Implementing the Store controller

Script start

The updated frontend portal you will explore in the next lesson includes a new store experience and to support it we need to add a new controller to our Trading microservice.

In Trading repo

1. Update Dtos:

```
namespace Play.Trading.Service.Dtos
{
…
   public record PurchaseDto(
  …);

   public record StoreItemDto(
      Guid Id,
      string Name,
      string Description,
      decimal Price,
      int OwnedQuantity);

   public record StoreDto(
      IEnumerable<StoreItemDto> Items,
      decimal UserGil);
}
```

2. Add StoreController:

```
namespace Play.Trading.Service.Controllers
{
   [ApiController]
   [Route("store")]
   [Authorize]
   public class StoreController : ControllerBase
   {
      private readonly IRepository<CatalogItem> catalogRepository;
      private readonly IRepository<ApplicationUser> usersRepository;
      private readonly IRepository<InventoryItem> inventoryRepository;

      public StoreController(
         IRepository<CatalogItem> catalogRepository,
```

```
            IRepository<ApplicationUser> userRepository,
            IRepository<InventoryItem> inventoryRepository)
        {
            this.catalogRepository = catalogRepository;
            this.usersRepository = userRepository;
            this.inventoryRepository = inventoryRepository;
        }

        [HttpGet]
        public async Task<ActionResult<StoreDto>> GetAsync()
        {
            string userId = User.FindFirstValue("sub");

            var catalogItems = await catalogRepository.GetAllAsync();
            var inventoryItems = await inventoryRepository.GetAllAsync(
                item => item.UserId == Guid.Parse(userId));
            var user = await usersRepository.GetAsync(Guid.Parse(userId));

            var storeDto = new StoreDto(
                catalogItems.Select(catalogItem =>
                new StoreItemDto
                (
                    catalogItem.Id,
                    catalogItem.Name,
                    catalogItem.Description,
                    catalogItem.Price,
                    inventoryItems.FirstOrDefault(
                        iventoryItem => iventoryItem.CatalogItemId == catalogItem.Id)?.Quantity ?? 0
                )),
                user?.Gil ?? 0);

            return Ok(storeDto);
        }
    }
}
```

3. Start the Trading service

4. Browse to the Swagger page:

   https://localhost:5007/swagger

5. Copy the swagger.json url

In Postman

6. Import the Trading collection
7. Update the baseUrl in the Variables
8. Call the Store GET API
9. Start the Catalog service
10. Do a PUT in any other Catalog items not showing up in Store
11. Call Store API again and confirm the updated items show up

In the next lesson we will use the updated Frontend portal to query the new Store API.