Assignment 1 Sales Board

Contents

[Introduction 2](#_Toc50392430)

[Phase 1: Creating MVC Application and Data 2](#_Toc50392431)

[Setting up Application 2](#_Toc50392432)

[Modifying Identity User 2](#_Toc50392433)

[Database Modelling 2](#_Toc50392434)

[Scaffolding 3](#_Toc50392435)

[Phase 2: Shopping Cart, View models 3](#_Toc50392436)

[Appendix 4](#_Toc50392437)

# Introduction

Github Link: <https://github.com/ZephyrGitCode/HIT339_Assign1_Salesboard>

# Phase 1: Creating MVC Application and Data

This encompasses setting up the application from scratch, creating database model files and scaffolding items.

## Setting up Application

I started by setting up a github repository for the application. Next, I created the MVC application, ensuring to use “User Authentication” functionality.

## Modifying Identity User

I then decided that I would start with modelling the database by creating empty classes for Items, Sales and Cart. I then started the Items class, during which I got side-tracked by trying to include multiple items under 1 user and add custom fields to the current user database. Therefore, I performed the following:

1. I created an applicationUser.cs class for my new custom fields.
2. I then modified the ApplicationContext.cs to support custom user data from the applicationUser class above.
   1. Made 1 custom field with no entry yet – to be added later.
3. Now I have my Identity, I then started to add some custom fields to the Identity.
   1. I added the following custom fields to the identity user: first name, last name, age, state, city, postcode, street, images. See [Figure 1.1](#_Appendix) For an image of the identity fields.
4. Having successfully made a migration with new custom user fields I then added a foreign key to users in Items and migrated.
   1. After I fixed Items class to reference Zephyr\_ApplicationUser I then migrated again.
   2. Then it worked and Item class is properly referencing the User.

## Database Modelling

Now that the Identity user table and Items table was able to reference it, then begun developing all the classes.

1. Next is to do the Items table.
   1. The items table includes the following fields: Id, SellerId (foreign key to Identity), Itemname, Itemdesc, Category, Price, Quantity, Itemimage, Postdate. See [Figure 1.2](#_Appendix) for image of Items class.
2. Then Sales table.
   1. Sale has the following: Id, ItemId and Quantity. See [Figure 1.3](#_Appendix) for image of Sales class.
3. Finally, Saleitems. This table is like Orderitems, it will handle item purchases with cart.
   1. Saleitems has the following: Id, CartId, Item, Quantity. See [Figure 1.4](#Figure1_4) for image of class.
   2. Time for migrations. I migrated the content in the following order: Identity, Items, Sales, Cart. See [Figure 1.5](#Figure1_4).
4. I then tested all the tables with two users, added some items and included some sales. This was achieved using the in-built database tool as I have not scaffolded my data yet, which is the next step.

## Scaffolding

1. Lastly, I performed scaffolding of my database classes to generate CRUD pages for them.

# Phase 2: Shopping Cart

This phase encompasses the creation of filtered pages, the addition of purchase button and creating the shopping cart.

1. New session helper with commands to set a key value pair in session and retrieve pairs from key.

# Appendix

Figure 1.

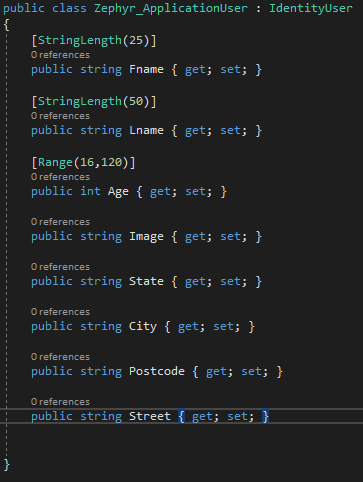


Figure 2.

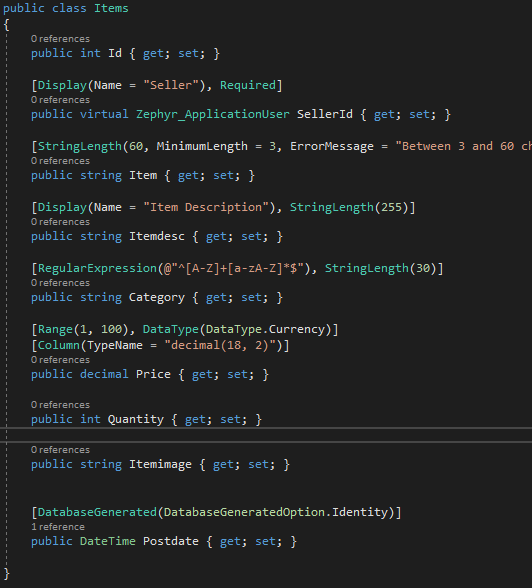


Figure 1.3

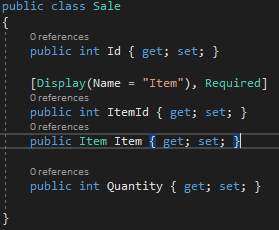


Figure 1.4

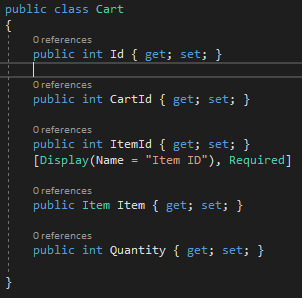


Figure 1.5

