BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2023**

Épreuve E5 - Conception et développement d'applications (option SLAM)

ANNEXE 7-1-B : Fiche descriptive de réalisation professionnelle (recto)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE	**N° réalisation : 02**
Nom, prénom : BOUTEVIN SANCÉ Lucas	N° candidat : 01948505199
Épreuve ponctuelle ☒ **Contrôle en cours de formation** ☐	**Date :** … / ….. / ……

Organisation support de la réalisation professionnelle

L'amphitryon

Intitulé de la réalisation professionnelle

Projet Application Mobile d'Amphitryon

Période de réalisation : 20/01/23 au 21/04/23 **Lieu :** Lycée Gustave Eiffel, Bordeaux
Modalité : ☐ **Seul(e)** ☒ **En équipe**

Compétences travaillées

☒ Concevoir et développer une solution applicative
☐ Assurer la maintenance corrective ou évolutive d'une solution applicative
☒ Gérer les données

Conditions de réalisation[1] (ressources fournies, résultats attendus)

Voici les ressources fournis au commencement du projet :

- Le sujet
- Le MEA à compléter

Objectif du projet :

L'amphitryon, une chaine de restaurants, souhaite disposer d'une application mobile afin de faciliter la gestion de salle, la prise de commandes et la facturation de ces commandes. Cette future application va être utilisée par le chef de salle, le chef cuisinier et les serveurs.

Ce travail ayant été fait en groupe, je me suis occupé de la partie : **Chef cuisinier**.

Le chef cuisinier doit pouvoir :

- Créer, modifier, supprimer et afficher les différents plats. Chaque plat porte un nom, est accompagné d'un descriptif.

- Proposer des plats pour un service donné. Le prix de vente d'un plat peut varier. Les plats sont classés en trois catégories (entrée, plat principal et dessert). Les plats proposés changent très fréquemment.

[1] En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO.

- Ne plus proposer un plat pour un service donné.

- Définir, modifier et afficher, pour chaque plat, les quantités disponibles pour chaque service.

- Afficher la liste des plats avec les quantités disponibles et les quantités vendues pour une journée donnée et un service donné.

Description des ressources documentaires, matérielles et logicielles utilisées[2]

Ce projet a été réalisé avec les outils suivant :

- Un ordinateur, un écran, un clavier, une souris
- Un navigateur web
- Une connexion internet
- AndroidStudio
- Visual Studio Code
- Wampserver64
- GitLab

Modalités d'accès aux productions[3] et à leur documentation[4]

Ouvrir l'explorateur de fichier, faire clique droite puis "Git Bash Here" et exécuter les commandes suivantes :

1. git clone https://gitlab.com/Majaaa/ap_android.git
2. Aller dans le dossier créer ap_android puis faire : git checkout Lucas

Ensuite lancer Wampserveur et faite :

1. Dans le dossier wamp64/www mettre le dossier API_a_utiliser.
2. Ensuite, dans PhpMyAdmin créer une base de données sous le nom "amphitryon" ; cliquer sur amphitryon, puis Importer et glisser le script qui se situe à ap_android/BDD/ScriptDeLucas.sql (Dans le code wamp64/www/API_a_utiliser/modeles/DAO/param.php mettre vos identifiant et mot de passe PhpMyAdmin)

Ensuite lancer Android Studio Code :

1. Ouvrir le dossier projet : ap_android/AP_Android_Amphitryon
2. Installer un SDK et un téléphone

Vous pouvez maintenant lancer le projet en cliquant sur le bouton "Run App" au haut à droite.

PS : Pour se connecter à ma partie rentrer l'identifiant Eren et le mot de passe 123.

[2] Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

[3] Conformément au référentiel du BTS SIO « *Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l'épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d'organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l'épreuve.* ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d'un espace de stockage et de la présentation de l'organisation du stockage.

[4] Lien vers la documentation complète, précisant et décrivant, si cela n'a été fait au verso de la fiche, la réalisation professionnelle, par exemples service fourni par la réalisation, interfaces utilisateurs, description des classes ou de la base de données.

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2023**

Épreuve E5 - Conception et développement d'applications (option SLAM)

ANNEXE 7-1-B : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

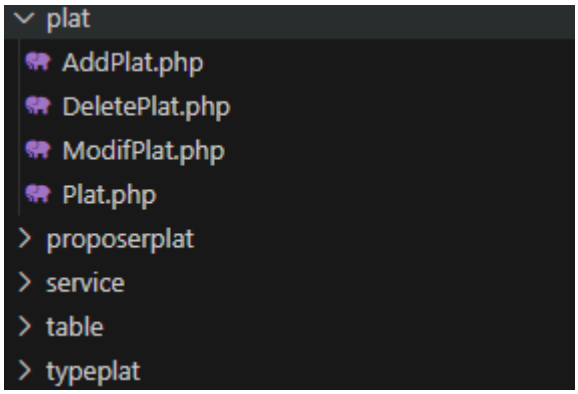
Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

Mon API, mes DAO, DTO (class singulière et pluriel), **FONCTIONNE TOUTE DE LA MÊME MANIERE** et voici leur fonctionnement :

Dans les DAO de mon API je dispose des méthodes all, add, modif et delete. Voici un exemple de la méthode allPlats de modeles/DAO/PlatDAO.php :

```php
// Permet de récupéré tous les plats
1 reference | 0 overrides
public static function allPlats(){
    try{
        $prepSql = "SELECT idPlat, libelleType, nomPlat, descriptionPlat
                    FROM plat AS P, typeplat AS T
                    WHERE P.codeType = T.codeType";

        $sql = DBConnex::getInstance()->prepare($prepSql);

        $sql->execute();

        return $sql->fetchAll(PDO::FETCH_ASSOC);
    } catch(Exception $e) {
        echo "Failed.";
    }

    return "Failed.";
}
```

Les méthodes "all" + nom de la classe, sont utilisées pour récupérer la liste de toutes les occurrences dans la BDD.
Toujours dans mon API, ces méthodes sont utilisées dans les contrôleurs, en voici un exemple :



(contrôleurs/plat/Plat.php)

```php
<?php

require_once '../../modeles/DAO/Param.php';
require_once '../../modeles/DAO/DBConnex.php';
require_once '../../modeles/DAO/PlatDAO.php';


print(json_encode(PlatDAO::allPlats()));

?>
```

Une fois les fonctions terminer dans mon API, je m'en sers dans mon code Java de Android Studio.

Dans AndroidSutdio dans un dossier nommé "lib" il y a la class "HttpRequest", elle contient une méthode qui permet d'exécuter une requête à l'API et de renvoyer son résultat. Elle à été créer pour faciliter le développement de l'application.

```java
/**
 * Méthode qui execute une requête en arrière-plan et renvoie la réponse.
 * @param endPoint (String : endPoint vers l'API.)
 * @param requestType (String : Type de la requête)
 * @param requestBody (Objet RequestBody)
 * @return String (Réponse de la requête)
 */
14 usages  lucas66166 +1
public static String executeRequest(String endPoint, String requestType, RequestBody requestBody){
```

Comme dans l'API, les DAO de AndroidStudio dispose des méthodes all, add, modif et delete. Voici un exemple de la méthode allPlats de modeles/DAO/PlatDAO.java :

```java
/**
 * Méthode qui enregistre tous les plats
 */
1 usage  lucas66166 +1
public static void allPlats() throws JsonProcessingException {

    // Appel de la méthode executent la requête.
    String responseBody = HttpRequest.executeRequest( endPoint: "plat/Plat.php",  requestType: "GET",  requestBody: null);

    // Si le couple login/mdp est correct.
    if (responseBody.compareTo("false") != 0) {
        // Utilisation de Jackson pour créer un objet Utilisateur grâce au JSON de la réponse de la requête (JSON -> Objet).
        ObjectMapper objectMapper = new ObjectMapper();
        // Convertir la chaîne JSON en tableau d'objets Plat
        lucas66166
        TypeReference<ArrayList<Plat>> typeRef = new TypeReference<ArrayList<Plat>>() {};

        // Créer des objets plats et les met dans l'ArrayList
        ArrayList<Plat> lesPlats = objectMapper.readValue(responseBody, typeRef);

        // Ajoute la liste des plats dans la class pluriel.
        Plats.setLesPlats(lesPlats);

    }
}
```

Ces méthodes all permettent de faire une requête à l'API à leur méthode all et de mettre le résultat dans la classe pluriel associé.

Les classes plurielles contiennent deux méthodes importantes à leur fonctionnement :

```
4 usages   lucas66166
public static ArrayList<Plat> getLesPlats() {
    if(lesPlats == null){
        initializePlats();
    }
    return lesPlats;
}
```

Cette méthode permet de renvoyer tous les plats dans une liste d'objet Plat.

```
/**
 * Méthode Méthode qui permet d'aller chercher les plats de la BDD et les mettre dans l'attribut lesPlats
 */
4 usages   lucas66166
public static void initializePlats(){
        // Fait un try/catch pour éviter les erreurs
        try {
            // Va cherche tous les plats
            PlatDAO.allPlats();
        } catch (JsonProcessingException e) {
            // Affiche les messages d'erreurs
            System.out.println("Erreur de initializePLats ligne 70 de Plats.java");
            e.printStackTrace();
        }
}
```

Cette méthode est utilisée pour aller chercher tous les plats dans la BDD et les mettre dans la classe plurielle. Ces méthodes sont aussi dans les autres classes plurielles.

Pour récupérer la liste des Plats dans les contrôleurs on aura juste à utiliser la méthode getLesPlats. Et lorsqu'on utilise les méthodes add, modif ou delete, la méthode initialize de sa classe plurielle est appelé pour mettre à jour la liste d'objet :

```
1 usage   lucas66166
public static void addPlat(String libelleTypePlat, String nomPlat, String descriptionPlat){

    // Prépare les argument de la requête
    RequestBody body = new FormBody.Builder()
            .add( name: "idTypePLat", TypesPlat.getIdTypePlatByLibelle(libelleTypePlat))
            .add( name: "nomPlat", nomPlat)
            .add( name: "descriptionPlat", descriptionPlat)
            .build();

    // Appel de la méthode executent la requête.
    HttpRequest.executeRequest( endPoint: "plat/AddPlat.php",  requestType: "POST", body);

    // Met à jour les plat de la classe pluriel
    Plats.initializePlats();

}
```

Ce travail ayant été fait en groupe, je me suis occupé de la partie : **Chef cuisinier**.

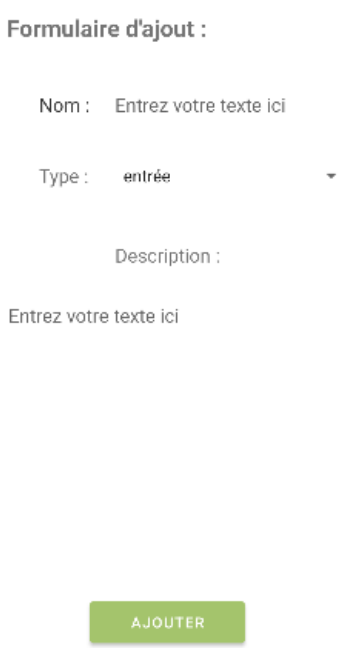
Le chef cuisinier peut donc :

- Créer, modifier, supprimer et afficher les différents plats. Chaque plat porte un nom, est accompagné d'un descriptif.

Lorsqu'on se connecte en tant que chef cuisinier on voit apparaitre la liste des plats :

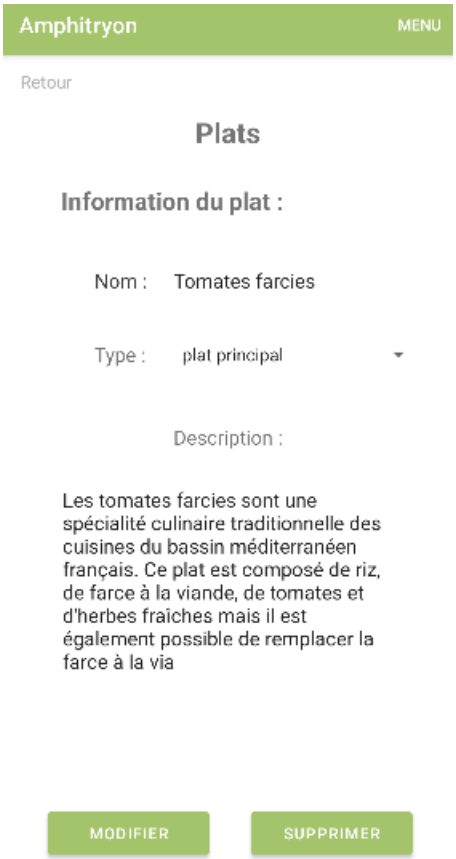
Tous les plats sont affichés et triées par type (entrée, plat, dessert).

On peut ajouter un plat en cliquant sur nouveau plat :



Il faut rentrer les informations nécessaires et cliquer sur ajouter. Vous serrez renvoyer à la liste des plats avec une notification d'ajout et votre plat apparaitra dans la catégorie indiquée.

Pour modifier ou supprimer un plat, lors de la liste des plats, il faut cliquer sur un plat :



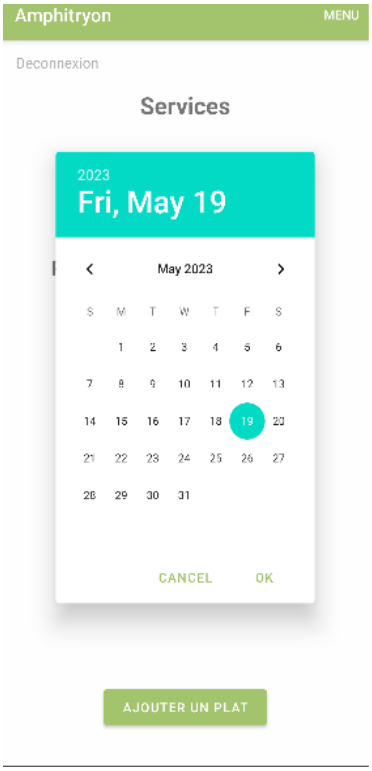
Vous pouvez alors modifier les valeurs souhaiter et cliquer sur modifier les modifiera. Si vous cliquez sur supprimer, ça supprimera le plat. Chacune des actions vous renverra sur la liste des plats avec une notification de modification ou de suppression.

- Proposer des plats pour un service donné. Le prix de vente d'un plat peut varier. Les plats sont classés en trois catégories (entrée, plat principal et dessert). Les plats proposés changent très fréquemment. Définir, modifier et afficher, pour chaque plat, les quantités disponibles pour chaque service. Afficher la liste des plats avec les quantités disponibles et les quantités vendues pour une journée donnée et un service donné. Ne plus proposer un plat pour un service donné.

L'application dispose d'un menu en haut à droite, cliquer dessus vous fera apparaitre les deux onglet "Plats" et "Services" :

Pour le moment nous étions dans "Plats", allons dans "Services" :

On peut voir à la date et au service choisie, les plats proposées. La date du jour est choisie automatiquement en arrivant sur la page. Il y a deux service "Midi" et "Soir", pour changer la date on clique sur la date puis, un calendrier apparait où l'on peut choisir la date :



On peut ajouter un plat pour se service en cliquant sur "Ajouter un plat" :

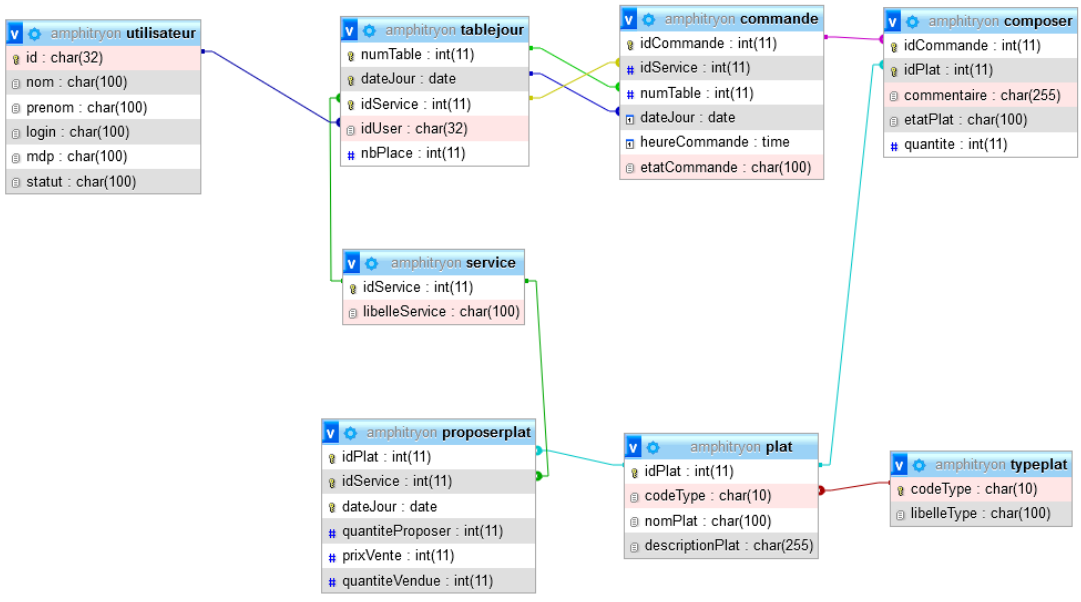
On peut choisir le prix de vente, la quantité proposée et la quantité vendue pour se service. Une fois appuyer sur Valider, le plat apparaitra dans la liste des plats pour ce service avec une notification d’ajout.

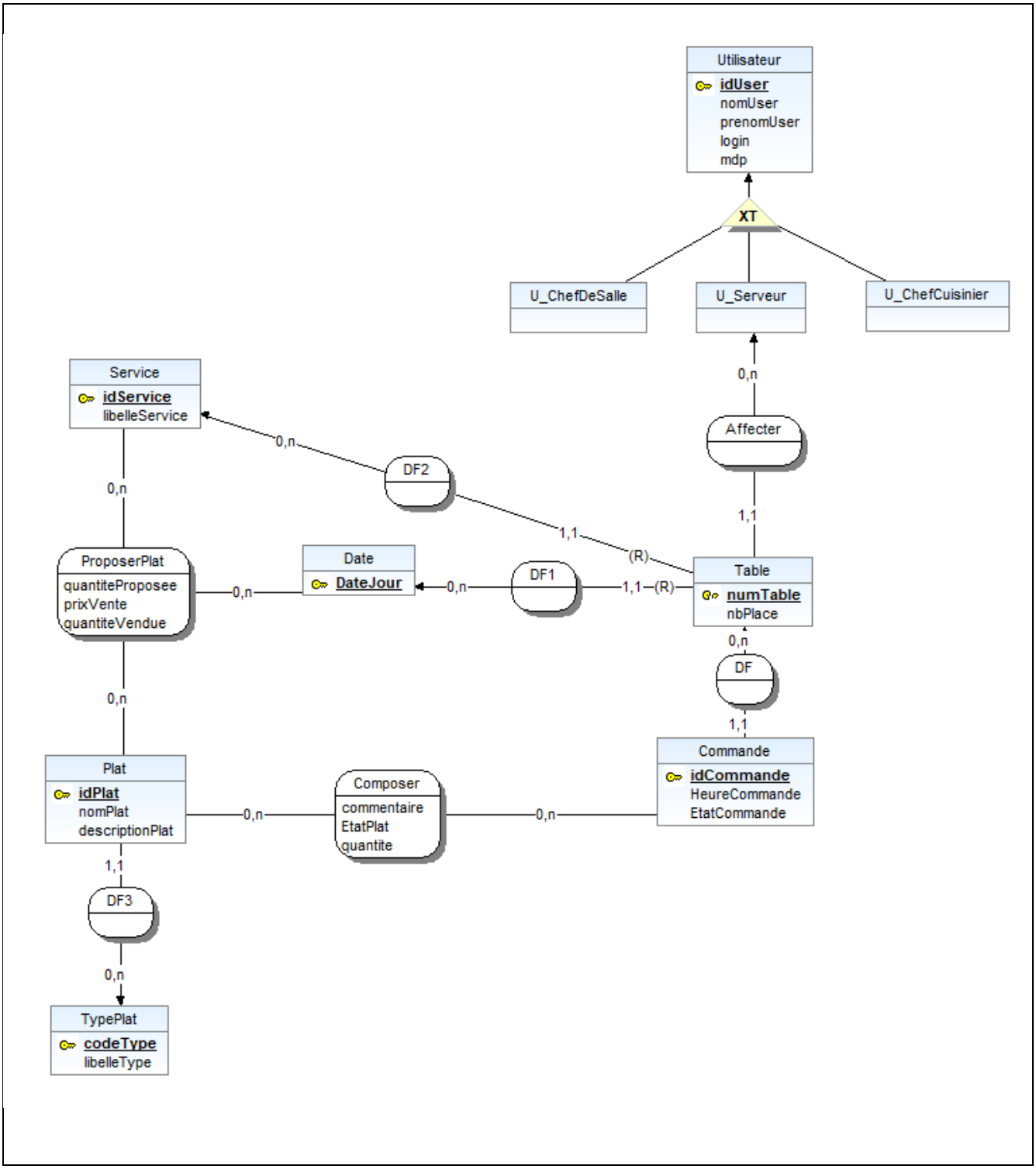
Une fois le plat ajouter au service lorsqu'on clique dessus, on peut modifier les valeurs du plat (prix, quantité proposée, quantité vendue) pour ce service ou le supprimer du service :

Vous pouvez alors modifier les valeurs souhaiter et cliquer sur modifier les modifiera. Si vous cliquez sur supprimer, ça supprimera le plat du service. Chacune des actions vous renverra sur la liste des plats du service avec une notification de modification ou de suppression.

Annexe :

Schémas de la BDD :




Utilisateur
idUser
nomUser
prenomUser
login
mdp
XT
U_ChefDeSalle
U_Serveur
U_ChefCuisinier
Service
idService
libelleService
0,n
Affecter
DF2
1,1
(R)
ProposerPlat
quantiteProposee
prixVente
quantiteVendue
Date
DateJour
DF1
Table
numTable
nbPlace
DF
Commande
idCommande
HeureCommande
EtatCommande
Plat
idPlat
nomPlat
descriptionPlat
Composer
commentaire
EtatPlat
quantite
DF3
TypePlat
codeType
libelleType