Projet Java – CIR3 : Application de gestion d'un restaurant

I. Présentation du projet

Introduction

Votre société a signé un contrat avec un grand compte de la restauration grâce à son expertise et ses compétences avérées (et de manigance), la conduisant à être sélectionnée pour le développement d'une nouvelle application utilisant des technologies de modernes.

Votre commercial a constitué un premier recueil de besoin. L'affaire n'est pas complètement conclue, mais le client insiste pour commencer le développement dès maintenant pour être prêt pour le centenaire de son anniversaire.

Un premier cahier des charges vous a été transmis. On vous demande de rédiger dès à présent la partie back, tandis qu'une autre équipe sera bientôt recrutée pour s'occuper de la partie front.

Que la force soit avec vous.

Recueil du besoin

L'application doit servir à la gestion globale du restaurant. Son utilisation ne doit concerner que le management et les employés et automatiser autant que possible toutes les tâches.

- Cycle de vie d'une transaction.
    1. Un groupe de client se présente à l'entrée et un serveur se charge de les accueillir. En fonction de leurs souhaits et de leur nombre, il les conduit jusqu'à une table dont il sera désormais responsable jusqu'à la fin de la transaction.
    2. Deux cartes leur sont proposées, une des boissons et l'autre des plats. Lorsque le client le souhaite, le serveur devrait pouvoir enregistrer la commande dans le logiciel
    3. La commande est directement envoyée au bar et en cuisine.
    4. Les cuisiniers prennent connaissance de la commande et commencent à la préparer. Pendant ce temps, le barman prépare les cocktails
    5. Le serveur amène la commande de nourriture et de boisson. Toutes les commandes de la table doivent être prêtes en même temps.
    6. Lorsque les clients terminent de manger, on leur apporte l'addition
    7. Les clients règlent la commande de la table en une ou plusieurs fois et une facture est sauvegardée en cas de contrôle fiscal.

- Cycle de vie d'une journée de travail

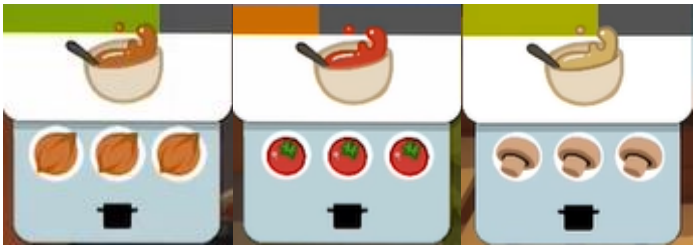
1. Le manager prévoit les employés qui travailleront avant l'ouverture de l'enseigne et reconstitue l'état des stocks.
2. Une fois le magasin ouvert, les employés effectuent des transactions jusqu'à l'horaire de fermeture.
3. Tous les employés nettoient le restaurant.
4. En fonction des ventes actuelles, le manager souhaiterait imprimer une liste de courses pour reconstituer les aliments dépensés.
5. Le manager aimerait avoir des outils de monitoring pour surveiller les performances de la journée.

- Au fur et à mesure des discussions, d'autres contraintes semblent émerger:
  - Pour que le restaurant ouvre, il faut obligatoirement avoir au moins une équipe avec 4 cuisiniers, 2 serveurs, 1 manager et un barman.
  - Aucun employé à l'exception du manager ne peut travailler 3 soirs de suite.
  - Comme son restaurant fonctionne beaucoup avec des contrats étudiants, il est vraiment nécessaire de pouvoir ajouter ou supprimer un élément. Heureusement les contrats de travail sont très simples, et on ne demande que le nom prénom et le salaire.
  - En revanche, il vous assure que la carte peut être complètement statique! Elle n'a changé qu'une seule fois en 100 ans!
  - Les clients choisissent parfois tous les mêmes plats, il est parfois question de rupture de stock. Il aimerait que les serveurs aient l'information au moment de donner la carte pour éviter les frustrations.

Carte du restaurant:

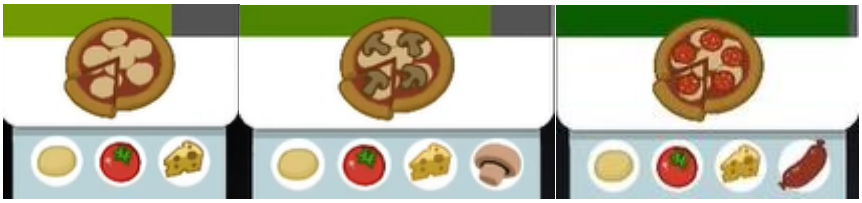
Salades: 9€

Potages: 8€



Burgers: 15€

Pizza: 12€



<u>Boissons</u>:

Limonade: 4€
Cidre doux: 5€
Bière sans alcool: 5€
Jus de fruit: 1€
Verre d'eau: Gratuit

Recommandations de l'UX Designer

L'expérience utilisateur est particulièrement importante pour la satisfaction cliente et votre designer participe à plusieurs ateliers pour confectionner un premier plan. Plusieurs écrans émergent rapidement de la discussion:

- Un écran destiné aux serveurs pour prendre les commandes de la table.
- Un écran destiné au barman pour prendre connaissance des cocktails à réaliser et indiquer lorsque la préparation est terminée.

- Un écran semblable pour les cuisiniers permettant de connaître la liste des plats à préparer et d'indiquer leur avancement.
- Un écran destiné au manager pour suivre les performances de son restaurant. et demander l'impression de la liste de course.
- Un écran pour ajouter un employé ou en supprimer un.
- Un écran pour mettre à jour les réserves d'aliments.
- Un écran pour programmer les employés qui seront présents pour cette soirée.

Impossible cependant de se mettre d'accord sur la solution pour imprimer les factures et la liste de course. Après plusieurs débats interminables, la deadline de la fonctionnalité a été reporté en attendant (le cours sur la gestion des fichiers en java) un futur atelier.

Solution des architectes

En attendant l'équipe front-end, l'architecte vous conseille de remplacer le front pour l'instant par des simples Scanner qui permettront de lancer les fonctions. Il propose de commencer le code par cette fonction par exemple:

```
public class App {
   public static void main(String[] args){
        System.out.println("Quel écran souhaitez vous afficher?");
        System.out.println("1- Ecran prise de commande");
        System.out.println("2- Ecran cuisine");
        System.out.println("3- Ecran bar");
        System.out.println("4- Ecran Monitoring");

        Scanner scanner = new Scanner(System.in);

        int choixEcran = scanner.nextInt();

        System.out.println("Vous avez choisi l'écran: " + choixEcran);

   }
}
```

II. Organisation et Livraison du projet

- Le projet doit être réalisé en trinôme.
- Livrables :
    - Un diagramme de classe qui représente la structure et les relations entre les principales classes de votre application. Ce diagramme de classe est essentiel pour comprendre l'architecture de votre projet et pour évaluer sa conception. Veuillez inclure les classes, les attributs, les méthodes et les relations clés entre

les classes. Assurez-vous que le diagramme est clair et lisible. Vous pouvez utiliser des outils de modélisation UML pour créer ce diagramme.
    - Code source et fichiers utilisés => bref, la solution doit être compilable et exécutable.
    - Documentation.

- Modalités de livraison :
    - Le premier livrable contiendra la première version du diagramme de classe, il doit être déposé dans le devoir (sur junia-learning) "Projet : diagramme de classe" avant le **mardi 07 novembre 23h.**
    - Le livrable final doit être compressé dans un répertoire Zip nommé nom1_nom2-nom3.zip, il contiendra le diagramme de classe final, le code source et la documentation de votre projet.
    - La version finale doit être déposée dans le devoir (sur junia-learning) " Projet : Gestion d'un restaurant" avant le **vendredi 08 décembre 23h00** (un rendu par trinôme).