sales, creating a bigger gap between T-shirt prices. While high-end sellers (like TFH) want to commit to their high prices, they don’t like the fact that the platform is offering really low prices from low-end sellers that makes the high-end sellers look really expensive.

Platforms/Products (Domain):

● nocollar.com: Your main website, which is the customer-facing app where they have a profile, shop from a wide range of T-shirts, and happily place their orders

● seller.nocollar.com: Where you manage a portal for T-shirt vendors to add their products to the website, along with their pricing and other selling management activities.

The ASK: “How to make the customer order the high-end products (Expensive) while capitalizing on the vendors’ price wars and maintaining the loyalty of the lower tier sellers” Think of solutions that will keep the vendors’ average pricing while allowing customers to buy the high-end products. And keep into consideration some vendors might not be really cooperative with you.

Deliverables:

1. Come up with 2-3 different solutions. Each explained in an average 3-line paragraph

2. Benchmarking: At least two benchmarks of similar solutions in other products

3. Now take just one of your solutions and perform the following

○ Break it down to user stories (Minimum of 5 user stories).

Each user story should be enough for the engineers to work and deliver a functional output.

○ Wireframes: Sketched mock ups ○ Prioritize the users stories explaining the logic behind it