EL MOKRETAR Malak
BTS SIO 2 Option B Solutions Logicielles et Applications Métiers

Dossier de conception :
Projet 1 site d’e-commerce Nabi

Numéro de candidat : 02148340887 Session 2025

Table des matières

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2025**

ANNEXE 9-1-B : Fiche descriptive de réalisation professionnelle (recto)

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

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE	**N° réalisation : 1**
Nom, prénom : EL MOKRETAR Malak	**N° candidat : 02148340887**
Épreuve ponctuelle ☐ **Contrôle en cours de formation** ☒	**Date :** 06 / 06 / 2025
Organisation support de la réalisation professionnelle	
Intitulé de la réalisation professionnelle Site d'e-commerce Nabi	
Période de réalisation : 2 semestres **Lieu :** Lycée Théodore Aubanel **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)

Ressources fournies :
- Création d'une application Web marchande (ou site d'e-commerce)
- Un cahier des charges avec les fonctionnalités demandées a été fourni

Résultats attendus :
Création d'un site Web marchand avec :
- Liste des produits
- Gestion des produits (création, modification, suppression)
- Gestion du panier
- Gestion des utilisateurs (création, modification, suppression)

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

Ressources logicielles :
- Machine Virtuelle Windows 10 Professionnel ;
- WAMPServer ;
- Base de données MySQL sur PHPMyAdmin ;
- Visual Studio Code ;
- Github ;
- https://app.diagrams.net ;

Ressources documentaires :
- TP fournis ;
- Documentations en ligne : https://www.php.net/docs.php, https://www.w3schools.com, https://getbootstrap.com/docs/5.3/getting-started/introduction, https://sql.sh
- Forums en ligne : https://stackoverflow.com/questions ;
- Validateur W3C (https://validator.w3.org/#validate_by_input+with_options)
- Éditeur de documents Markdown (.md) : https://readme.so/fr/editor

Ressources matérielles :
- Ordinateur fixe ;
- Ordinateur portable ;

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

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

Modalité d'accès à la production :

I. Installation :

1. Installer WAMP SERVER
2. Cloner le projet :

```
git clone https://github.com/malak-elmokretar/projet_flowershop.git
```

3. Aller dans le répertoire du projet :

```
cd projet_flowershop
```

4. Installer les dépendances dans le dossier **.\lib**

```
cd .\lib
```

5. Installer Composer

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') ===
'dac665fdc30fdd8ec78b38b9800061b4150413ff2e3b6f88543c636f7cd84f6db9189d43a81e5503cda447da73c7e5b6') { echo
'Installer verified'.PHP_EOL; } else { echo 'Installer corrupt'.PHP_EOL; unlink('composer-setup.php'); exit(1); }"
php composer-setup.php
php -r "unlink('composer-setup.php');"
```

6. Installer TWIG

```
composer require "twig/twig:^3.0"
```

7. Démarrer le serveur

II. Déploiement de la base de données :

1. Dans **PHPMyAdmin**, créer un utilisateur autre que root :

```
CREATE USER 'malak'@'%' IDENTIFIED WITH caching_sha2_password BY 'S98-p[Phwf6pxRL(';
-- insérer le nom de l'utilisateur et son mot de passe que vous souhaitez, vous devrez cependant modifier les valeurs des
variables $config["login"] et $config["mdp"] dans le fichier .\config\parametres.php si vous utilisez un nom d'utilisateur et un
mot de passe différents de ceux indiqués
 GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, FILE, REFERENCES, ALTER, CREATE VIEW, EVENT,
TRIGGER, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EXECUTE ON *.* TO 'test'@'%';ALTER USER 'test'@'%'
REQUIRE NONE WITH MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0;
```

2. Récupérer le fichier .\bd\flowers.sql
3. Dans **PHPMyAdmin**, créer une base de données flowers :

```
CREATE DATABASE "flowers";
```

4. Cliquer sur la base de données créée puis cliquer sur *Importer* et importer le fichier **.\bd\flowers.sql**

III. Déploiement du site

1. Depuis la racine du projet, coller le projet dans **C:\wamp64\www**

```
xcopy . C:\wamp64\www\projet_flowershop /E
```

2. Dans votre **navigateur**, rechercher l'adresse **http://localhost/add_vhost.php**
   a. Remplir l'input "Nom du Virtual Host" : Flowershop
   b. Remplir l'input "Chemin complet absolu du dossier VirtualHost" : **C:/wamp64/www/projet_flowershop/public**
   c. Cliquer sur "Démarrer la création ou la modification du VirtualHost"

3. Dans Outils, cliquer sur Redémarrage DNS
4. Si ce n'est pas déjà fait, modifier les valeurs des variables $config["server"], $config["login"] et $config["mdp"] en y associant votre adresse IP et les informations de l'utilisateur créé dans la partie 1.
5. Rechercher dans votre **navigateur** http://projet_flowershop/

IV. Identifiants et mots de passe pour accéder à toutes les fonctionnalités

Base de données :
Identifiant : malak

---

[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.

Mot de passe : S98-p[Phwf6pxRL(

Visiteur :
Pas besoin de se connecter

Client :
Adresse e-mail : jurybts02@lyceeaubanel.fr
Mot de passe : pWy1#ffL7TPYAF36c

Administrateur :
Adresse e-mail : jurybts01@lyceeaubanel.fr
Mot de passe : 5grX2nKd4#m0QUKi

Modalité d'accès à la documentation :

README.md : présent à la racine du projet
https://github.com/malak-elmokretar/projet_flowershop

Vous pourrez également retrouver ce fichier dans le dossier **.\documentation**.

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

Le but de cette réalisation professionnelle était de créer une application Web de e-commerce.

Côté Front-end :

Des pages d'authentification ont été mises en place afin que l'utilisateur puisse créer un compte et s'y connecter.
L'application liste tous les produits leur titre, leur description et leur prix, qu'un connecté avec un compte administrateur peut modifier ou supprimer. Un visiteur (utilisateur non connecté) peut également contacter les administrateurs via un formulaire de contact.

Côté back-end :

- Tous les mots de passes sont cryptés.
- Les modifications apportées par les administrateurs sont automatiquement mises à jour dans la base de données.
- Toutes les requêtes SQL effectuées (sauf certains cas particuliers) sont des procédures stockées, des vues ou bien des déclencheurs (triggers) afin de minimiser les risques liés à des requêtes en clair dans le code source.

Toutes les données de l'application sont stockées en base de données dont voici la liste des tables :

	Table	Action	Lignes	Type	Interclassement	Taille	Perte
☐	anciensmdp	Parcourir Structure Rechercher Insérer Vider Supprimer	29	MyISAM	utf8mb4_0900_ai_ci	4,7 kio	-
☐	commande	Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
☐	composer	Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
☐	fournisseur	Parcourir Structure Rechercher Insérer Vider Supprimer	7	MyISAM	utf8mb4_0900_ai_ci	2,9 kio	-
☐	listerproduits	Parcourir Structure Rechercher Insérer Éditer Supprimer	~0	Vue	---	-	-
☐	listerroles	Parcourir Structure Rechercher Insérer Éditer Supprimer	~0	Vue	---	-	-
☐	listertype	Parcourir Structure Rechercher Insérer Éditer Supprimer	~0	Vue	---	-	-
☐	listerutilisateurs	Parcourir Structure Rechercher Insérer Éditer Supprimer	~0	Vue	---	-	-
☐	occasion	Parcourir Structure Rechercher Insérer Vider Supprimer	11	MyISAM	utf8mb4_0900_ai_ci	3,3 kio	-
☐	produit	Parcourir Structure Rechercher Insérer Vider Supprimer	5	MyISAM	utf8mb4_0900_ai_ci	4,8 kio	408 o
☐	role	Parcourir Structure Rechercher Insérer Vider Supprimer	2	MyISAM	utf8mb4_0900_ai_ci	2,0 kio	-
☐	saison	Parcourir Structure Rechercher Insérer Vider Supprimer	7	MyISAM	utf8mb4_0900_ai_ci	2,2 kio	-
☐	type	Parcourir Structure Rechercher Insérer Vider Supprimer	4	MyISAM	utf8mb4_0900_ai_ci	2,1 kio	-
☐	utilisateur	Parcourir Structure Rechercher Insérer Vider Supprimer	15	MyISAM	utf8mb4_0900_ai_ci	8,0 kio	136 o
☐	utilisateur_supprime	Parcourir Structure Rechercher Insérer Vider Supprimer	3	MyISAM	utf8mb4_0900_ai_ci	4,4 kio	-
	15 tables	Somme	~83	MyISAM	utf8mb4_0900_ai_ci	36,4 kio	544 o

Figure 1: Ensemble des tables de la base de données utilisée par l'application

PRÉSENTATION DU BESOIN MÉTIER

Description du site

Nabi est une plateforme de vente en ligne proposant un large catalogue de bouquets de fleurs locales. Pour des questions éthiques, le site a été soumis à plusieurs tests d'accessibilité (couleurs, affichage sur des écrans de différentes tailles).

Cadre logique du projet

L'application doit permettre à l'utilisateur de :

- Gérer le catalogue de produits en listant, ajoutant, modifiant ou supprimant des produits, tout en s'assurant que les informations (prix, descriptions, images) sont à jour.
- Assurer la sécurité des données grâce à un système d'authentification sécurisé.
- Gérer le panier
- Gestion des utilisateurs (création, modification, suppression

Résultats attendus :

- Amélioration de l'efficacité : L'application permet d'automatiser et de centraliser des tâches administratives, ce qui améliore la productivité de l'équipe.
- Gains financiers : l'application Web permet à l'entreprise d'attirer davantage de clients, qui seront tentés de commander des fleurs via le site plutôt que de se déplacer.
- Gain de notoriété : L'application en ligne permet à l'entreprise de se faire connaître davantage, en attirant une clientèle plus large grâce à sa visibilité accrue sur le web, notamment grâce à une interface utilisateur réfléchie pour optimiser l'expérience utilisateur.
- Sécurisation des données : L'application assure que seules les personnes autorisées accèdent aux informations sensibles, renforçant la sécurité de l'entreprise.

PRÉSENTATION DE L'ARCHITECTURE ET DE L'ARBORESCENCE DU PROJET

Architecture du projet

L'architecture utilisée pour ce projet est une architecture Modèle-Vue-Contrôleur (ou MVC), conçue pour séparer la logique métier et l'affichage du logiciel, ce qui une meilleure répartition du travail et une maintenance améliorée. Cela sécurise également le site car les requêtes SQL utilisées ne sont pas visibles par les utilisateurs.

Le Modèle : contient les classes. Ces dernières contiennent les méthodes qui accèdent à la base de données
La Vue : contient ce qui est visible par l'utilisateur (les pages .twig)
Le Contrôleur : Contient les fonctions qui permettent d'afficher les vues et d'appeler les classes

Source : https://developer.mozilla.org/fr/docs/Glossary/MVC

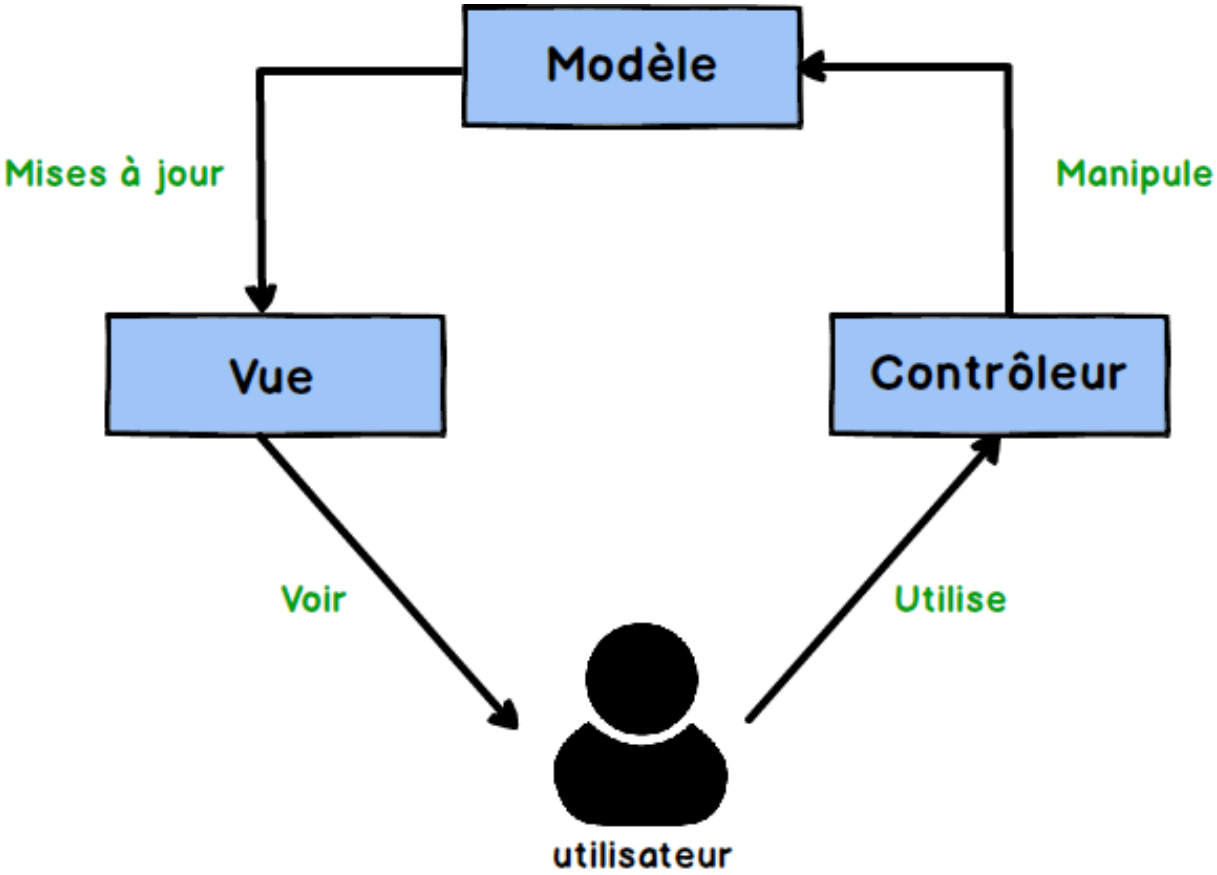


Figure 2 Schéma explicatif de l'architecture MVC. Source: https://waytolearnx.com/2020/06/difference-entre-mvc-et-mvvm.html

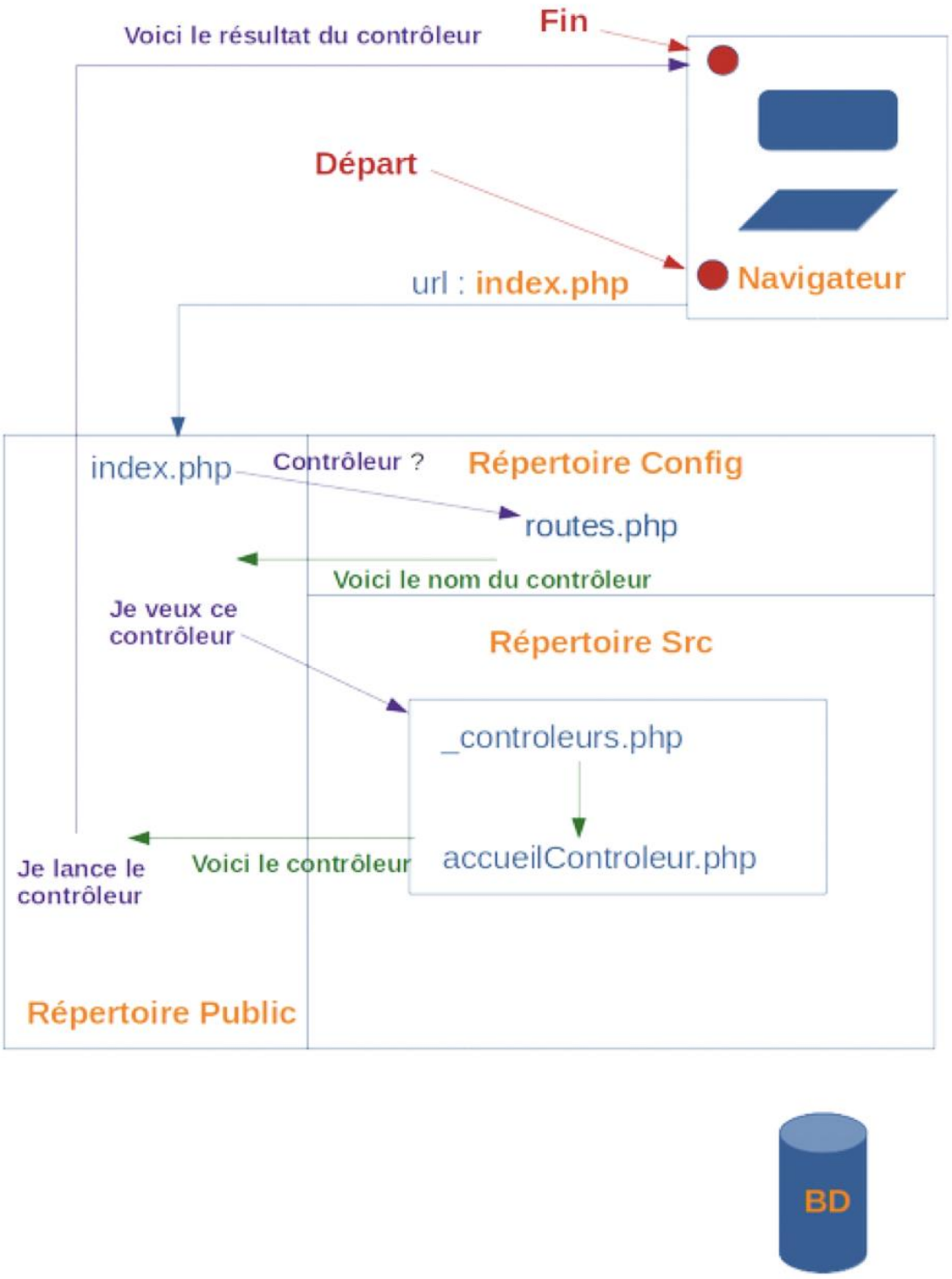
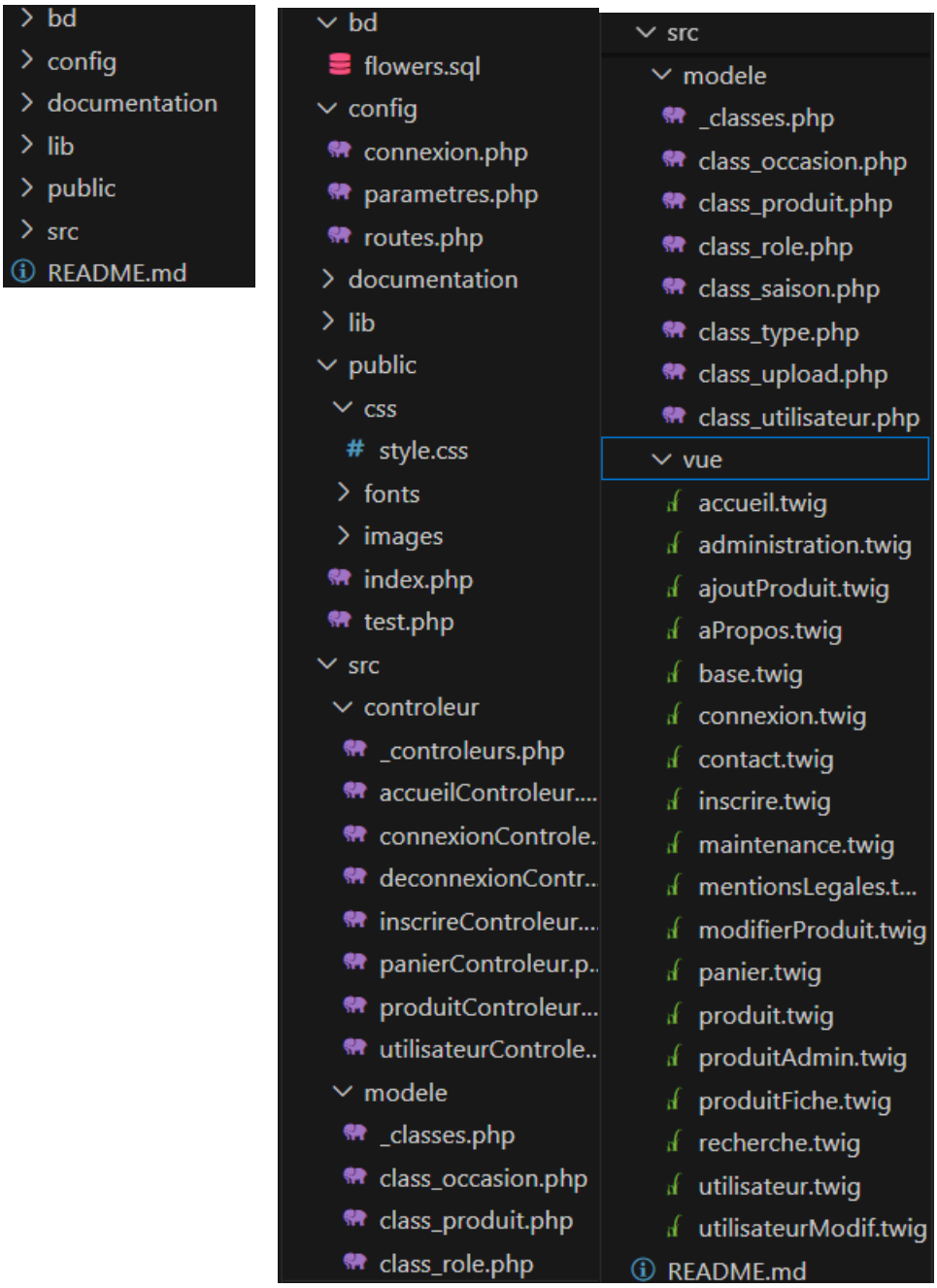


Figure 3: Schéma explicatif de l'architecture MVC

Arborescence du projet



FONCTIONNALITÉS :

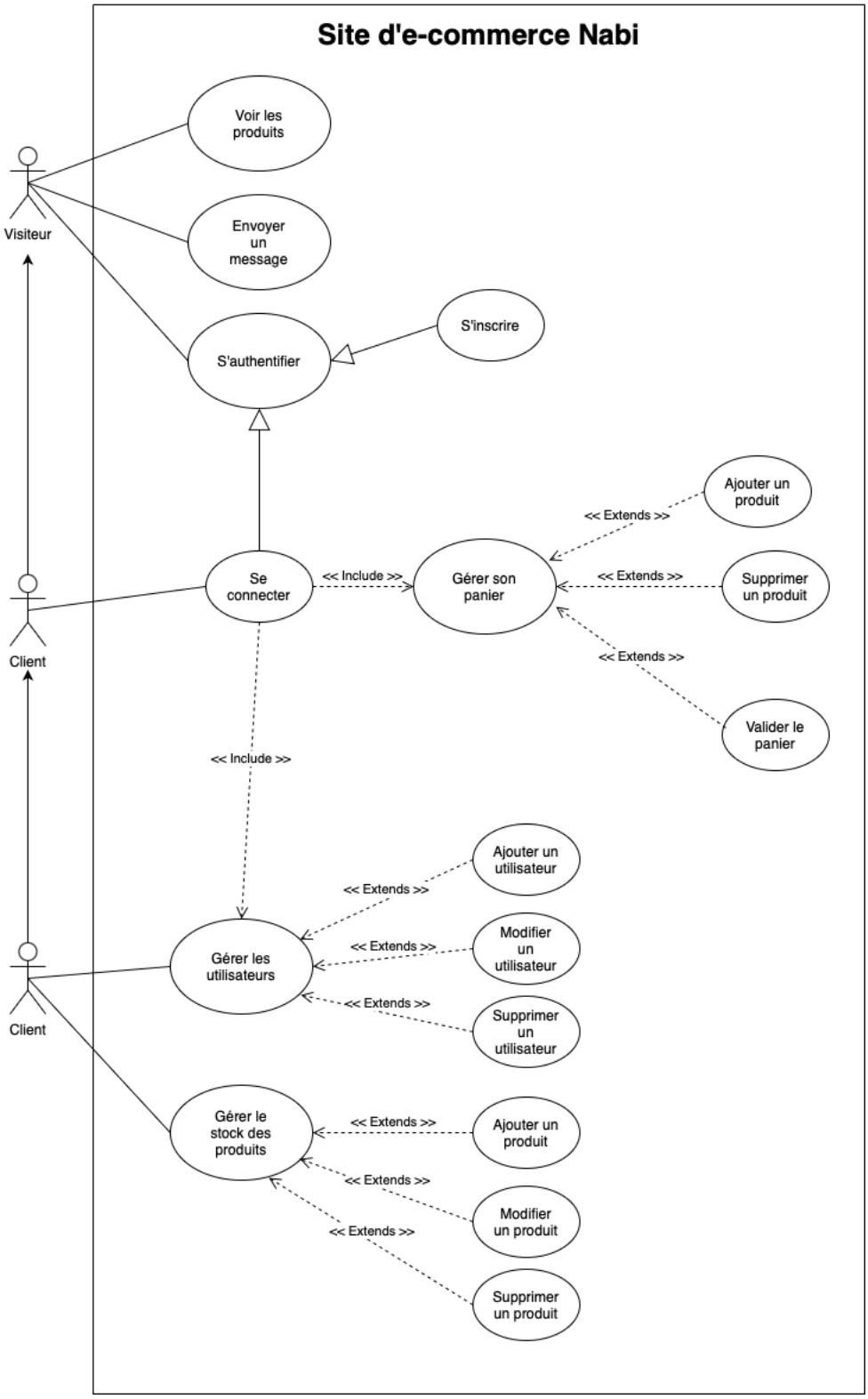


Figure 4: Modélisation du cas d'utilisation de l'application

LANGAGES UTILISÉS :

Front-End :

- HTML
- CSS
- JavaScript
- TWIG

Back-End :

- PHP
- SQL

STRUCTURE DE LA BASE DE DONNÉES ET REQUÊTES UTILISÉES :

Structure de la base de données

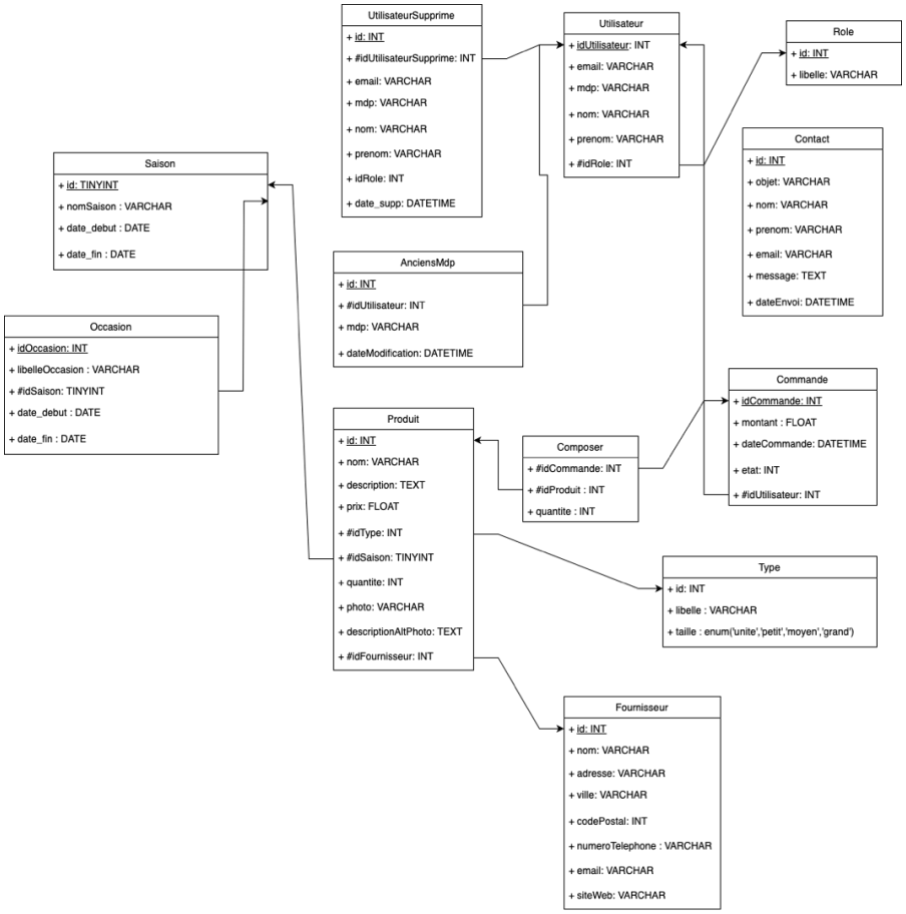


Figure 5: Schéma de la structure de la base de données

Légende :
Attribut : Clé primaire
Attribut : clé étrangère

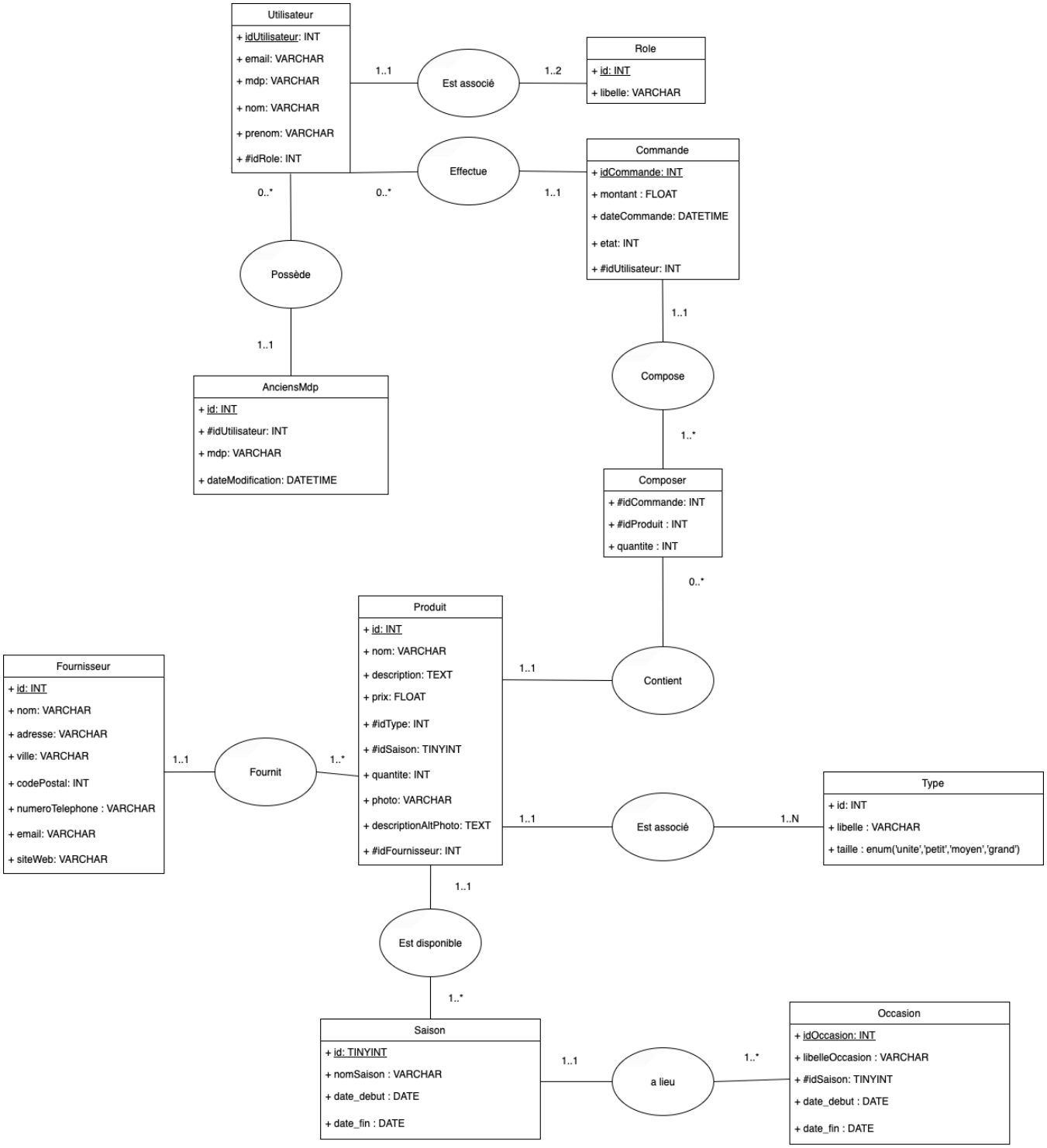


Figure 6: Modèle Conceptuel des Données de la base de données

Requêtes utilisées

1. Création des tables

Table utilisateur :

```
CREATE TABLE IF NOT EXISTS `utilisateur` (
  `idUtilisateur` INT PRIMARY KEY AUTO_INCREMENT,
  `email` VARCHAR(100) UNIQUE,
  `mdp` VARCHAR(256),
  `nom` VARCHAR(100),
  `prenom` VARCHAR(100),
  `idRole` INT DEFAULT '2',
  FOREIGN KEY idRole
  REFERENCES role(id)
) ;
```

Table Role :

```
CREATE TABLE IF NOT EXISTS `role` (
  `id` int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `libelle` VARCHAR(50) DEFAULT NULL,
  PRIMARY KEY (`id`)
);
```

Table utilisateur_supprime :

```
CREATE TABLE IF NOT EXISTS `utilisateur_supprime` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `idUtilisateurSupprime` INT DEFAULT NULL,
  FOREIGN KEY idUtilisateurSupprime
  REFERENCES utilisateur(idUtilisateur),
  `email` VARCHAR(100) DEFAULT NULL,
  `mdp` VARCHAR(256) DEFAULT NULL,
  `nom` VARCHAR(100) DEFAULT NULL,
  `prenom` VARCHAR(100) DEFAULT NULL,
  `idRole` INT DEFAULT NULL,
  FOREIGN KEY idRole
  REFERENCES role(id),
  `date_supp` DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

Table anciensMdp:

```
CREATE TABLE IF NOT EXISTS `anciensmdp` (
```

```
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `idUtilisateur` INT DEFAULT NULL,
  FOREIGN KEY idUtilisateur
  REFERENCES utilisateur(idUtilisateur),
  `mdp` VARCHAR(255) DEFAULT NULL,
  `dateModification` DATETIME DEFAULT CURRENT_TIMESTAMP()
);
```

Table Produit :

```
CREATE TABLE IF NOT EXISTS `produit` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `nom` VARCHAR(100) DEFAULT NULL,
  `description` TEXT,
  `prix` FLOAT DEFAULT NULL,
  `idType` INT DEFAULT NULL,
  FOREIGN KEY idType
  REFERENCES type(id),
  `idSaison` TINYINT DEFAULT NULL,
  FOREIGN KEY idSaison
  REFERENCES saison(id),
  `quantite` INT NOT NULL,
  `photo` VARCHAR(255),
  `descriptionPhotoAlt` TEXT NOT NULL,
  `idFournisseur` INT,
  FOREIGN KEY idFournisseur
  REFERENCES fournisseur(id)
);
```

Table Type :

```
CREATE TABLE IF NOT EXISTS `type` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `libelle` VARCHAR(100) DEFAULT NULL,
  `taille` ENUM('unite','petit','moyen','grand') DEFAULT NULL
) ;
```

Table saison :

```
CREATE TABLE IF NOT EXISTS `saison` (
  `id` TINYINT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `nomSaison` VARCHAR(100) DEFAULT NULL,
  `date_debut` DATE DEFAULT NULL,
  `date_fin` DATE DEFAULT NULL
);
```

Table occasion :

```
CREATE TABLE IF NOT EXISTS `occasion` (
  `idOccasion` int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `libelleOccasion` varchar(100) DEFAULT NULL,
  `idSaison` TINYINT DEFAULT NULL,
  FOREIGN KEY idSaison
```

```
  REFERENCES saison(id),
  `dateDebut` date DEFAULT NULL,
  `dateFin` date DEFAULT NULL
);
```

Table fournisseur :

```
CREATE TABLE IF NOT EXISTS `fournisseur` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `nom` VARCHAR(100) DEFAULT NULL,
  `adresse` VARCHAR(100) DEFAULT NULL,
  `ville` VARCHAR(100) DEFAULT NULL,
  `codePostal` INT DEFAULT NULL,
  `numeroTelephone` VARCHAR(10) DEFAULT NULL,
  `email` VARCHAR(100) DEFAULT NULL,
  `siteWeb` VARCHAR(255) DEFAULT NULL
);
```

Table commande :

```
CREATE TABLE IF NOT EXISTS `commande` (
  `idCommande` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `montant` FLOAT DEFAULT NULL,
  `dateCommande` DATETIME DEFAULT NULL,
  `etat` INT DEFAULT NULL,
  `idUtilisateur` INT DEFAULT NULL,
  FOREIGN KEY idUtilisateur
  REFERENCES utilisateur(idUtilisateur)
);
```

Table composer :

```
CREATE TABLE IF NOT EXISTS `composer` (
  `idCommande` INT DEFAULT NULL,
  FOREIGN KEY idCommande
  REFERENCES commande(id),
  `idProduit` INT DEFAULT NULL,
  FOREIGN KEY idProduit
  REFERENCES produit(id),
  `quantite` INT DEFAULT NULL
);
```

2. Création des vues :

Vue listerOccasions :

```
CREATE VIEW listerOccasions AS
SELECT idOccasion, libelleOccasion, nomSaison, dateDebut, dateFin
FROM occasion
```

```
JOIN saison ON occasion.idSaison = saison.id;
```

Vue selectNombre :

```
CREATE VIEW selectNombre
AS SELECT COUNT(*) AS nb FROM produit;
```

Vue listerProduits :

```
CREATE VIEW listerProduits AS
SELECT description, descriptionPhotoAlt, id, libelle, libelleOccasion, nom,
nomSaison, photo, prix, quantite, taille
FROM `produit`
LEFT JOIN saison ON produit.idSaison = saison.id
LEFT JOIN occasion ON saison.id = occasion.idSaison
LEFT JOIN type ON produit.idType = type.id;
```

Vue listerType

```
CREATE VIEW listerType AS
SELECT *
FROM type;
CREATE VIEW listerRoles AS
SELECT *
FROM role;
```

Vue listerUtilisateurs

```
CREATE VIEW listerutilisateurs AS
SELECT email, mdp, nom, prenom, libelle
FROM utilisateur
JOIN role ON utilisateur.idRole = role.id;
```

3. Création des procédures stockées :

Procédure ajouterProduit :

```
CREATE PROCEDURE ajouterProduit(IN p_nom VARCHAR(100), p_description TEXT,
p_prix FLOAT, p_idType INT, p_idSaison INT, p_quantite INT,
p_descriptionPhotoAlt TEXT)
BEGIN
    INSERT INTO produit (nom, description, prix, idType, idSaison, quantite,
descriptionPhotoAlt)
        VALUES (p_nom, p_description, p_prix, p_idType, p_idSaison,
p_quantite, p_descriptionPhotoAlt);
END;;
```

Procédure connexion :

```
CREATE PROCEDURE connexion(IN p_email VARCHAR(100))
BEGIN
    SELECT email, idRole, mdp
    FROM utilisateur
    WHERE email = p_email;
END;;
```

Procédure inscription:

```
CREATE PROCEDURE inscription(IN p_email VARCHAR(100), p_mdp VARCHAR(256),
p_nom VARCHAR(100), p_prenom VARCHAR(100), p_idRole INT)
BEGIN
    INSERT INTO utilisateur(email, mdp, nom, prenom, idRole) VALUES (p_email,
p_mdp, p_nom, p_prenom, p_idRole);
END;;
```

Procédure listerProduitsParId

```
CREATE PROCEDURE listerProduitsParId(IN p_idProduit INT)
BEGIN
    SELECT produit.id, nom, description, prix, type.libelle, taille,
nomSaison, quantite, photo, descriptionPhotoAlt
    FROM produit
    LEFT JOIN type ON produit.idType = type.id
    LEFT JOIN saison ON produit.idSaison = saison.id
    WHERE produit.id = p_idProduit;
END;;
```

Procédure listerUtilisateursParId :

```
CREATE PROCEDURE listerUtilisateursParId(IN p_idUtilisateur INT)
BEGIN
    SELECT idUtilisateur, email, nom, prenom, idRole
    FROM utilisateur
    WHERE idUtilisateur=p_idUtilisateur
    ORDER BY idUtilisateur;
END;;
```

Procédure modifierMDP :

```
CREATE PROCEDURE modifierMDP(IN p_idUtilisateur INT, p_mdp VARCHAR(256))
BEGIN
    UPDATE utilisateur
    SET mdp = p_mdp
    WHERE idUtilisateur = p_idUtilisateur;
END;;
```

Procédure modifierProduit :

```
CREATE PROCEDURE modifierProduit (IN p_idProduit INT, p_nom VARCHAR(100),
p_description TEXT, p_prix FLOAT, p_idType INT, p_idSaison INT, p_quantite
INT, p_descriptionPhotoAlt TEXT)
BEGIN
    UPDATE produit
    SET
        nom = p_nom,
        description = p_description,
        prix = p_prix,
        idType = p_idType,
        idSaison = p_idSaison,
        quantite = p_quantite,
        descriptionPhotoAlt = p_descriptionPhotoAlt
    WHERE id = p_id;
END;;
```

Procédure modifierProduit

```
CREATE PROCEDURE modifierProduit (IN p_idProduit INT, p_nom VARCHAR(100),
p_description TEXT, p_prix FLOAT, p_idType INT, p_idSaison INT, p_quantite
INT, p_descriptionPhotoAlt TEXT)
BEGIN
    UPDATE produit
    SET
        nom = p_nom,
        description = p_description,
        prix = p_prix,
        idType = p_idType,
        idSaison = p_idSaison,
        quantite = p_quantite,
        descriptionPhotoAlt = p_descriptionPhotoAlt
    WHERE id = p_id;
END;;
```

4. Déclencheur :

```
CREATE TRIGGER historique_anciens_utilisateurs
AFTER UPDATE
ON utilisateur
FOR EACH ROW
BEGIN
    INSERT INTO utilisateur_supprime (idUtilisateurSupprime, email, mdp, nom,
prenom, idRole, date_supp)
    VALUES (OLD.idUtilisateur, OLD.email, OLD.mdp, OLD.nom, OLD.prenom,
OLD.idRole, NOW());
END;;
```