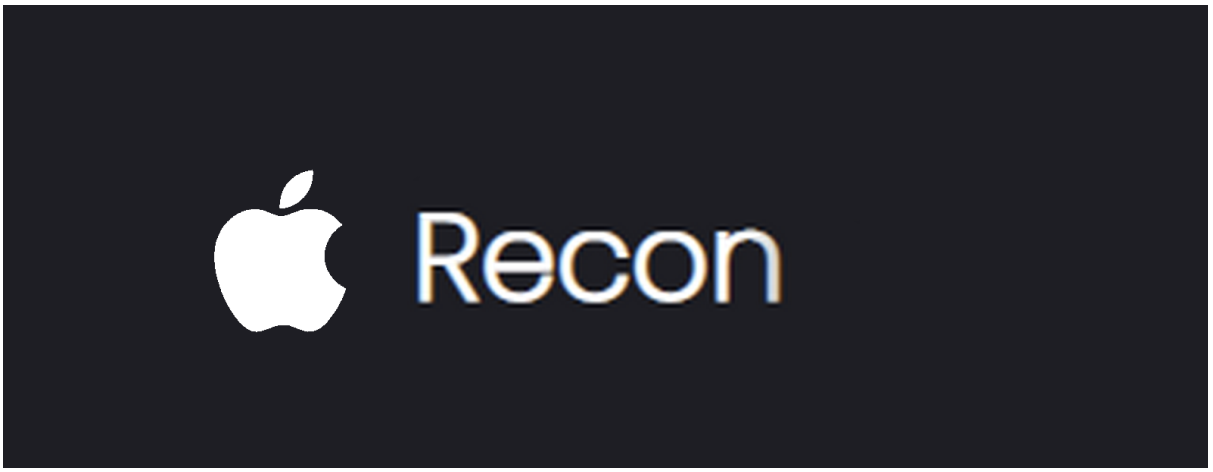
DOSSIER PROJET PROFESSIONNEL



Titre professionnel :

Développeur Web / Web Mobile

LOUNI Samir

Table des matières

Compétences du référentiel couvertes par le projet

Le projet couvre les compétences énoncées ci-dessous.

Pour l'activité 1, "Développer la partie front-end d'une application web et web
mobile en intégrant les recommandations de sécurité":

- Maquetter une application
- Réaliser une interface utilisateur web ou mobile statique et adaptable
- Développer une interface utilisateur web dynamique
- Réaliser une interface utilisateur avec une solution de gestion de contenu ou
- e-commerce

Pour l'activité 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 la partie back-end d'une application web ou web mobile
- Élaborer et mettre en œuvre des composants dans une application de gestion
- de contenu ou e-commerce.

Résumé

Recon est une boutique en ligne fictive créée en 2021 et basée à Marseille.
Cette boutique en ligne fait partie de mon travail effectué en tant que développeur web / d'application web en tant qu'étudiant à La Plateforme.
L'entreprise Recon vend des téléphones, accessoires, macbook,et Ipad .
Sa particularité est que tous les produits proposés sont reconditionnés et à des prix défiant toute concurrence.
L'enjeu pour Recon est d'accroître ses parts de marché ainsi que sa visibilité sur internet grâce à cette boutique en ligne.
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 en binôme au cours de ma formation de développeur web à l'école La Plateforme à Marseille.

Périmètre du projet

Le site sera réalisé en français et ce dernier devra être accessible sur différents supports, à savoir mobile, tablette et ordinateur.

Cible adressée par le site internet

Apple comme la marque d'une image de luxe, devenu un exigence de nos jours, le site de l'entreprise Recon s'adresse aux personnes à petit budget désirant avoir des produits Apple.

Description des fonctionnalité

1.Authentification classique

Sur le site, il a été convenu de pouvoir s'inscrire, et se connecter de manière classique via un formulaire d'inscription/connexion pour les clients. Le client peut se connecter avec son adresse mail, ou son identifiant.

2.Produit catégorie et filtre

Chaque page doit permettre d'afficher l'ensemble des produits disponibles selon leurs catégories.
Cet affichage comprend la photo du produit, le nom du produit ainsi que son prix.
Un filtre y est implémenté afin de trier les articles en fonction de leur mise en ligne sur le site.

3.Fiche produit

L'utilisateur devra être en mesure d'accéder à une fiche produit.
Cette dernière devra comprendre :

- La photo du produit
- Le nom du produit
- Le prix
- La description du produit

4.Panier client

L'utilisateur devra être en mesure de sélectionner un article et de le mettre dans son panier dans le but final de passer commande sur le site.
Ce panier devra afficher les éléments suivants:

- Une photo de l'article
- Le prix de l'article
- La quantité
- Le total des articles sélectionnés
- Un bouton de validation du panier débouchant sur le processus de commande.

Le client aura aussi la possibilité de supprimer un article du panier s'il le souhaite.

5. Fonctionnalité de recherche produit

Le client devra être en mesure d'effectuer une recherche de produit dans un champ prévu à cet effet.
Afin de faciliter la recherche, un système de recherche avec autocompletion doit être mis en place.

6.Espace client

L'utilisateur aura accès à un espace client dans lequel il lui sera possible de consulter et modifier ses informations.
Son nom, son prénom, son adresse, son email, son mot de passe, son identifiant.
Il aura également la capacité de consulter ses précédentes commandes.

7.Back office

L'administrateur du site devra avoir accès à un espace sécurisé lui permettant d'administrer le site grâce au panel administrateur.

7.1.Ajout de produits

L'administrateur aura également la capacité de créer des produits.
Lors de la création du produit, ce dernier sera en mesure de:

- *Donner un titre au produit*
- *Définir le prix du produit*
- *Donner une description au produit*
- *Uploader une image du produit*
- *Sélectionner la catégorie du produit*

7.2.Suppression de produits

L'administrateur aura également la capacité de supprimer des produits.

7.3.Modification du produit

L'administrateur du site sera en mesure de modifier le prix d'un article.

7.4.Ajout de catégorie

L'administrateur sera en mesure de créer des catégories.

7.5.Suppression de catégorie

L'administrateur sera en mesure de supprimer une catégorie ainsi que les articles auxquels ils sont liés .

7.6.Suivi des commandes

L'administrateur aura la possibilité de visualiser les commandes qui ont été effectuées sur le site. Ceci signifie que l'administrateur aura accès aux informations suivantes:

- La date de la commande
- Le nom et le prénom de la personne ayant passé commande
- Son adresse
- Son numéro de téléphone
- Son pays , sa ville , ainsi que son arrondissement
- Son email, et son identifiant de log
- Le nom de l'article
- La quantité
- Le prix total de la commande

L’administrateur aura aussi la possibilité de voir si la commande a simplement été validé dans le panier, puis n’a plus donner suite, ou si la commande a été finalisée, donc payer.

7.7.Gestion des droits

L’administrateur sera en mesure de gérer les droits de chaque membre inscrit .Celui-ci pourra faire passer un utilisateur ayant le statut ‘membre’, au statut ‘administrateur, ce qui lui permettra d’avoir accès au panel administrateur à son tour.

7.8.Gestion du produit phare

L’administrateur se verra attribuer la modification s’il le souhaite du produit phare affiché sur l’index du site.
Il devra choisir :

- L’image (Grande taille).
- L’article qui sera lié à l'image
- Le texte au dessus du bouton (exemple : ’Nouvelle arrivage‘)

8.Solution de paiement

La solution de paiement choisie est celle proposée par Stripe. Cette solution permettra au client de procéder au paiement de manière sécurisée par carte bancaire.

9.Page qui sommes-nous ?

Présentation de l’entreprise, de nos services, etc ..

10.Site responsive

Le site devra être disponible sur Pc, mobile, tablette et devra s’adapter en fonction de la taille de l’écran de l’utilisateur.

11.Site dynamique

Le site devra être dynamique avec l’ajout d’interactions avec l'utilisateur (afficher , cacher un texte, etc ..)

Spécifications techniques

Technologies utilisées pour la partie back-end :

- *Le projet sera réalisé avec le langage PHP, en programmation orientée objet (POO)*
- *Base de données SQL*
- *WampServer*

Technologies utilisées pour la partie front-end :

- *Le projet sera réalisé avec du HTML et CSS.*
- *Javascript , et plus précisément JQuery afin de dynamiser le site et d'améliorer l'expérience*

L'environnement de développement est le suivant:

- Editeur de code: Visual Studio Code
- Outil de versioning: GIT, Github Desktop.
- Maquettage: Figma, Balsamiq Wireframes

Du point de vue de l'organisation, j'ai utilisé MICROSOFT TO-DO *afin de découper le projet en une multitude de tâches à réaliser et de définir leur ordre de priorité.*

WAMP

Pour mon site j'utilise WampServer.
WampServer est une plateforme de développement web sous Windows. Elle permet principalement de gérer les services tel que Apache MySQL, ou bien d'ouvrir des pages web d'extension '.php' par exemple.
L'avantage de ce logiciel est d'éviter de passer par un hébergeur pour avoir un aperçu de ses scripts codé en PHP par exemple
Car en ouvrant un fichier .php sans des logiciels tel que WAMP, la page afficherait trop d’erreurs .
De plus on a une base de données local (MySQL) qui permet d'exécuter tout type de script ayant besoin d'une base de données

MySQL®

Pour la base de données du site, j'ai utilisé MySQL.
MySQL est le système de gestion de base de données relationnelle le plus populaire.
Une base de données relationnelle permet d'entreposer et d'organiser les données structurées sous forme de tableaux liés entre eux.
Le système MySQL présente plusieurs particularités. Tout d'abord, il s'agit d'un logiciel Open Source utilisable librement et modifiable à volonté. N'importe qui peut l'installer et personnaliser le code source pour répondre à ses besoins spécifiques

IDE

Visual studio code

Pour le projet, j'ai choisis Visual studio code comme éditeur de code

Celui-ci présenté comme avantage de pouvoir coder en PHP , en JavaScript.

l'IDE est aussi gratuit dans sa version d'apprentissage et pour les projets open-source. C'est aussi l'IDE que j'ai le plus utilisé dans l'année.

Visual Studio Code est un éditeur de code extensible développé par Microsoft.

Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré. Les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires.

GitHub

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git.

Gestion du projet

Pour la gestion de projet et le travail d’équipe avec mon binôme, nous avons utilisé divers outils afin de se répartir les tâches avec l’envoi de mail, partage de fichier sur le drive, et travail en présentiel.
Avec Github Desktop, nous avons pu apporter nos modifications au code chacun de notre côté.
Nous avions une branche Master, c’est la branche de base.
Reliée à celle-ci, la branche ‘DEV’. Autour de ces branches, nous avons plusieurs branches pour chaque fonctionnalité que j’envoie sur la branche ‘DEV’ qui permet à mon camarade de voir le code avant de le valider et l'envoyer sur la branche principale ‘MASTER’.

Modélisation de la base de données

J’ai modélisé la base de données en plusieurs étapes :

- En identifiant le besoin (spécifications techniques, besoin des utilisateurs, des administrateur, et des clients.)
- Choisir le système de gestion de base de données (SGBD), Pour le cas du site Recon, le SGBD est MySQL
- Application de la méthode Merise

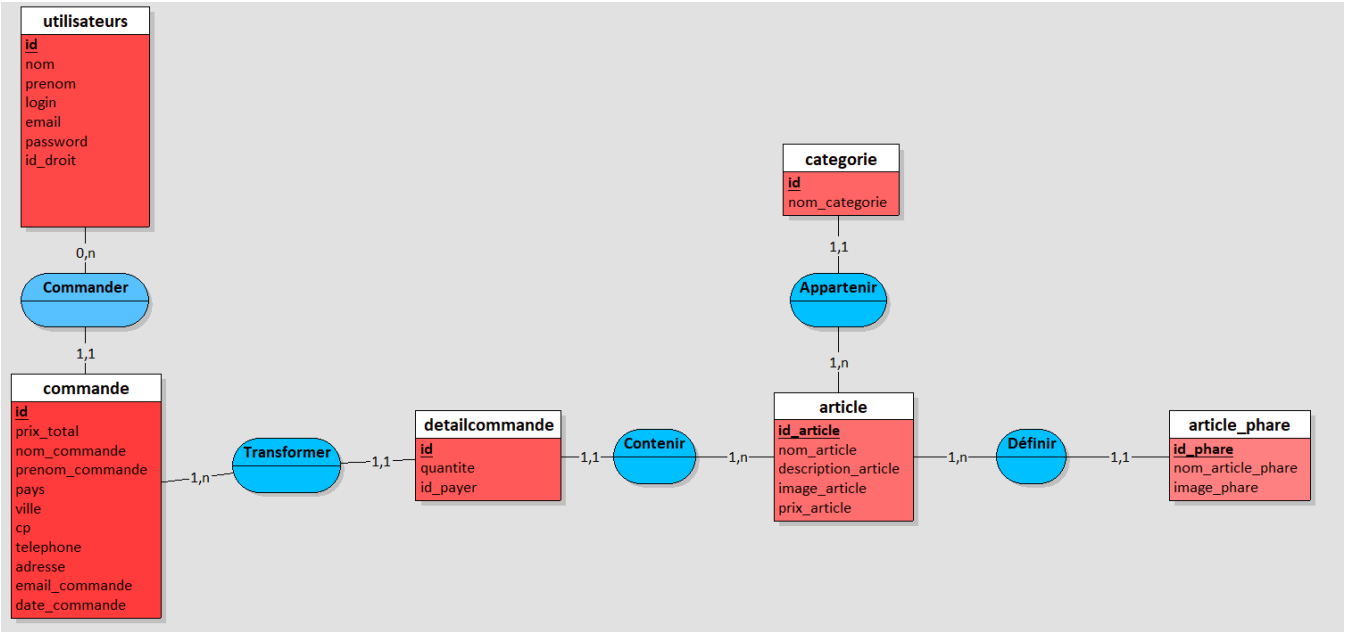
Méthode Merise

Merise est une méthode de conception, de développement et de réalisation de projets informatiques.
Le but de cette méthode est d'arriver à concevoir un système d'information.
La méthode MERISE est basée sur la séparation des données et des traitements à effectuer en plusieurs modèles conceptuels et physiques.

Modèle conceptuel de données (MCD)

Le MCD est une représentation graphique de haut niveau qui permet facilement et simplement de comprendre comment les différents éléments sont liés entre eux à l’aide de diagrammes codifiés.
Il faut définir les données élémentaires qui vont être conservées en base de données et définir certaines caractéristiques qui figureront dans le MCD, par exemple : son type , identifiant unique etc ..

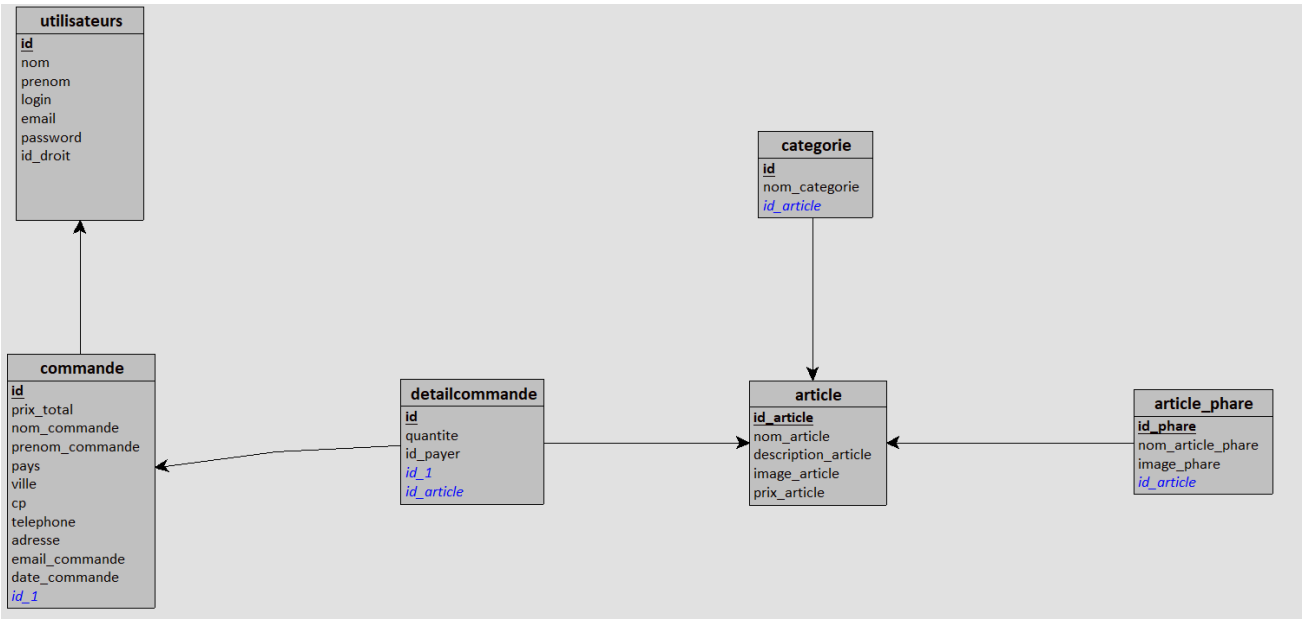
MCD DU SITE RECON



Modèle logique de données (MLD)

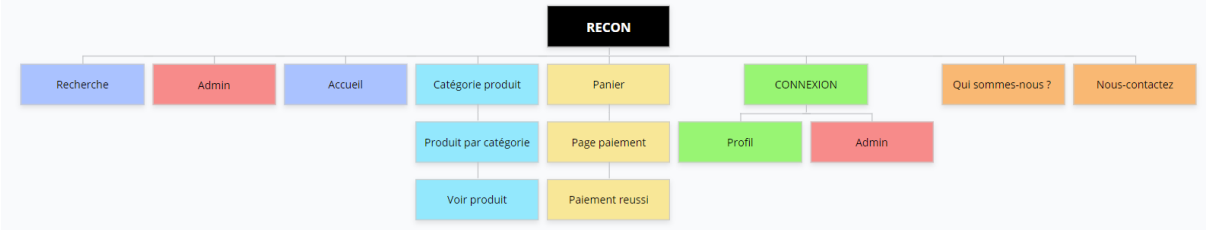
Un modèle logique de données (MLD) est la représentation des données d'un système d'information. Les données sont représentées en prenant en compte le modèle technologique qui sera utilisé pour leur gestion.

MLD DU SITE RECON



Réalisations

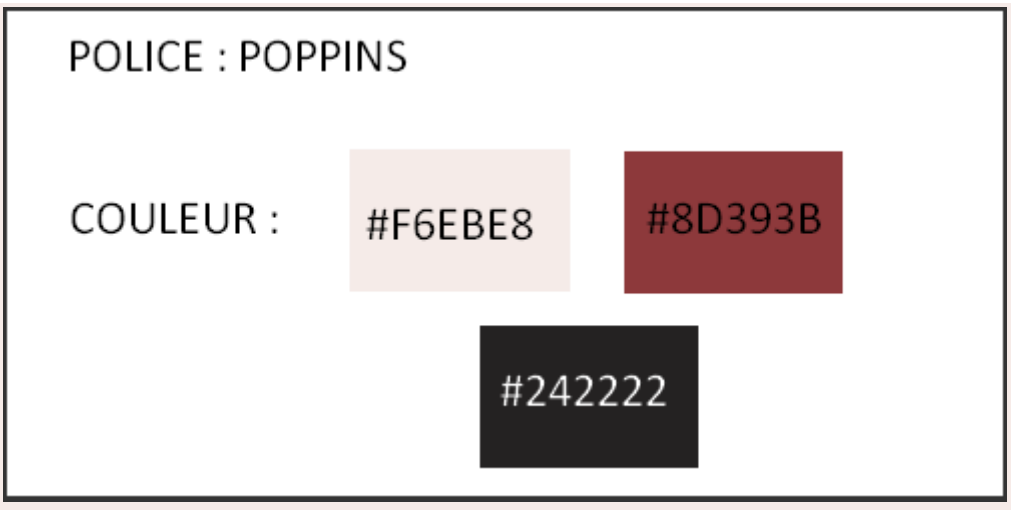
1.Arborescence du site



L'arborescence d'un site web permet d'avoir une vision globale du site et des parcours de navigation avant de réaliser les maquettes des pages principales.
Ci-dessus, l'arborescence du site RECON réalisé sur GlooMaps.

2. Charte graphique

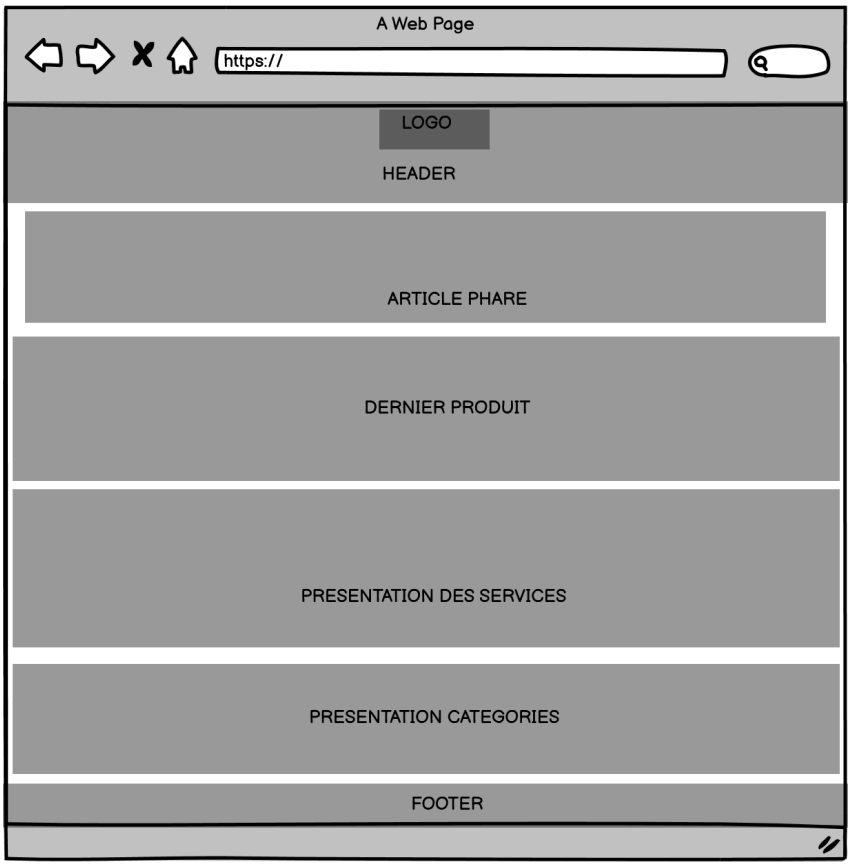
La polices d'écriture est la suivante: 'Poppins'
Les couleurs dominante sont les suivantes : #F6EBE8 #8D393B #242222



3.Maquette

2.1 Zoning

Pour commencer la maquette, j’ai dans un premier temps établi le zoning.
Le zoning est la pratique qui consiste à « découper » et représenter les différentes zones et les types de contenus qui leur sont affectés.

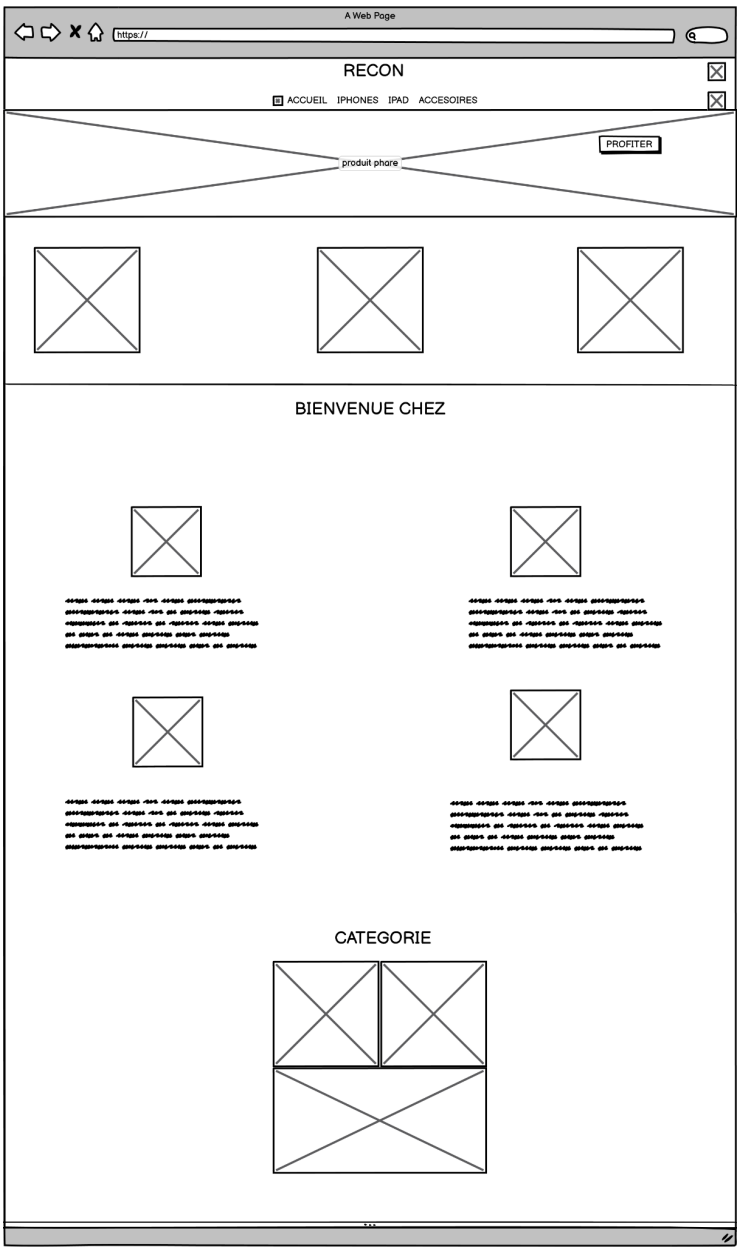


2.2 Wireframe

Après avoir établi un zoning des pages du site web, la suite logique est de spécifier ce que chaque bloc devra contenir précisément. Il s’agit de préciser:

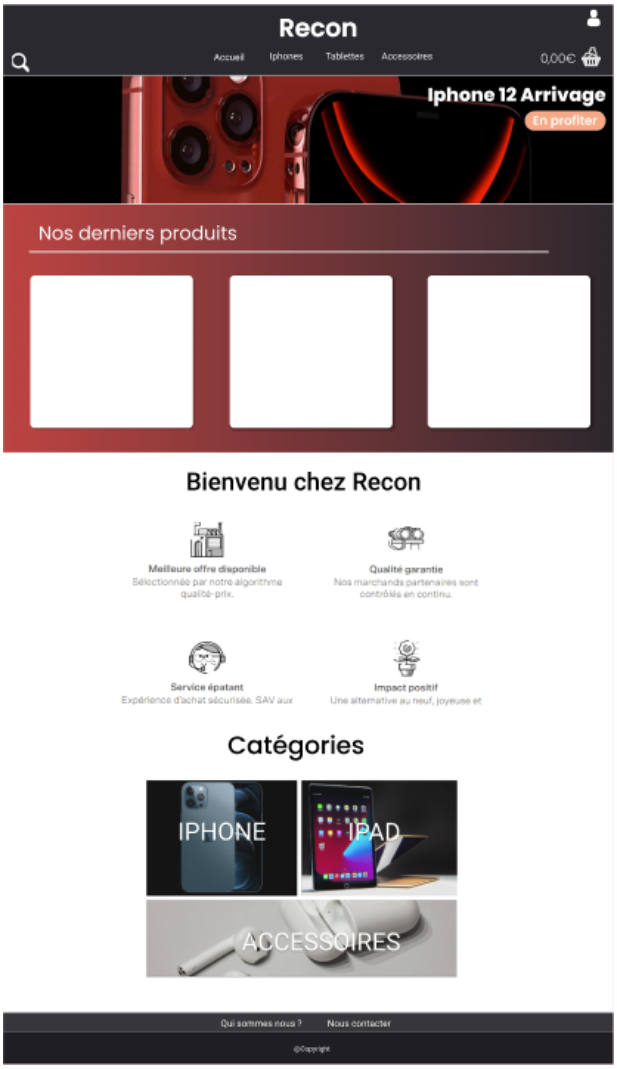
- Ce que contient la page
- à quel endroit sont placés les éléments
- Leurs tailles approximative
- Leurs comportements
- Etc ..

Ce que j’ai donc réalisé à l’aide de l'application ‘Balsamiq Wireframe’.



2.3.Prototype

Enfin, après avoir effectué le Zoning, puis le Wireframe, nous passons au prototype.
Il s'agit à cette étape de transformer chacune des pages réalisées à l'étape précédente en pages interactives, ajout des couleurs, etc …
Ce prototype a été réalisé à l'aide de Figma



Extrait de code significatif

INCLUDE

La grande majorité des sites web dynamiques ou des applications ont besoin de réutiliser des parties de code identiques à plusieurs endroits d'une même page, ou bien dans plusieurs pages différentes. C'est pourquoi j'ai décidé d'utiliser la fonction include sur mon site.
J'ai inclus le header sur chaque page afin d'éviter de le reproduire sur chacune de mes pages.

```php
<?php include 'header.php';?>
```

Programmation orienté objet

La programmation orientée objet offre de nombreux avantages:

- La possibilité de réutiliser le code dans différents projets
- Une conception de l'algorithme plus claire et organisée.

Toutes les requêtes en base de données sont gérées par des classes (user , admin , panier , ...).

Ces classes possèdent des méthodes permettant d'exécuter des requêtes préparées.
L'utilisation de requêtes préparées, rendues possibles grâce à l'extension PDO

Les requêtes préparées sont principalement utilisées pour deux raisons :

- Protéger son application des injections SQL ;
- Gagner en performance dans le cas d'une requête exécutée plusieurs fois par la même session.

Dans un premier temps, je commence par créer ma classe et j'y ajoute des attribut, public ou private.
En programmation orientée objet, un attribut n'est ni plus ni moins qu'une variable.
On peut y stocker par exemple les paramètres de la connexion à la base de données afin de ne pas avoir à le refaire à chaque fois qu'on a besoin de se connecter à la base de données .

Public : La variable sera accessible en dehors de la classe.
Private : La variable ne sera pas accessible en dehors de la classe uniquement à l'intérieur.

```php
class user
{
private $_id;
private $_email;
public $_login;
public $_password;
```

Ici par exemple, la class ‘user’.
Nous ne pouvons pas appeler la variable **$_id** en dehors de la classe, par contre pour **$_login** oui.

Maintenant, il faut inclure la page, la ou je souhaite faire appel au fonction contenu dans celle-ci. J’utilise la fonction **include()**.

```php
include("class/classUser.php");
```

Une fois réalisé, je dois instancié la classe :

```php
include("class/classUser.php");
$user = new user;
```

L'opérateur ‘new’ se charge de créer une instance de la classe et de l'associer à une variable, ici ‘**$user**’.
Une fois cela effectué, nous pouvons désormais grâce à la variable **$user** appeler les méthodes contenu dans la classes user.

Appeler les méthodes de l'objet

Pour appeler une méthode d'un objet, il va falloir utiliser un opérateur : il s'agit de l'opérateur ‘**->**’

Nous allons appelé ici la connexion à la base de données

```php
5   $user->dbconnect();
```

La ligne de code ci-dessus signifie donc “va chercher la variable **$user**, et fait appel à la fonction dbconnect()”.

Allons voir à quoi ressemble cette fonction que je viens d'appeler .

```
15  public function dbconnect()
16  {
17      $db = new PDO("mysql:host=localhost; dbname=boutique", 'root', '');
18      $this->_db = $db;
19  }
```

C’est la connexion a ma base de donnée, celle-ci attend plusieurs paramètre :

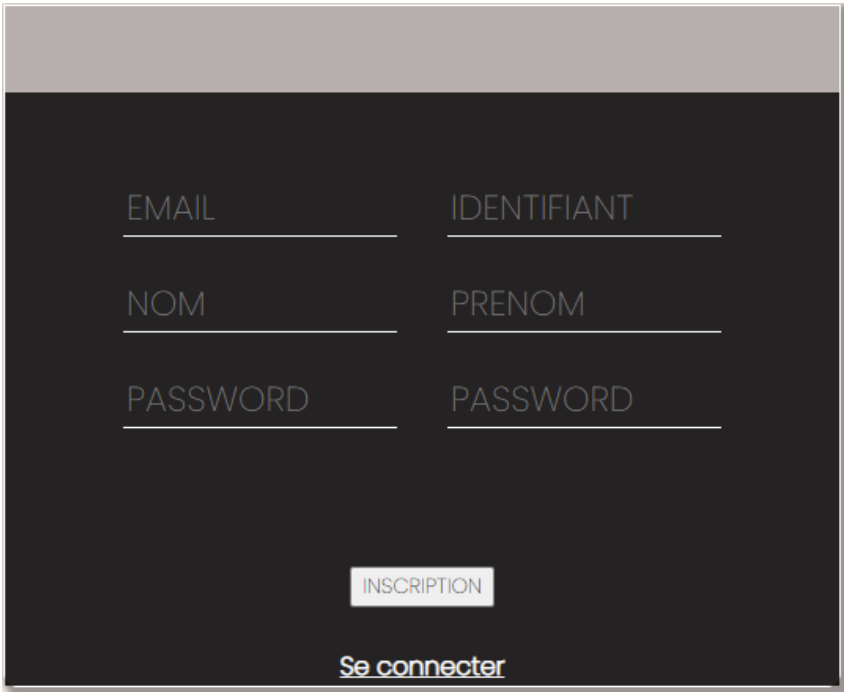
- Le nom du serveur (localhost, car j’ai travaillé en local,
- Le nom de la base de donnée (boutique ici)
- L’identifiant et mot de passe pour accéder à la base de donnée

Maintenant que nous avons inclus la page, la ou je souhaite faire appel au fonction, instancié la classe , et fait appel à dbconnect().

Nous sommes désormais connecté à la base de données, nous pouvons maintenant commencer l’écriture de nos différentes 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***).

CREATE :
Cette opération est gérée par la requête “INSERT” .
Ici, l’exemple d’utilisation de cette requête. Dans le cadre du projet, avoir un compte utilisateur était indispensable pour le site. J’ai donc créé une fonction inscription.



(Apparence de l'inscription)

Code HTML de celui-ci :

```
<form action = '' method = 'post' autocomplete = 'off'>
    <input type = 'email' placeholder = 'EMAIL' id ='mail' name = 'mail'>
    <input type = 'text' placeholder = 'IDENTIFIANT' id = 'identifiant' name = 'identifiant ' ><br>
    <input  type = 'text' placeholder = 'NOM' id = 'nom' name = 'nom'>
    <input   type = 'text' placeholder = 'PRENOM' id = 'prenom' name = 'prenom'><br>
    <input type = 'password' placeholder = 'PASSWORD' id ='password-inscription' name = 'password-inscription'>
    <input type = 'password' placeholder = 'PASSWORD' id = 'confirm-password' name = 'confirm-password'> <br>
    <input type = "submit" name = 'inscription' value = 'Inscription'>
</form>
```

Il existe 2 type de method pour un formulaire :

- GET (*Avec la méthode GET, les données à envoyer au serveur sont écrites directement dans l'URL. Dans la fenêtre de votre navigateur*)
- POST (*La méthode POST écrit les paramètres URL dans la requête HTTP pour le serveur. Les paramètres ne sont donc pas visibles pour les utilisateurs*)

Les données ici sont récupérées en méthode 'POST' dans mon formulaire ci dessus :
Afin d'ajouter dans la base de données les informations entrée dans le formulaire je dis que :

```php
<?php
    if(isset($_POST['inscription'])){
        $user->inscription($_POST['mail'], $_POST['identifiant'], $_POST['nom'],$_POST['prenom'],
                           $_POST['password-inscription'],$_POST['confirm-password']);
    }
?>
```

Lorsque l'utilisateur clique sur l'input 'inscription'. Je fais appel à la fonction **inscription()**.
Alors les informations entrée dans le formulaire sont entrée dans les paramètres de la fonction **inscription()**

$_POST['prenom'] pour l'input qui porte comme nom: 'prenom',
$_POST['password-inscription'] pour l'input qui porte comme nom 'password-inscription', ainsi de suite.

Allons voir à quoi ressemble la fonction **inscription()**.

```php
public function inscription($email, $login, $nom, $prenom, $motdepasse, $confirm)
{
    $db = $this->_db;


    $droit = 1;
    $_nom = htmlspecialchars(trim($nom));
    $_login = htmlspecialchars(trim($login));
    $_prenom = htmlspecialchars(trim($prenom));
    $_email = htmlspecialchars(trim($email));
    $_motdepasse = htmlspecialchars(trim($motdepasse));
    $_confirm = htmlspecialchars(trim($confirm));
    $cryptage = password_hash($_motdepasse, PASSWORD_BCRYPT);

    $loginexistant = "SELECT `login` FROM utilisateurs WHERE login = '$_login'";
    $verification = $db->query($loginexistant);
    $emailexistant = "SELECT `email` FROM utilisateurs WHERE email = '$_email'";
    $verification2 = $db->query($emailexistant);


    if(!empty($_prenom) && !empty($_email) && !empty($_login) && !empty($_nom) && !empty($_motdepasse) && !empty($_confirm)){
        if($verification2->fetch(PDO::FETCH_ASSOC) == 0 ){
            if($verification->fetch(PDO::FETCH_ASSOC) == 0 ){
                if($motdepasse == $confirm){
                    $requete = $db->prepare("INSERT INTO `utilisateurs` (`email`, `login`, `nom`, `prenom`, `password`, `id_droit`) VALUES ('$_email', '$_login', '$_nom', '$_prenom', '$cryptage
                    $requete->execute();
                    echo  'Bienvenue ! ';
                }else{
                    echo 'Les mots de passe ne correspondent pas';
                }

            }else{
                echo  'Cette identifiant éxiste déjà';
            }
        }else{
            echo  'Cette email est déja utilisé';
        }
    }else{
        echo "Remplissez tout les champs ! ";
```

Nous pouvons voir que celle-ci prend en paramètre 6 variables, l'email , le login, le nom, le prénom, le mot de passe , et la confirmation du mot de passe,
mot de passe que je vais hasher par soucis de sécurité à l'aide de la fonction **password_hash()** qui crée une clé de hachage pour un mot de passe.
Et que je vérifierai à l'aide de **password_verify()** qui vérifie qu'un mot de passe correspond à un hachage

Je commence par récupérer la connexion a ma base de données

(ici **$db**)
J’ajoute des fonctions pour chaque paramètre entrée, ici : **htmlspecialchars()**, qui permet de convertir les caractères spéciaux en entités HTML (Pratique pour éviter que des données fournies par les utilisateurs contiennent des balises HTML)

trim() qui supprime les espaces (ou d'autres caractères) en début et fin de chaîne

Nous commençons par vérifier qu’aucune des 6 variables n’est vide à l’aide de la fonction **empty(),** empty détermine si une variable est vide.
Une fois cette vérification effectuée nous vérifions qu’il n’éxiste pas déja un compte avec le même login ou la même adresse mail à l’aide d’une requête contenant ‘SELECT’ que nous verrons plus tard.

Après ces vérifications nous controlons que la variable **$motdepasse** est égale a la variable **$confirm** (pour la confirmation du mot de passe) .
Une fois toute ces vérifications effectuées, nous pouvons maintenant prépare la requête SQL qui est un INSERT, dans cette INSERT je lui spécifie de cibler la table “utilisateurs” et les colonnes dans lesquelles je veux les stocker et pour finir les valeurs que je veux insérer (VALUES).
Enfin, j'exécute la requête.

READ

READ, cette opération est gérée par la requête SELECT, pour récupérer des éléments en base de données.
Dans le cadre de ma boutique en ligne, je m’en suis servi pour afficher les catégories présentes dans ma base de données .

```php
public function affichercategorie()
{
    $db = $this->_db;
    $requete = $db->prepare('SELECT * FROM categorie ORDER BY id ASC');
    $requete->execute();
    $resultat = $requete->fetchall();

    foreach ($resultat as $key) {
        $id = $key['id'];
        echo "<li>"."<a href='produit.php?id=$id'>".$key['nom_categorie']. '</a>' . "</li>";
    }
}
```

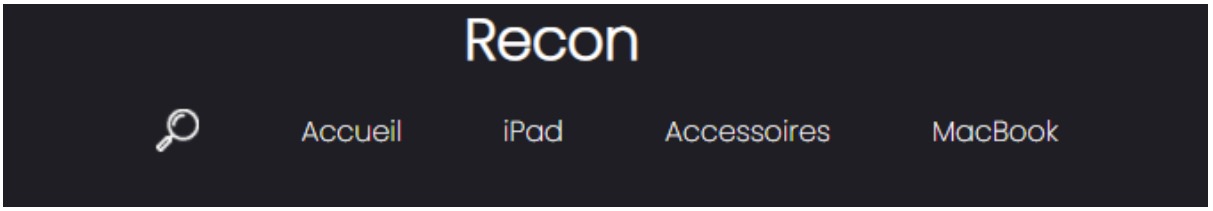
Cette fonction ne comprend aucun paramètre pour être exécuté.
Comme pour CREATE ,je récupère la connexion a ma base de données (ici **$db**)
Ensuite je prépare ma requête.
Je rédige ma fonction, qui est un SELECT suivi d'une astérisque qui signifie que je souhaite récupérer tous les résultats que cette requête aura en en base de donnée, je l'exécute ensuite pour que cette requête soit traité par la base de données.
Ensuite je rédige un **fetchall()** de ma requête que je stock dans une variable **$resultat**, **fetchall()** retourne un tableau contenant tous les résultats de ma requête précédente (ici, tout ce qui concerne la table 'categorie').
Afin de pouvoir avoir accès aux informations renvoyer, je rédige une boucle **foreach ()** qui permet de parcourir un tableau ,elle réitère chaque valeur du tableau une par une jusqu'à qu'il soit parcouru en entier.
Ici je lui demande de m'afficher chaque 'nom_categorie' qui existe dans le tableau.

Maintenant il me suffit juste d'appeler la fonction **affichercategorie()** la ou je souhaite afficher chacune des catégories pour que celle-ci s'affiche.

Cette fonction est contenu dans la variable **$affiche.**

```php
$affiche->affichercategorie();
```

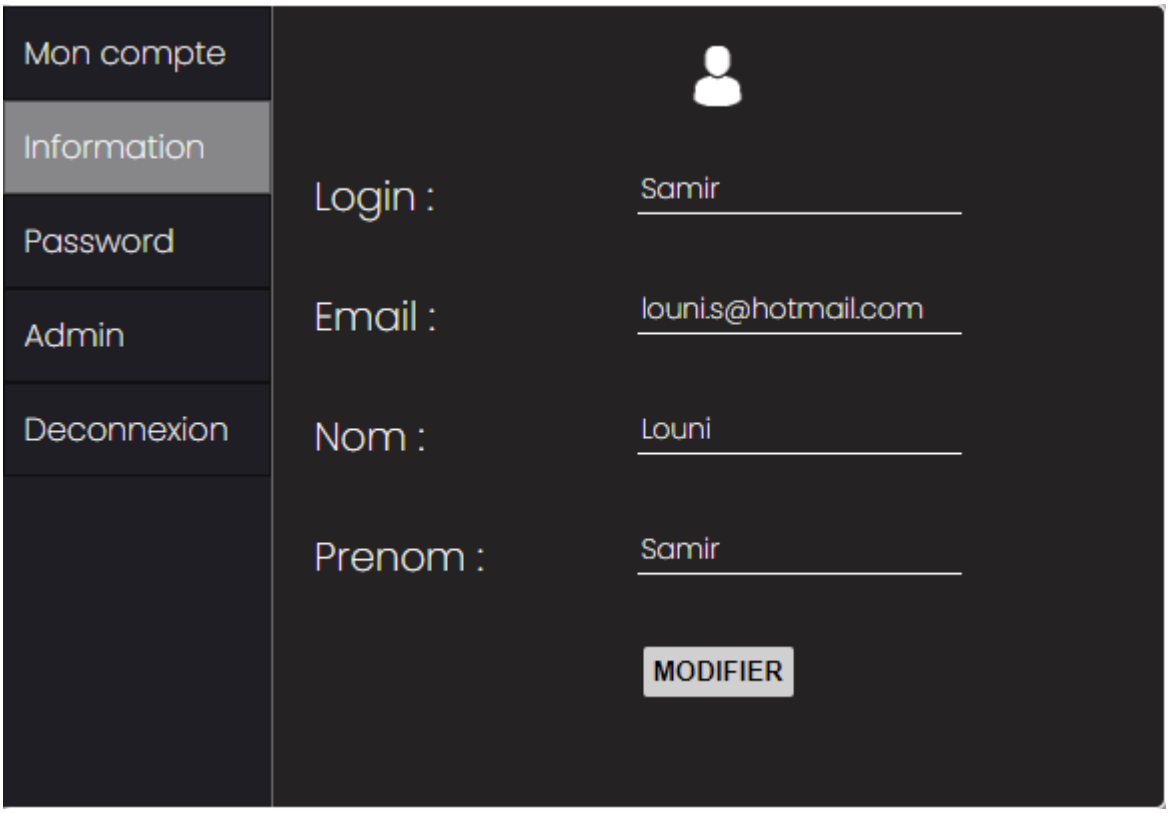
J'ai appelé la fonction dans le header allons vérifier :



Le header affiche bien les catégories que j'ai dans ma base de données.

UPDATE

Cette opération est gérée par la requête qui porte le nom UPDATE, elle permet de 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 l'utilisateur de modifier ses informations.



Je rédige ma fonction, qui est un UPDATE

```php
public function modifierinfo($login, $email, $nom, $prenom)
{
        $db = $this->_db;
        $id = $_SESSION['id'];
        if(!empty($login) && !empty($email) && !empty($nom) && !empty($prenom))
        $requete = $db->prepare("UPDATE utilisateurs SET nom='$nom' , prenom='$prenom' `login`='$login' `email`='$email' WHERE id=$id");
        $requete->execute();
        $_SESSION['nom'] = $nom;
        $_SESSION['prenom'] = $prenom;
        $_SESSION['email'] = $email;
        $_SESSION['login'] = $login;
        echo 'Informations mise a jours ! ';
}
```

Comme pour les précédentes requête je récupère l'attribut qui met permet de me connecter a ma base de données et je vérifie que les paramètres ne sont pas vides

j'attribue une variable a **$_SESSION['id']** afin de pouvoir la réutiliser après.
$_SESSION est une variable globale automatique. Cela signifie simplement que cette variable est disponible dans tous les contextes du script.
Je prépare ensuite ma requête, et demande un UPDATE de la table 'utilisateurs' et annonce de remplacer la valeur de la colonne 'nom' par ce qui a été rentrer dans la variable **$nom** depuis le formulaire, pareil pour prenom, login et email, LA ou l'id correspond à **$id** (la variable de session vu précédemment).
Une fois cette requête préparée et exécutée, on remplace la valeur des variables de sessions par ce qui a été mit dans les variables **$nom, $prenom,** etc .. afin de ne pas avoir à déconnecter et reconnecter l'utilisateur pour changer la valeur de ses variables de sessions.

Bien sûr, je fais appel à la fonction sur la page concernée (*ici la page profil*).

DELETE

Cette opération est gérée par la requête qui porte le 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 des produits en base de données.

nom de l'article	Prix	Supprimer
iPad Pro 12,9 4e génération (2020) 256 Go	1044 €	Supprimer
iPad 9,7 5e génération (2017) 128 Go	325 €	Supprimer
iPad Pro 12,9" 4e génération (2020) 256 Go	1080 €	Supprimer
iPad Pro 11 1e génération (2018) 64 Go	612 €	Supprimer
iPad 9,7 5e génération (2017) 32 Go	249 €	Supprimer
AirPods (2ème génération) avec boitier de charge	129 €	Supprimer

```php
public function delete_article($id)
{
    $db = $this->_db;

    $requete2 = $db->prepare("DELETE FROM article WHERE id_article='$id'");
    $requete2->execute();
}
```

Je commence par récupérer la connexion a ma base de données (ici **$db**)
La fonction prend en paramètre 1 variable , ici l'id de l'article que je souhaite supprimée, ensuite je rédige ma requête SQL, je lui spécifie que je souhaite supprimer de la table 'article' le produit qui à pour id celui que je récupère et que je passe en paramètre dans ma fonction, je récupère cet id dans ma page admin via une autre fonction qui me permet d'afficher tous les articles (voir READ), ensuite grâce à Javascript et plus précisément JQuery et Ajax (Voir AJAX) je supprime l'article que je veux en cliquant sur 'supprimer' à droite du nom et du prix de l'article.

Intégration CSS

L'un des objectifs majeurs de CSS est de permettre la mise en forme hors des documents. Il est par exemple possible de ne décrire que la structure d'un document en HTML, et de décrire toute la présentation dans une feuille de style CSS séparée.

Les styles sont appliqués au dernier moment, dans le navigateur web des visiteurs qui consultent le document. Cette séparation fournit un certain nombre de bénéfices, permettant d'améliorer l'accessibilité, de changer plus facilement de présentation, et de réduire la complexité de l'architecture d'un document.

L'intégration consiste à transformer la charte graphique composées d'images en langage HTML et CSS.

Avant toute chose, il faut savoir que le CSS peut être inséré à trois endroits très distincts d'une page web :

- Dans l'attribut style de la balise nécessitant une mise en page particulière
- Entre deux balises
- Dans un fichier externe ayant généralement pour extension .css

Ces deux dernières méthodes permettent d'appliquer un style à toutes les balises d'un même type (des liens par exemple) sans avoir à le spécifier pour chaque balise.

Pour mon cas l'intégration s'est faite dans l'attribut style.

```
34      <link rel="stylesheet" href="css/about.css">
35      <link rel="stylesheet" href="css/admin.css">
```

Ensuite j'attribue une 'class' a une div, une section, etc ..

Ici une div sans class :

```
5   <div>
6   ADMIN
7   </div>
```

Résultat:

ADMIN

Maintenant on attribue une class a cette div :

```
5   <div class="case_admin">
6   ADMIN
7   </div>
```

On va changer la taille du texte , la couleur , la police ,etc .., depuis le fichier css, linké au préalable (about.css)

```
470   .case_admin
471   {
472       font-size: 25px;
473       font-weight: 800;
474       font-family: 'Poppins', sans-serif;
475       color: rgb(1, 113, 187);
476   }
```

Résultat :

ADMIN

RESPONSIVE

Un site responsive est un site qui est conçu et développé de façon à pouvoir s'adapter à toutes les résolutions d'écran. C'est donc un seul et même site qui peut être consulté sur ordinateur, sur smartphone ou sur tablette.

Grâce à l'utilisation des media queries, j'ai mis en place des éléments "responsive". Cela permet une utilisation du site aussi bien sur ordinateur que sur mobile.

MEDIA QUERIES :

```
@media screen and (max-width: 929px) {

  .produit-phare {

      width: 200px;
      height: 230px;
  }

  .produit-phare2 {

      width: 200px;
      height: 200px;

  }

}

@media screen and (max-width: 700px) {

  .bloc-phare{
      flex-direction: column;

  }

  .produit-phare
  {
      margin: 10px;
}

}
```

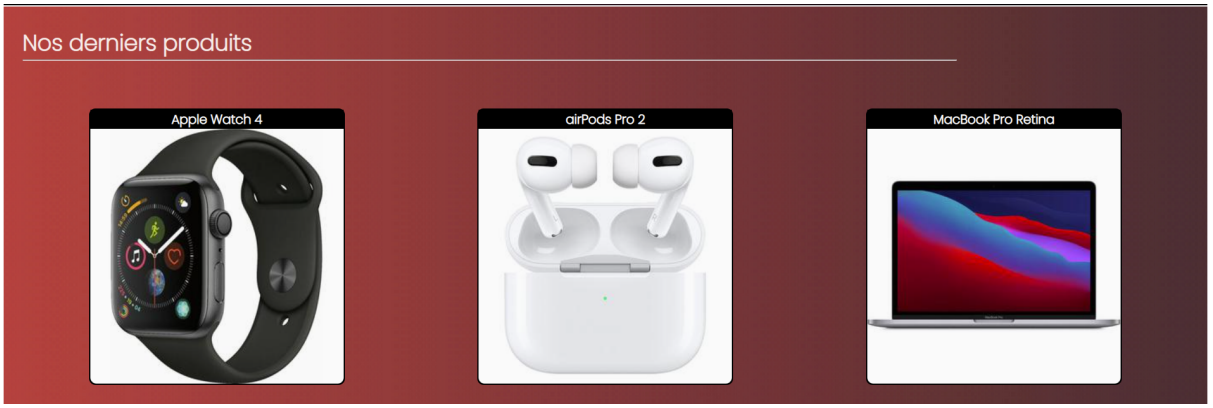
Ici, je défini que la largeur maximale que je souhaite est de **929px** donc ces paramètres seront affectés aux classes ciblé qu'une fois que la largeur d'écran passe en dessous de **929px**, la taille des images seront réduite à 200px.

Ensuite, pour ce qui est des écrans encore plus petits, je défini que la largeur maximum que je souhaite est **700px**,

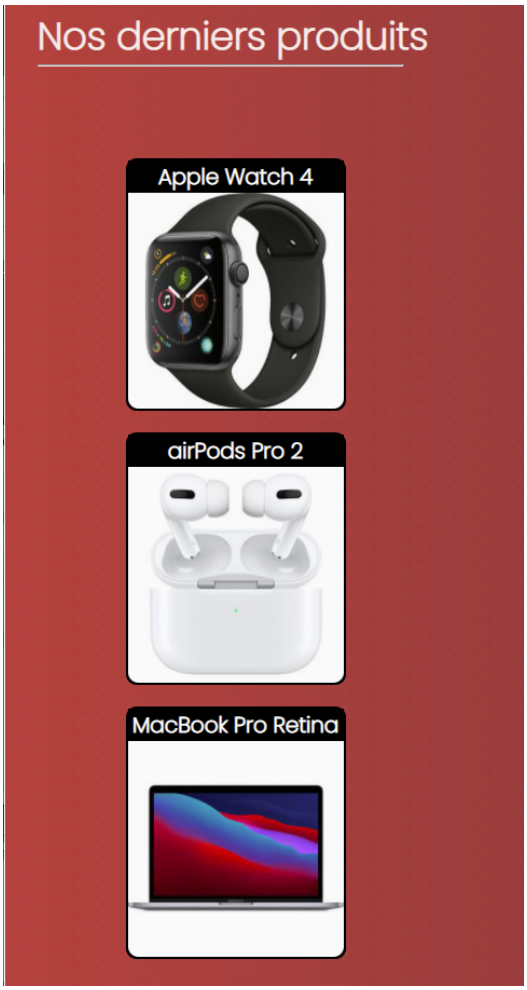
a partir de la, l'affichage des images ne se fera plus horizontalement mais verticalement.

Exemple :

Taille supérieur à **700px** :



Taille inférieur à **700px** :



JAVASCRIPT

Le JavaScript est un langage de programmation utilisé notamment lors de la conception de sites web et d'applications. Il est particulièrement utile pour concevoir des sites dynamiques.

Dans le but d'agrémenter l'expérience utilisateur et d'apporter du dynamisme au site, nous avons eu recours au langage Javascript, plus précisément de JQuery qui est une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web.

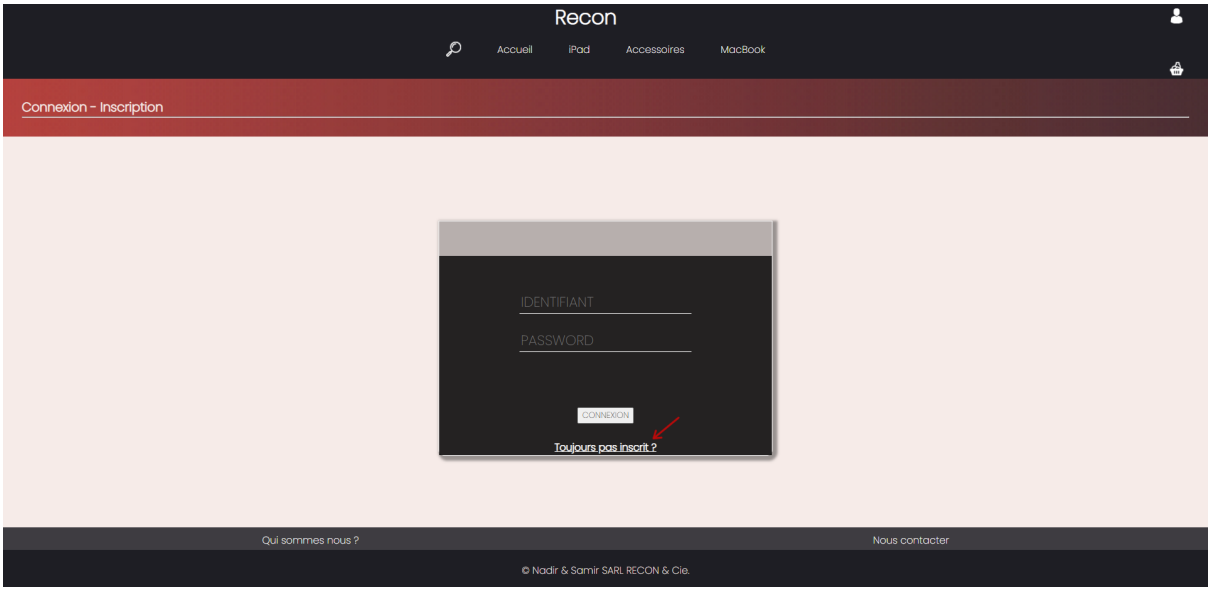
Il existe plusieurs façons de commencer à utiliser jQuery sur votre site Web. Vous pouvez:

- Téléchargez la bibliothèque jQuery sur jQuery.com
- Inclure jQuery à partir d'un CDN, comme Google

J'ai optez pour la deuxième option, j'ai inclu la bibliothèque jQuery dans mon header dans la balise <head>

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

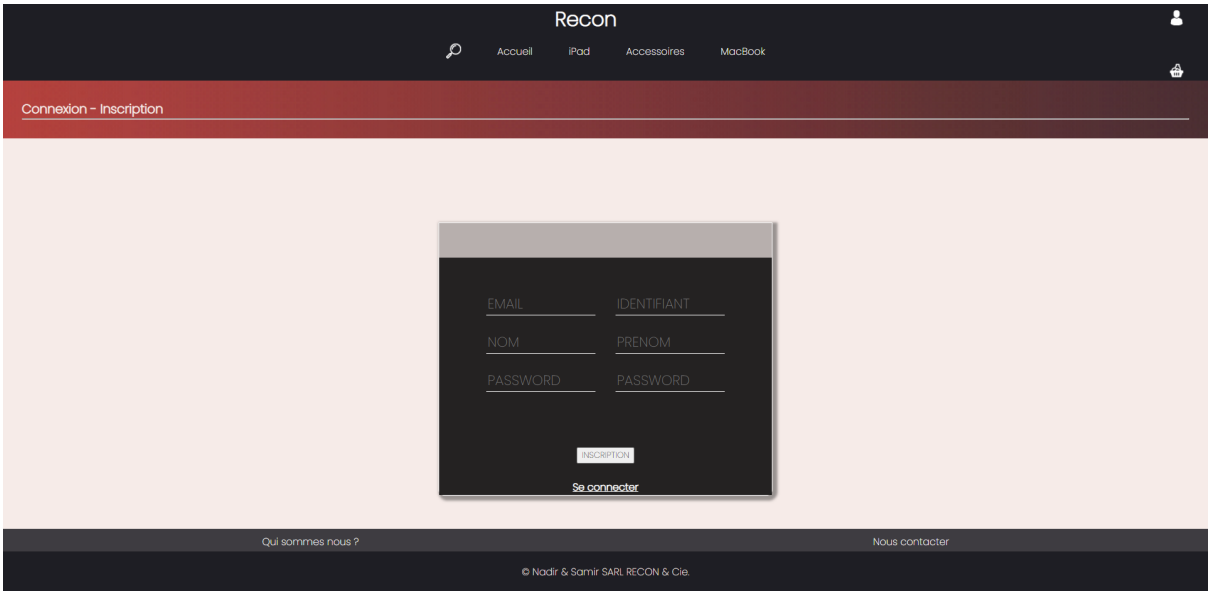
L'utilisateur se retrouve donc avec une expérience plus agréable avec un site dynamique, par exemple il peut switcher entre le formulaire de connexion et d'inscription sans changer de page, sans rechargement.



En cliquant sur '***toujours pas inscrit***' le formulaire de connexion va disparaître grâce à la fonction **css()** de jQuery, ou je le fais disparaître

Et je fais apparaitre le formulaire d'inscription avec la fonction **slideDown()**

slideDown() *fait défiler les éléments sélectionnés.*



Si l'utilisateur clique à nouveau sur '***Se connecter***', le formulaire d'inscription disparaîtra, et le formulaire de connexion apparaîtra, ainsi de suite.

Extrait du code :

HTML :

```
            <div class="form_connexion" id = 'form_connexion'  autocomplete = 'off'>
                <form action = '' method = 'post'>

                <input type = 'text'  placeholder = 'IDENTIFIANT'  id ='login'><br>

                <input type = 'password'  placeholder = 'PASSWORD' id ='pass'> <br>

                </form>
                <div class = 'button_center'>
                    <button type = 'submit' id = 'connexion_button'>CONNEXION</button>
                </div>
                <p id = 'inscrivez-vous'>Toujours pas inscrit ?</p>
            </div>

<div id = 'form_inscription'>
            <form action = '' method = 'post' autocomplete = 'off'>
                <input type = 'email' placeholder = 'EMAIL' id ='mail' name = 'mail'>
                <input type = 'text' placeholder = 'IDENTIFIANT' id = 'identifiant' name = 'identifiant ' ><br>
                <input  type = 'text' placeholder = 'NOM' id = 'nom' name = 'nom'>
                <input   type = 'text' placeholder = 'PRENOM' id = 'prenom' name = 'prenom'><br>
                <input type = 'password' placeholder = 'PASSWORD' id ='password-inscription' name = 'password-inscription'>
                <input type = 'password' placeholder = 'PASSWORD' id = 'confirm-password' name = 'confirm-password'> <br>
                <input type = "submit" name = 'inscription' value = 'Inscription'>
            </form>

            <div class = 'button_center'>
            <button type = 'submit' id = 'inscription_button'>INSCRIPTION</button>
            </div>

            <p id = 'connectez-vous'>Se connecter</p>
</div>
```

J'attribue à la div contenant le formulaire de connexion un id = ***'form_connexion'***, et au message ***'toujours pas inscrit?'*** un id = ***'inscrivez-vous'***

J'attribue à la div contenant le formulaire d'inscription un id = ***'form_inscription'*** et au message ***'se connecter'*** un id=***'connectez-vous'***

Du côté CSS : J'attribue un ***display : none*** au formulaire d'inscription qui permet de le rendre invisible à l'ouverture de la page de connexion.

jQuery :

```
2    /* FAIRE APPARAITRE LE FORMULAIRE D'INSCRIPTION */
3    $('#inscrivez-vous').click(function(){
4        $('#form_connexion').css("display", "none")
5        $('#form_inscription').slideDown(1000)
6
7    })
8
9    /* FAIRE APPARAITRE LE FORMULAIRE DE CONNEXION*/
10       $('#connectez-vous').click(function(){
11           $('#form_inscription').css("display", "none")
12           $('#form_connexion').slideDown(1000);
13   })
```

Je récupère l'id '**inscrivez-vous**' et j'indique que lorsque je clique sur ce texte, alors le formulaire de connexion (récupérer grâce à son id) se voit attribuer un ***display : none*** qui le fait disparaître et je fais apparaitre le formulaire d'inscription (recuperer grace a son id) à l'aide de **slideDown()**.

Je fais l'inverse pour afficher le formulaire de connexion si le formulaire d'inscription est affiché.

Ces fonctionnalités sont aussi présente sur d'autre page du site (admin , profil ..)

FORCE DE MOT DE PASSE

La création d'un mot de passe fort répond à plusieurs critères :

- Il doit comporter au moins huit caractères.
- Il faut également qu'il combine des lettres minuscules, majuscules, des chiffres.
- etc…

A l'aide de jQuery j'ai réuni certaines de ces conditions lors de l'inscription de l'utilisateur.

Code HTML :

```
<input type = 'password' placeholder = 'PASSWORD' id ='password-inscription' name = 'password-inscription'>
```

Code jQuery :

```
68          $('#inscription_button').click(function(){
69
70          var pass = $('#password-inscription').val();
71          var passlong = $('#password-inscription').val().length;
72
73              if(passlong > 7){
74                  if(/[A-Z]/.test(pass)){
75                      if(/[0-9]/.test(pass)){
```

Ici lorsque l'utilisateur clique sur le bouton pour s'inscrire, je récupère la valeur entrée dans l'input password ainsi que sa taille (encore grâce à son id).

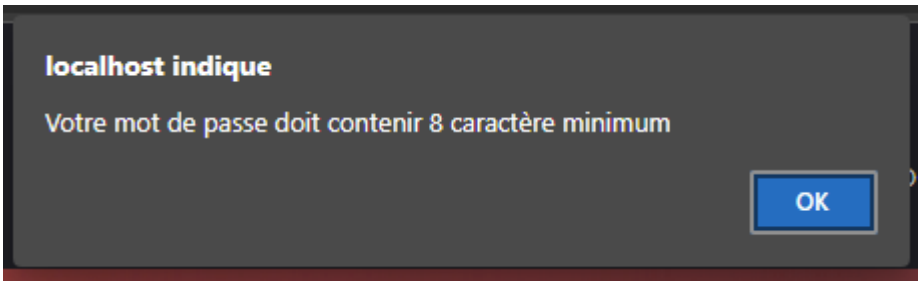
je stock la valeur dans une variable 'pass' et la taille dans la variable '**passlong**'

Ensuite je vérifie si '**passlong**' comprend plus de 7 caractères.
Si oui, d'autre vérification sont mises en place,
je vérifie si le mot de passe contient une majuscule, si oui, je vérifie si le mot de passe contient un chiffre, si oui , alors la requête est exécutée.
Si non, j'envoi une alerte avec la fonction **alert()** contenant un message que j'ai prédéfini en fonction de l'erreur, je vide aussi les inputs correspondant.

La fonction **alert()** affiche une boîte d'alerte avec un message spécifié et un bouton OK, celle-ci change d'apparence en fonction du navigateur et de l'appareil (mobile , PC …)

```
107                     }else{
108                         alert('Votre mot de passes doit contenir au moins un chiffres')
109                         $('#password-inscription').val('');
110                         $('#confirm-password').val('');
111                     }
112 
113                 }else{
114                     alert('Votre mot de passes doit contenir au moins une majuscule')
115                     $('#password-inscription').val('');
116                     $('#confirm-password').val('');
117                 }
118             }else{
119                 alert('Votre mot de passe doit contenir 8 caractère minimum')
120                 $('#password-inscription').val('');
121                 $('#confirm-password').val('');
```

Exemple d’alerte :



AJAX

AJAX consiste à charger des données en arrière-plan et à les afficher sur la page Web, sans recharger toute la page.

C'était impératif pour moi d'intégrer du ajax dans ma boutique, afin d'éviter un chargement de la page a chaque fois qu'une requête été effectué.

Code HTML :

```
<form action = '' method = 'post'>

<input type = 'text'  placeholder = 'IDENTIFIANT'  id ='login'><br>

<input type = 'password'  placeholder = 'PASSWORD' id ='pass'> <br>

</form>
<div class = 'button_center'>
    <button type = 'submit' id = 'connexion_button'>CONNEXION</button>
</div>
```

J 'attribue des “id” aux balises que je souhaite cibler dans mon fichier JS.

Code jQuery :

```
133     $('#connexion_button').click(function(){
134         if($('#login').val() != '' && $('#pass').val() != ''){
135             $.ajax({
136                 type:'POST',
137                 url:'ajax/utilisateurs.php',
138                 data:{
139                     login:$('#login').val(),
140                     pass:$('#pass').val(),
141                      connect:'connect'
142                 },
143                 datatype: 'text',
144                 success:function(response2){
145                     console.log(response2);
146                     alert(response2);
147                     if(response2 == "Connexion établie !")
148                     window.location.replace('profil.php');
149                 }
150             })
151 
152         }else{
153             alert('Veuillez remplir tout les caractères');
154         }
155     })
```

Ici, lorsque l'utilisateur cliquera sur l'élément qui contient l'id '**connexion_button**' (ici le button , voir code html en haut)

Une vérification est d'abord faite pour savoir si les inputs ne sont pas vides.

Si les inputs ne sont pas vide, je lance AJAX

je spécifiée dans '**url :**' la page sur laquelle je veux que l'AJAX envoie les données, ici je me redirige dans le dossier '**ajax**' et je sélectionne le fichier '**utilisateur.php**' qui est une page intermédiaire qui traitera les données pour les envoyer à ma fonction PHP dédiée à la connexion.

Dans '**data :** ' je specifie le nom de chaque valeur entrée dans les inputs, '**login**' pour la valeur entrée dans l'input contenant l'id '**login**', de même pour le reste.

'**connect**' jouera ici le rôle de vérification lorsque l'on appelera la fonction avec php.

```php
// CONNEXION DE L'UTILISATEURS

if(isset($_POST['connect']))
{
    $user->connexion($_POST['login'], $_POST['pass']);
}
```

Nous nous trouvons dans la page mentionné dans url : ajax/utilisateurs.php c'est ici que nous faisons appel à notre fonction, les paramètres $_POST['login'], $_POST['pass'] correspondent à ce qui a été entré dans les inputs plus haut. Enfin ces données parcourt ma fonction php connexion() :

```php
public function connexion($login, $password)
{

    $db = $this->_db;

    $login = htmlspecialchars($login);
    $password = htmlspecialchars($password);

    $_login = trim($login);
    $_password = trim($password);

    $requete = $db->prepare("SELECT * FROM utilisateurs WHERE login = '$_login'");
    $requete->execute();

    $verification = $requete->RowCount();
            if($verification == 1){
                $info = $requete->fetch();
                if( $_password == password_verify($_password, $info['password'])){
                    $_SESSION['login'] = $info['login'];
                    $_SESSION['password'] = $info['password'];
                    $_SESSION['nom'] = $info['nom'];
                    $_SESSION['prenom'] = $info['prenom'];
                    $_SESSION['email'] = $info['email'];
                    $_SESSION['id'] = $info['id'];
                    $_SESSION['id_droit'] = $info['id_droit'];
                    echo "Connexion établie !";
                }else{
                echo "Mauvais mot de passe !";
                }
            }else{
            echo "Cette identifiant n'existe pas !";
            }
}
```

Une fois que toute les vérification sont effectuées, je stocke les informations dans des variables '**$_SESSION**'

Ensuite ma fonction me retourne le message qui correspond à mon resultat

Si mauvais mot de passe : "mauvais mot de passe !"

le message sera renvoyer dans la variable response2 (voir plus haut AJAX)

qui sera envoyé en avec la fonction **alert()** .

```
146                     alert(response2);
147                     if(response2 == "Connexion établie !")
148                     window.location.replace('profil.php');
149                 }
```

J'ajoute la condition que si la fonction me retourne "Connexion établie !"

Alors j'utilise la fonction replace, qui permet de rediriger l'utilisateur connecté sur la page profil.php.

Veille sur les vulnérabilités de sécurité.

La sécurité d'un site web exige de la vigilance dans tous les aspects de sa conception et de son utilisation.

INJECTION SQL

Cette vulnérabilité permet à un attaquant d’injecter des données non maîtrisées qui seront exécutées par l’application et qui permettent d’effectuer des actions qui ne sont normalement pas autorisées.
Ce type d’attaque s’effectue généralement grâce aux champs présents dans les formulaires.
Dans le cas d’une attaque par injection SQL, au lieu de mettre un nom d'utilisateur et un mot de passe sur une page de connexion, un utilisateur malveillant entrera des données directement interprétées par le moteur SQL, ce qui lui permettra de modifier
le comportement de votre application.

Comment s’en proteger ? :

- Validez les entrées

Cela consiste à limiter ce que l’utilisateur peut mettre dans la zone de texte.

- utiliser des fonction du genre htmlspecialchars() ,

Cela n'empêchera pas l'injection, mais c'est une mesure que vous pouvez mettre en place pour limiter des attaques de base. En effet, les caractères spéciaux spécifiques à certains langages ne pourront pas être utilisés.

- Préparez les requêtes SQL (vue plus haut)

FAILLE XSS

Une faille XSS consiste à injecter du code directement interprétable par le navigateur Web, comme, par exemple, du JavaScript ou du HTML. Cette attaque ne vise pas directement le site comme le ferait une injection SQL mais concerne plutôt la partie client c'est-à-dire vous (ou plutôt votre navigateur). Ce dernier ne fera aucune différence entre le code du site et celui injecté par le pirate, il va donc l'exécuter sans broncher. Les possibilités sont nombreuses : redirection vers un autre site, vol de cookies, modification du code HTML de la page, exécution d'exploits contre le navigateur : en bref, tout ce que ces langages de script vous permettent de faire.

Comment s'en proteger ? :

La meilleure défense contre les vulnérabilités XSS est de supprimer ou désactiver toutes les balises qui peuvent potentiellement contenir des instructions pour exécuter du code. Pour HTML cela inclut les tags comme <script>, <object>, <embed>, et <link>.

FAILLE INCLUDE

Cette faille est par définition présente sur la plupart des sites web utilisant la fonction include « sans précaution ». En effet, le langage php est l'un, si ce n'est LE langage le plus utilisé sur le web mais aussi le plus puissant car celui-ci permet un contrôle quasi total sur l'ensemble du serveur.
Parmi ses fonctions on compte donc le fameux « include ». Cette fonction permet, comme son nom l'indique, d'inclure un fichier dans un autre. Elle est généralement utilisée sur les sites nécessitant un appel de page réutilisé. Par exemple, si toutes vos pages requièrent la page mysql.php dans laquelle sont contenues vos informations mysql, il convient de l'inclure dans chacune de vos pages plutôt que de retaper l'équivalent de mysql.php dans toutes celles-ci.

Pour détecter une faille PHP include sur un site web, il suffit de tester tous les paramètres possibles d'une URL et d'observer le résultat.

La force brute

L'attaque par force brute permet à un pirate de tester des identifiants et mots de passe de façon très rapide au moyen d'un script qui envoie des milliers de tentatives sur un formulaire jusqu'à arriver à ses fins.

Comment s'en proteger ? :

Pour ralentir cette attaque, il est possible de mettre une ligne de code très simple

sleep(1)

Cette ligne ralentit les tentatives en ajoutant un délai d'une seconde au traitement du formulaire, délai imperceptible pour un utilisateur.

La seconde méthode permet de définir un nombre maximal de tentatives dans un délai donné en stockant un timestamp dans la base de données.

L'Upload

Lors de l'upload de fichiers, il serait aisé pour un pirate de faire passer un fichier PHP pour une image en le renommant par exemple "scriptpirate.php.jpg".

De nombreux sites se feraient piéger par ce type de fichier.

Comment s'en proteger ? :

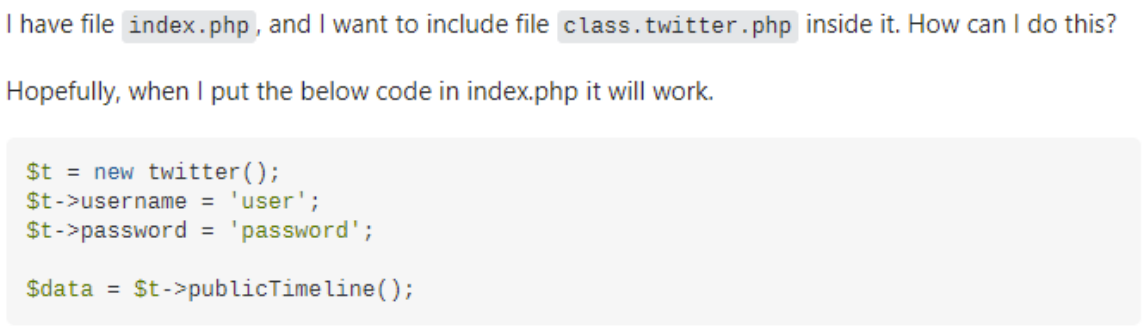
- Vérifier le type mime du fichier

- Toujours renommer le fichier en utilisant, par exemple, un timestamp
- Ne pas faire confiance à l'extension du fichier
- Utiliser un dossier d'upload qui n'est pas à la racine du site et dont vous maîtrisez les permissions

Recherche effectuées à partir d'un site anglophone

Lors de mon apprentissage avec la programmation orientée objet je ne savais pas comment appeler une classe depuis une autre page, j'ai donc effectué une recherche sur google " include class PHP"

Je suis tombé sur ce cette page ou un utilisateur rencontre le même soucis que moi



I have file `index.php`, and I want to include file `class.twitter.php` inside it. How can I do this?

Hopefully, when I put the below code in index.php it will work.

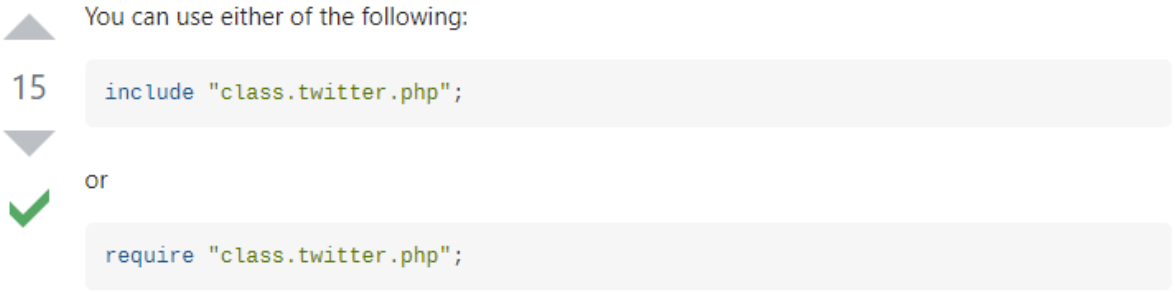
```
$t = new twitter();
$t->username = 'user';
$t->password = 'password';

$data = $t->publicTimeline();
```

(Capture d'ecran issu de stackoverflow.com)

Ici l'utilisateur dit qu'il a un fichier 'index.php' et qu'il veut y inclure son fichier 'class.twitter.php' et nous montre un extrait de son code.

Un utilisateur lui répond :



15

You can use either of the following:

```
include "class.twitter.php";
```

or

```
require "class.twitter.php";
```

Il lui dit qu'il peut utiliser la fonction include() ou la fonction require().

Conclusion

Ce projet est l’aboutissement de cette année, afin d'acquérir toute ces connaissance, j’ai effectué de nombreux projet qui avait tous un liens avec ce que j’ai fais sur cette boutique en passant par :

- La gestion de contenu
- Le système de connexion
- AJAX
- Le Responsive
- Utilisation de POO
- Maquetter un site
- Modéliser une base de données
- etc …