Dossier de projet

Phone Market

Vente de téléphones reconditionnés

Titre professionnel :

DÉVELOPPEUR WEB / WEB MOBILE

Nom : SHEIKHO

Prénom : Mohamad

Compétence couverte par le projet 3

Présentation de l'entreprise 6

Cahier des charges 6
- Les objectifs 6
- La cible 6
- Le périmètre du projet 7
- Description fonctionnelle 7
- L'arborescence du site 7
- Charte graphique 8
- Maquette design 8

Environnement de développement et organisation 10
- Technologies 10
- Workflow 10

Modélisation de la base de données 11
- Méthode MERISE 11
- Modèle conceptuel de données (MCD) 11
- Modèle Logique de Données (MLD) 12

Architecture logicielle : Backend 14
- Extraits de code significatifs 14

Intégration 26

Responsive design 27

Sécurité 30
- Sources pour rester informé 30

Faille Include 31

Injection SQL 32

Faille Upload 33

Faille XSS 34

Attaque force brute 35

Recherches Anglophones 37

Conclusion 38

Introduction :

Compétences couvertes par le projet
Le projet que je vais présenter est un site e-commerce réalisé seul, je vous présenterai les compétences référentielles et du développement.

● *Activité type 1 :*

“Développer la partie Front-end d’une application web ou web-mobile en intégrant les recommandations de sécurité. “

- Maquetter un site internet
- Réaliser une interface web statique et adaptable
- Développer une interface utilisateur web dynamique

Les compétences liées à l’activité de type 1 sont abordées au travers de :

- Réflexions apportées à l’UI/UX pour le parcours utilisateur
- La réalisation d’une maquette et d’un prototype
- L’intégration de la maquette dans le projet

● *Activité type 2 :*

“Développer la partie back-end d’une application web ou web-mobile en intégrant les recommandations de sécurité. “

- Créer une base de données
- Développer les composants d’accès aux données
- Développer le back-end d’une application web / web mobile

Les compétences liées à l’activité de type 2 sont abordées au travers de :

- La conception et la modélisation de base de données utilisant la méthode MERISE.
- L’utilisation de la POO permettant l’interaction avec les données présentes en BDD pour l’utilisateur

Résumé du dossier de projet :

Ce dossier présente entièrement la partie du travail que j'ai effectué, à savoir la création du site web Phone Market, une boutique en ligne proposant des produits des marques (Apple, Samsung, Xiaomi) reconditionnés.

Le site web de Phone Market a pour objectif de permettre à ses utilisateurs d'avoir accès aux produits et les différentes catégories de produits, commander différents produits, pouvoir contacter le fournisseur, elle permet aussi aux utilisateurs de s'inscrire, se connecter, avoir accès à son historique de Commande.

Le site pourra être utilisé par les utilisateurs inscrits ou non mais uniquement les utilisateurs inscrits pourront commander.
Il y a également une partie administrateur qui sert au management du site, de ses produits, des offres misent en avant et un suivi d'activité et des commandes.
Le site devrait donc être composé de plusieurs pages web statiques et dynamiques et selon le rôle attribué à l'utilisateur il pourra accéder à différentes pages.

Si la personne connectée à un rôle d'utilisateur normale il aura accès à ces pages suivantes :

- Une page connexion qui servirait également de page d'accueil,
- Une page d'inscription
- Une page profil récapitulant les informations de l'utilisateur et lui permettant aussi de les modifier,
- Une page pour les produits et catégorie
- Une page infos-contact accessible depuis le bouton en savoir plus qui permet-elle de visualiser les informations du contact qui fait le lien entre l'entreprise partenaire et l'agence,
- Une page pour le panier

Ce site web comptant comme l’un des deux plus grands projets de l’année pour le passage du titre professionnel de développeur web et web mobile, a été réalisé Seul au cours de ma formation de développeur web à l’école La Plateforme à Marseille.

Cahier des charges

Présentation de l'entreprise

C'est une entreprise fictive que j'ai dû créer pour la réalisation de ce projet de site e-commerce.
J'ai décidé de choisir comme domaine d'activité la vente de téléphone toute catégorie confondue mais avec une volonté plus écologique portée sur la vente de reconditionné et non le changement systématique par l'acquisition d'un nouvel appareil.
En effet, en achetant un appareil **reconditionné** on lutte contre l'obsolescence programmée et on favorise le recyclage de produit encore en état de fonctionnement. Car en reconditionnant un appareil on répare le **téléphone** en remplaçant les pièces défectueuses par des pièces neuves de qualité.

Les besoins clients

L'objectif

L'objectif est de mettre en place une marketplace permettant à des particuliers ou entreprises de bénéficier de produits à prix réduits et de manière plus responsable et écologique, avec la vente de produits uniquement reconditionnées.
Prévue pour être utilisée uniquement sur le web, un travail en amont sera réalisé pour que le site soit compatible sur tous les types d'appareils que ce soit mobile, tablette ou ordinateur.

Les cibles

Sur le site il y aura principalement trois types de cible :
Les visiteurs : Ce sont les utilisateurs non-inscrits sur le site, qui pourront consulter les produits et potentiellement devenir des nouveaux inscrits, voir qui nous sommes, prendre contact avec le fournisseur.

Les inscrits : ce sont les utilisateurs qui pourront consulter les produits, voir qui somme nous, prendre contact avec le fournisseur, avoir accès au panier, prendre commande auprès du site et avoir un historique des commandes passées.

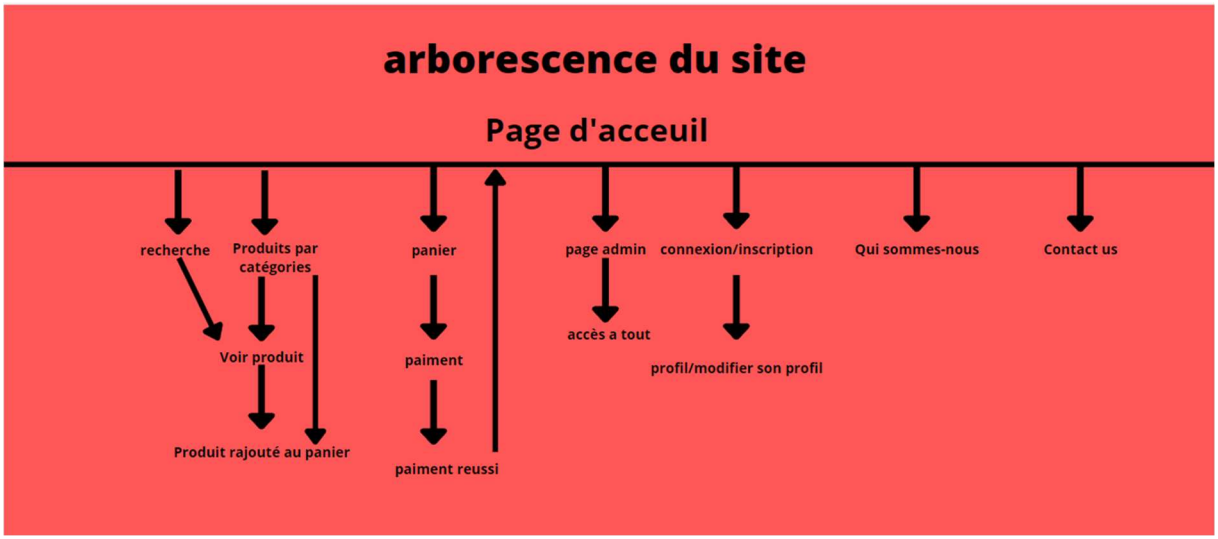
Les administrateurs : ce qui comprend les utilisateurs avec des droit d'accès spéciaux au site pour le manager, c'est-à-dire avoir un suivi des commandes réalisées, changer les produits mis en avant sur le site, ajouter, supprimer, modifier un produit, changer les droits d'accès d'un utilisateur.

Le périmètre du projet

Le site sera disponible uniquement en français.
Par ailleurs, il devra être adapté à tous les supports (ordinateurs, tablettes, mobiles) pour permettre à tous les utilisateurs d'avoir une expérience de navigation optimale.

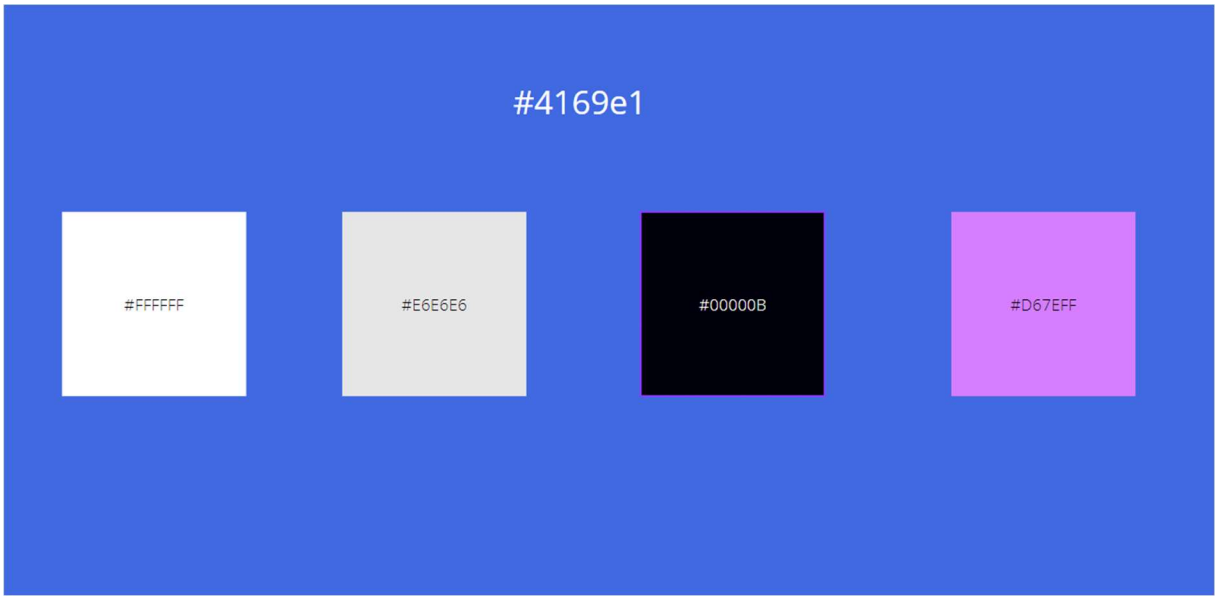
L'arborescence

J'ai utilisé la méthode « story », définir le parcours de l'utilisateur, c'est-à-dire identifier comment se déroule le parcours et toutes les possibilités d'actions que l'utilisateur aura sur le site, je l'ai modélisé sous forme d'arborescence.



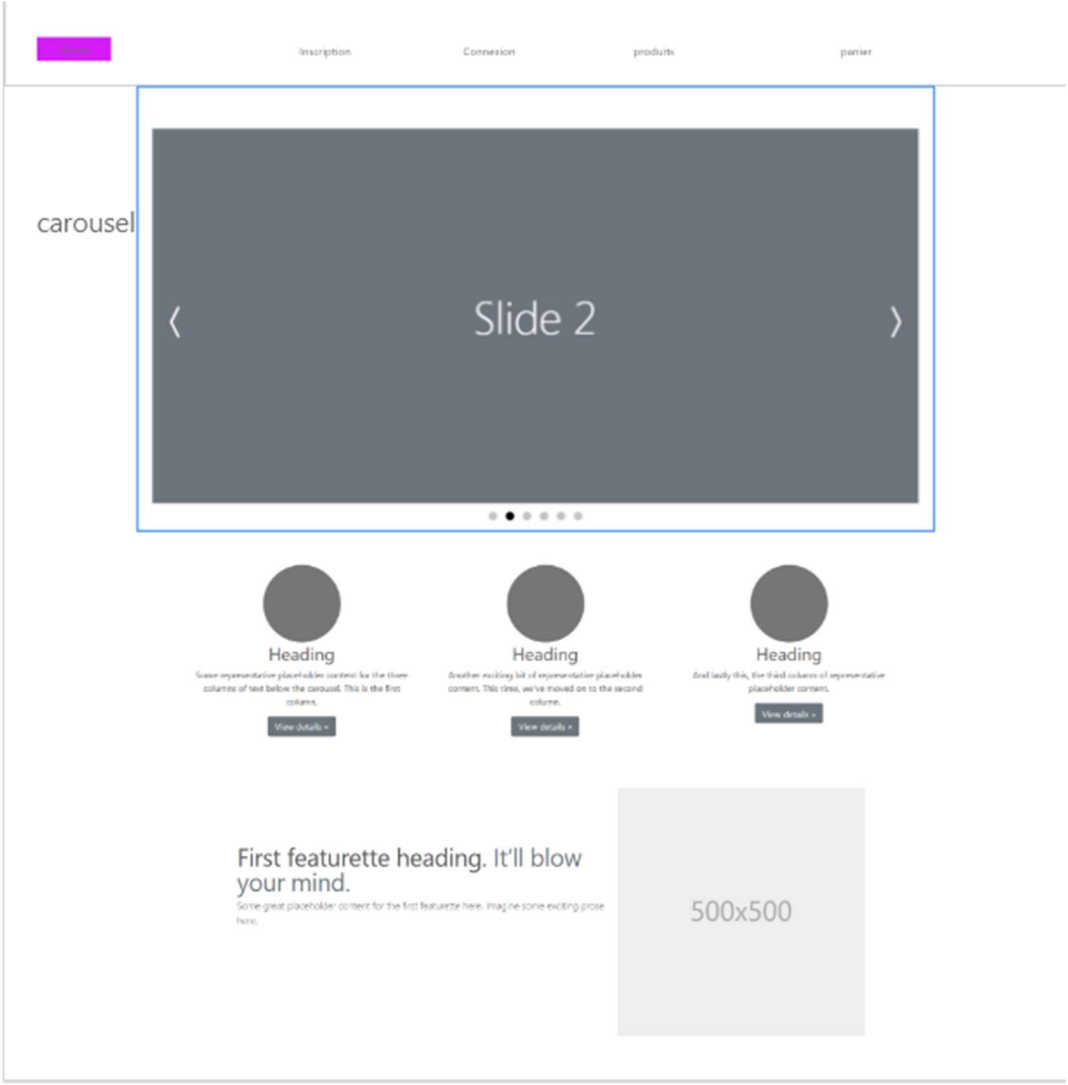
La charte graphique

Pour ce qu'il s'agit de la charte graphique, je défini entre autres les couleurs et les polices utilisées pour la typographie, pour que le site ait une identité visuelle, qu'il soit attrayant visuellement pour l'utilisateur. Puisque, l'utilisateur ne voit et ne se fie qu'à l'apparence du site et ne peut pas voir le Back-office.



Pour ce qu'il s'agit de la typographie, j'ai décidé de prendre la police Poppins de Google Font, son esthétique donne une écriture élégante, simple, mais à la forte personnalité qui est épurée et pas trop tape à l'œil pour que la police soit passe partout et qu'elle passe avec du flat design. Elle propose une vaste gamme de 9 gras.

J’ai appris à faire un wireframe pour visualiser comment seront agencé les éléments sur les différentes pages, j’ai choisi un design minimaliste, qui est très en vogue actuellement dans l’univers du web.



Spécificités techniques : Backend

Environnement de développement et organisation

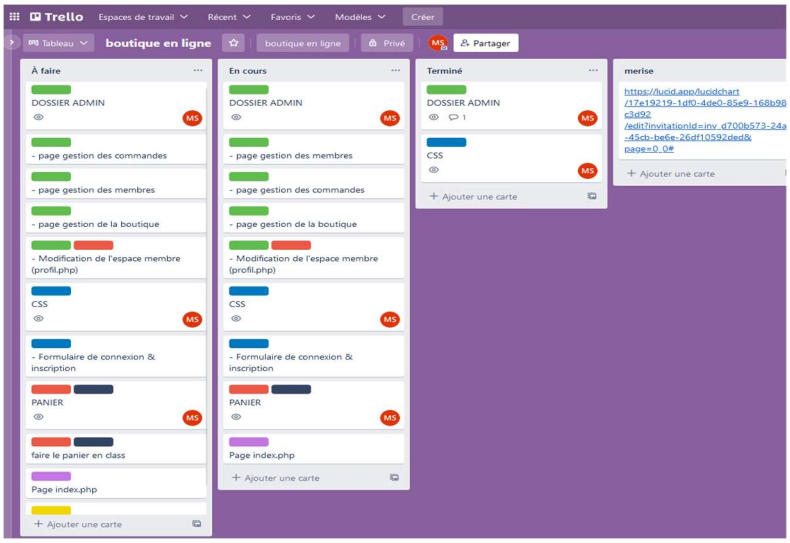
Technologies :

Pour mener à bien le développement de ce projet, j'ai opté pour un environnement de développement sous la stack technique XAMPP.
XAMPP : est un ensemble de logiciels permettant de mettre en place un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres (**X** (cross) **A**pache **M**ariaDB **P**erl **P**HP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

Workflow :

Pour la gestion de projet et le travail que j'ai effectué, j'ai utilisé divers outils permettant de m'organiser et répartir les tâches de développement.

Pour la gestion de projet j'ai utilisées tous les outils mis à disposition par G-suite le service de Google, le drive pour la documentation, j'ai utilisé Trello afin de déterminé ce qui est à faire, en cours, terminé...etc.



En ce qui concerne la gestion des modifications apportées,
J'ai utilisé le logiciel Git Kraken qui permet aux développeurs de stocker et de partager, publiquement ou non, le code qu'ils créent.
Le travail sur GitHub se fait autour d'une seule branche principale, la branche "Master".

Modélisation de la base de données

Méthode MERISE

MERISE (Méthode d'Etude et de Réalisation Informatique pour les Systèmes d'Entreprise), c'est une méthode française qui a émergée dans les années 70 en France qui permet la modélisation et la conception de S.I. Parmi les ressources informatiques de ces S.I., il y a en particulier les fichiers de données, bases de données et système de gestion de bases de données (S.G.B.D.).
C'est sur ce dernier point que la méthode MERISE m'a été utile, car c'est en se basant sur ses principes de modélisation que j'ai conçu la base de données de Phone Market.

Modèle conceptuel de données (MCD)

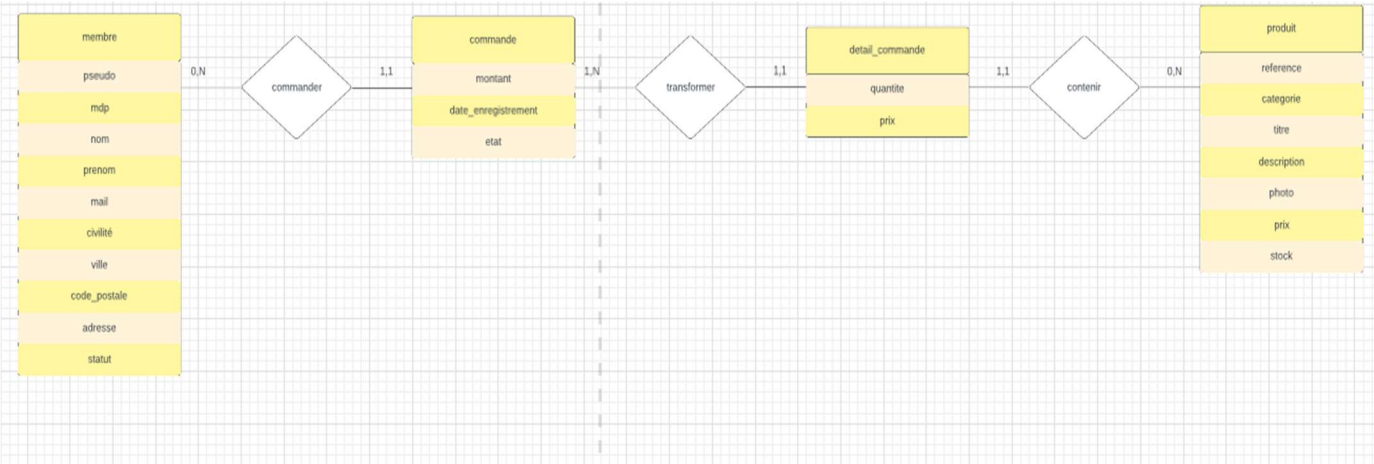
C'est un modèle simplifié de la base de données, ou les tables sont seulement au stade d'entités (c'est l'équivalent de la table en MCD), entre les différentes entités il y a des liaisons que l'on appelle association, qui est le verbe d'action qui décrit l'opération qui se fait entre les deux entités, donc nous distinguons principalement les entités et les associations, d'où son second nom de schéma Entité/Association.

D'abord il a fallu imaginer tous les besoins des futurs utilisateurs et admin de Phone Market. A partir de ces besoins, j'ai été en mesure d'établir les règles de gestion des données à conserver.

Ensuite il faut définir le dictionnaire des données, c'est-à-dire toutes les données élémentaires qui vont être conservées en base de données et définir certaines caractéristiques qui figureront dans le MCD. Parmi ces caractéristiques, on retrouve par exemple la référence d'une donnée et notamment un identifiant unique, sa désignation, son type etc…

Étape finale à sa conception, j'ai pu à partir des informations Précédemment recueillies, créer chaque entité, unique et décrite par un Ensemble de propriétés ; Et leur association permettant de définir les liens et cardinalités entre les entités.

Le MCD de Phone Market



Modèle Logique de données (MLD)

Le modèle logique de données (MLD), est une étape intermédiaire entre le modèle conceptuel de données et le modèle physique de données.

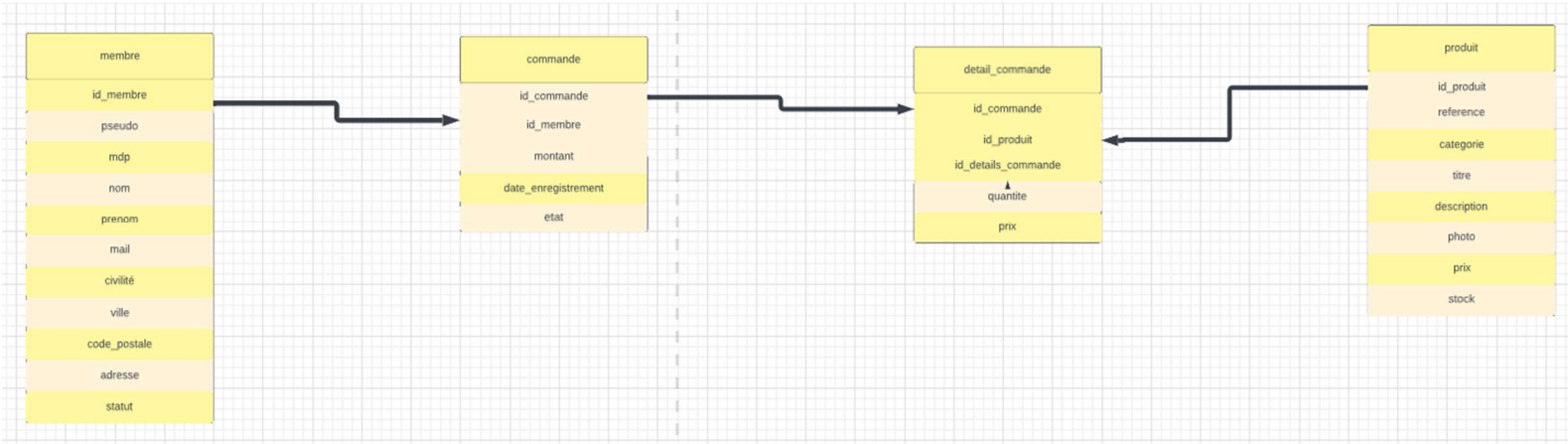
Le passage d'un MCD en MLD s'effectue selon quelques règles de conversion précise :

Une entité du MCD devient une table. Dans un SGBD de type relationnel, une table est une structure tabulaire dont chaque ligne correspond aux données d'un objet enregistré et où chaque colonne correspond à une propriété de cet objet.
Ces colonnes font notamment référence aux caractéristiques définies dans le dictionnaire de données du MCD.

Les identifiants respectifs de chaque entité deviennent des clefs primaires et toutes les autres propriétés définies dans le MCD deviennent des attributs. Les clefs primaires permettent d'identifier de façon unique chaque enregistrement dans une table et ne peuvent pas avoir de valeur nulle.

Les cardinalités de type "0:n" / "1:n" sont représentées à travers le référencement de la clef primaire de la table qui la possède, en clef étrangère au sein de la table auquel elle est liée. Si deux tables liées possèdent une cardinalité de type "0:n / 1:n", la relation sera traduite par la création d'une table de jonction, dont la clef primaire de chaque table deviendra une clef étrangère au sein de la table de jonction.

Le MLD de Phone Market



Ici, nous voyons bien que les clés primaires de chaque table associée deviennent des clés étrangères dans chacune des tables qui reçoivent les données.

Extrait de code significatif

Toutes les requêtes en base de données sont gérées par la classe boutique. Cette classe possède des méthodes permettant d'exécuter des requêtes préparées.
L'utilisation de requêtes préparées, rendues possibles grâce à l'extension PDO, présentent un double avantage par rapport à l'exécution directe de requêtes SQL. Premièrement, peu importe le nombre d'exécution des requêtes, nous n'avons besoin de les préparer qu'une seule fois pour qu'elles soient réutilisables à volonté avec des paramètres identiques ou différents.
En un second temps, elles présentent un avantage majeur en termes de sécurité notamment pour prévenir des injections SQL.
Nos requêtes étant préformatées, elles empêchent le passage de code malicieux en paramètre, nous n'avons donc pas besoin de protéger nos paramètres ou valeurs manuellement.

Dans un premier temps, je commence par créer ma classe et y ajouter des attributs si on le souhaite.

```php
<?php

class boutique
{
private $_link;
private $_id;
private $_id_utilisateur;
private $_id_sujet;
public $_login;
public $_password;
```

Un attribut est une variable qui est partagée par toutes les instances de la classe, on peut y stocker par exemples les paramètres de la connexion à la base de données pour ne pas avoir à les recopier à chaque fois que l'on a besoin de faire un appel à la base de données.

Pour que je puisse faire appel à mes fonctions sur cette page, il faut que je J'inclus ma page de fonction dans la page ou je souhaite utiliser ces fonctions et ensuite instancier la classe que je veux instancier, cela ce fait de cette manière :

```php
<?php
session_start();
include("../function/fonctions.php");
$panier = new panier();
$panier->dbconnect();
?>
```

Concernant la connexion à la base de données j'ai procédé de la manière suivante.

J'ai codé une class **connexionDB** qui se connecte à la base de données dans _CONSTRUCT directement, j'ai utilisé un "**try and catch**" la fonction php qui gère les erreurs. La gestion d'une erreur via une exception se fait en deux temps.

On va utiliser un bloc try dans lequel le code qui peut potentiellement retourner une **erreur** va être exécuté. On crée à l'intérieur une nouvelle connexion grâce à l'objet **new PDO**.

On va créer un bloc catch dont le but va être d'attraper l'exception si celle-ci a été lancée et de définir la façon dont doit être gérée l'erreur. La fonction est appelée dans les autres classes afin de réaliser différentes requêtes.

Grâce au principe d'héritage de la programmation orientée objet de PHP, je fais hériter cette class connexionDB à toutes mes autres classes qui ont

besoin d'une connexion à la base de données pour ne pas répéter mon code, et pouvoir réaliser différentes requêtes.

Extrait du code permettant la connexion à la base de données

```php
<?php
  // Déclaration d'une nouvelle classe
  class connexionDB {
    private $host    = 'localhost';   // nom de l'host
    private $name    = 'boutique-en-ligne';     // nom de la base de donnée
    private $user    = 'root';        // utilisateur
    private $pass    = '';        // mot de passe
    //private $pass    = '';          // Ne rien mettre si on est sous windows
    private $connexion;

    function __construct($host = null, $name = null, $user = null, $pass = null){
      if($host != null){
        $this->host = $host;
        $this->name = $name;
        $this->user = $user;
        $this->pass = $pass;
      }
      try{
        $this->connexion = new PDO('mysql:host=' . $this->host . ';dbname=' . $this->name,
          $this->user, $this->pass, array(PDO::MYSQL_ATTR_INIT_COMMAND =>'SET NAMES UTF8',
          PDO::ATTR_ERRMODE => PDO::ERRMODE_WARNING));
      }catch (PDOException $e){
        echo 'Erreur : Impossible de se connecter  à la BDD !';
        die();
      }
    }

    public function query($sql, $data = array()){
      $req = $this->connexion->prepare($sql);
      $req->execute($data);
      return $req;
    }

    public function insert($sql, $data = array()){
      $req = $this->connexion->prepare($sql);
      $req->execute($data);
    }
  }

  // Faire une connexion à votre fonction
  $DB = new connexionDB();
?>
```

Après avoir fait cela, on peut s'attaquer à l'écriture de toutes nos fonctions. Les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données est nommée le CRUD (pour *Create*, *Read*, *Update*, *Delete*).

Je vais vous présenter l'ensemble des requêtes couvertes par le CRUD :

En ce qui concerne le Create, cette opération est gérée par la requête "INSERT", pour envoyer des éléments en base de données.
Dans le cadre de ma boutique en ligne, je m'en suis servi pour envoyer en base de données les articles que je souhaite présenter.

Je commence déjà par récupérer les données en méthode 'POST' dans mon formulaire ci-dessous :

Ici le code HTML de mon formulaire :

```php
<h1> Formulaire Produits </h1>
<form method="post" enctype="multipart/form-data" action="">

    <input type="hidden" id="id_produit" name="id_produit" value="'; if(isset($produit_actuel['id_produit'])) echo $produit_actuel['id_produit']; echo '">

    <label for="reference">reference</label><br>
    <input type="text" id="reference" name="reference" placeholder="la référence de produit" value="'; if(isset($produit_actuel['reference'])) echo $produit_actuel['reference']; echo

    <label for="categorie">categorie</label><br>
    <input type="text" id="categorie" name="categorie" placeholder="la categorie de produit" value="'; if(isset($produit_actuel['categorie'])) echo $produit_actuel['categorie']; echo

    <label for="titre">titre</label><br>
    <input type="text" id="titre" name="titre" placeholder="le titre du produit" value="'; if(isset($produit_actuel['titre'])) echo $produit_actuel['titre']; echo '" > <br><br>

    <label for="description">description</label><br>
    <textarea name="description" id="description" placeholder="la description du produit">'; if(isset($produit_actuel['description'])) echo $produit_actuel['description']; echo '</te

    <label for="photo">photo</label><br>
    <input type="file" id="photo" name="photo"><br><br>';
    if(isset($produit_actuel))
    {
        echo '<i>Vous pouvez uplaoder une nouvelle photo si vous souhaitez la changer</i><br>';
        echo '<img src="' . $produit_actuel['photo'] . '"  ="30" height="30"><br>';
        echo '<input type="hidden" name="photo_actuelle" value="' . $produit_actuel['photo'] . '"><br>';
    }

    echo '
    <label for="prix">prix</label><br>
    <input type="text" id="prix" name="prix" placeholder="le prix du produit"  value="'; if(isset($produit_actuel['prix'])) echo $produit_actuel['prix']; echo '"><br><br>

    <label for="stock">stock</label><br>
    <input type="text" id="stock" name="stock" placeholder="le stock du produit"  value="'; if(isset($produit_actuel['stock'])) echo $produit_actuel['stock']; echo '"><br><br>

    <input type="submit" value="'; echo ucfirst($_GET['action']) . ' du produit">
</form>';
}
require_once("../inc/footer.inc.php");

?>
```

Donc pour envoyer des données en BDD, la méthode est la suivante, je commence par vérifier que Cette partie ne doit être accessible que par un internaute connecté ayant son statut fixé à 1 (autrement dis : un administrateur). Je vais utiliser la fonction utilisateurEstConnecteEtEstAdmin puisque nous avions prévu un code permettant de renvoyer TRUE (vrai tu es admin) ou FALSE (faux tu n'es pas admin).

```php
//--- VERIFICATION ADMIN ---//
if(!internauteEstConnecteEtEstAdmin())
{
    header("location:../view/connexion.php");
    exit();
}

//--- SUPPRESSION PRODUIT ---//
if(isset($_GET['action']) && $_GET['action'] == "suppression")
{   // $contenu .= $_GET['id_produit']
    $resultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
    $produit_a_supprimer = $resultat->fetch_assoc();
    $chemin_photo_a_supprimer = $_SERVER['DOCUMENT_ROOT'] . $produit_a_supprimer['photo'];
    if(!empty($produit_a_supprimer['photo']) && file_exists($chemin_photo_a_supprimer)) unlink($chemin_photo_a_supprimer);
    $contenu .= '<div class="validation">Suppression du produit : ' . $_GET['id_produit'] . '</div>';
    executeRequete("DELETE FROM produit WHERE id_produit=$_GET[id_produit]");
    $_GET['action'] = 'affichage';
}
//--- ENREGISTREMENT PRODUIT ---//
if(!empty($_POST))
{   // debug($_POST);
    $photo_bdd = "";
    if(isset($_GET['action']) && $_GET['action'] == 'modification')
    {
        $photo_bdd = $_POST['photo_actuelle'];
    }
    if(!empty($_FILES['photo']['name']))
    {   // debug($_FILES);
        $nom_photo = $_POST['reference'] . '_' . $_FILES['photo']['name'];
        $photo_bdd = RACINE_SITE . "photo/$nom_photo";
        $photo_dossier = $_SERVER['DOCUMENT_ROOT'] . RACINE_SITE . "/photo/$nom_photo";
        copy($_FILES['photo']['tmp_name'],$photo_dossier);
    }
    foreach($_POST as $indice => $valeur)
    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("REPLACE INTO produit (id_produit, reference, categorie, titre, description, photo, prix, stock) values
    ('$_POST[id_produit]', '$_POST[reference]', '$_POST[categorie]', '$_POST[titre]', '$_POST[description]','$photo_bdd', '$_POST[prix]', '$_POST[stock]')");
    $contenu .= '<div class="validation">Le produit a été ajouté</div>';
    $_GET['action'] = 'affichage';
}
```

Nous faisons évoluer la requête SQL afin que celle-ci devienne REPLACE.

Le traitement PHP pour l'ajout de produit

Nous faisons évoluer la requête SQL afin que celle-ci devienne REPLACE.
REPLACE se comporte en INSERT (ajout) s'il n'y a pas d'id_produit connu

Nous prévoyons un traitement PHP si le contenu du POST n'est pas vide if(!empty($_POST)) (autrement dis s'il a été rempli par un clic sur le bouton submit).

Nous initialisons une variable $photo_bdd à vide pour éviter plus tard une erreur undefined si aucune photo n'est ajoutée.

Un fichier transmis par upload ne se récupère pas avec $_POST mais avec $_FILES (il s'agit d'une autre superglobale).

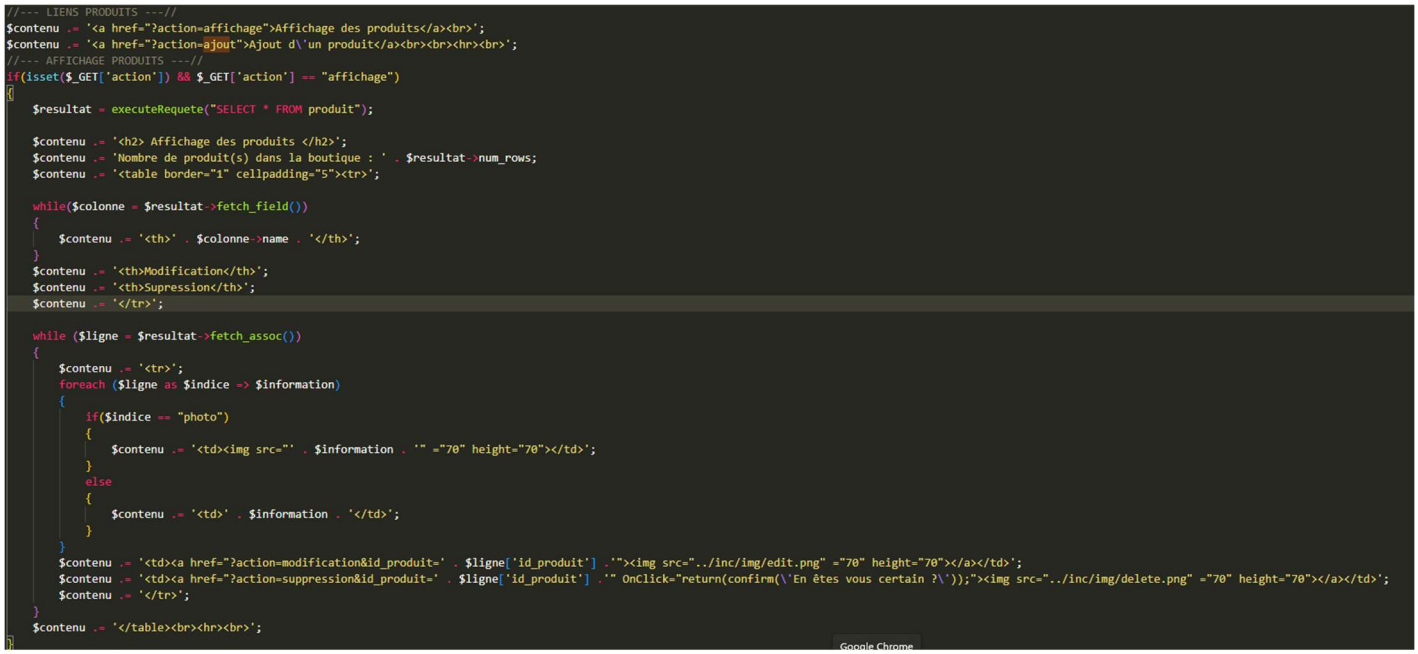
Si une photo a été uplaodée if(!empty($_FILES['photo']['name'])), nous changeons le nom de la photo $nom_photo = $_POST['reference'] . '_' .$_FILES['photo']['name']; car par défaut 2 photos du même nom s'écrasent et se remplacent.
Il est nécessaire que le dossier /photo/ soit existant sur le serveur.

Il y a une sauvegarde du chemin de la photo dans la base de données, et une sauvegarde physique du fichier photo sur le serveur dans le dossier prévu à cet effet.

Dans l'idéal, il serait intéressant d'engager plusieurs traitements sur une photo uplaodée :

- Nom : donner 1 nom plus cohérent
- Dimension : faire attention à ce qu'on ne vous envoie pas des photos miniatures ou au contraire une taille trop grande
- Poids : des poids trop volumineux peuvent encombrer votre serveur
- Extension : attention, on peut très bien vous envoyer des .exe ou des fichiers d'attaques) (même si effectivement l'administrateur n'est pas censé pirater son propre site).

J’ai prévu 2 liens permettant soit d'afficher les produits, soit d'ajouter 1 produit.



```php
//--- LIENS PRODUITS ---//
$contenu .= '<a href="?action=affichage">Affichage des produits</a><br>';
$contenu .= '<a href="?action=ajout">Ajout d\'un produit</a><br><br><hr><br>';
//--- AFFICHAGE PRODUITS ---//
if(isset($_GET['action']) && $_GET['action'] == "affichage")
{
    $resultat = executeRequete("SELECT * FROM produit");

    $contenu .= '<h2> Affichage des produits </h2>';
    $contenu .= 'Nombre de produit(s) dans la boutique : ' . $resultat->num_rows;
    $contenu .= '<table border="1" cellpadding="5"><tr>';

    while($colonne = $resultat->fetch_field())
    {
        $contenu .= '<th>' . $colonne->name . '</th>';
    }
    $contenu .= '<th>Modification</th>';
    $contenu .= '<th>Supression</th>';
    $contenu .= '</tr>';

    while ($ligne = $resultat->fetch_assoc())
    {
        $contenu .= '<tr>';
        foreach ($ligne as $indice => $information)
        {
            if($indice == "photo")
            {
                $contenu .= '<td><img src="' . $information . '" ="70" height="70"></td>';
            }
            else
            {
                $contenu .= '<td>' . $information . '</td>';
            }
        }
        $contenu .= '<td><a href="?action=modification&id_produit=' . $ligne['id_produit'] .'"><img src="../inc/img/edit.png" ="70" height="70"></a></td>';
        $contenu .= '<td><a href="?action=suppression&id_produit=' . $ligne['id_produit'] .'" OnClick="return(confirm(\'En êtes vous certain ?\'));"><img src="../inc/img/delete.png" ="70" height="70"></a></td>';
        $contenu .= '</tr>';
    }
    $contenu .= '</table><br><hr><br>';
}
```

Techniquement, cela permet de passer dans l'url **?action=affichage** ou **?action=ajout** et donc en récupérant l'information véhiculer dans l'url (via $_GET), le site peut détecter l'action à déclencher.

L'administrateur peut donc choisir l'action qu'il souhaite mettre en œuvre : ajout ou affichage.

L'affichage des produits se fait dans 1 table (1 tableau).

J’ajoute 2 liens (représentés d'images pour l'occasion) afin de proposer la modification et la suppression de produits pour l'administrateur (le commerçant).

Pour la modification et la suppression, nous passerons par l'action par l'url ainsi que l'id du produit correspondant.

L'action nous permettra de savoir que nous devons supprimer/modifier un produit et l'id nous permettra de savoir du quel il s'agit.

Voici l’affichage des produits ajoutés :

Affichage des produits

Nombre de produit(s) dans la boutique : 14

id_produit	reference	categorie	titre	description	photo	prix	stock	Modification	Supression
13	04-AA-04	Apple	iphone 12	samsung s22		800	50		
14	05-AA-05	Apple	iphone 11	iphone 11		800	20		
15	01-AA-01	Apple	iphone 8	iphone 8		500	5		
16	06-AA-06	Apple	iphone 8 plus	iphone 8 plus		600	9		
17	07-AA-07	Apple	Iphone 7	Iphone 7		500	15		
18	08-AA-08	Apple	Iphone 7 plus	Iphone 7 plus		550	54		
19	09-AA-09	Apple	Iphone 12 mini	Iphone 12 mini 256Go		700	10		

Accueil / Iphone / Iphone 8 plus

Iphone 8 plus

$189.00

Iphone 8 plus 128GO

1 AJOUTER AU PANIER

Catégorie : Iphone

Description Avis (0)

Tous les produits vendus sur Phone Market proviennent de reconditionneurs experts et vérifiés, qui s'engagent à tester chaque appareil selon notre charte qualité. Chaque produit est 100% fonctionnel, parfaitement nettoyé et garanti.

En ce qui concerne “l’update”, cette opération est gérée par la requête qui porte le même nom UPDATE, pour modifier des éléments en base de données.
Dans le cadre de ma boutique en ligne, je m’en suis servi pour donner la possibilité à chaque utilisateur de mettre à jour leurs information personnelles (là en l'occurrence leurs nom et prénom) présents en base de données.

Je commence par récupérer les données de l’utilisateurs

```php
// On récupère les informations de l'utilisateur connecté
$afficher_profil = $DB->query("SELECT *  FROM utilisateur  WHERE id = ?", array($_SESSION['id']));
$afficher_profil = $afficher_profil->fetch();

if(!empty($_POST)){
    extract($_POST);
    $valid = true;

    if (isset($_POST['modification'])){
        $nom = htmlentities(trim($nom));
        $prenom = htmlentities(trim($prenom));
        $mail = htmlentities(strtolower(trim($mail)));

        if(empty($nom)){
            $valid = false;
            $er_nom = "Il faut mettre un nom";
        }

        if(empty($prenom)){
            $valid = false;
            $er_prenom = "Il faut mettre un prénom";
        }

        if(empty($mail)){
            $valid = false;
            $er_mail = "Il faut mettre un mail";

        }elseif(!preg_match("/^[a-z0-9\-_.]+@[a-z]+\.[a-z]{2,3}$/i", $mail)){
            $valid = false;
            $er_mail = "Le mail n'est pas valide";

        }else{
            $req_mail = $DB->query("SELECT mail FROM utilisateur   WHERE mail = ?",
                array($mail));
            $req_mail = $req_mail->fetch();

            if ($req_mail['mail'] <> "" && $_SESSION['mail'] != $req_mail['mail']){
                $valid = false;
                $er_mail = "Ce mail existe déjà";
            }
```

Je commence par récupérer l'attribut qui contient la connexion à la BDD pour que je puisse communiquer avec et lui donner des instructions, après avoir fait ça je récupère l'id de l'utilisateur connecté qui est stocké dans une variable dite superglobale ce qui signifie qu'elles sont disponibles quel que soit le contexte du script, la en l'occurrence c'est la variable "$_SESSION" qui est utilisée pour stocker des tableaux associatifs.

Après avoir fait ça je rédige ma requête préparée, je dis de mettre à jour le nom, prénom et le mail. Et je lui dis par quelles valeurs les remplacés et quel élément ciblé avec l'id, après avoir fait ça je l'exécute et je remplace les valeurs stockés ma variable superglobale par les nouvelles que l'utilisateur à entrer tout en transférant l'utilisateur sur la page profil.php, avec un message comme quoi les informations ont bien été mises à jour.

L'utilisateur pourra ensuite voir sur son profil que les informations ont bien été modifiées.

En ce qui concerne le "DELETE", cette opération est gérée par la requête qui porte le même nom DELETE, pour supprimer des éléments en base de données.
Dans le cadre de ma boutique en ligne, je m'en suis servi pour donner la possibilité à chaque administrateur de supprimer un ou des produits en base de données pour qu'ils ne soient plus présents sur la boutique Phone Market.
Je commence par rédiger la fonction qui est destinée à supprimer ces données en BDD.

```php
//--- SUPPRESSION PRODUIT ---//
if(isset($_GET['action']) && $_GET['action'] == "suppression")
{   // $contenu .= $_GET['id_produit']
    $resultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
    $produit_a_supprimer = $resultat->fetch_assoc();
    $chemin_photo_a_supprimer = $_SERVER['DOCUMENT_ROOT'] . $produit_a_supprimer['photo'];
    if(!empty($produit_a_supprimer['photo']) && file_exists($chemin_photo_a_supprimer)) unlink($chemin_photo_a_supprimer);
    $contenu .= '<div class="validation">Suppression du produit : ' . $_GET['id_produit'] . '</div>';
    executeRequete("DELETE FROM produit WHERE id_produit=$_GET[id_produit]");
    $_GET['action'] = 'affichage';
}
```

Donc pour le commerçant (administrateur) peut ajouter et observer ses produits mais pour lui offrir une gestion complète il est important de lui proposer les options de suppression et de modification de produit.

Nous détectons l'action suppression dans l'url if(isset($_GET['action']) && $_GET['action'] == "suppression").

Nous allons d'abord sélectionner toutes les informations sur ce produit dans la base de donnée, avec notamment le chemin vers la photo afin de la supprimer de notre serveur (puisque nous ne gérons pas l'attribution d'une même image pour plusieurs produits).

Nous formulons une 2e requête pour supprimer réellement le produit de notre base de donnée et nous rebasculons vers l'action d'affichage pour observer tous les produits.

Nous devons également proposer la modification de produits, pour cela nous aurons besoin d'un formulaire permettant d'accueillir les données en vue d'une modification.
Pour factoriser le code (factoriser = réduire le code, comme ici on ré-utilise du code pour plusieurs choses différentes), nous pouvons nous servir d'un seul formulaire pour l'ajout et la modification.

Et oui, qu'on ajoute ou que l'on modifie, il s'agit des mêmes champs.

Seule différence, lorsque l'on modifie il nous faudra les champs de formulaire pré-saisis avec les informations actuelles.

```php
if(!empty($_POST))
{   // debug($_POST);
    $photo_bdd = "";
    if(isset($_GET['action']) && $_GET['action'] == 'modification')
    {
        $photo_bdd = $_POST['photo_actuelle'];
    }
```

```php
 executeRequete("REPLACE INTO produit (id_produit, reference, categorie,
titre, description, photo, prix, stock) values
    ('$_POST[id_produit]', '$_POST[reference]', '$_POST[categorie]',
'$_POST[titre]',
'$_POST[description]','$photo_bdd',  '$_POST[prix]',  '$_POST[stock]')");
```

Mon formulaire html :

```php
if(isset($_GET['action']) && ($_GET['action'] == 'ajout' || $_GET['action'] == 'modification'))
{
    if(isset($_GET['id_produit']))
    {
        $resultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
        $produit_actuel = $resultat->fetch_assoc();
    }

    echo '
    <h1> Formulaire Produits </h1>
    <form method="post" enctype="multipart/form-data" action="">

        <input type="hidden" id="id_produit" name="id_produit" value="'; if(isset($produit_actuel['id_produit'])) echo $produit_actuel['id_produit']; echo '">

        <label for="reference">reference</label><br>
        <input type="text" id="reference" name="reference" placeholder="la référence de produit" value="'; if(isset($produit_actuel['reference'])) echo $produit_actuel['reference']; echo '"><br><br>

        <label for="categorie">categorie</label><br>
        <input type="text" id="categorie" name="categorie" placeholder="la categorie de produit" value="'; if(isset($produit_actuel['categorie'])) echo $produit_actuel['categorie']; echo '" ><br><br>

        <label for="titre">titre</label><br>
        <input type="text" id="titre" name="titre" placeholder="le titre du produit" value="'; if(isset($produit_actuel['titre'])) echo $produit_actuel['titre']; echo '" > <br><br>

        <label for="description">description</label><br>
        <textarea name="description" id="description" placeholder="la description du produit">'; if(isset($produit_actuel['description'])) echo $produit_actuel['description']; echo '</textarea><br><br>

        <label for="photo">photo</label><br>
        <input type="file" id="photo" name="photo"><br><br>';
        if(isset($produit_actuel))
        {
            echo '<i>Vous pouvez uplaoder une nouvelle photo si vous souhaitez la changer</i><br>';
            echo '<img src="' . $produit_actuel['photo'] . '"  ="30" height="30"><br>';
            echo '<input type="hidden" name="photo_actuelle" value="' . $produit_actuel['photo'] . '"><br>';
        }

        echo '
        <label for="prix">prix</label><br>
        <input type="text" id="prix" name="prix" placeholder="le prix du produit"  value="'; if(isset($produit_actuel['prix'])) echo $produit_actuel['prix']; echo '"><br><br>

        <label for="stock">stock</label><br>
        <input type="text" id="stock" name="stock" placeholder="le stock du produit"  value="'; if(isset($produit_actuel['stock'])) echo $produit_actuel['stock']; echo '"><br><br>

        <input type="submit" value="'; echo ucfirst($_GET['action']) . ' du produit">
    </form>';
}
```

Je demande à ce que le formulaire produit s'affiche également en cas de modification.

Je recupere les informations actuelles d'un produit pour les disposer et les pré-saisir dans le formulaire afin que le commerçant puisse modifier seulement ce qu'il souhaite (sans pour autant perdre les autres informations).

On utilise l'action se trouvant dans l'url ($_GET) pour définir le texte du bouton submit.

Je crée un champ hidden (caché) pour transmettre l'id du produit devant être modifié, cela n'aura pas d'impact en cas d'ajout car ce champ n'enverra pas d'id (puisque le produit n'existera pas encore au moment de la création).

En ce qui concerne la photo, on la recupere dans cet ordre : la photo actuelle du produit, et ensuite la photo qui aura été uplaodée (si elle a été uplaodé) pour écraser la photo actuelle.

Je fais évoluer la requête SQL afin que celle-ci devienne REPLACE.
➤REPLACE se comporte en INSERT (ajout) s'il n'y a pas d'id_produit connu
➤REPLACE se comporte en UPDATE (modification) s'il y a 1 id_produit connu (transmis par le champ id_produit caché en hidden)

Autant dire que le code est assez diffus mais nous avons l'avantage d'utiliser 1 seul formulaire et 1 seule requête SQL pour 2 choses différentes (l'ajout et la modification).

Intégration

L'intégration de la maquette a été faite en utilisant le langage de balisage HTML. HTML étant limité à la création de simples représentations structurées des pages, c'est avec le langage CSS que je défini le style désiré en accord avec la maquette.
Je le fais par le biais de feuilles de style personnalisées propres à chaque page. Ce choix à l'avantage d'offrir une maintenabilité du code plus aisé car les modifications apportées impactent seulement la page liée à la feuille de style concernée.

Pour ajouter du CSS à une balise HTML il suffit de lui donner une "class" que l'on nommera comme on le souhaite (de préférence avec un nom bien explicit pour pouvoir mieux s'y retrouver après), avec cela je peux cibler cette balise via le nom qu'elle porte directement dans mon fichier CSS.

je commence par relié mon fichier CSS sur la page dans laquelle je désire ajouter du style, cela se fait avec une balise <link> à l'intérieur de la balise <head>

```
<link href="inc/css/carousel.css" rel="stylesheet">
```

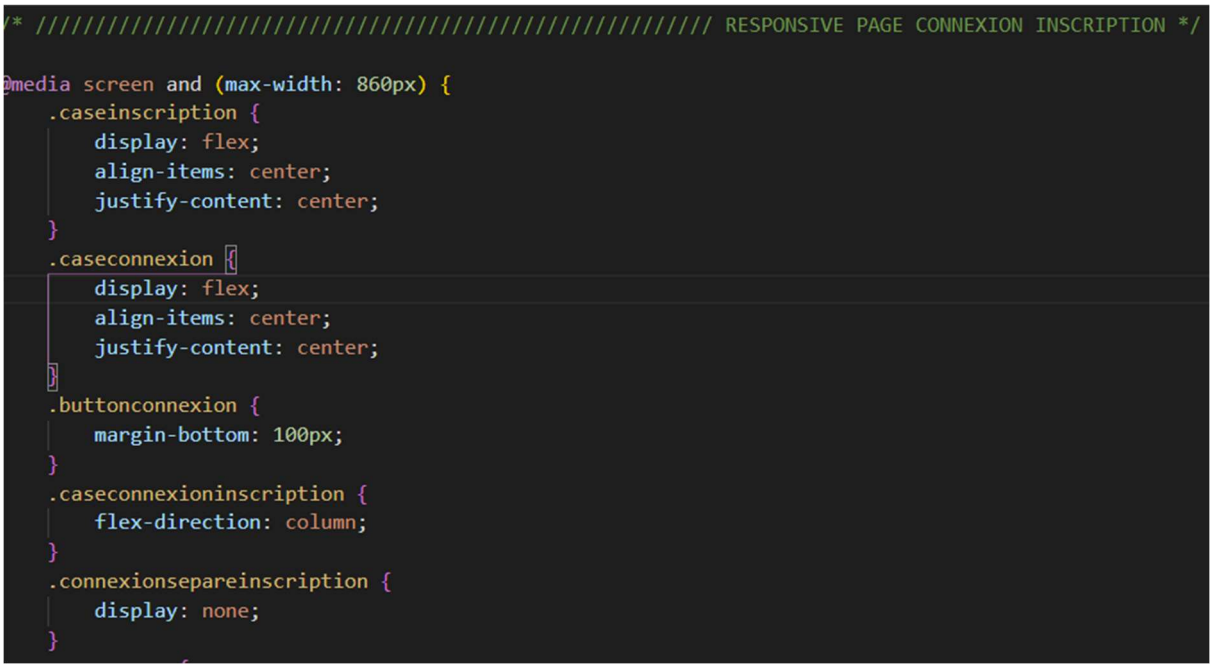
Après avoir fait cela, je donne une classe à n'importe quelle balise et c'est comme ça que je lui attribue du style.

```
<div class="carousel-item">
  <img class="bd-placeholder-img" width="100%" height="100%" src="inc/img/xiaomi.png" aria-hidden="true" preserveAspectRatio="xMidYMid slice" focusable="false">
  <rect width="100%" height="100%" fill="#777"/></img>
```

```
.carousel-item {
  height: 32rem;
}
.carousel-item > img {
  position: absolute;
  top: 0;
  right: 0;
  min-width: 100%;
  height: 32rem;
}
```

Responsive design

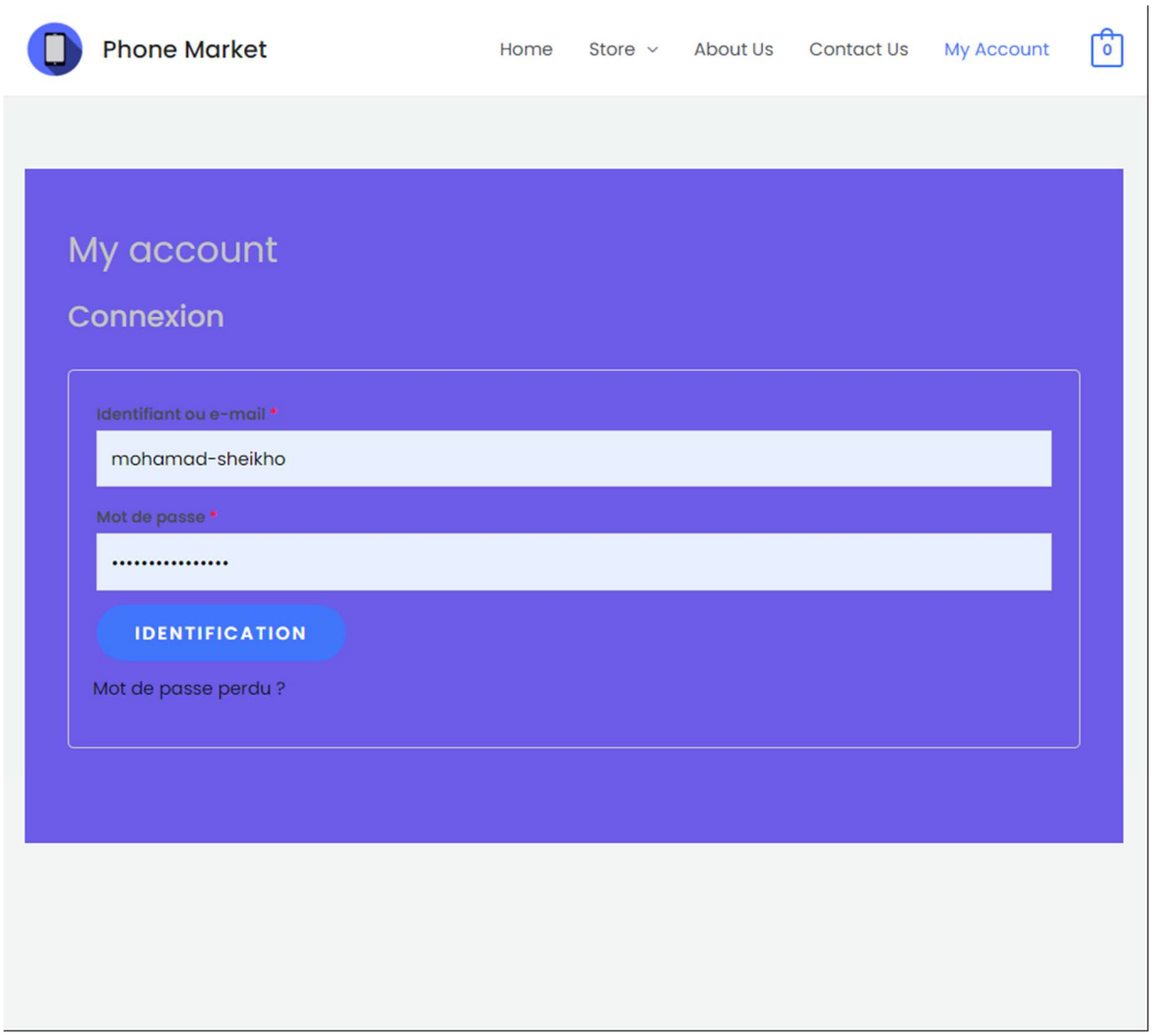
Une grande importance est portée sur l'adaptation de l'interface pour que la plateforme puisse être utilisée à partir de tous types de support.
Ces derniers regroupent principalement les ordinateurs, tablettes et smartphones.
Pour ce faire, j'utilise des Media Queries. En appelant les classes que je veux cibler et en définissant les largeurs d'écran maximal ou minimal, je peux facilement adapter le contenu affiché en changeant sa taille, masquer ou afficher du contenu ciblé, voici un exemple des media queries que j'ai fait sur Phone Market :



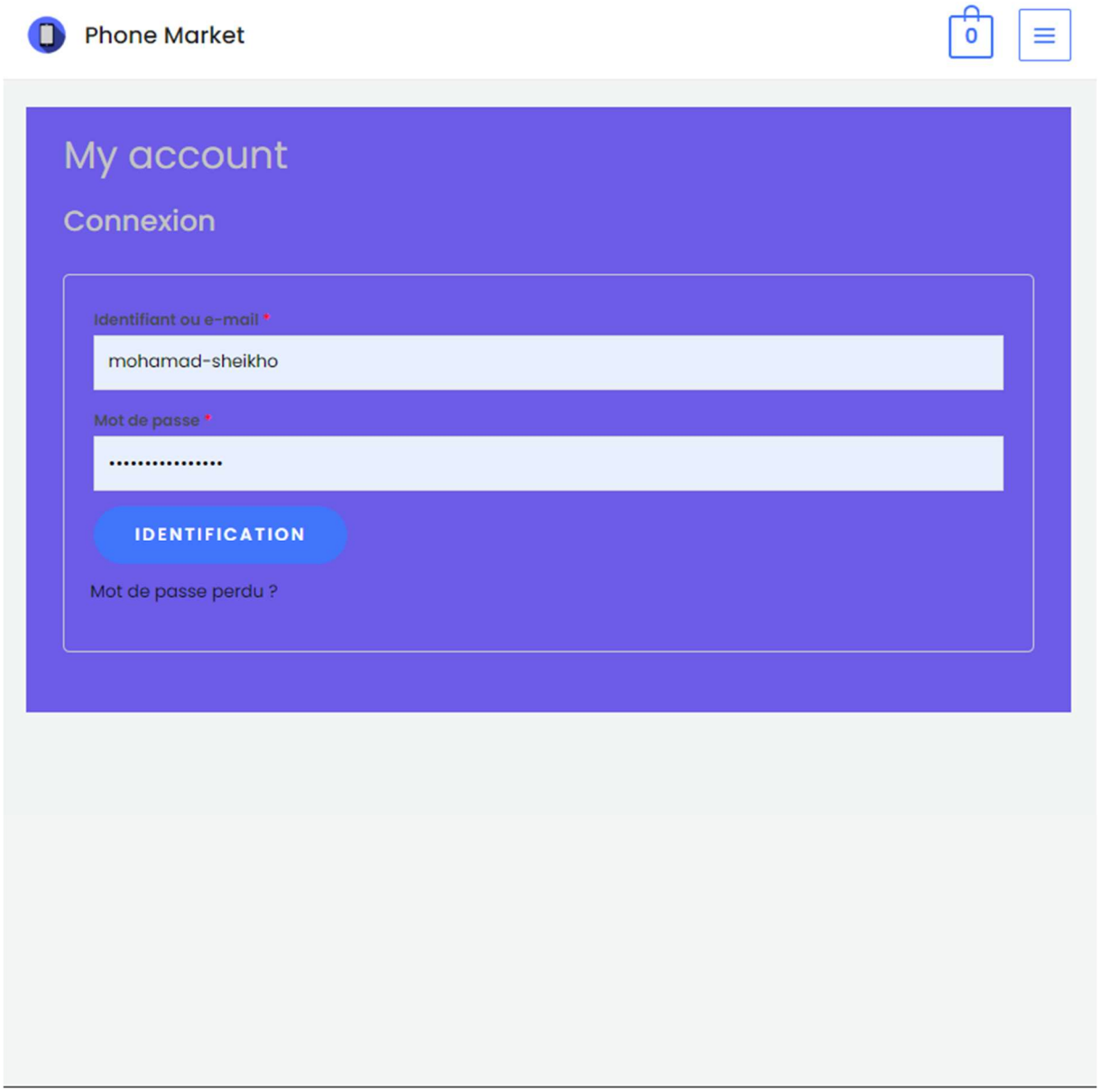
```css
/* ////////////////////////////////////////////////////// RESPONSIVE PAGE CONNEXION INSCRIPTION */

@media screen and (max-width: 860px) {
    .caseinscription {
        display: flex;
        align-items: center;
        justify-content: center;
    }
    .caseconnexion {
        display: flex;
        align-items: center;
        justify-content: center;
    }
    .buttonconnexion {
        margin-bottom: 100px;
    }
    .caseconnexioninscription {
        flex-direction: column;
    }
    .connexionsepareinscription {
        display: none;
    }
```

je défini que la largeur maximale que je souhaite est de 860px donc ces paramètres seront affectés aux classes ciblé qu'une fois que la largeur d'écran passe en dessous de 860px, voici un exemple :

Plus de 860px



Moins de 860px



Sécurité

Une veille concernant les failles de sécurité web a été faite durant la réalisation du site. J'ai parcouru le web à travers différents sites internet pour recouper les informations sur les failles web les plus connues.

Voici les sources dont je me suis servis :

https://www.cert.ssi.gouv.fr/

https://vigilance.fr/?langue=1

https://security.stackexchange.com/

https://owasp.org/

Faille INCLUDE

Il s'agit d'une faille très dangereuse. Comme son nom l'indique, elle exploite une mauvaise utilisation de la fonction include.

La plupart du temps, cette fonction est utilisée pour exécuter du code PHP qui se situe dans une autre page, permettant de se connecter à une base de données.

Il existe deux types de failles include :

A distance : il s'agit de la faille include par excellence. C'est à la fois la plus courante et la plus facilement exploitable.

En local : cela revient à inclure des fichiers qui se trouvent sur le serveur du site. Une personne mal intentionnée pourra s'emparer facilement de votre fichier contenant vos mots de passes.

Pour se protéger de cette faille, rien de mieux que de la tester ! Il vous suffit d'inclure une page qui
n'existe pas. Si l'URL de celle-ci est vulnérable, un message d'erreur vous sera transmis venant de PHP.

Injection SQL

Cette faille survient lors de la modification d'une requête SQL et consiste à injecter des morceaux de code non filtrés, généralement par le biais d'un formulaire. Cela revient à détourner la requête et lui faire faire autre chose que ce pour quoi elle a été conçue. Cette manipulation donne donc accès à vos données telles que les login, mots de passe ou adresses e-mail.

Faille UPLOAD

Cette faille peut apparaître lors de l'upload de fichiers sur un site : photo de profil, document pdf, image dans un message, etc. Elle profite de l'action effectuée pour mettre en ligne des fichiers malveillants PHP qui vont permettre au « hacker » de prendre le contrôle total de notre site.

Pour éviter cette vulnérabilité, il est important de :

Empêcher les utilisateurs d'envoyer des fichiers lorsque cela n'est pas une fonction primordiale pour votre site ou application

Interdire l'exécution de code depuis le dossier dans lequel sont stockés les fichiers uploadés sur votre site

Vérifier et autoriser l'extension des fichiers que vous tolérez via une liste blanche.

Faille XSS

Une faille XSS consiste à injecter du code qui pourra être interprété directement par le navigateur Web. Ce dernier ne fera ainsi aucune différence entre le code du site et celui injecté par le pirate.
Les effets sont bien entendu assez embêtants puisque vous risquez de faire face à des redirections vers un autre site, du vol de cookies ou encore une modification du code de votre page.

Pour vous protéger des XSS, vous devez remplacer les caractères pouvant être compris par le navigateur comme des balises par leur entité HTML. En procédant ainsi, le navigateur affichera mot à mot le caractère et ne cherchera plus à l'interpréter. En PHP, vous pouvez utiliser les fonctions htmlentities ou htmlspecialchar.

ATTAQUE PAR FORCE BRUTE

Cette méthode consiste à trouver le mot de passe ou la clé cryptographique d'une personne afin de pouvoir accéder à un service en ligne, à des données personnelles, voire à un ordinateur.
Il est donc indispensable d'utiliser des mots de passe forts pour vos sites et comptes utilisateurs afin de rendre complexe l'attaque par une personne tiers.

3 conseils utiles pour renforcer vos mots de passe :

Utiliser des lettres minuscules, des majuscules, des chiffres et des caractères spéciaux (notez qu'il existe des générateurs automatiques de mots de passe sur internet)

Renouveler souvent ses mots de passe.

Utiliser des mots de passe différents pour chaque site.

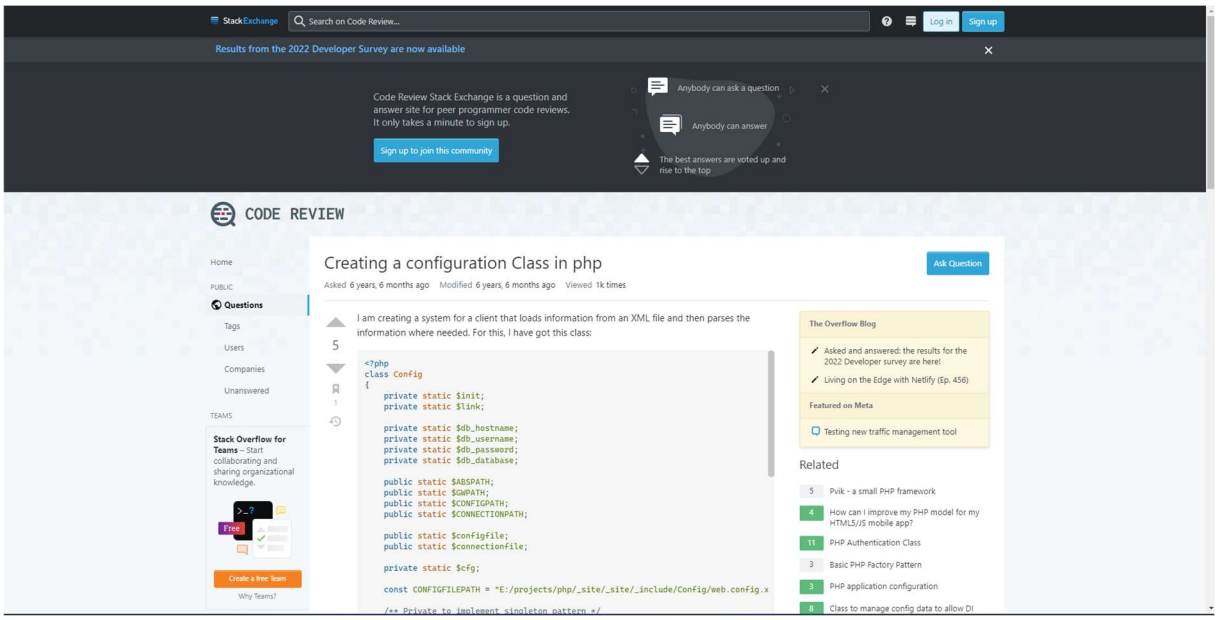
Recherche Anglophone

How to create a configuration class with php

Lors de la création de mon projet Phone Market, je devais créer un fichier connexionDB.php en classe, mais je ne savais pas comment faire, pour cela j'ai effectué une recherche d'un tutoriel en anglais sur un site auquel j'avais déjà effectué des recherches auparavant et ou j'avais trouvé des réponse à mes questions

Voici un lien vers le site : https://stackexchange.com.

J'ai donc trouvé une solution Pour ma problématique.



Conclusion

Ce projet est l'aboutissement de toutes les connaissances que j'ai emmagasiné durant cette année, plein de projet m'ont aidé à en arriver là, comme par exemple faire un module de connexion, de la gestion de projet pour connaître la démarche à adopter et quel procédé suivre, faire du maquettage, du css, du responsive, du JS....

Merci pour votre attention.