# Building the store backend

In this lesson you will get an overview of the new store experience enabled in the updated FrontEnd portal and what backend pieces need to be built to support it.

## The store experience

In this module you will integrate an updated version of the FrontEnd portal with your microservices. Specifically, the new Store section.

The list of items presented in this section is similar to the list in the Catalog section, but in this case the user can also tell how much of each item it already has in his inventory bag. There is also a Purchase button that the user can now use to purchase a specific item. And also, there is an indicator in the top-right that tells the user how much gil is still available in his wallet.

We could certainly stand up a brand new microservice to support the Store section, but just to keep things simple, we will keep using our existing Trading microservice for this. The frontend will send a GET request to a new store endpoint, to which Trading will reply with all the data needed to populate the page.

Now, before Trading can provide this data, it will need the help of other microservices. Any time an item is updated in the Inventory microservice, Inventory will publish a new Inventory item updated event, that will notify Trading how much of an item a specific user has. Trading will store this in its local InventoryItems database collection.

Also, any time a user is updated in the Identity microservice, especially the amount of gil in his wallet, Identity will fire a new User updated event, that Trading will also consume. And Trading will store this data in its local Users database collection.

In the next few lessons you will create these new events, the corresponding consumers, and the new Store controller.