Programmation système

JUSTIFICATION DES CHOIX : DESIGN PATTERN

MEMBRES DU GROUPE :

- DEMANOU Maéva
- LOCK Pauline
- MANFOUO Fredy
- WAMBO Harley

GROUPE 3

PROGRAMMATION SYSTEME

[Date]

a. DESIGN PATTERN MEMENTO

Le design pattern Adapter est un patron de conception de type comportement. Il est utilisé pour restaurer un état précédent d'un objet (retour arrière) sans violer le principe d'encapsulation.

Application : nous l'utiliserons pour mettre en pause notre simulateur ?

b. DESIGN PATTERN ABSTRACT FACTORY

Le design pattern Abstract Factory est un patron de conception de type création. Il est utilisé pour créer à partir d'une interface, une famille d'objets liés ou interdépendant sans toutefois spécifier de classes concrètes à utiliser.

Application : nous l'utiliserons dans notre cas pour créer les ustensiles de cuisine.

c. DESIGN PATTERN SINGLETON

Le design pattern Singleton est un patron de conception de type création. Son objectif est de restreindre l'instanciation d'une classe à un seul objet. Simplement, il permet de s'assurer qu'une classe ne possède qu'une seule instance.

Application : on l'utilise dans notre cas pour créer une seule instance du maître d'hôtel et du chef de rang car ils sont uniques dans la salle du restaurant.

d. DESIGN PATTERN OBSERVER

Le design pattern Observer est un patron de conception de type comportemental. Il est utilisé pour envoyer un signal à des modules qui jouent le rôle des observateurs. Utilisé lorsqu'il existe une relation un-à-plusieurs entre des objets. Par exemple, si un objet est modifié, ses objets dépendants doivent être notifiés automatiquement.

Application : nous l'utiliserons pour la gestion des stocks. Lorsque le stock devient insuffisant, le chef de cuisine est notifié.

e. DESIGN PATTERN BUILDER

Le design pattern Builder permet de créer un objet complexe à partir de plusieurs autres objets.

Application : pour faire les commandes des clients.

f. DESIGN PATTERN MVC

Le design pattern MVC pour Model-View-Controller ou Modèle-Vue-Contrôleur. C'est un motif d'architecture logicielle permettant de séparer l'affichage des informations, les actions de l'utilisateur et l'accès aux données.

Le modèle gère les données. Il récupère les informations de la base de données, de les organiser et de les assembler pour qu'elles puissent être traitées par le contrôleur. On y trouve les requêtes SQL.

La vue, elle gère l'affichage. Elle se contente de récupérer des variables pour savoir ce qu'elle doit afficher.

Le contrôleur c'est l'intermédiaire entre le modèle et la vue. Il gère la logique du code. Le contrôleur récupère au modèle les données, les analyser, prendre des décisions et renvoyer ce qu'il faut afficher à la vue.

Justification : on l'utilise pour la réutilisabilité, la bonne présentation, la maintenabilité du code