Advanced Java OOP

Capstone 2: DELI-cious

Java CLI Application

Capstone Setup

In this capstone you will work as a team. One team member should create a public GitHub repo for this Capstone project. Make sure that the capstone repo has a meaningful name, like `DELI-dious` (do not just give it a generic name like `capstone-2`)

Ensure that all team members are granted admin access to this GitHub repo, since all team members will need to contribute to the code.

Clone the `DELI-cious` repo into the `C:/pluralsight` folder. Ensure that each team member has cloned the repo and that they have the ability to push changes.

Create a GitHub project board to manage your teams work. Use the project requirements to create user stories on your board so that you can manage your time throughout the project.

Read all project requirements individually, and then discuss high level concepts as a team before beginning to code.

Capstone Description

This project is the point of sales application for DELI-cious, a custom sandwich shop. Currently at DELI-cious our customers can fully customize their sandwich orders. Until recently we have been managing all orders in person and are currently taking all orders on paper. But our business is growing, and we need a way to automate the order process (and eventually even make it available online).

This application should take full advantage of Object Oriented Analysis and Design. In other words you should make use of OOP concepts throughout this process. Create classes and interfaces as necessary to build this solution. As you begin this project you should start by creating a class diagram of the classes, interfaces that you will need. This will be your starting point, and it may change as your application progresses throughout the week. You should keep your diagram current. You should also save the diagram to your repository.

HINT: Use the requirements below to look for nouns and verbs to help you create your class diagram.

Application Requirements

The sandwich is the core of our business. Customers can order sandwiches in 3 sizes (4", 8" and 12"). When they order, they choose the type of bread that they would like (white, wheat, rye, or wrap). The customer can then choose their toppings. Toppings are categorized as regular and premium. Meats and cheeses are premium toppings, but most other toppings are considered regular.

Customers can request for extra toppings, but premium toppings come at an additional cost. Each sandwich can also be toasted.

A customer can place an order with 1 or more sandwiches on the order. When a customer places the order, they should be prompted to customize each sandwich one at a time.

A customer should also be able to add drinks and chips to their order.

When they have completed their order, the application should display the order details, including the list of sandwiches that were ordered with all of the toppings so that the customer can verify that the order is correct. The screen should also display the total cost of the order.

When the customer completes the order, the order details should be saved to a receipts folder. Each order should have it's own receipt file, and it should be named by the date and time that the order was placed (`yyyyMMdd-hhmmss.txt` - i.e. `20230329-121523.txt`)

Sandwich Prices	4"	8"	12"
Bread - white - wheat - rye - wrap	5.50	7.00	8.50
Toppings	4"	8"	12"
Meats - steak - ham - salami - roast beef - chicken - bacon	1.00	2.00	3.00
Extra Meat	.50	1.00	1.50
Cheese - american - provolone - cheddar - swiss	.75	1.50	2.25
Extra Cheese	.30	.60	.90
Regular Toppings - lettuce - peppers - onions - tomatoes - jalepenos - cucumbers	Included	Included	included

- pickles - guacamole - mushrooms			
Sauces - mayo - mustard - ketchup - ranch - thousand islands - vinaigrette	Included	Included	Included
Sides - au jus - sauce	Included	Included	Included
Other Products	**Small**	**Medium**	**Large**
Drinks	2.00	2.50	3.00
Chips	1.50		

Your application must include several screens with the listed features in order to be considered complete:

- **Home Screen**
  - The home screen should give the user the following options. The application should continue to run until the user chooses to exit.
    - **1) New Order**
    - **0) Exit** - exit the application
- **Order Screen** - All entries should show the newest entries first
  - **1) Add Sandwich**
  - **2) Add Drink**
  - **3) Add Chips**
  - **4) Checkout**
  - **0) Cancel Order** - delete the order and go back to the home page
- **Add Sandwich** - the add sandwich screen will walk the user through several options to create the sandwich
  - **Select your bread:**
  - **Sandwich size:**
  - **Toppings:** - the user should be able to add extras of each topping
    - **Meat:**
    - **Cheese:**
    - **Other toppings:**
    - **Select sauces:**
  - **Would you like the sandwich toasted?**
- **Add Drink** - select drink size and flavor
- **Add Chips** - select chip type
- **Checkout** - display the order details and the price
  - **Confirm** - create the receipt file and go back to the home screen
  - **Cancel** - delete order and go back to the home screen

(Optional Bonus) Challenge Yourselves

If you have time and want to challenge yourself, consider the following:

Create several Signature Sandwiches that are a template for what the customer can order. The customer could then customize the existing toppings to either remove or add more toppings to the sandwich.

HINT: Signature Sandwiches could be managed by creating custom classes that inherit from the Sandwich class.

Signature Sandwiches might include:

- **BLT**
    - 8" white bread
    - Bacon
    - Cheddar
    - Lettuce
    - Tomato
    - Ranch
    - Toasted
- **Philly Cheese Steak**
    - 8" white bread
    - Steak
    - American Cheese
    - Peppers
    - Mayo
    - Toasted

Other General Project Requirements

Your project must also meet the following requirements:

Repository

- Your code must be in a public GitHub repository
- You must use Git branches appropriately
- The repository must contain an appropriate Git commit history
  - **Minimally**, you should have a commit for each meaningful piece of work completed
- It must contain an informative README file that:
  - Describes your project
  - Includes images of your application screens
  - Describes/shows one interesting piece of code from your project
- Include exports or screenshots of your diagrams

Class Demonstrations

Each team will be given 10 minutes to demonstrate their project to the class on "project demonstration day". During this time you will:

- Present your application - run through the different screens and scenarios
  - **Each team member must demonstrate part** of the application
- Describe / show one interesting piece of code that you wrote
  - Each team member should share one piece of code
- Answer questions from the audience if time permits