

# Rapport de projet de session

« Conception et réalisation d'un site web E-commerce sous nom : luxforAll »

**Réalisé par :**

Kamal Chokrane  
Ayoub El Bouinany

**Encadré par :**

MME. Fatimaezzahra sadraoui  
MME.Meryeme Faouzi

Année universitaire 2020-2021

## Site e-Commerce

---

Conception et réalisation d'un site web dynamique (site E-commerce) [Outils: HTML, CSS, JavaScript ,PHP et MySQL ]

# Contexte

Des ventes de mains en mains, vers des ventes virtuelles, passent les priorités des opérations de ventes des biens et des services, ce qui nous rend obligés de donner plus d'importance à la vente électronique. Les sites de vente en ligne permettent aux clients de profiter d'une foire virtuelle disponible est quotidiennement mise à jour sans la moindre contrainte, ce qui leur permettra de ne jamais rater les coups de cœur, ainsi une foire sans problèmes de distance géographique, ni d'horaire de travail ni de disponibilité de transport. D'une autre part ces sites offrent à la société de profiter de cet espace pour exposer ses produits à une plus large base de clientèle. Grâce à un site web e-commerce, on peut choisir et payer des articles comme dans un magasin réel. Pour acheter un produit d'une boutique virtuelle, il suffit le plus souvent de choisir les produits désirés puis de les mettre dans un panier d'achat. La commande sera livrée en fonction du choix de l'internaute et selon les modalités définies par le responsable de la boutique. Notre projet est réalisé dans le cadre d'améliorer nos compétences en langage de programmation web PHP ayant comme objectif principal : la conception et la création d'un site web e-commerce « luxforall » pour la vente des chemises en ligne.

## Fonctionnalités

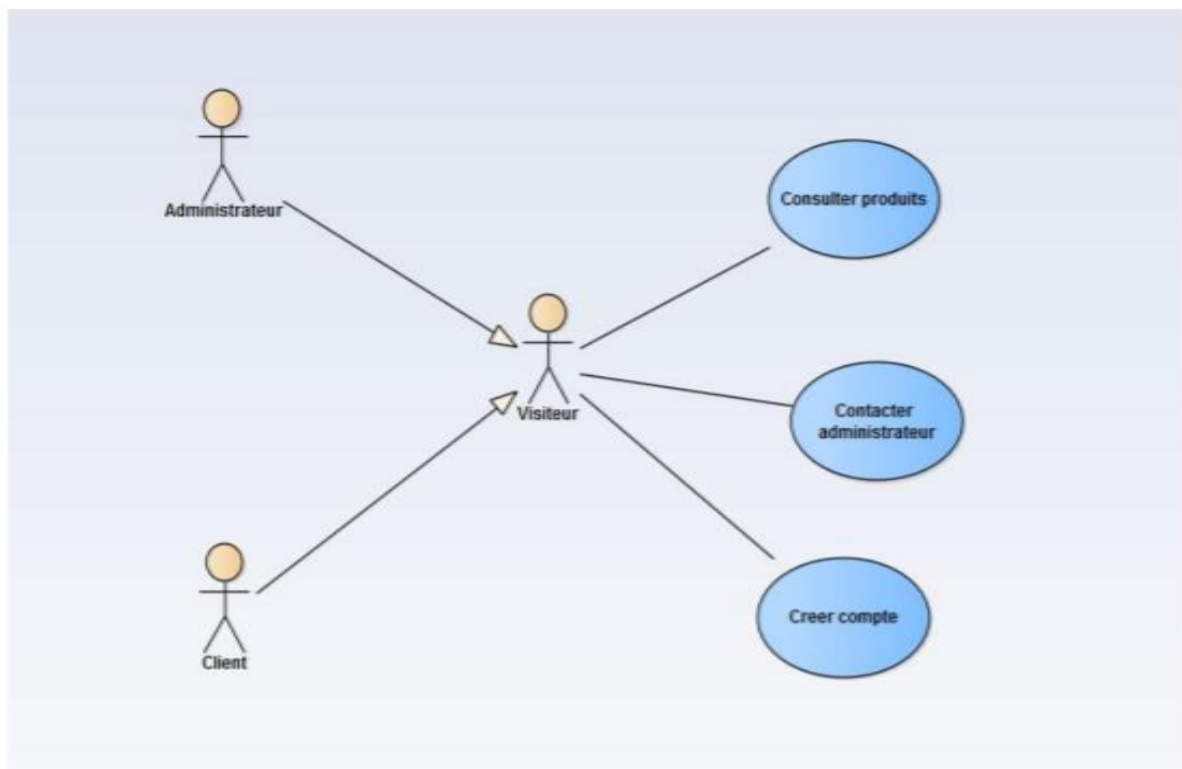
Ce projet consiste à la mise en place d'un site Web dynamique qui gère la commercialisation des vêtements et accessoires utilisés. Ceci est possible à travers des catalogues en ligne proposant ces produits aux meilleurs prix par rapport aux prix initiaux. La société n'aura donc qu'à agencer ses produits et bien sûr de mettre sa base de données à jour. Le client est toujours anonyme mais pour pouvoir passer à un stade plus rigoureux, il faut qu'il s'inscrive, cela se fait uniquement pour la première commande mais après, notre client peut s'authentifier avec son E-mail et son mot de passe pour passer d'autres commandes. Le client a le droit de consulter tous les produits et ajouter ce qu'il veut à son panier. Et après le choix d'un produit le client doit mentionner la quantité qui s'ajoute automatiquement à son panier avec le prix unitaire et le prix total afin de finir sa commande. Alors ce site web offre beaucoup de services à savoir : - Consulter les produits. -Laisser des commentaires sur les produits. -Modifier le panier en ajoutant ou supprimant des produits. -Consulter le profil. - Lancer une commande en ligne contient ces différents produits, qui sont par la suite livrés à domicile. -Contacter l'administrateur.

# La conception

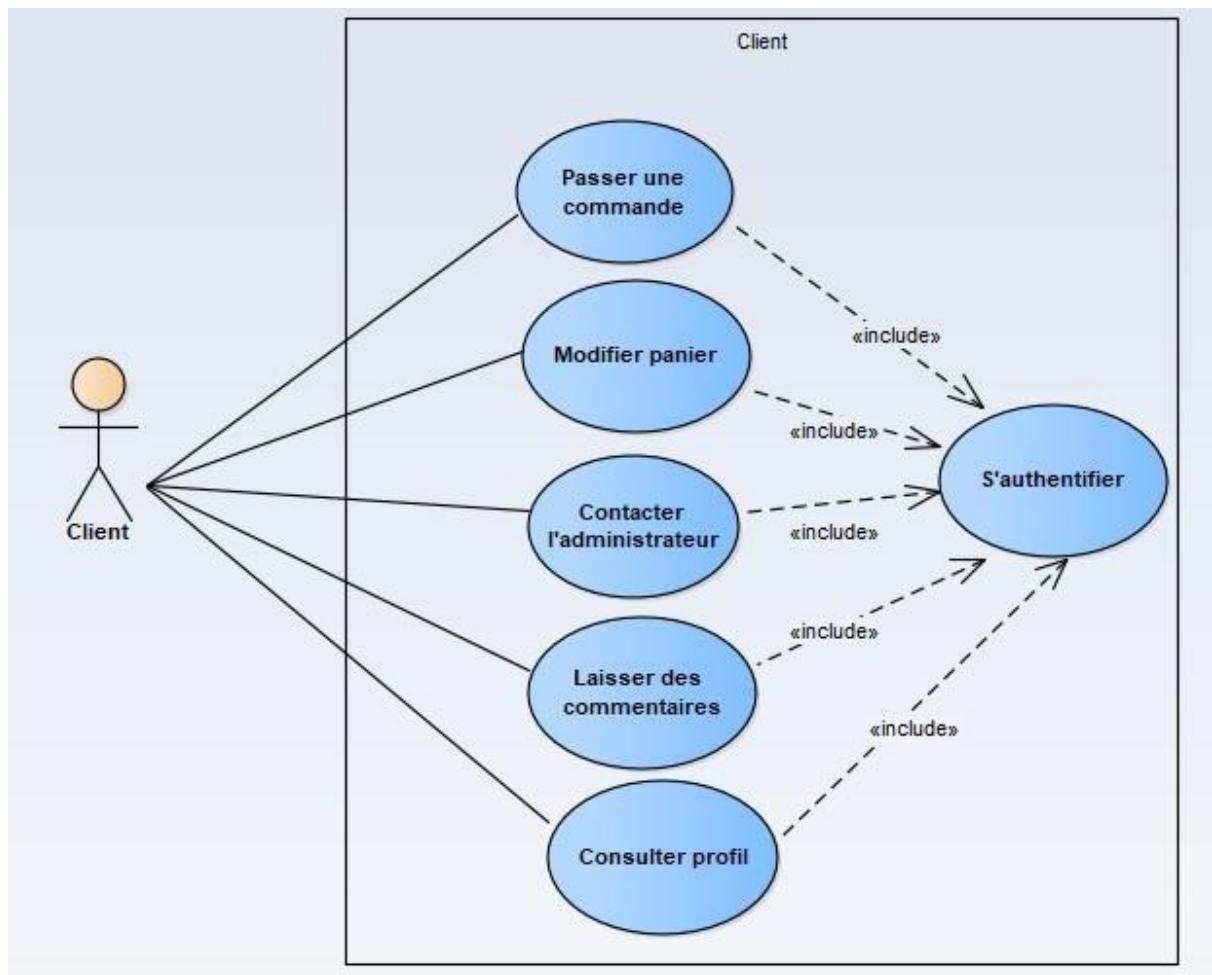
## 1- Diagramme de cas d'utilisation :

Les rôles des diagrammes de cas d'utilisation sont de recueillir, d'analyser et d'organiser les besoins, ainsi que de recenser les grandes fonctionnalités d'un système.

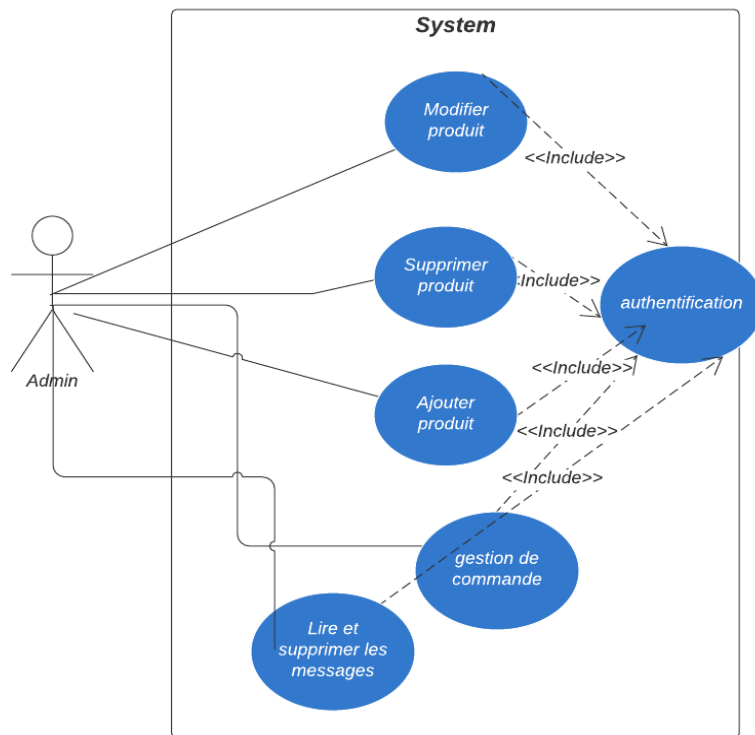
**Le visiteur** : c'est un individu qui est en train de chercher un produit pour l'acheter ou pour avoir une idée sur les modèles et les prix. C'est un utilisateur inconnu donc il n'est pas encore un client.



**Le client** : Cet acteur est un visiteur ayant déjà créé un compte sur notre site, il peut donc suivre le processus d'achat des produits en toute sécurité sachant que notre système doit être l'unique responsable de la confidentialité des données personnelles de ses clients.

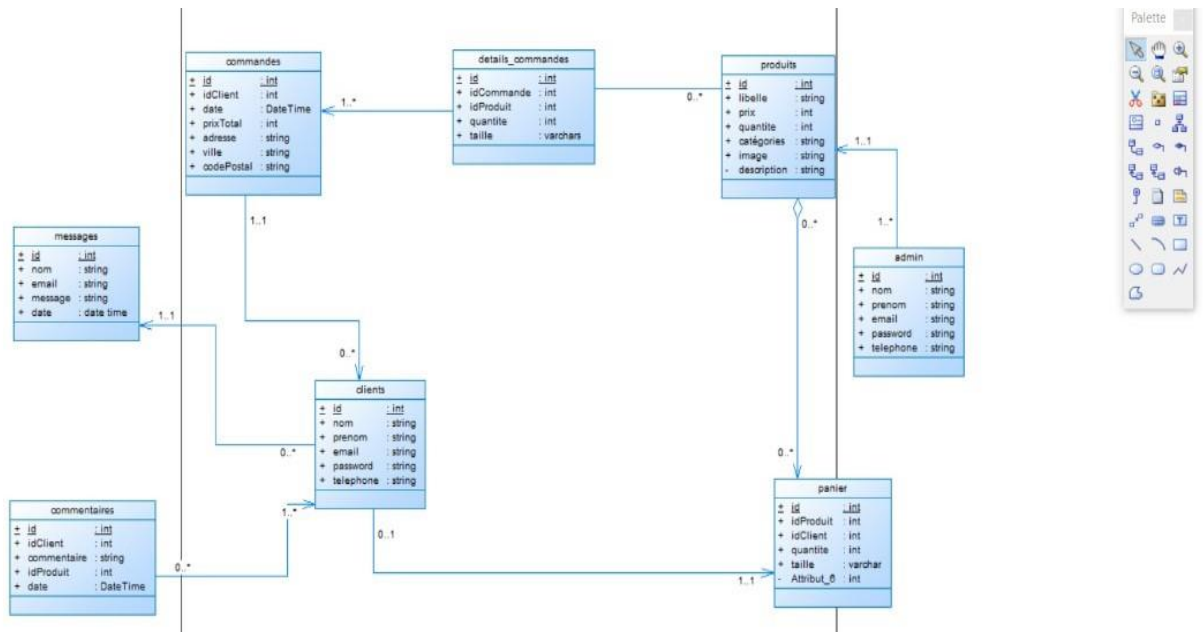


**L'admin :** L'administrateur E-commerce/webstore gère le déroulement des activités commerciales électroniques et les promotions en ligne auprès de la clientèle afin d'augmenter l'utilisation de ce service. Il vérifie également l'exécution des commandes et l'exactitude des produits énumérés, analyse toutes les ventes de produits, mène les promotions approuvées, maintient les comptes de clients et gère la circulation des produits tout au long de la chaîne d'approvisionnement.



## 2- Le Diagramme de classe:

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Ce diagramme fait partie de la partie statique d'UML car il fait abstraction des aspects temporels et dynamiques.



# Réalisation

## 1- Les techniques et les fonctions utilisées :

**1.1. L'utilisation des sessions et des variables superglobales :** On a utilisé plusieurs variables superglobales (\$\_GET, \$\_POST, \$\_SESSION ...).

-Les superglobales \$\_GET et \$\_POST : sont utilisées pour manipuler les informations envoyées via un formulaire HTML. \$\_GET stockera les valeurs lorsque le formulaire sera envoyé via la méthode GET tandis que \$\_POST stockera les valeurs lorsque le formulaire sera envoyé via la méthode POST.

```

if(isset($_POST['inscrire']) && !empty($_POST['inscrire']))
{

```

```

$nom =htmlspecialchars($_POST['nom']);
$prenom = htmlspecialchars($_POST['prenom']);
$email = htmlspecialchars($_POST['email']);
$telephone =htmlspecialchars($_POST['telephone']);
$password =htmlspecialchars($_POST['password']);
}

```

Ici, on définit la méthode POST comme méthode d'envoi du formulaire et la page inscrire.php comme page de traitement des données (la page où les données du formulaire devront être envoyées).

Dans le cas présent, cette page est la même que la page dans laquelle se situe le formulaire, ce qui signifie que les données seront envoyées vers notre page courante, et ils seront insérées dans notre base de données.

-\$SESSION : Cette superglobale est un tableau associatif qui stocke les différentes variables de sessions avec leurs noms en index du tableau et leurs valeurs en valeurs du tableau. On a créé quelques variables de sessions dans notre fichier connexion.php :

```

$_SESSION['client_id'] = $client['id'];
$_SESSION['client_nom'] = $client['nom'];
$_SESSION['client_prenom'] = $client['prenom'];
$_SESSION['client_email'] = $client['email'];
$_SESSION['client_telephone'] = $client['telephone'];

```

Pour activer les sessions on va utiliser la fonction session\_start(); :

```

<?php require 'dbconn.php'; session_start();
?>

```

Et pour terminer les sessions on va utiliser ces deux fonctions session\_destroy() , session\_unset() .

session\_destroy() : détruit toutes les données associées à la session courante session\_unset() : détruit toutes les variables d'une session.

```

<?php
include('config.php');

```

```
session_unset  
session_destroy();  
header("location: ../index.php");  
?>
```

## 1.2. La connexion à la base de données MySQL : On peut utiliser l'extension PDO pour se connecter à la base de données MySQL.

```
$db = new PDO('mysql:host='.$DATABASE_HOST.';dbname='.$DATABASE_NAME  
.';charset=utf8mb4', $DATABASE_USER, $DATABASE_PASSWORD);
```

## 1.3. L'utilisation des exceptions :

Utiliser des exceptions va nous permettre de gérer les erreurs de manière plus fluide et de personnaliser la façon dont un script doit gérer certaines erreurs.

Ici on commence par établir la connexion vers la base de données MySQL en utilisant le PDO. Si la connexion est échouée une exception de la classe PDOException est également lancée. On capture cette erreur dans le bloc catch juste en dessous et on affiche les informations relatives à l'erreur avec la méthode getMessage() de la classe PDOException.

```
try {  
    $db = new PDO('mysql:host='.$DATABASE_HOST.';dbname='.$DATABASE_N  
AME.';charset=utf8mb4', $DATABASE_USER, $DATABASE_PASSWORD);  
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
    $db->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);  
} catch (PDOException $e) {  
    echo  
    "Connection failed : ". $e->getMessage();  
}
```

## 1.4. Les structures conditionnelles :

La condition if est l'une des plus importantes et des plus utilisées



dans l'ensemble des langages de programmation, elle va nous permettre d'exécuter un bloc de code si et seulement si le résultat d'un test vaut true.

Ici, on utilise deux conditions if dans le fichier connexion.php, pour vérifier si le client saisie ses informations correctement, on utilisant ses données dans la table client dans notre base de données.

```
if($client === false){
    $erreur .= "Email insérée n'existe pas<br>";
} else{
    $validPassword = password_verify($password,
    $client['password']);          if($validPassword){

        $_SESSION['client_id'] = $client['id'];
        $_SESSION['client_nom'] = $client['nom'];
        $_SESSION['client_prenom'] = $client['prenom'];
        $_SESSION['client_email'] = $client['email'];
        $_SESSION['client_telephone'] = $client['telephone'];

header('Location: produits.php');
    } else{
        $erreur .= "Mot de passe est incorrect<br>";
    }
}
```

## 1.5. Boucle de parcours (Foreach) :

```
<?php foreach ($commandes as $commande) { ?>
```

```
<div class="row">
<div class="col-md-3">
<p>COMM#<?php echo $commande['id'] ?></p>
</div>
<div class="col-md-3">
<p><?php echo $commande['adresse'] ?></p>
</div>
<div class="col-md-3">
<p><?php echo $commande['ville'] ?></p>
</div>
<div class="col-md-3">
<p><?php echo $commande['codePostal'] ?></p>
</div>
</div>
<?php } ?>
```

La boucle foreach nous permettra de parcourir les tableaux.

Ici, on parcourt le tableau qui contient les commandes d'un client correspondant et les afficher.

## 1.6. Les expressions régulières et les fonctions PCRE :

Preg\_match() : Compare une regex à une chaîne de caractères

```
$pattern_password = '#^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*\W){8,}$#';
$pattern_email = "/^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,})$/";
if($row['num'] > 0)
    $erreur .= "Email déjà utilisé<br>";
else if(!preg_match($pattern_email, $email))
    $erreur .= "Email n'est pas valide<br>";

if($password != $_POST['password2'])
    $erreur .= "Mot de passe n'est pas identique<br>";
else if(!preg_match($pattern_password, $password))
    $erreur .= "Mot de passe n'est pas très fort<br>";
```

## 1.7. Application Panier :

Le fichier functions\_panier.php contient toutes les fonctions effectuées sur le panier (l'ajout, la suppression, la modification...)

```
require 'config.php';
//changerQuantite
function changerQuantite($idp, $idc,$quantite, $db)
{

    $sql = "UPDATE panier SET quantite = :quantite WHERE idProduit = :idp AND
idClient = :idc";
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':quantite', $quantite);
    $stmt->bindValue(':idp', $idp);
    $stmt->bindValue(':idc', $idc);

    if($stmt->execute())
        return true;
}
```

//Ajouter Produit dans Panier

```
function ajouterProduit($idp, $idc, $quantity, $taille, $db)
```

```

{
    if(produitExiste($idp, $idc, $db)) {
        $quantite = quantiteProduit($idp, $idc, $db);
        if(changerQuantite($idp, $idc, $quantite+$quantity, $db))
            return true;
    }
    else {
        $data = [
            'idProduit' => $idp,
            'idClient' => $idc,
            'taille'=>$taille,
            'quantite' =>$quantity,
        ];
        $sql = "INSERT INTO panier (idProduit, idClient,taille, quantite) VALUES (:idProduit, :idClient,:taille, :quantite)";
        $stat= $db->prepare($sql);
        if($stat->execute($data)) {
            return true;
        }
    }
}

// Produit Existe dans panier
function produitExiste($idp, $idc, $db)
{
    $sql = "SELECT COUNT(id) AS num FROM panier WHERE idProduit = :idp AND idClient = :idc";
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':idp', $idp);
    $stmt->bindValue(':idc', $idc);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    if($row['num'] > 0)
        return true;
    else
        return false;
}

function quantiteProduit($idp, $idc, $db)
{
    $sql = "SELECT quantite FROM panier WHERE idProduit = :idp AND idClient = :idc";
    ;
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':idp', $idp);
    $stmt->bindValue(':idc', $idc);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row['quantite'];
}

function supprimerProduit($idp, $idc, $db)
{
    $sql = "DELETE FROM panier WHERE idProduit = :idp AND idClient = :idc";
    $stat= $db->prepare($sql);
}

```

```

    $stat->bindValue(':idp', $idp);
    $stat->bindValue(':idc', $idc);
    if($stat->execute()) {
        return true;
    }
}
// Nombre Produits dans Panier
function nombreProduits($idc, $db)
{
    $sql = "SELECT COUNT(id) AS num FROM panier WHERE idClient = :idc";
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':idc', $idc);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row['num'];
}
//
function totalPrixPanier($idc, $db)
{
    $total = 0;
    $rows=lesProduitsExistent($idc, $db);
    foreach ($rows as $row) {
        $row2 = infosProduit($row['idProduit'], $db);
        $total += ($row2['prix'] * $row['quantite']);
    }
    return $total;
}

function lesProduitsExistent($idc, $db)
{
    $sql = "SELECT * FROM panier WHERE idClient = :idc";
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':idc', $idc);
    $stmt->execute();
    $produits = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $produits;
}

function infosProduit($idp, $db)
{
    $sql = "SELECT * FROM produits WHERE id = :idp";
    $stmt = $db->prepare($sql);
    $stmt->bindValue(':idp', $idp);
    $stmt->execute();
    $produit = $stmt->fetch(PDO::FETCH_ASSOC);
    return $produit;
}

function supprimerPanier($idc, $db)
{
    $sql = "DELETE FROM panier WHERE idClient = :idc";
    $stat= $db->prepare($sql);

```

```

$stat->bindValue(':idc', $idc);
if($stat->execute())
    return true;
}

```

## 1.8. Requête d'insertion d'un client :

```

$sql = "INSERT INTO clients (nom, prenom, email, password, telephone) VALUES (:nom
, :prenom, :email, :password, :telephone)";

```

## 1.9. Requête d'insertion des produits achetés par un client :

```

$sql = "INSERT INTO commandes (idClient, date, prixTotal, adresse, ville, codal)
VALUES (:idClient, :date, :prixTotal, :adresse, :ville, :codePostal)";

```

```

$sql2 = "INSERT INTO details_commandes (idCommande, idProduit, taille, quantite)
VALUES (:idCommande, :idProduit, :taille, :quantite)";

```

## 1.10. Formulaire pour laisser des commentaires :

Chaque client peut laisser des commentaires sur un produit.

```

<div class="comment-form-wrap pt-5">
    <h3 class="mb-5">Laissez un commentaire</h3>
    <?php if(!isset($_SESSION['client_id'])){ ?>
        <div class="row justify-content-center">
            <a class="btn btn-primary py-3 px-
4" href="connexion.php">S'identifier</a>
        </div>
    <?php } else { ?>
        <form action="" method="POST" class="p-5 bg-light">
            <div class="form-group">
                <label for="commentaire">Commentaire </label>
                <textarea class="form-
control" cols="30" id="commentaire" name="commentaire" rows="5"></textarea>
            </div>
            <div class="form-group">
                <input class="btn py-3 px-4 btn-
primary" name="ajouter-commentaire" type="submit" value="Poster le commentaire">
            </div>
        </form>
    <?php } ?>
</div>

```

## 1.11. Backoffice:

-\$\_SESSION : Cette superglobale est un tableau associatif qui stocke les différentes variables de sessions avec leurs noms en index du tableau et leurs valeurs en valeurs du tableau. On a créé quelques variables de sessions dans notre fichier connexion\_admin.php

:

```
$_SESSION['admin_id'] = $admin['id'];
$_SESSION['admin_nom'] = $admin['nom'];
$_SESSION['admin_prenom'] = $admin['prenom'];
$_SESSION['admin_email'] = $admin['email'];
$_SESSION['admin_telephone'] = $admin['telephone'];
```

Le fichier admin\_function.php contient toutes les fonctions effectuées sur la gestion des produits(l'ajout, la suppression, la modification, list des catégories, nombre des messages)

```
require '../inc/config.php';
//récupérer tous les catégories des produits depuis la base de données
function categories($db){
    $query1 = $db-
>prepare("SELECT DISTINCT categorie FROM produits where quantite>0");
    $query1->execute();
    $categories = $query1->fetchAll();
    return $categories;
}
//Ajouter Produit
function AjouterProduit($libelle,$prix,$quantite,$image,$categorie,$description, $db)
{
    $sql = "INSERT INTO `produits`(`libelle`, `prix`, `quantite`, `image`,`categorie`, `description`) VALUES (:libelle,:prix,:quantite,:image,:categorie,:description)";
    $stat= $db->prepare($sql);
    $data=[
        'libelle' => $libelle,
        'prix' => $prix,
        'quantite' => $quantite,
        'image'=>$image,
        'categorie'=>$categorie,
        'description' => $description
    ];
    if($stat->execute($data)) {
        return true;
    }
}
//Modifier Produit
```

```

function ModifierProduit($idp,$libelle,$prix,$quantite,$categorie,$description, $db)
{
    $sql = "UPDATE `produits` SET `libelle`=:libelle,`prix`=:prix,`quantite`=:quantite,`categorie`=:categorie,`description`=:description WHERE id=:idp";
    $stat= $db->prepare($sql);
    $data=[
        'libelle' => $libelle,
        'prix' => $prix,
        'quantite' => $quantite,
        'categorie'=>$categorie,
        'description' => $description,
        'idp'=>$idp
    ];
    if($stat->execute($data)) {
        return true;
    }
}

// supprimer produit
function SupprimerProduit($idp, $db)
{
    $sql = "DELETE FROM produits WHERE id = :idp";
    $stat= $db->prepare($sql);
    $stat->bindValue(':idp', $idp);
    if($stat->execute()) {
        $sql2 = "DELETE FROM panier WHERE idProduit = :idp";
        $stat2= $db->prepare($sql2);
        $stat2->bindValue(':idp', $idp);
        if($stat2->execute()) {
            return true;
        }
    }
}

//récupérer le nombre de message
function nombreMessages($db){
    $sql = "SELECT COUNT(id) AS num FROM messages";
    $stmt = $db->prepare($sql);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row['num'];
}

```

## 1.12. Boucle de parcours list des commandes a été confirmé (Foreach) :

```

<div class="row">
    <div class="col-md-12">
        <div class="tab-content profile-tab" id="myTabContent">

```

```

        <div class="tab-
pane fade show active" id="home" role="tabpanel" aria-labelledby="home-tab">
        <div class="row">
            <div class="col-md-3">
                <label>Référence</label>
            </div>
            <div class="col-md-3">
                <label>Date</label>
            </div>
            <div class="col-md-3">
                <label>Prix total</label>
            </div>
            <div class="col-md-3">
                <label>Nombre des produits</label>
            </div>
        </div>
        <?php foreach ($commandes as $commande) { ?>
            <div class="row">
                <div class="col-md-3">
                    <p>COMM#<?php echo $commande['id'] ?><
/p>
                    </div>
                    <div class="col-md-3">
                        <p><?php echo $commande['date'] ?></p>
                    </div>
                    <div class="col-md-3">
                        <p><?php echo $commande['prixTotal'] ?
>.00 DHs</p>
                    </div>
                    <div class="col-md-3">
                        <p><?php echo nombreProduitsCommande($
commande['id'], $db) ?></p>
                    </div>
                </div>
            <?php } ?>
        </div>

```

## 1.8. Requête d'afficher la List des commandes a était confirmé:

```
$sql = "SELECT * FROM commandes ";
```

## 1.8. Requête pour récupérer nom des client pour chaque commande

```
$sql = "SELECT nom,prenom FROM clients AS c,commandes AS cd
WHERE c.id = cd.idClient AND cd.id=:idcom";
```



# Conclusion

Ce projet consiste à concevoir un site web dynamique qui permet de réaliser le commerce électronique des produits déjà utilisés.

Nous avons présenté les différentes étapes de la réalisation de notre projet en commençant par le contexte général de projet, puis ses fonctionnalités, et après la conception, et enfin la réalisation et la description des techniques et des fonctions utilisées.

Des améliorations pourraient aussi être apportées à ce site par exemple la possibilité de modifier les informations personnelles de client et l'ajout d'une photo de profil et l'adresse de chaque commande, de vérifier l'email, l'envoi d'un SMS au client indique que sa commande a été confirmée et développer un espace pour l'administrateur pour gérer le site.

Concernant les obstacles que nous avons affrontés, on peut citer : la possibilité de changer la quantité d'un produit dans le panier, c'est pour cela que nous avons créé une fonction en JavaScript permettant de actualiser la page de panier une fois que la valeur de quantité a été changé par le client. Cette fonction envoie un ordre au PHP à travers l'url de la page pour effectuer les modifications nécessaires dans la base de données et changer la quantité ainsi le prix total du produit et par conséquent de la commande.