Builder Pattern

The Builder pattern is a creational design pattern that allows the construction of complex objects step by step. It provides a flexible solution for object creation by allowing optional parameters without needing to define multiple constructors. The Builder pattern is particularly useful when an object can be created with various configurations or optional attributes.

Code Overview In this example, we have implemented the Builder pattern to create a Menu object, which represents a custom meal menu in a restaurant chain. The Menu class has required attributes such as the main dish (mainDish) and optional attributes like the side dish (sideDish), drink (drink), and dessert (dessert).

The pattern consists of two main classes:

- Menu.java: Represents the product (the object being built).

- MenuBuilder.java: Implements the Builder pattern, allowing the creation of Menu objects with different configurations. The MenuBuilder provides methods to set each attribute of the menu and a build() method to create the Menu object.

The unit tests have been implemented using JUnit. The goal is to ensure that all possible menu configurations behave correctly. To achieve at least 85% coverage, the tests verify both complete menus and menus with missing optional attributes.