Kata: Sweet Shop Management System

Objective

Create a simple **Sweet Shop Management System** following TDD that allows users to perform basic operations such as adding sweets, updating sweet details, deleting sweets, searching, sorting, and viewing available sweets. Optionally, you may implement a basic front-end interface.

Requirements

1. Operations

Add Sweets:

- Users should be able to add new sweets to the shop.
    - Each sweet should have a unique identifier (e.g., ID), name, category (e.g., chocolate, candy, pastry), price, and quantity in stock.

Delete Sweets:

- Users should be able to remove sweets from the shop.

View Sweets:

- Users should be able to view a list of all sweets currently available in the shop.

2. Search & Sort Features

Search:

- Users should be able to search for sweets by name, category, or price range.

3. Inventory Management

Purchase Sweets:

- Users can purchase sweets, which decreases the quantity of them in stock.
- The system should ensure enough stock is available before allowing a purchase.
- If there is not enough stock, the system should raise an appropriate error.

Restock Sweets:

- Users can restock sweets, increasing their quantity of stock.

4. (Optional) Frontend

- You may implement a simple front-end interface (webapp) to interact with the system, extra points for making it look pretty, and show us your creativity.

Instructions

Code Only

- This is a code-only kata. Focus on writing clean, maintainable code and implementing the required features.

Test-Driven Development (TDD)

- Write tests before implementing the functionality. Follow the three laws of TDD.
- Ensure that all tests pass before considering the implementation complete.
- Aim for high test coverage and meaningful test cases.

Clean Coding Practices

- Write clean, readable, and maintainable code.
- Follow SOLID principles and other best practices in software design.
- Ensure the code is well-documented with meaningful comments and clear variable/method names.

Git Usage

- Use Git for version control.
- Create a Git repository for your project.
- Commit your changes frequently with meaningful commit messages to show your TDD journey.
- Push your code to a public remote repository (e.g., GitHub, GitLab, Bitbucket) and share the repository link.

Deliverables

- A Git repository link containing the source code for the sweet shop management system.
- A README file explaining how to set up and run the project with any other relevant details you want to include.
- A test report showing the results of the test cases.
- (Optional) Screenshots or demo of the front-end, if implemented.

- Avoid copying solutions from other repositories; we are strictly against plagiarism; any plagiarized code will be automatically rejected.

Sample Steps and Git Workflow

1. Initialize Git Repository

```
git init
```

2. Create Initial README

```
echo "# Sweet Shop Management System" > README.md
git add README.md
git commit -m "Initial commit with README"
```

3. Write Tests for Adding Sweets

- Create a new test file (e.g., `test_sweetshop.py`).
- Write a test for the add sweet feature.
- Commit your changes:

```
git add .
git commit -m "Add test for adding sweets"
```

4. Implement Adding Sweets

- Implement the add sweet functionality.
- Ensure the test passes.
- Commit your changes:

```
git add .
git commit -m "Implement add sweet feature"
```

5. Repeat Steps 3-4 for Delete, Search, Sort, Purchase, and Restock Features

6. Push to Remote Repository

```
git remote add origin <remote-repository-url>
git push -u origin master
```

Implementation Guidelines

- **Languages:** Implement the system in one of the following languages:
  - Ruby
  - Java
  - Typescript / Javascript
  - Python
- **Assessment Focus:**
  - Adherence to TDD principles.
  - Frequency and quality of Git commits.
  - Clean coding practices.
  - Proper use of language-specific features and idioms is important.

Note: Incubyte is an AI first company. We not only encourage the use of AI, but we also think it's a must in SDLC. Feel free to use your favorite AI tools. Make sure to mark the AI commits separately.

Example Sweet Data Table

ID	Name	Category	Price	Quantity
1001	Kaju Katli	Nut-Based	50	20
1002	Gajar Halwa	Vegetable-Based	30	15
1003	Gulab Jamun	Milk-Based	10	50