Nos choix de Designs Patterns :

# Observer

L’observation de classe permet par exemple à la vue d’observer le comportement du modèle pour s’adapter aux différents évènements des employée présent sur la simulation.

L’observation permet aussi à un serveur de prévenir de la présence de couvert salle sur le comptoir d’objet salle mais aussi au chef de cuisine d’être au courant des nouvelles commandes.

# Memento

Le mémento permet de conserver en mémoire l’état d’un objet pour pouvoir le réinitialiser à cet état plus tard. Le mémento sera exploité pour la gestion de table pour la réinitialiser la table à son état d’origine lorsque le chef de rang ira dresser cette table.

# Factory

Le Factory (ou fabrique) permet l’instanciation d’objet générique à partir de classe spécifique, permettant de créer des objets au nom générique sans être obligé d’appeler chaque classe.

# Singleton

L’instanciation unique de singleton sera utile pour l’instanciation des classe uniques tel que le SQLConnector, le Chef, la cuisine, le restaurant, le socketManagement, Maître d’hôtel, … Il permet d’éviter d’instancier en multiple, certaines classes qui ne doivent pas l’être, tel que les clients de connexion au réseau.

# Strategy

L’utilisation de Strategy permet la généralisation d’attribut et de méthodes communes à plusieurs classes, réduisant ainsi la taille des classes en évitant la répétition de code. Le strategy est ici exploité pour les employées ayant des fonctions communes tel que le chef de rang et le commis (cuisiner) ou encore le chef de rang, le commis et le plongeur (préparer les légumes).

# Model Vue Contrôleur

Le MVC permet une meilleure organisation de son code, permettant la bonne gestion du code, ainsi qu’une reprise plus efficace de celui-ci. La vue est responsable de la gestion de l’interface graphique, le model gère l’intégralité des classes, des connections. Enfin, le contrôleur permet de lier la vue à l’interface et de gérer les entrées utilisateurs.