OC Pizza

Solution technique

Dossier de spécifications techniques

Arnaud KIEFER
Parcours DA iOS – Projet 7
OpenClassRooms

1. Votre entreprise : Contexte et besoins

1.1. Contexte :

« OC Pizza » est un jeune groupe de pizzeria en plein essor. Créé par Franck et Lola, le groupe est spécialisé dans les pizzas livrées ou à emporter. Il compte déjà 5 points de vente et prévoit d'en ouvrir au moins 3 de plus d'ici 6 mois.

Le système informatique actuel ne correspond plus aux besoins du groupe car il ne permet pas une gestion centralisée de toutes les pizzerias.

De plus, il est très difficile pour les responsables de suivre ce qui se passe dans les points de ventes. Enfin, les livreurs ne peuvent pas indiquer « en live » que la livraison est effectuée.

Pour tout cela, ils nous demandent de leur proposer un nouveau système informatique qui répondra à leurs attentes et qui puisse être déployé dans l'ensemble des pizzerias du groupe.

1.2. Besoins attendus par le client :

- Être plus efficace dans la gestion des commandes, de leur réception à leur livraison en passant par leur préparation ;
- Suivre en temps réel les commandes passées, en préparation et en livraison ;
- Suivre en temps réel le stock d'ingrédients restants pour savoir quelles pizzas peuvent encore être réalisées ;
- Proposer un aide-mémoire aux pizzaiolos indiquant la recette de chaque pizza
- Proposer un site Internet pour que les clients puissent :
    - Passer leurs commandes, en plus de la prise de commande par téléphone ou sur place ;
    - Payer en ligne leur commande s'ils le souhaitent – sinon, ils paieront directement à la livraison ;
    - Modifier ou annuler leur commande tant que celle-ci n'a pas été préparée.

1.3. La conception de la solution :

Partant de ces besoins exprimés par le client, nous allons dérouler la conception de notre solution. Mais avant, pour proposer une solution la plus adaptée possible, nous avons analysé les besoins complémentaires qui étaient requis.

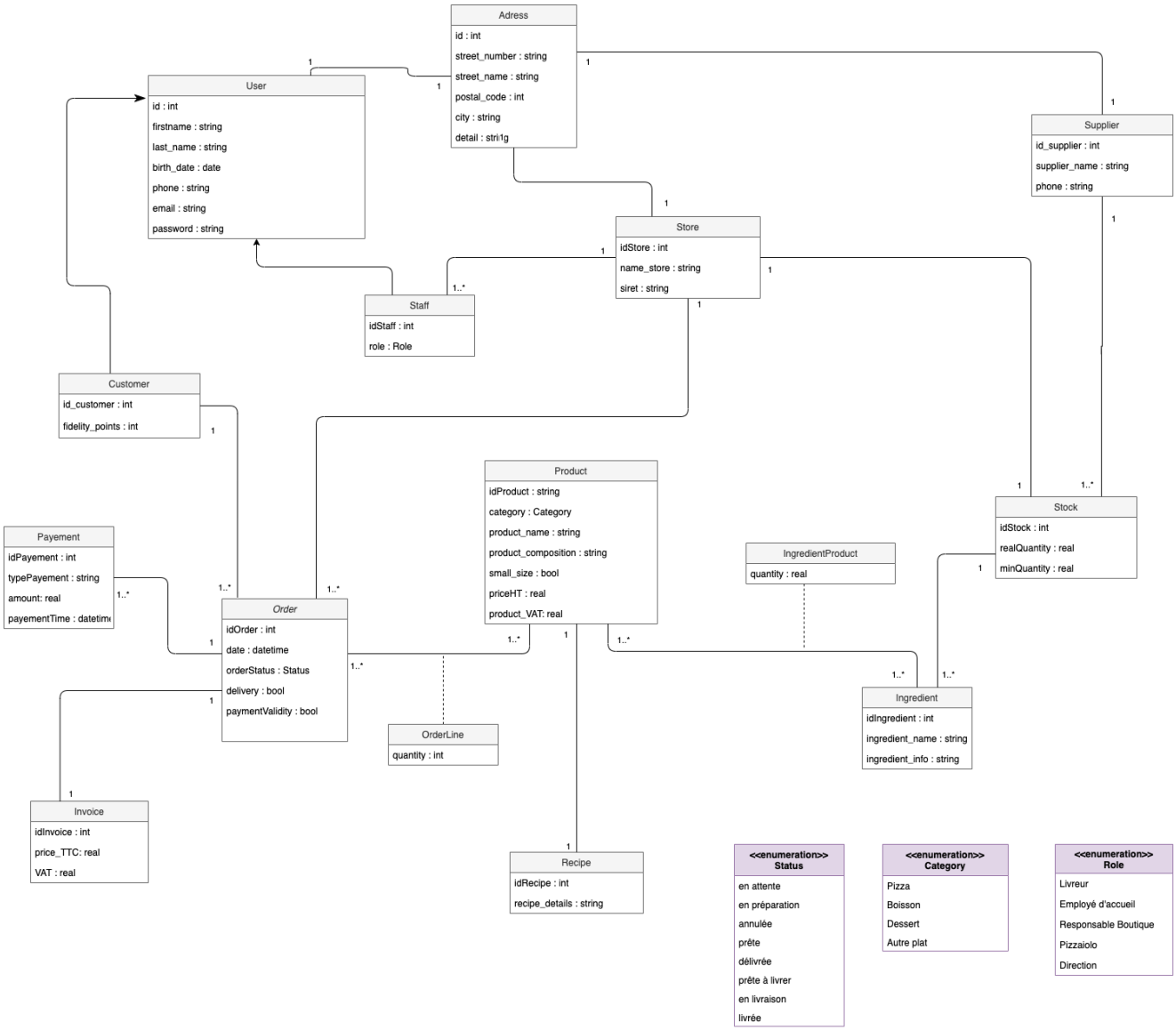
Nous avons donc réalisé une analyse des personnes qui utiliseront le système, de leurs attentes et de fonctionnalités souhaitées par ces acteurs.

La conception est guidée par les besoins du client

2. Le Domaine fonctionnel

Pour décrire le domaine fonctionnel, nous allons tout d'abord voir l'ensemble des classes dont nous aurons besoin pour construire notre modèle, et comment elles sont liées entre elles. Nous allons représenter tout cela avec un diagramme de classes. Ensuite nous définirons ces classes dans un dictionnaire de données.

2.1. Le diagramme de classe



2.2.Le dictionnaire de données

Le dictionnaire de données suivant détaille les classes et leurs attributs qui seront utilisées dans notre solution.

Classe User		
Nom des attributs	*Type des attributs*	*Détails*
idUser	Int	Numéro d'identifiant de l'utilisateur
last_name	String	Nom de l'utilisateur
firstname	String	Prénom de l'utilisateur
birth_date	Date	Date de naissance de l'utilisateur
phone	String	Numéro de téléphone portable
adress	Adress	Adresse postale de l'utilisateur
email	String	Email de l'utilisateur
password	String	Mot de passe de l'utilisateur

User :

C'est la classe qui définit un utilisateur avec toutes les caractéristiques communes à un profil client et un profil employé.

Classe Customer		
Nom des attributs	*Type des attributs*	*Détails*
id-customer	Int	Numéro de client
fidelity_points	Int	Point fidélité acquis

Customer :
C'est la classe qui définit un client.

Classe Adress		
Nom des attributs	*Type des attributs*	*Détails*
idAdress	Int	Numéro d'identification de l'adresse
street_number	String	Numéro de l'adresse
street_name	String	Nom de la voie
postal_code	Int	Code postal de la commune
city	String	Commune d'habitation
detail	String	Précision sur l'adresse

Adress :

C'est la classe qui définit une adresse qui sera utilisée pour uniformiser les adresses du programme.

Classe Staff		
Nom des attributs	*Type des attributs*	*Détails*
idStaff	Int	Numéro d'identification du personnel
role	Role	Poste du personnel

Staff :
C'est la classe qui définit un profil du personnel de la boutique.

Classe Store		
Nom des attributs	*Type des attributs*	*Détails*
idStore	Int	Numéro d'identification du point de vente
name_store	String	Nom du point de vente
siret	String	Siret de la boutique

Store :
C'est la classe qui définit un point de vente boutique. Elle permettra de créer plusieurs point de vente suivant le même modèle.

Classe Order		
Nom des attributs	*Type des attributs*	*Détails*
idOrder	Int	Numéro de commande
date	Date	Date d'enregistrement de la commande
delivery	Boolean	Délivré ou pas ?
orderStatus	Status	Statut de la commande (en attente, en livraison, en préparation, prête…)
etatCommande	String	Numéro de téléphone portable
payementValidity	Boolean	Paiement validé ou pas ?

Order :

C'est la classe qui définit une commande.

Classe Invoice		
Nom des attributs	*Type des attributs*	*Détails*
idInvoice	Int	Numéro de facture
price_TTC	Real	Montant totale de la facture
VAT	Real	TVA appliquée

Invoice :
C'est la classe qui définit la facture qui correspondra à la commande concernée.

Classe OrderLine		
Nom des attributs	*Type des attributs*	*Détails*
quantity	Int	Quantité de produit commandé

OrderLine :
C'est la classe qui fera la liaison entre les classes commande et produit.

Classe Payement		
Nom des attributs	*Type des attributs*	*Détails*
idPayement	Int	Identifiant du paiement
typePayement	String	Type de paiement (CB, liquide, …)
amount	Real	Montant de la transaction
payementTime	DateTime	Date et heure de validation du paiement

Payement :

C'est la classe qui gèrera les paiements associés à chaque commande.

Classe Product		
Nom des attributs	*Type des attributs*	*Détails*
idProduct	String	Nom du produit
category	Category	Catégorie du produit (pizza, boisson, dessert…)
product_name	String	Libellé du produit
product_composition	String	Composition du produit
small_size	Boolean	Petite taille ou pas ?
prixUnitaireHT	Real	Prix unitaire du produit Hors taxes
product_VAT	Real	TVA appliquée sur le produit

Product :

C'est la classe qui définit chaque produit à vendre. Il peut être une pizza, une boisson, un dessert ou un autre plat proposé par la boutique.

Classe Stock		
Nom des attributs	*Type des attributs*	*Détails*
idStock	Int	Identifiant du stock
realQuantity	Real	Quantité restante
minQuantity	Real	Quantité minimal avant lancement d'une alerte

Stock :

C'est la classe qui définit le stock de chaque ingrédient.

Classe Ingredient		
Nom des attributs	*Type des attributs*	*Détails*
idIngredient	Int	Identifiant de l'ingrédient
ingredient_name	String	Nom de l'ingrédient
ingredient_info	String	Information sur l'ingrédient

Ingredient :

C'est la classe qui définit les ingrédients qui composent les produits.

Classe IngredientProduct		
Nom des attributs	*Type des attributs*	*Détails*
quantity	Int	Quantité d'ingrédient dans le produit

IngredientProduct :
C'est la classe qui fait la liaison entre les produits et les ingrédients.

Classe Supplier		
Nom des attributs	*Type des attributs*	*Détails*
idSupplier	Int	Identifiant du fournisseur
supplier_name	String	Nom du fournisseur
phone	String	Numéro de téléphone du fournisseur

Supplier :

C'est la classe qui définit les fournisseurs de la boutique.

Classe Recipe		
Nom des attributs	*Type des attributs*	*Détails*
idRecipe	Int	Identifiant de la recette
recipe_details	String	Description de la recette en détail

Recipe :

C'est la classe qui contient la recette du produit.

Enum Role	
Nom des attributs	*Détails*
livreur	La personne est un livreur
employé d'accueil	La personne est un employé d'accueil
responsable boutique	La personne est un responsable de la boutique
pizzaiolo	La personne est un pizzaiolo
direction	La personne fait partie de la direction du groupe

Role :

C'est l'énumération qui définit le poste occupé par le membre de la boutique.

Enum Status	
Nom des attributs	*Détails*
en attente	La commande est passé et la pizza est en attente de préparation
en préparation	Le pizzaiolo est en train de faire la pizza
annulée	La commande a été annulée
prête	La commande est prête à être donnée au client (retrait boutique)
délivrée	Le client est venu chercher sa commande
prête à livrée	La commande est en attente du livreur qui va livrer
en livraison	Le livreur est en route avec la commande
livrée	La commande a été livrée

Status :

C'est l'énumération qui définit le statut dans lequel se trouve la commande. Il va changer en fonction l'avancement de la commande.

Enum Category	
Nom des attributs	*Détails*
Boisson	Le produit fait partie de la catégorie Boisson
Dessert	Le produit fait partie de la catégorie Dessert
Pizza	Le produit fait partie de la catégorie Pizza
Autre plat	Le produit fait partie de la catégorie Autre plat

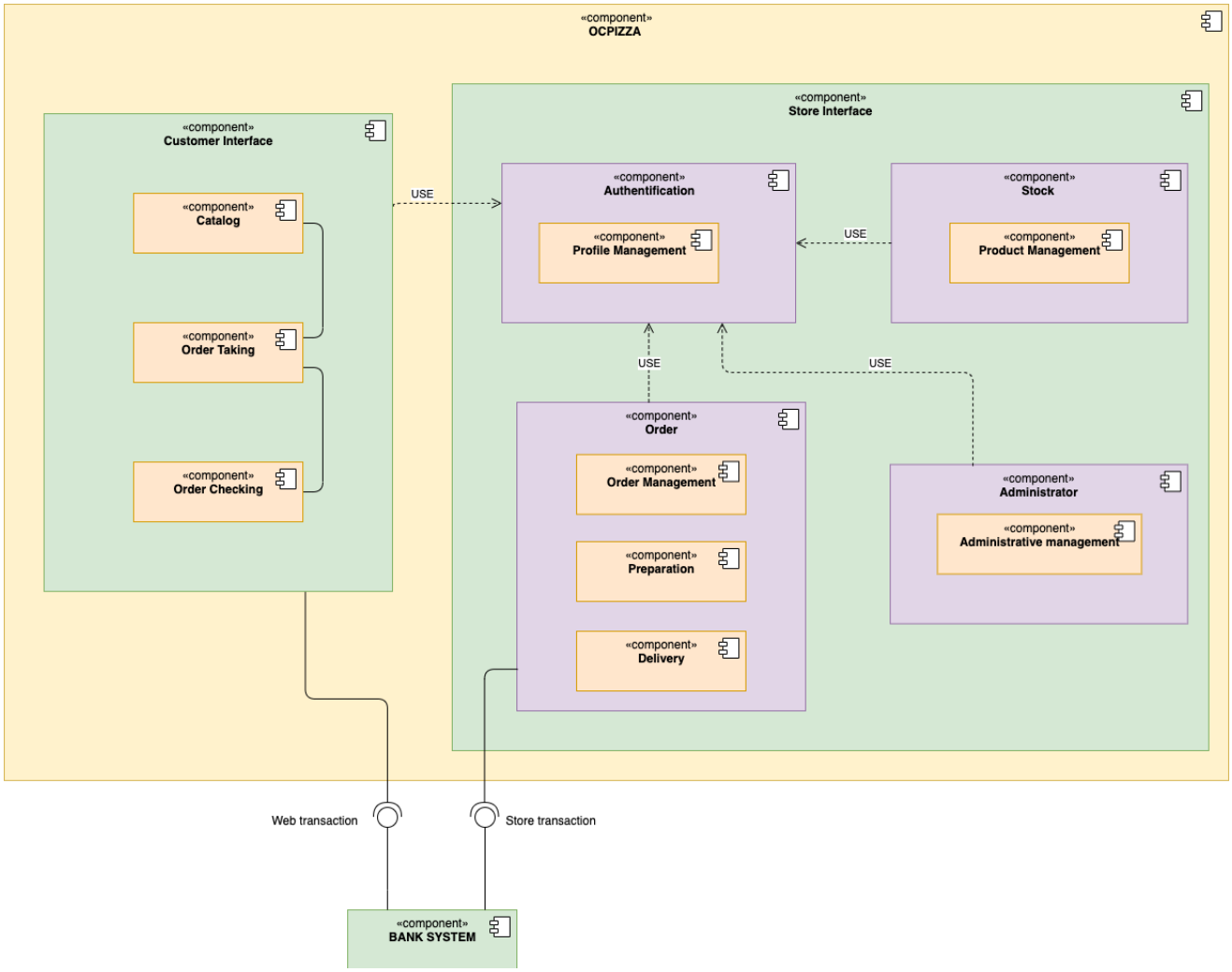
Category :

C'est l'énumération qui définit la catégorie d'un produit.

3. Les Composants du modèle

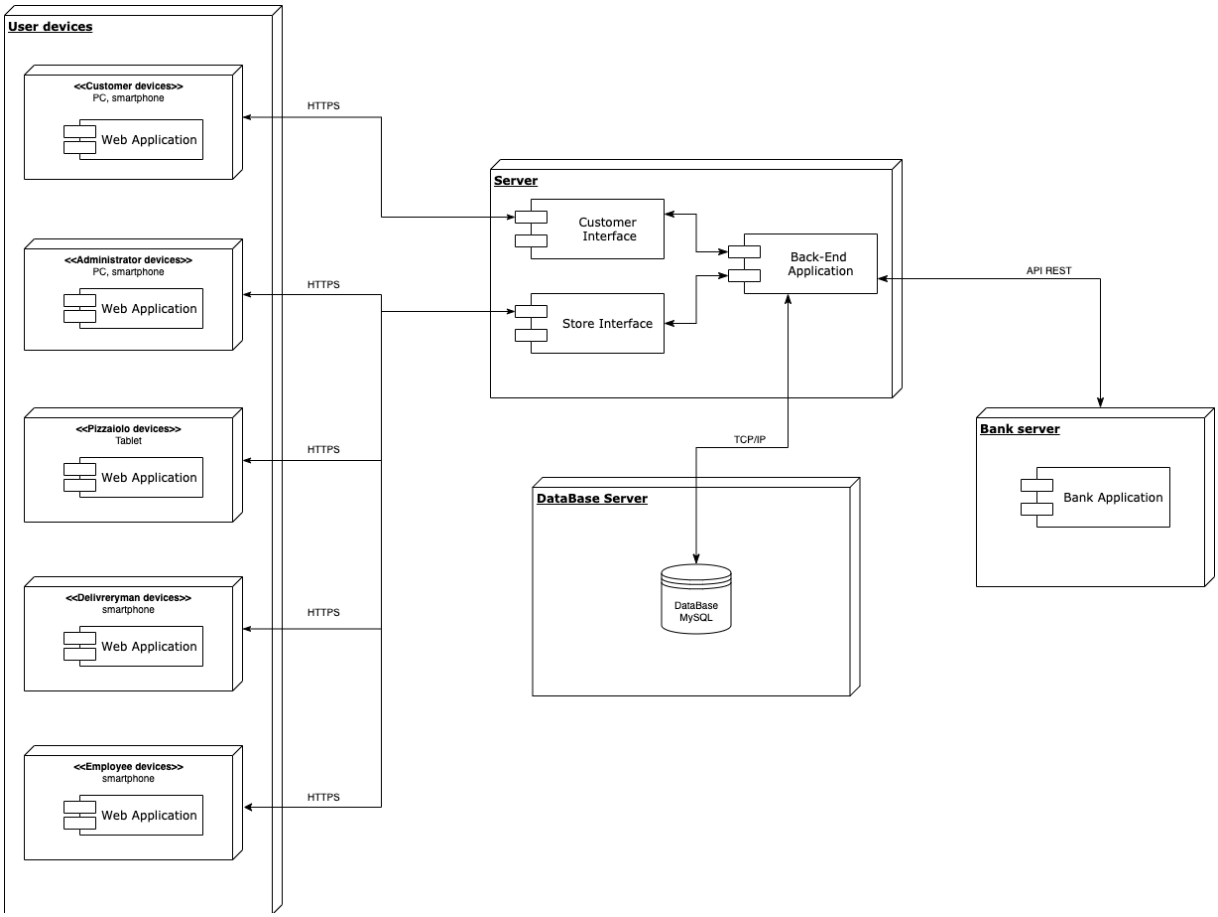
3.1. Les Composants logiciels

Le diagramme de composants ci-dessous nous montre les différents composants qui composeront le système du point de vue logiciel. Nous y retrouvons les interfaces client et magasin, eux-mêmes composée de différentes parties.
Nous voyons également que le système bancaire (pour le paiement) est également présent comme composant externe.



3.2.Les Composants physiques

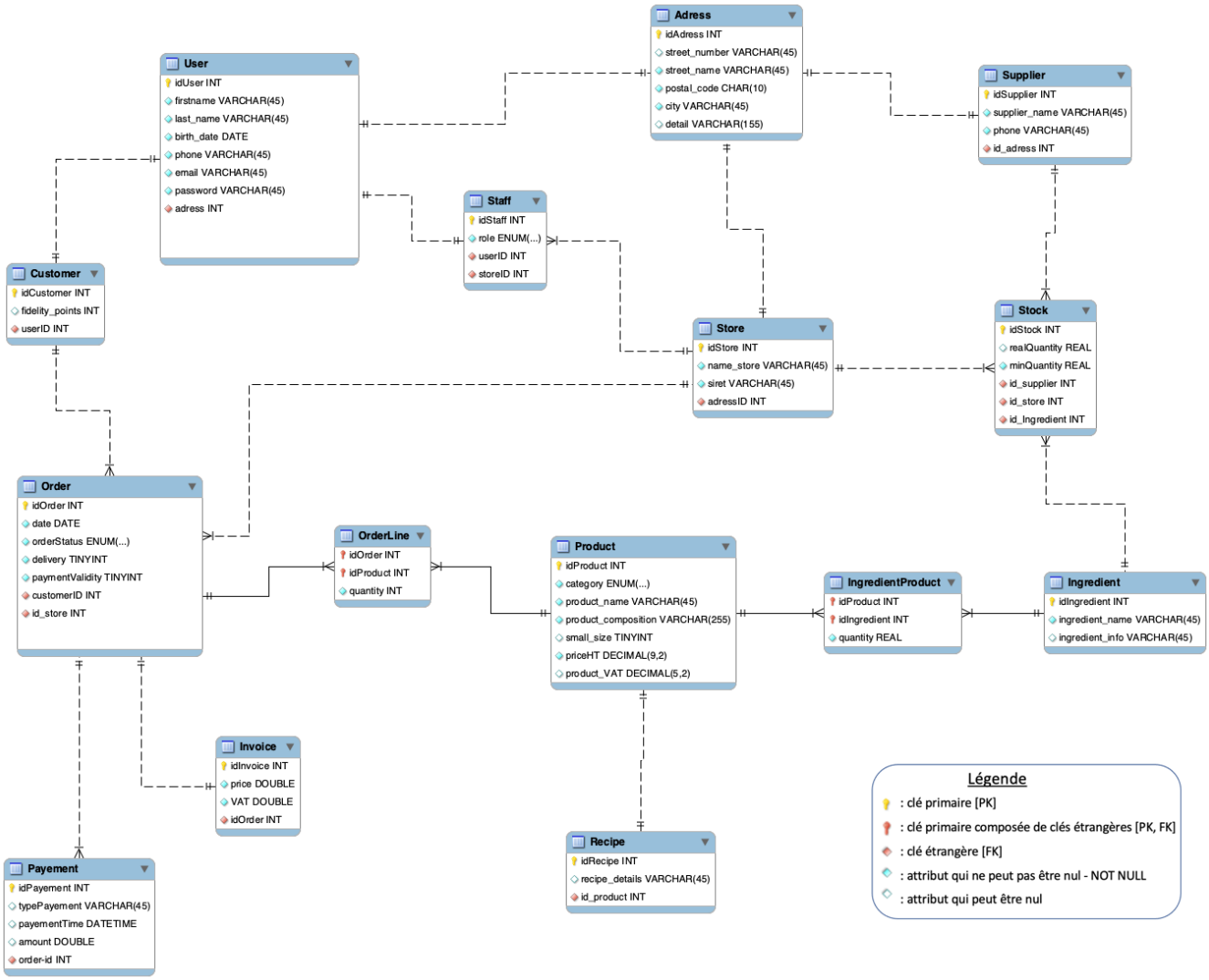
Le diagramme de déploiement se concentre sur les éléments physiques qui composeront le système. Nous y retrouvons les différents appareils des utilisateurs (PC, smartphone, tablette…) sur lesquels sera installée l'application web. Ces différents appareils communiqueront avec le serveur qui contiendra les deux interfaces et le cœur de l'application. C'est elle qui communiquera avec les serveurs de la banque pour les paiement. Elle communiquera également avec la base de données qui stockera les informations dans le serveur de données.



4. La Base de données

4.1. Le Modèle physique de données

Partant du diagramme de classe, nous pouvons décrire le modèle physique de données de notre système.
Il permet de visualiser les tables de données nécessaires, leurs liens et les clés qui permettront de trouver la donnée qu'il nous faut.



Avec ce modèle nous pouvons construire notre base de données. Pour cela, nous allons écrire un script SQL de création de la base. Puis nous alimenterons notre base pour pouvoir la tester.

4.2.Le script SQL de création de la base

Avec le modèle Physique de données, nous pouvons écrire le script suivant pour créer notre base de données.

```
CREATE SCHEMA IF NOT EXISTS `ocpizza` ;
USE `ocpizza` ;

-- -----------------------------------------------------
-- Table `ocpizza`.`Adress`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Adress` (
  `idAdress` INT NOT NULL AUTO_INCREMENT,
  `street_number` VARCHAR(45) NULL DEFAULT NULL,
  `street_name` VARCHAR(45) NOT NULL,
  `postal_code` CHAR(10) NOT NULL,
  `city` VARCHAR(45) NOT NULL,
 `detail` VARCHAR(155) NULL DEFAULT NULL,
  PRIMARY KEY (`idAdress`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ocpizza`.`User`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`User` (
  `idUser` INT NOT NULL AUTO_INCREMENT,
  `firstname` VARCHAR(45) NOT NULL,
  `last_name` VARCHAR(45) NOT NULL,
  `birth_date` DATE NOT NULL,
  `phone` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `adress` INT NOT NULL,
  PRIMARY KEY (`idUser`),
  INDEX `fk_Utilisateur_Adresse1_idx` (`adress` ASC) VISIBLE,
  CONSTRAINT `fk_Utilisateur_Adresse1`
    FOREIGN KEY (`adress`)
    REFERENCES `ocpizza`.`Adress` (`idAdress`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ocpizza`.`Customer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Customer` (
  `idCustomer` INT NOT NULL AUTO_INCREMENT,
  `fidelity_points` INT NULL DEFAULT NULL,
  `userID` INT NOT NULL,
  PRIMARY KEY (`idCustomer`),
  INDEX `fk_Client_Utilisateur1_idx` (`userID` ASC) VISIBLE,
  CONSTRAINT `fk_Client_Utilisateur1`
    FOREIGN KEY (`userID`)
    REFERENCES `ocpizza`.`User` (`idUser`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ocpizza`.`store`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Store` (
  `idStore` INT NOT NULL AUTO_INCREMENT,
  `name_store` VARCHAR(45) NOT NULL,
  `siret` VARCHAR(45) NOT NULL,
  `adressID` INT NOT NULL,
  PRIMARY KEY (`idStore`),
  INDEX `fk_pointDeVente_Adresse1_idx` (`adressID` ASC)
VISIBLE,
  CONSTRAINT `fk_pointDeVente_Adresse1`
    FOREIGN KEY (`adressID`)
    REFERENCES `ocpizza`.`Adress` (`idAdress`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `ocpizza`.`staff`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Staff` (
  `idStaff` INT NOT NULL AUTO_INCREMENT,
  `role` ENUM('pizzaiolo', 'livreur', 'responsable') NOT NULL,
  `userID` INT NOT NULL,
  `storeID` INT NOT NULL,
  PRIMARY KEY (`idStaff`),
  INDEX `fk_Personnel_Utilisateur1_idx` (`userID` ASC) VISIBLE,
  INDEX `fk_Personnel_pointDeVente1_idx` (`storeID` ASC) VISIBLE,
  CONSTRAINT `fk_Personnel_Utilisateur1`
    FOREIGN KEY (`userID`)
    REFERENCES `ocpizza`.`User` (`idUser`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Personnel_pointDeVente1`
    FOREIGN KEY (`storeID`)
    REFERENCES `ocpizza`.`Store` (`idStore`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ocpizza`.`Order`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Order` (
  `idOrder` INT NOT NULL AUTO_INCREMENT,
  `date` DATE NOT NULL,
  `orderStatus` ENUM('en attente', 'en préparation', 'annulée', 'prête',
'délivrée', 'prête à livrer', 'en livraison', 'livrée') NOT NULL,
  `delivery` TINYINT NOT NULL DEFAULT 0,
  `paymentValidity` TINYINT NOT NULL DEFAULT 0,
  `customerID` INT NOT NULL,
  `id_store` INT NOT NULL,
  PRIMARY KEY (`idOrder`),
  INDEX `fk_Order_Customer1_idx` (`customerID` ASC) VISIBLE,
  INDEX `fk_Order_store1_idx` (`id_store` ASC) VISIBLE,
  CONSTRAINT `fk_Order_Customer1`
    FOREIGN KEY (`customerID`)
    REFERENCES `ocpizza`.`Customer` (`idCustomer`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Order_store1`
    FOREIGN KEY (`id_store`)
    REFERENCES `ocpizza`.`Store` (`idStore`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ocpizza`.`Payement`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ocpizza`.`Payement` (
  `idPayement` INT NOT NULL AUTO_INCREMENT,
  `typePayement` VARCHAR(45) NULL DEFAULT NULL,
  `payementTime` DATETIME NULL DEFAULT NULL,
  `amount` DOUBLE NULL DEFAULT NULL,
  `order-id` INT NOT NULL,
  PRIMARY KEY (`idPayement`),
  INDEX `fk_Payement_Commande1_idx` (`order-id` ASC) VISIBLE,
  CONSTRAINT `fk_Payement_Commande1`
    FOREIGN KEY (`order-id`)
    REFERENCES `ocpizza`.`Order` (`idOrder`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

… (suite dans le fichier createDB_ocpizza.sql)

4.3.Insertion de données dans la base – Jeu de données de démo

Suite à sa création, voyons comment insérer des données dans notre base avec le script suivant :

```
-- ---------------
-- insertion données Adress
-- ---------------
INSERT INTO `ocpizza`.`Adress` (`idAdress`, `street_number`, `street_name`, `postal_code`, `city`, `detail`) VALUES ('1', '1', 'RUE JEAN LUC LAGARCE', '25000', 'Besançon', '1er étage');
INSERT INTO `ocpizza`.`Adress` (`idAdress`, `street_number`, `street_name`, `postal_code`, `city`, `detail`) VALUES ('2', '34', 'RUE CHARLES SEILER', '21000', 'Dijon', 'RDC');
INSERT INTO `ocpizza`.`Adress` (`idAdress`, `street_number`, `street_name`, `postal_code`, `city`, `detail`) VALUES ('3', '45', 'ALLEE ADRIEN NICKLES', '21000', 'Dijon', '');
INSERT INTO `ocpizza`.`Adress` (`idAdress`, `street_number`, `street_name`, `postal_code`, `city`, `detail`) VALUES ('4', '6', 'RUE SCHLUMBERGER', '25000', 'Besançon', '2ème étage');
INSERT INTO `ocpizza`.`Adress` (`idAdress`, `street_number`, `street_name`, `postal_code`, `city`, `detail`) VALUES ('5', '67', 'GRAND CHAMP DU CERISIER', '25300', 'Pontarlier', '');

-- ---------------
-- insertion données Ingredient
-- ---------------
INSERT INTO ocpizza.Ingredient
   (idIngredient, ingredient_name, ingredient_info)
VALUES
   (1, 'Pate à pizza', 'pate fine'),
   (2, 'crème fraiche', 'crème fraiche épaisse'),
   (3, 'Mozzarella', 'Mozzarella rapé'),
   (4, 'Sauce tomate', 'sauce tomate fraiche'),
   (5, 'Champignon', 'champignon de paris'),
   (6, 'jambon', 'Jambon en tranche'),
   (7, 'huile d''olive', 'huile d''olive extra vierge'),
   (8, 'origan', null),

-- ---------------
-- insertion données Product
-- ---------------
INSERT INTO ocpizza.Product
   (idProduct, category, product_name, small_size, product_composition, priceHT, product_VAT)
VALUES
   (1, 'pizza', 'Calzone', '1', 'Base Sauce tomate, oeuf, mozzarella, champignons frais, jambon, huile d olive, origan', 9.00, 10.0),
   (2, 'pizza', 'Norvégienne', '1', 'Base crème, mozzarella, saumon, persillade, origan', 13.00, 10.0),
   (3, 'pizza', 'Végétarienne', '0', 'Base sauce tomate, mozzarella, poivrons, tomates fraîches, coeur d''artichaut, olives à la Grecque, herbes de Provence', 10.00, 10.0),
   (4, 'autre plat', 'Burger', '1', '2 steaks hachés, double cheddar fondu, oignons, salade,', 7.50, 10.0),
   (5, 'autre plat', 'Wrap', '1', 'Du poulet croustillant et un délicieux bacon fumé. Ajoutez du cheddar fondu et une sauce onctueuse et ça y est : vous êtes à States !', 6.00, 10.0),
   (6, 'boisson', 'Eau', '1', 'Bouteille de 50cl', 1.50, 10.00),
   (7, 'boisson', 'Coca-Cola', '1', 'Cannette de 33cl', 1.90, 10.00),
   (8, 'dessert', 'Tiramisu', '1', 'Une fabrication maison pour ce dessert italien par excellence ! Saveur café et chocolat, un délice !', 2.10, 10.00),
   (9, 'pizza', 'Regina', '0', 'Base Sauce tomate, mozzarella, origan', 9.50, 10.0),
   (10, 'dessert', 'Mousse au Chocolat', '1', 'Fabrication maison', 1, 10.00),
   (11,'pizza','Montbéliarde', '0', 'Avec ses pommes de terres et sa cancoillotte : Une vraie pizza de la région !', 13.00,10.00);

-- ---------------
-- insertion données User
-- ---------------
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('1', 'Natasha', 'ROMANOFF', '1966-05-16', '1 02 03 04 05', 'monemail1@email.com', 'MDP1', '1');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('2', 'Han', 'SOLO', '1979-05-02', '1 02 03 04 06', 'monemail2@email.com', 'MDP2', '2');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('3', 'Luke', 'SKYWALKER', '1971-03-14', '1 02 03 04 07', 'monemail3@email.com', 'MDP3', '3');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('4', 'Wanda', 'MAXIMOFF', '1973-06-06', '1 02 03 04 08', 'email2@email.com', 'MDP4', '4');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('5', 'Peter', 'PARKER', '1971-11-15', '1 02 03 04 09', 'email3@email.com', 'MDP5', '5');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('6', 'Tony', 'STARK', '1966-02-14', '1 02 03 04 10', 'email4@email.com', 'MDP6', '6');
INSERT INTO `ocpizza`.`User` (`idUser`, `firstname`, `last_name`, `birth_date`, `phone`, `email`, `password`, `adress`) VALUES ('7', 'Bruce', 'BANNER', '1986-09-13', '1 02 03 04 11', 'email5@email.com', 'MDP7', '7');

-- ---------------
-- insertion données Store
-- ---------------
INSERT INTO `ocpizza`.`Store` (`idStore`, `name_store`, `siret`, `adressID`) VALUES ('1', 'OCPIZZA Pontarlier', '12345678901111', '5');
INSERT INTO `ocpizza`.`Store` (`idStore`, `name_store`, `siret`, `adressID`) VALUES ('2', 'OCPIZZA Montbéliard', '12345678901112', '14');
INSERT INTO `ocpizza`.`Store` (`idStore`, `name_store`, `siret`, `adressID`) VALUES ('3', 'OC PIZZA Besançon', '12345678901113', '8');

-- ---------------
-- insertion données Supplier
-- ---------------
INSERT INTO `ocpizza`.`Supplier` (`idSupplier`, `supplier_name`, `phone`, `id_adress`) VALUES ('1', 'PrimeurLocal', '03 45 67 89 01', '16');
INSERT INTO `ocpizza`.`Supplier` (`idSupplier`, `supplier_name`, `phone`, `id_adress`) VALUES ('2', 'Moulin de la montagne', '03 45 67 89 03', '17');
INSERT INTO `ocpizza`.`Supplier` (`idSupplier`, `supplier_name`, `phone`, `id_adress`) VALUES ('3', 'Viandes Doubs', '03 45 67 89 05', '18');
INSERT INTO `ocpizza`.`Supplier` (`idSupplier`, `supplier_name`, `phone`, `id_adress`) VALUES ('4', 'Grossiste.com', '03 45 67 89 07', '19');

-- ---------------
-- insertion données Recipe
-- ---------------
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('1', 'Étalez la pâte finement, garnissez-en la moitié de dés de mozzarella, de sauce tomate, de champignons et de lanières de jambon. Laissez le bord net pour permettre de refermer la calzone. Saupoudrez d'origan, assaisonnez et arrosez d'un peu d'huile d'olive. Cassez l'oeuf délicatement et déposez-le au centre de la garniture. Humectez les bords de la pizza et repliez-la en pressant les bords. Conseil : pour un aspect plus joli, vous pouvez dorer le chausson de pâte avec un peu de jaune d'oeuf délayé dans de l'eau avant de mettre au four.', '1');
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('2', 'Etalez la pâte finement, garnissez avec votre crème fraiche, la mozzarella puis le saumon en fines lamelles. Ajoutez quelques noix de persillade. Saupoudrez d\'origan. Mettre au four.', '2');
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('3', 'Etalez la pâte finement, garnissez avec votre sauce tomate, la mozzarella, les poivrons, quelques morceaux de tomates fraîches et d\'artichaut. Saupoudrez d\'herbes de Provence. Ajouter quelques olives. Mettre au four.', '3');
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('4', 'Faire revenir les oignons tranchés en fines lamelles dans une poêle avec de l\'huile d\'olive. Faire cuire les steaks. En fin de cuisson ajouter le cheddar sur le steak pour qu il fonde. Déposez la sauce, les oignons, la salade, les steaks avec les cheddar puis refermer le burger avec le couvercle.', '4');
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('5', 'Etalez la pâte finement, garnissez avec votre sauce tomate, la mozzarella. Saupoudrez d\'origan. Mettre au four.', '9');
INSERT INTO `ocpizza`.`Recipe` (`idRecipe`, `recipe_details`, `id_product`) VALUES ('6', 'Etalez la pâte finement, garnissez avec votre sauce tomate, quelques tranches de jambon et de pommes de terre. Ajoutez un cuillère à soupe de cancoillotte. Mettre au four.', '11');

-- ---------------
-- insertion données Staff
-- ---------------
INSERT INTO `ocpizza`.`Staff` (`idStaff`, `role`, `userID`, `storeID`) VALUES ('1', 'responsable', '5', '1');
INSERT INTO `ocpizza`.`Staff` (`idStaff`, `role`, `userID`, `storeID`) VALUES ('2', 'livreur', '10', '2');
INSERT INTO `ocpizza`.`Staff` (`idStaff`, `role`, `userID`, `storeID`) VALUES ('3', 'pizzaiolo', '15', '3');
INSERT INTO `ocpizza`.`Staff` (`idStaff`, `role`, `userID`, `storeID`) VALUES ('4', 'livreur', '20', '3');

-- ---------------
-- insertion données Order
-- ---------------
INSERT INTO `ocpizza`.`Order` (`idOrder`, `date`, `orderStatus`, `delivery`, `paymentValidity`, `customerID`, `id_store`) VALUES ('1', '2021-07-01', 'livrée', '1', '1', '3', '2');
INSERT INTO `ocpizza`.`Order` (`idOrder`, `date`, `orderStatus`, `delivery`, `paymentValidity`, `customerID`, `id_store`) VALUES ('2', '2021-07-01', 'délivrée', '1', '1', '4', '1');
INSERT INTO `ocpizza`.`Order` (`idOrder`, `date`, `orderStatus`, `delivery`, `paymentValidity`, `customerID`, `id_store`) VALUES ('3', '2021-07-02', 'livrée', '1', '1', '6', '1');
INSERT INTO `ocpizza`.`Order` (`idOrder`, `date`, `orderStatus`, `delivery`, `paymentValidity`, `customerID`, `id_store`) VALUES ('4', '2021-07-02', 'livrée', '1', '1', '8', '2');
INSERT INTO `ocpizza`.`Order` (`idOrder`, `date`, `orderStatus`, `delivery`, `paymentValidity`, `customerID`, `id_store`) VALUES ('5', '2021-07-03', 'en livraison', '0', '0', '2', '3');
```

… (La suite est dans le fichier InserAllData.sql)

4.4. Quelques requêtes pour tester la base

Pour vérifier notre base, nous pouvons lancer quelques requêtes et vérifier le résultat.

- *Afficher la recette des pizzas*

```
-- ––––––––––––––
-- recette des pizzas
-- ––––––––––––––
SELECT Product.product_name AS Produit,
recipe.recipe_details AS Recette
FROM Recipe
INNER JOIN Product
ON id_product = idProduct
WHERE category = 'Pizza'
Order by product_name;
```

Produit	Recette
Calzone	Étalez la pâte finement, garnissez-en la moitié de dés de mozzarella, de sauce tomate, de champignons e...
Montbéliarde	Etalez la pâte finement, garnissez avec votre sauce tomate, quelques tranches de jambon et de pommes...
Norvégienne	Etalez la pâte finement, garnissez avec votre crème fraiche, la mozzarella puis le saumon en fines lamelle...
Regina	Etalez la pâte finement, garnissez avec votre sauce tomate, la mozzarella. Saupoudrez d'origan. Mettre a...
Végétarienne	Etalez la pâte finement, garnissez avec votre sauce tomate, la mozzarella, les poivrons, quelques morcea...

- *Afficher les commandes du client nommé Wayne*

```
-- ––––––––––––––
-- rechercher : commande de M.WAYNE avec détails et quantités
-- ––––––––––––––
SELECT Order.idOrder AS Commande, User.last_name AS Nom,
User.firstname AS Prénom, Order.date, Product.product_name AS
Produit, OrderLine.quantity AS Quantité
FROM ocpizza.Order
INNER JOIN Customer ON customerID = idCustomer
INNER JOIN User ON userID = idUser
INNER JOIN OrderLine ON OrderLine.idOrder = Order.idOrder
INNER JOIN Product ON OrderLine.idProduct = Product.idProduct
WHERE last_name = 'Wayne'
ORDER BY Order.idOrder;
```

Commande	Nom	Prénom	date	Produit	Quantité
5	WAYNE	Bruce	2021-07-03	Burger	2
5	WAYNE	Bruce	2021-07-03	Eau	2
5	WAYNE	Bruce	2021-07-03	Coca-Cola	3
5	WAYNE	Bruce	2021-07-03	Montbéliarde	3

- *Afficher le nom du fournisseur du produit Mozzarella*

```
-- ––––––––––––––
-- recherche nom du fournisseur de la mozzarella
-- ––––––––––––––
SELECT DISTINCT Supplier.supplier_name AS Fournisseur,
Ingredient.ingredient_name AS Ingrédient
FROM Supplier
Inner JOIN Stock ON id_supplier = idSupplier
inner join Ingredient ON id_Ingredient = idIngredient
WHERE ingredient_name = 'Mozzarella';
```

Fournisseur	Ingrédient
PrimeurLocal	Mozzarella