|  |  |
| --- | --- |
| **Practicum Case** |  |
| COMP6122 | COMP6122001  Framework Layer Architecture |
| **Computer Science** | **O221-COMP6122-CT01-03** |
| ***Valid on*** *Odd Semester Year 2021/2022* | **Revision 00** |

**Learning Outcome**

* LO1 – describe use of design pattern in java
* LO2 – apply design pattern in java

**Topic**

* Session 3 – Creational Design Pattern II

## Sub Topics

* Builder
* Prototype

1. **Builder**

Builder is a creational design pattern that lets you construct complex objects step by step. The pattern allows you to produce different types and representations of an object using the same construction code. The Builder pattern suggests that you extract the object construction code out of its own class and move it to separate objects called builders.

1. **Prototype**

Prototype is a creational design pattern that lets you copy existing objects without making your code dependent on their classes. The Prototype pattern delegates the cloning process to the actual objects that are being cloned. The pattern declares a common interface for all objects that support cloning. This interface lets you clone an object without coupling your code to the class of that object. Usually, such an interface contains just a single clone method.

## Soal

*Case*

**Angelic Toppo Shop**

Angelic Toppo Shop is a popular burger shop in Jakarta. Every day the Angelic Toppo shop sells a **burgers**. The burgers are selling well because the burgers are inexpensive. Behind the inexpensive burgers, the customer must buy a packet menu that has been available in the shop where each menu sells more than 1 burger, because the Angelic Toppo shop does not want to lose money on oil and roasting of meats. Because of that, the business has rapidly grown. Angel as the business owner of the shop, she asks you to create a program. Below are the following details of the program:

* **Main Menu**

At first, the program will show the menu. In this menu, the program will ask the user to choose between **0 and 4 (inclusive)**.

1. **Chicken Burger (4 pieces)**
2. **Big Chicken Burger (3 pieces)**
3. **Beef Burger (3 pieces)**
4. **Special Burger (3 pieces)**
5. **Exit**



**Figure 1. Main Menu**

* 1. **Generate Burger**

If the customer **chooses** **between 1 to 4**, then the program will:

* The program will **generate** the **burgers** based on the of his/her input.
* And **show** **the** **total price** of the generated burgers.
* To generate the burgers, the program must follow the **formula** below:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Burger Name** | **Type of Bread** | **Meat** | **Has Cucumber** | **Has Egg** | **Has Tomato** | **Has Onion** | **Has Lettuce** | **Price** |
| Chicken Burger | English Muffin | Chicken | No | No | Yes | No | Yes | 28000 |
| Big Chicken Burger | English Muffin | Two Chickens | No | Yes | Yes | No | Yes | 40000 |
| Beef Burger | Ciabatta Roll | Beef | No | No | Yes | Yes | Yes | 35000 |
| Special Burger | Ciabatta Roll | Chicken and Burger | Yes | No | Yes | Yes | No | 50000 |



**Figure 2. Generate Two Chicken Burgers**

Background pattern

Description automatically generated with low confidence

**Figure 3. Generate Three Special Burgers**

* 1. **Exit**

If the customer **choose 0,** then the program will be **terminated**.