

Date : 08/01/2023

Rapport - Web

Requêtes préparées et pages principales

Par **FERNANDEZ** Mickael, **DOURLENT** Maxime, **STRAPUTICARI** Luca et
VIGNAL Alexandre (Groupe G2B-10)



Sommaire :

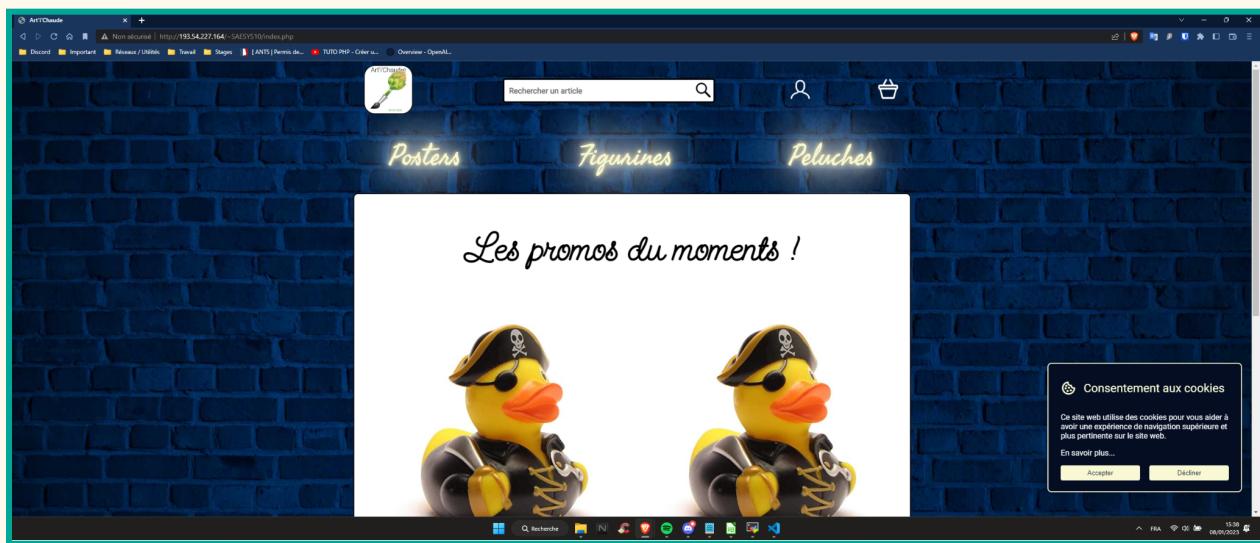
Sommaire :	2
Améliorations des pages existantes	3
1. Cookies - aspect visuel	3
2. Cookies - aspect fonctionnel	5
3. Formulaire de connexion - fonctionnel & visuel	8
4. Page index.php - ensemble	10
Ajout de pages et différentes requêtes	12
1. Pages du footer - visuel & fonctionnel	12
2. Récupération des articles en fonction de la catégorie correspondante - visuel & fonctionnel	18
3. Barre de recherche concernant les articles - visuel & fonctionnel	19
4. Redirection des images de produits dans la page produit.php	21
5. Page produit.php et requêtes	22

Améliorations des pages existantes

1. Cookies - aspect visuel

Auparavant, sur le sprint précédent, le cookie n'était alors constitué uniquement d'une "checkbox" qui permettait à l'utilisateur, s'il le souhaitait, à ce qu'on se souvienne de lui s'il venait à se déconnecter. Ce cookie permettait alors de remplir automatiquement le champ de l'adresse mail sans avoir à le renseigner, de n'importe quel utilisateur. Néanmoins, le consentement de l'utilisateur à utiliser les cookies n'était pas formel. Autrement dit, rien ne lui était indiqué sur son utilisation à valider les cookies et sur les données recueillies par celui-ci.

C'est pourquoi, lorsqu'un internaute vient à consulter le site web, une pop-up dédiée à ce consentement est automatiquement mise à disposition que voici :

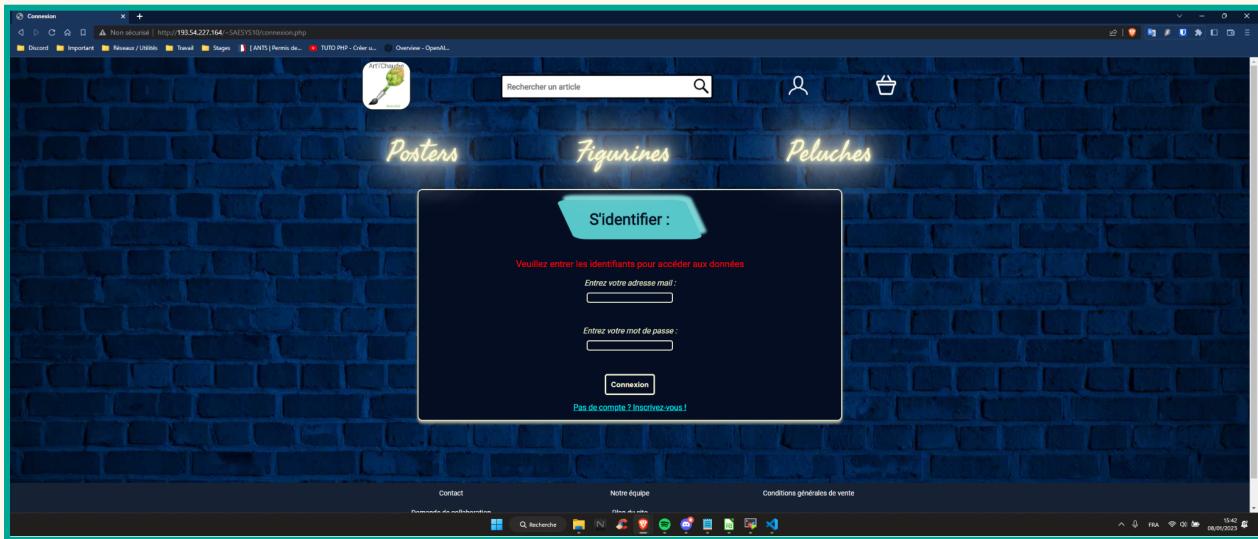


Ainsi, plusieurs possibilités sont disponibles :

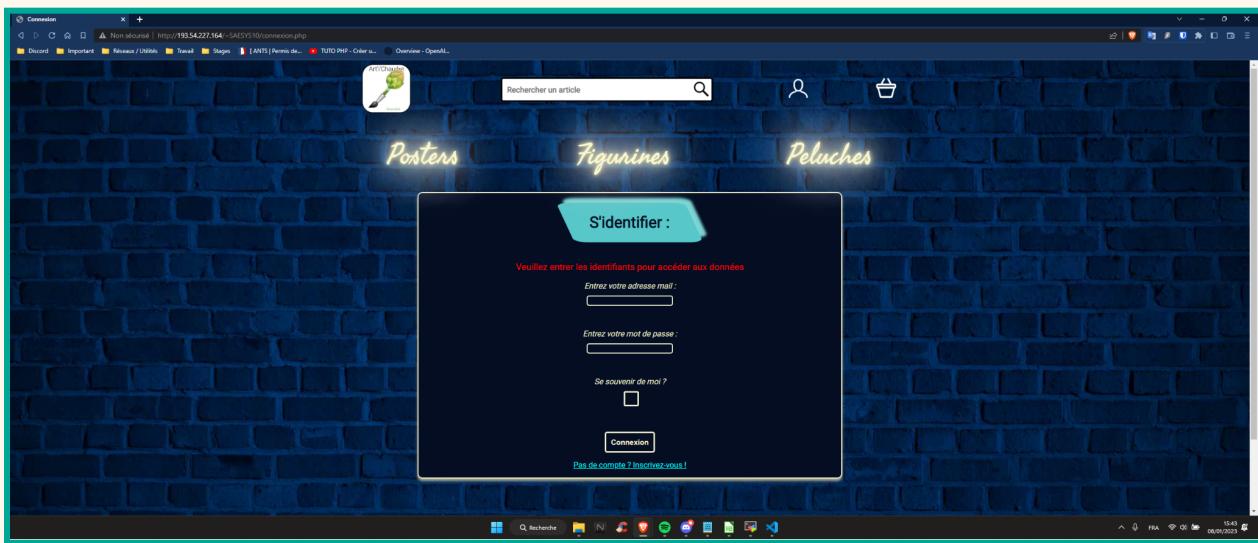
- Si l'utilisateur décline la pop-up, celle-ci réapparaîtra à chaque fois que celui-ci reviendra sur la page principale et la checkbox dédiée à la mémorisation de l'adresse mail de l'utilisateur ne sera pas disponible;
- Si celui-ci ignore la pop-up, il en sera de même.
- Si celui-ci accepte la pop-up, celle-ci sera effective pendant quelques minutes et la checkbox sera alors affichée, ayant alors consenti à l'utilisation des cookies. S'il souhaite en savoir plus, il peut à la fois cliquer sur le lien hypertexte de "En savoir plus" ou accéder directement à la page traitant des cookies depuis le footer.

En voici un exemple d'utilisation du formulaire de connexion, lorsque l'utilisation consent ou non aux cookies :

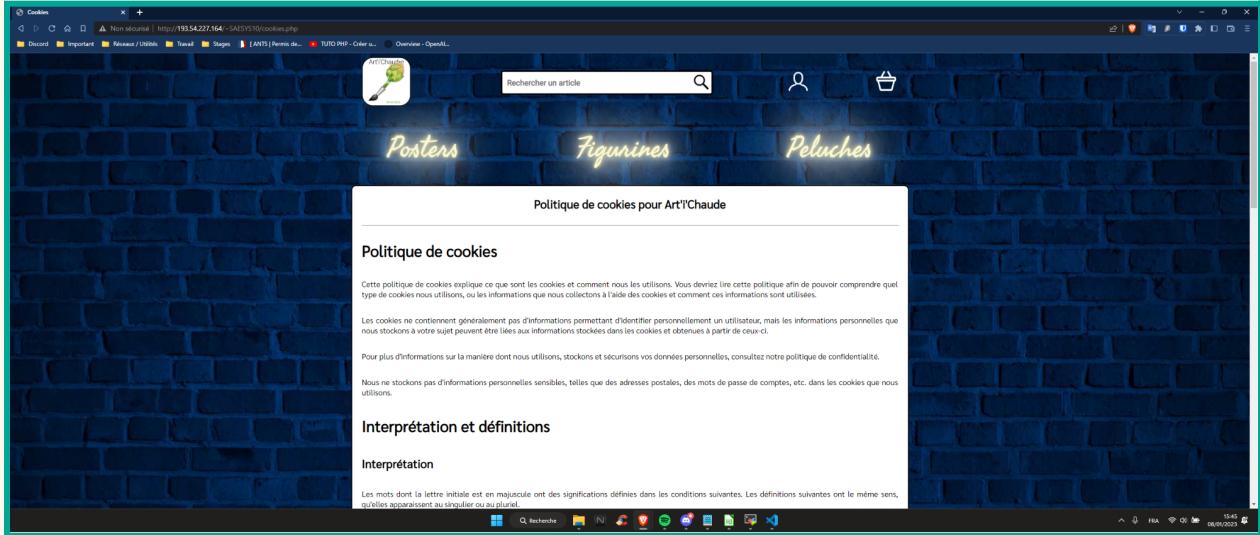
Sans consentement des cookies :



Avec consentement des cookies :



Exemple de la page des cookies :



2. Cookies - aspect fonctionnel

Concernant l'aspect fonctionnel et rendre le cookie de la pop-up fonctionnel, nous avons varié sa création en le réalisant en JavaScript. Cela n'a aucune contrainte d'utilisation car les cookies étant stockés dans la même partie "Application", une interaction JavaScript-PHP est alors possible. En voici son code HTML, CSS & JavaScript :

HTML :

```
<script src="script.js" defer></script>
```

```
<div id="cookieContainer">
  <div id="cookieTitle">
    <i class="bx bx-cookie"></i>
    <h2>Consentement aux cookies</h2>
  </div>

  <div id="dataCookie">
    <p>Ce site web utilise des cookies pour vous aider à avoir une expérience de navigation supérieure et plus pertinente sur le site web.</p>
    <p><a href="cookies.php">En savoir plus...</a></p>
  </div>

  <div id="buttonsContainer">
    <button class="buttonCookie" id="acceptBtn">Accepter</button>
    <button class="buttonCookie" id="declineBtn">Décliner</button>
  </div>
</div>
```

CSS :

```

/*---| Cookie |---*/

#cookieContainer {
    position:fixed;
    bottom:50px;
    right:-400px;
    max-width:345px;
    width:100%;
    background-color: ##040d22;
    border-radius: 8px;
    border:2px solid #f8f9d4;
    padding:15px 25px 22px;
    transition: right 0.3s ease;
    box-shadow:0 5px 10px rgba(0,0,0,0.1);
}

#cookieContainer.show {
    right:20px;
}

#cookieTitle {
    display:flex;
    align-items:center;
    column-gap:15px;
}

#cookieTitle > i {
    color: #f8f9d4;
    font-size:32px;
}

#cookieTitle > h2 {
    color: #f8f9d4;
    font-weight:500;
}

#dataCookie > p{
    color: white;
    font-size:16px;
}

#dataCookie > p > a {
    color: white;
    text-decoration:none;
}

#dataCookie > p > a:hover {
    color: #f8f9d4;
    text-decoration:underline;
}

#buttonsContainer {
    display:flex;
    align-items:center;
    justify-content:space-between;
    margin-top:16px;
    width:100%;
}

.buttonCookie {
    cursor:pointer;
    width: calc(100% / 2 - 10px);
    padding:8px 0;

    border:none;
    border-radius: 4px;

    background-color: #f8f9d4;
    color: ##040d22;
}

.buttonCookie:hover {
    background-color: white;
}

```

JavaScript :

```

1  const cookieBox = document.querySelector('#cookieContainer'), buttons = document.querySelectorAll('.buttonCookie');
2
3  const executeCodes = () => {
4      //If cookie contains Artichaude it will be returned and below of this code will not run
5      if(document.cookie.includes('memorisedArtichaude')) return;
6
7      cookieBox.classList.add('show');
8      buttons.forEach((button) => {
9          button.addEventListener('click', () => {
10
11              cookieBox.classList.remove('show');
12
13              //If button has acceptBtn id
14              if(button.id == "acceptBtn") {
15                  //Set cookie for 1h
16                  document.cookie = "cookieFrom= memorisedArtichaude; max-age=" + 300;
17              }
18          });
19      });
20  };
21
22  window.addEventListener("load", executeCodes);

```

Concrètement, cette partie est intéressante à commenter car rien de particulier ou de différent par rapport aux anciens sprints est à mentionner sur les parties “frontend”.

Ici, dans deux constantes nommées “cookieBox” et “buttons”, on vient récupérer la classe des boutons et l’ID du cookie, correspondant en réalité, à l’intégralité de celui-ci.

Dans une dernière constante nommée “executeCodes”, celle-ci permettra d’enclencher le processus de fonctionnement du cookie. Pour ce faire, il vérifie dans un premier temps si le cookie existe déjà. Si c’est le cas, le code s’arrête ici et le reste ne sera jamais exécuté. Sinon, il ajoute dans la constante “cookieBox”, une classe nommée “show” qui permettra, par défaut, d’afficher le cookie lorsque l’internaute sera situé sur la page principale. Lors du retrait de cette classe par l’intermédiaire d’une interaction avec l’un des deux boutons, cette classe sera retirée et permettra de ne plus afficher la pop-up.

Enfin, s’il accepte les cookies par le biais du bouton “acceptBtn”, celui-ci sera effectif pendant quelques minutes (ici, le nombre correspond au nombre de secondes). A la fin, la constante sera exécutée.

3. Formulaire de connexion - fonctionnel & visuel

Dans l'ancien sprint, la présence d'une “checkbox” destinée à confirmer si l'utilisateur était un administrateur était présente. Pour autant, cela alourdissait le code PHP qui devait prendre en compte cette checkbox et vérifier à l'aide d'une requête préparée que le rôle de l'utilisateur correspondait au rôle d'administrateur ayant été coché, en plus de la vérification de l'existence du compte dans cette même base de données.

Après refonte de celle-ci et mentionné dans le rapport BD de la semaine, cette checkbox n'existe plus. Ainsi, la vérification s'effectuera uniquement sur l'existence du compte. Ainsi, la session prendra en compte les informations nécessaires de l'utilisateur qui seront figurés dans sa page de profil, dont son nom et prénom ou encore son rôle.

Le formulaire de connexion ayant déjà été montré auparavant, en voici l'aspect visuel de cette page de profil et fonctionnel :

Pour un administrateur & artiste :



Pour un client :



HTML & CSS : Simple retrait de la checkbox et de ses anciennes propriétés utilisées pour sa forme.

PHP : Retrait de la vérification de l'interaction avec la checkbox et ajout dans la variable superglobale de session, du rôle de l'utilisateur et traitement à partir d'une table "Acteur" :

```

1  <?php
2      // on inclut le fichier de connexion à la base Oracle
3      require_once('connect.inc.php');
4
5      session_start();
6
7      if(isset($_POST['Connexion'])) {
8
9          if(isset($_POST['adresseMail']) && isset($_POST['motDePasse'])) {
10
11              //On crée une requête paramétrée
12              $req = "SELECT * FROM ACTEUR WHERE MAILACTEUR = :pMailActeur";
13
14              //On prépare la requête
15              $clientRecup = oci_parse($connect, $req);
16
17              //On récupère les valeurs renseignées par l'utilisateur dans les champs de texte
18              $mailActeur = htmlspecialchars($_POST['adresseMail']);
19              $mdpActeur = htmlspecialchars($_POST['motDePasse']);
20
21              //On lie les valeurs aux paramètres de la requête
22              oci_bind_by_name($clientRecup, ":pMailActeur", $mailActeur);
23
24              //On exécute la requête
25              $result = oci_execute($clientRecup);
26
27              // si erreur de requête alors affichage...
28              if (!$result) {
29                  $e = oci_error($clientRecup); // on récupère l'exception liée au pb d'exécution de la requête (Violation PK par exemple)
30                  print htmlentities($e['message']). ' pour cette requête : '. $e['sqltext'];
31              } else {
32
33                  $clientDatabase = oci_fetch_assoc($clientRecup);
34
35                  if(empty($clientDatabase)){
36
37                      header('Location: ./connexion.php?msgErreur=Compte inexistant');
38                      exit;
39                  }
40
41                  if(password_verify($mdpActeur, $clientDatabase['MDPACTEUR'])) {
42
43                      $_SESSION['utilisateur'] = $clientDatabase['ROLEACTEUR'];
44                      $_SESSION['prenom'] = $clientDatabase['PRENOMACTEUR'];
45                      $_SESSION['nom'] = $clientDatabase['NOMACTEUR'];
46
47                      if(isset($_POST['memo'])) {
48                          setcookie('cookIdentifiant', $_POST['adresseMail'], time() + 60);
49                      } else {
50                          setcookie('cookIdentifiant', false, -1);
51                      }
52
53                      // Libère toutes les ressources réservées par un résultat Oracle
54                      oci_free_statement($clientRecup);
55
56                      header('Location: ./index.php');
57                      exit;
58
59                  } else {
60
61                      header('Location: ./connexion.php?msgErreur=Mot de passe invalide');
62                      exit;
63
64                  }
65
66              }
67
68              header('Location: ./connexion.php?msgErreur=Mot de passe incorrect');
69              exit;
70
71      }
72
73      header('Location: ./connexion.php?msgErreur=Veuillez remplir les champs');
74      exit;
75
76  ?>
```

```

46
47          if(isset($_POST['memo'])) {
48              setcookie('cookIdentifiant', $_POST['adresseMail'], time() + 60);
49          } else {
50              setcookie('cookIdentifiant', false, -1);
51          }
52
53          // Libère toutes les ressources réservées par un résultat Oracle
54          oci_free_statement($clientRecup);
55
56          header('Location: ./index.php');
57          exit;
58
59      } else {
60
61          header('Location: ./connexion.php?msgErreur=Mot de passe invalide');
62          exit;
63
64      }
65
66  }
67
68  header('Location: ./connexion.php?msgErreur=Mot de passe incorrect');
69  exit;
70
71 }
72
73 header('Location: ./connexion.php?msgErreur=Veuillez remplir les champs');
74 exit;
75
76 ?>
```

4. Page index.php - ensemble

Afin d'avoir une idée de comment seront positionnés les articles en promo sur la page index, nous avons décidé d'afficher une seule image de canard. Nous l'utiliserons afin d'indiquer plus facilement la place que prendra les articles sur le site. Bien sûr la page index est basée sur les maquettes du client et ressemble exactement à ce qui nous a été transmis.

Site actuel :



Maquette :



Code :

```
<div class="infosContainer">
    <h2 class="text">Les promos du moments !</h2>

    <div id="gridMembers">
        <div class="promoIndex">
            
        </div>

        <div class="promoIndex">
            
        </div>

        <div class="promoIndex">
            
        </div>

        <div class="promoIndex">
            
        </div>
    </div>
</div>
```

Le code actuel est juste une mise en forme des images de la page index.

CSS :

```
#gridContainer, #gridContact, #gridMembers, #gridCollaboration {
    display: grid;
    grid-template-columns: auto auto; /*columns widths*/
}
```

```
.promoIndex {
    display:flex;
    flex-direction:column;
    justify-content:center;
    align-items:center;
    margin-bottom:20px;
}
```

Le CSS nous permet d'avoir des images centrées et un visuel attractif.

Ajout de pages et différentes requêtes

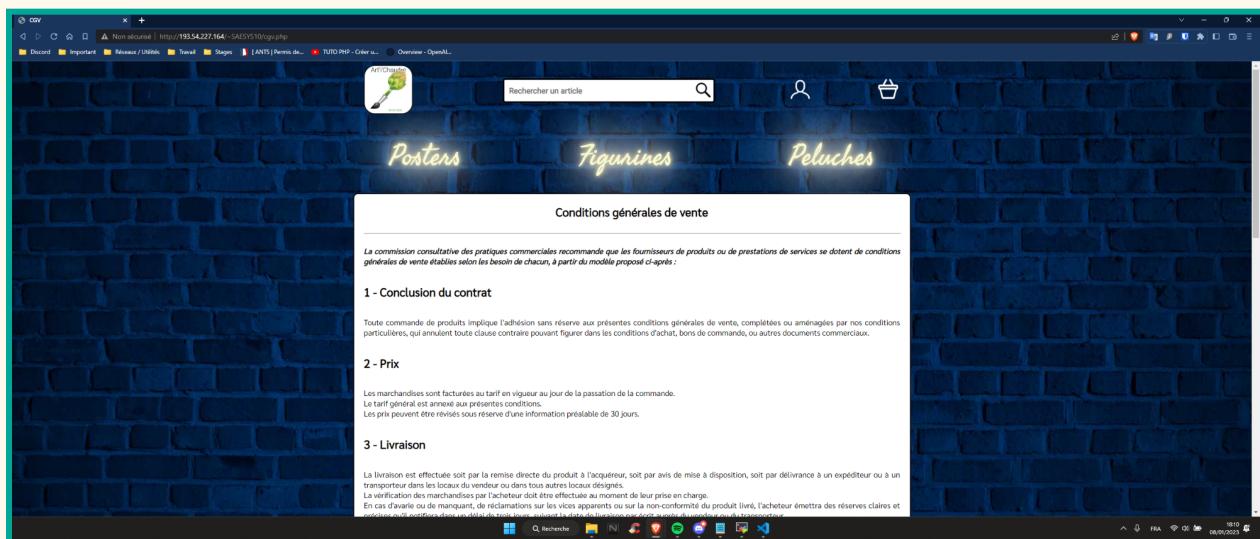
1. Pages du footer - visuel & fonctionnel

Dans notre charte graphique réalisée au premier sprint d'implémentation du site web, celui-ci inclut huit pages à implémenter. La plupart de ces pages utilisent les mêmes propriétés étant donné qu'elles ne nécessitent pas d'être distinguées au niveau visuel. Il en va de même sur la structure (fond) de ces pages où celles-ci sont souvent répétitives. Pour d'autres, d'autres propriétés (classes & id) sont utilisées, notamment pour la réalisation du formulaire de collaboration ou encore de la présentation de l'équipe.

En voici ces différentes pages (certaines ne seront pas figurées de part leur forte ressemblance) ainsi que leur fonctionnement :

Plan du site : simplement constitué d'une image.

CGV (également valable pour la page 'Cookie', 'Conditions générales de vente', 'Politique de confidentialité' et 'Mentions légales') :



Code HTML & CSS :

```
<div class="infosContainer">

<h2 class="infoTitle">Politique de confidentialit  pour le site web d'Art'l'Chade</h2>
<br>





<h3>I - Les donn es que nous collectons</h3>

<p>En utilisant le site d'Art'l'Chade, vous  tes amen s   nous transmettre des informations, dont certaines sont de nature   vous identifier et constituent de ce fait des donn es   caract re personnel (ci-apr s d nomm es les "donn es").</p>
<p>Ces informations contiennent notamment les donn es suivantes :</p>

<ul style="list-style-type: none; padding-left: 0;">
    <li><small><input checked="" type="checkbox"/></small> Donn es du compte : <small>indiquent les donn es que vous renseignez lors de la cr ation d'un compte en remplissant le formulaire d'inscription.</small></li>
    <li><small><input checked="" type="checkbox"/></small> Donn es rendues publiques : <small>indiquent l'ensemble des informations que vous affichez volontairement sur le site telles que notamment les commentaires sur les blogs et forums, photos, discussions sur les forums, et profil de compte.</small></li>
    <li><small><input checked="" type="checkbox"/></small> Donn es sur les transactions : <small>cas  ch t, indiquent les donn es que vous renseignez lorsque vous effectuez des achats par le biais du site telles que notamment les renseignements relatifs   votre moyen de paiement.</small><br/>
        Les donn es bancaires collect es sont transmises   des tiers qui contribuent   traiter et   satisfaire vos demandes. </small></li>
    <li><small><input checked="" type="checkbox"/></small> Donn es relatives   la navigation : <small>indiquent les donn es que nous collectons lors de votre navigation sur le site telles que notamment la date, heure de la connexion et/ou navigation, le type de navigateur, la langue du navigateur, son adresse IP. </small></li>
</ul>

<h3>II - Comment utilisons-nous les donn es que nous collectons ?</h3>
<p><strong>Id="test">Nous utilisons les donn es que nous recueillons afin de :</strong></p>
```

	Finalités	Base légale
	<ul style="list-style-type: none">Créer votre compte en ligne,Exécuter les contrats conclus entre vous et nous,Gérer la relation commerciale (livraisons, factures, services après vente)	
	<ul style="list-style-type: none">Exécution d'un contrat ou exécution de mesures précontractuelles,Consentement,Respect d'une obligation légale à laquelle le responsable du traitement est soumis,Sauvegarde des intérêts vitaux de la personne concernée,Exécution d'une mission d'intérêt public ou relevant de l'exercice de l'autorité publique dont est investi le responsable du traitement,Intérêts légitimes poursuivis par le responsable du traitement à moins que ne prévalent les intérêts ou les libertés et droits fondamentaux de la personne concernée.	

<small><input type="checkbox"/> Un droit d'accès et d'information :</small> vous avez le droit d'être informé de manière concise, transparente, intelligible et facilement accessible de la manière dont vos données sont traitées.
 Vous avez également le droit d'obtenir la confirmation que les données vous concernant sont traitées, et, le cas échéant, d'accéder à ces données et d'en obtenir une copie.<small><input type="checkbox"/> Un droit de rectification :</small> vous avez le droit d'obtenir la rectification des données inexactes vous concernant.
 Vous avez également le droit de compléter les données incomplètes vous concernant, en réalisant une déclaration complémentaire.
 Vous avez également le droit de demander la suppression des données traitées à l'exception de celles nécessaires au traitement de vos données.<small><input type="checkbox"/> Un droit d'effacement :</small> dans certains cas, vous avez le droit d'obtenir l'effacement de vos données.
 Cependant, ceci n'est pas un droit absolu et nous pouvons, avec le droit d'effacement, limiter la conservation de vos données.<small><input type="checkbox"/> Un droit à la limitation du traitement :</small> vous avez le droit de demander la limitation du traitement de vos données dans un format structuré, couramment utilisé et lisible par une machine, pour votre usage personnel ou pour les transmettre à un tiers de votre choix.
 Ce droit s'applique lorsque le traitement de vos données est basé sur votre consentement, sur un contrat et que ce traitement est effectué par des moyens automatisés.<small><input type="checkbox"/> Un droit d'opposition au traitement :</small> vous avez le droit de vous opposer à tout moment au traitement de vos données pour les traitements basés sur notre intérêt légitime, une mission d'intérêt public et ceux à des fins de prospection commerciale.
 Ce droit s'applique lorsque le traitement de vos données est basé sur notre intérêt légitime, une mission d'intérêt public et ceux à des fins de prospection commerciale.

 <small><input type="checkbox"/> Le droit de retirer votre consentement à tout moment :</small> vous pouvez retirer votre consentement au traitement de vos données lorsque le traitement est basé sur votre consentement.
 Le retrait du consentement ne compromet pas la légalité du traitement fondé sur le consentement effectué avant ce retrait.

 <small><input type="checkbox"/> Le droit de déposer une plainte auprès d'une autorité de contrôle :</small> vous avez le droit de contacter votre autorité de protection des données pour vous plaindre de nos pratiques de protection des données personnelles.<small><input type="checkbox"/> Le droit de donner des directives concernant le sort de vos données après votre décès :</small> vous avez le droit de nous donner des directives concernant l'utilisation de vos données après votre décès.

A la différence de la majorité des pages HTML et par surcharge potentielle du code CSS, l'utilisation de certaines balises sont privilégiées telles que les balises ``, `<i>` ou `<u>` signifiant respectivement ‘gras’ (`bold`), ‘italique’ (`italic`) et ‘souligné’ (`underline`).

Les balises `<table>` ainsi que leur cellule ayant déjà été évoquées dans les TP de PHP, il n'est nécessaire de les mentionner à nouveau.

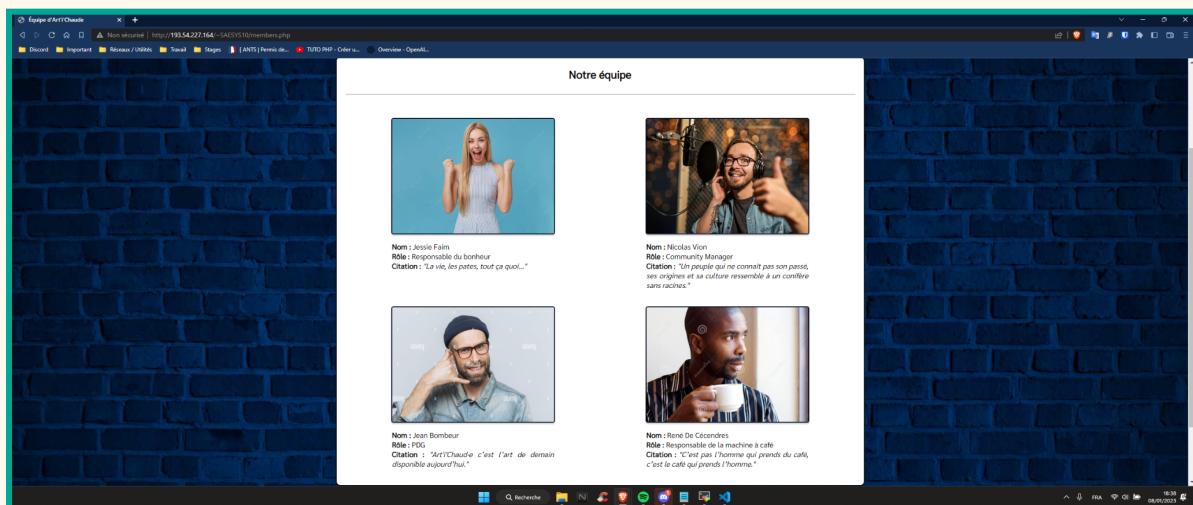
```

782 /*---| Containers, titles, links and tables |---*/
783
784 .infosContainer {
785   max-width:1100px;
786   width:100%;
787   padding-left:0px;
788   padding-right:20px;
789   border-radius:10px;
790   border:2px solid black;
791   background-color:#white;
792   text-align:justify;
793   text-justify:inter-word;
794   font-family: 'Sarabun', sans-serif;
795 }
796 .infosTitle {
797   text-align:center;
798 }
799
800 hr {
801   color:#black;
802   width:100%;
803 }
804
805 .linkInfos {
806   color:#rgb(0, 155, 0);
807 }
808 .linkInfos:hover {
809   text-decoration: none;
810 }
811 .infosTable {
812   border:2px solid black;
813 }
814
815 table {
816   border-collapse:collapse;
817   margin:auto;
818 }
819 .cellsInfos {
820   width:50%;
821   padding:20px;
822   border:3px solid black;
823 }
824 .headInfos {
825   text-align:center;
826   background-color:##040d22;
827   color:#white;
828 }

```

Ce code permet d'avoir un texte justifié et possédant un même espace entre chaque mot dans le conteneur principal nommé ‘infosContainer’, ou encore border-collapse:collapse qui permet de fusionner les bordures de chacune des cellules.

Page ‘Notre équipe’ :



Code HTML & CSS :

```
<div class="infosContainer">
    <h2 class="infosTitle">Notre équipe</h2>
    <hr>

    <div id="gridMembers">
        <div class="memberInfo">
            
            <p><b>Nom :</b> Jessie Faim <br/>
                <b>Role :</b> Responsable du bonheur <br/>
                <b>Citation :</b> <i>"La vie, les pates, tout ça quoi..."</i></p>
        </div>

        <div class="memberInfo">
            
            <p><b>Nom :</b> Nicolas Vion <br/>
                <b>Role :</b> Community Manager <br/>
                <b>Citation :</b> <i>"Un peuple qui ne connaît pas son passé, ses origines et sa culture ressemble à un conifère sans racines."</i></p>
        </div>

        <div class="memberInfo">
            
            <p><b>Nom :</b> Jean Bombeur <br/>
                <b>Role :</b> PDG <br/>
                <b>Citation :</b> <i>"Art's i'Chaud-e c'est l'art de demain disponible aujourd'hui."</i></p>
        </div>

        <div class="memberInfo">
            
            <p><b>Nom :</b> René De Cérendres <br/>
                <b>Role :</b> Responsable de la machine à café <br/>
                <b>Citation :</b> <i>"C'est pas l'homme qui prends du café, c'est le café qui prends l'homme."</i></p>
        </div>
    </div>
</div>
```

```
/*---| Spécificités de la page members |---*/
.memberInfo {
    display:flex;
    flex-direction:column;
    justify-content:center;
    align-items:center;
    margin-bottom:20px;
}

.memberInfo > p {
    width:350px;
}

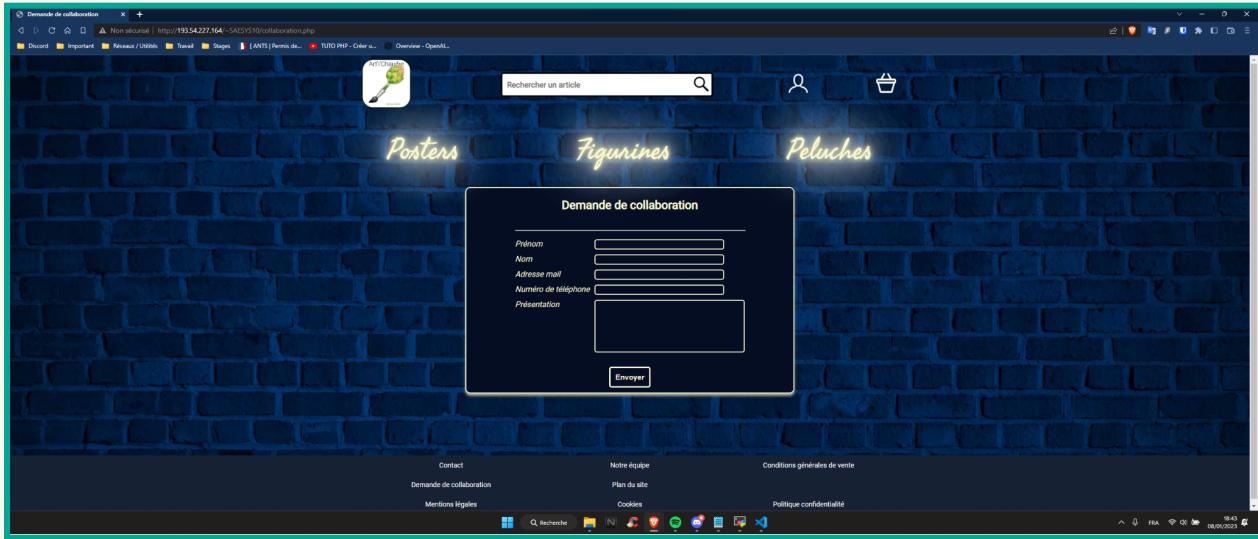
.memberInfo > img {
    width:350px;
    height:250px;

    border-radius:5px;
    border:2px solid #040d22;

    filter: drop-shadow(0 1mm 0.3mm lightgrey);
}

#alignImage {
    margin-top:42px;
}
```

Page ‘Formulaire de collaboration’ (traitée par la page PHP ‘traitCollaboration.php’):



Code HTML, CSS et PHP :

```
<form method="post" id="formCollaboration" action="traitCollaboration.php">
    <h2 class="infostitle">Demande de collaboration</h2>
    <hr>
    <?php
        if(isset($_GET['msgErreur'])) {
            echo "<h3 id='errorCollaboration'>".$_GET[' msgErreur']. "</h3>";
        }
    ?>
    <div id="gridCollaboration">
        <label class="formLabels labelsCollab">Prénom</label>
        <input type="text" class="champsForm champsCollab" name="prenom" required minlength="2"/>

        <label class="formLabels labelsCollab">Nom</label>
        <input type="text" class="champsForm champsCollab" name="nom" required minlength="2"/>

        <label class="formLabels labelsCollab">Adresse mail</label>
        <input type="mail" class="champsForm champsCollab" name="adresseMail" required minlength="5"/>

        <label class="formLabels labelsCollab">Numéro de téléphone</label>
        <input type="text" class="champsForm champsCollab" name="telephone" required maxlength="10"/>

        <label class="formLabels labelsCollab">Présentation</label>
        <textarea class="champsForm" id="zoneCollab" name="Presentation" required maxlength="400"></textarea>
    </div>
    <div id="zoneBtnCollab">
        <input type="submit" name="Envoyer" value="Envoyer" id="btnCollaboration"/>
    </div>
</form>
```

```
/*---| Spécificités de la page de collaboration |---*/
.labelsCollab {
    padding-right:10px;
}

#btnCollaboration {
    margin-top:10px;
    margin-bottom:10px;
}

#zoneBtnCollab {
    display:flex;
    justify-content:center;
    align-items:center;
}

.champsCollab {
    width:250px;
    padding-left:10px;
    margin-bottom:10px;
    font-weight:bold;
}

#zoneCollab [
    resize:none;
    height:100px;
    width:300px;
]
```

```
1 <?php
2
3     //On se contentera également d'une simulation d'un envoi d'un message
4
5     if(isset($_POST['Envoyer'])) {
6
7         if(isset($_POST['prenom']) && isset($_POST['nom']) && isset($_POST['adresseMail']) && isset($_POST['telephone']) && isset($_POST['Presentation'])) {
8
9             $adresseMail = htmlspecialchars($_POST['adresseMail']);
10            $numTelephone = htmlspecialchars($_POST['telephone']);
11
12            if(!preg_match('#^a-zA-Z-]{2,}\.{a-zA-Z}{2,}$#', $adresseMail)) {
13
14                header('Location: ./collaboration.php?msgErreur=Vérifiez la saisie de votre adresse mail');
15                exit;
16
17            } else if(!preg_match('#^\d{9}$#', $numTelephone)) {
18
19                header('Location: ./collaboration.php?msgErreur=Vérifiez la saisie de votre numéro de téléphone');
20                exit;
21
22            } else {
23
24                echo('<script language="JavaScript" type="text/javascript">
25                    alert("Demande de collaboration envoyée.");
26                    location.href = "./index.php";
27                </script>');
28
29            }
30
31        } else {
32
33            echo('<script language="JavaScript" type="text/javascript">
34                alert("Veuillez terminer de remplir les champs.");
35                location.href = "./collaboration.php";
36            </script>');
37
38        }
39
40    }
41
42 ?>
```

Le plus intéressant à mentionner est le code PHP. En réalité, il est impossible, par le biais du formulaire de contact ou de collaboration, de pouvoir envoyer un message à une adresse mail spécifique : le serveur SMTP n'est pas paramétré pour l'envoi d'emails à un destinataire. Ainsi, l'utilisation de la librairie *PHPMailer* (ayant été utilisée auparavant) ne sera plus utilisée. C'est pourquoi une simple simulation d'un envoi d'un mail sera privilégiée. Pour autant, le

traitement du formulaire est réalisé et il en va de même pour le formulaire de contact à l'entreprise : celui-ci évite les failles XSS en convertissant les caractères en entités HTML et vérifie les REGEX d'une adresse mail ou encore d'un numéro de téléphone afin d'éviter toute tentative de contournement au renseignement attendu. Si une erreur apparaît, celle-ci sera passée par l'URL et mentionnée dans le formulaire. Autrement, une pop-up JavaScript apparaîtra, signifiant que l'envoi est effectué, en redirigeant l'internaute sur la page principale.

2. Récupération des articles en fonction de la catégorie correspondante - visuel & fonctionnel

On doit pouvoir afficher tous les articles en fonction de la catégorie à laquelle il appartient. Pour cela, dans un premier temps, on va effectuer une requête afin de récupérer les noms des articles qui nous intéressent.

Code :

On récupère donc dans la base de données les articles avec comme nom de catégorie “*Posters*” puis on exécute la requête (cela est ainsi valable pour chacune des pages traitant des autres catégories).

```
<?php
require_once("connect.inc.php");

$recherche = 'POSTERS';
$requete = "Select * FROM ARTICLE WHERE NOMCATEGORIE LIKE '%' || :articles || '%'";
$req1 = oci_parse($connect, $requete);
oci_bind_by_name($req1, ':articles', $recherche);
$result= oci_execute($req1);

?>
```

Par la suite, on va se contenter d'un simple écho pour afficher le/les article(s) concerné(s).

Pour ce faire, dans le while, on vérifie que le fetch de la requête n'est pas vide. Puis, on affiche le contenu de la requête ligne par ligne en indiquant l'attribut (“NOMARTICLE”) qui nous intéresse, afin d'avoir le nom de tous les articles contenus dans la catégorie “*Posters*”.

```
<section class="afficher_article">
    <?php
        while (($article = oci_fetch_assoc($req1))!= false) {
            echo $article['NOMARTICLE'];
            echo "<br/>";
        }
    </?>
</section>
```

Site :



Nous avons utilisé le même principe pour les catégories “*Figurines*” et “*Peluches*”. Cependant, étant donné que nous avons une barre de recherche, on va devoir récupérer le contenu de la barre de recherche.

3. Barre de recherche concernant les articles - visuel & fonctionnel

La requête est la même que celle réalisée précédemment. Ici, on va comparer les articles dans la base de données avec le contenu de la barre de recherche :

```

require_once("connect.inc.php");
if(isset($_POST['s']) AND !empty($_POST['s'])){
    $recherche = htmlspecialchars($_POST['s']);
    $requete = "Select * FROM ARTICLE WHERE NOMARTICLE LIKE '%' || :articles || '%'";
    $req1 = oci_parse($connect, $requete);
    oci_bind_by_name($req1,':articles',$recherche);
    $result= oci_execute($req1);
}

```

Il faut que la barre de recherche qui se nomme 's' soit **set ??** c'est-à-dire, qu'on a interagi avec et que le contenu ne soit pas vide.

Par ailleurs, on va récupérer le contenu de la barre de recherche puis en fonction du contenu de celle-ci :

- On affiche les articles correspondant à ce qu'il est exactement figuré (ou plus ou moins) dans la barre de recherche (le cas où il n'y a pas d'erreur(s)) :

```

if (isset($_POST['s']) AND !empty($_POST['s'])){
    if((!isset($_POST['s'])) AND ($article = oci_fetch_assoc($req1))!= True) {
        echo "Aucun produit ne corespond à votre recherche : ".$_POST['s'];
    } else {
        ?>Résultats trouvés pour "<?php echo $_POST['s'] ; ?>" <?php
    }
} else {
    header("Location: ./index.php");
}

```

- On affiche un message d'erreur si aucun(s) article(s) ne correspond au contenu de la barre de recherche :

```

if (isset($_POST['s']) AND !empty($_POST['s'])){
    if((!isset($_POST['s'])) AND ($article = oci_fetch_assoc($req1))!= True) {
        echo "Aucun produit ne corespond à votre recherche : ".$_POST['s'];
    } else {
        ?>Résultats trouvés pour "<?php echo $_POST['s'] ; ?>" <?php
    }
} else {
    header("Location: ./index.php");
}

```

- On renvoie vers la page index.php si aucun contenu n'est présent dans la barre de recherche :

```

if (isset($_POST['s']) AND !empty($_POST['s'])){
    if((!isset($_POST['s'])) AND ($article = oci_fetch_assoc($req1))!= True) {
        echo "Aucun produit ne correspond à votre recherche : ".$_POST['s'];
    } else {
        ?>Résultats trouvés pour "<?php echo $_POST['s'] ; ?>" <?php
    }
} else {
    header("Location: ./index.php");
}

```

4. Redirection des images de produits dans la page produit.php

Ici-même, nous allons afficher une image pour chaque article et mettre en forme les articles afin de rendre la page “*Figurines*”, par exemple plus attractive :

```

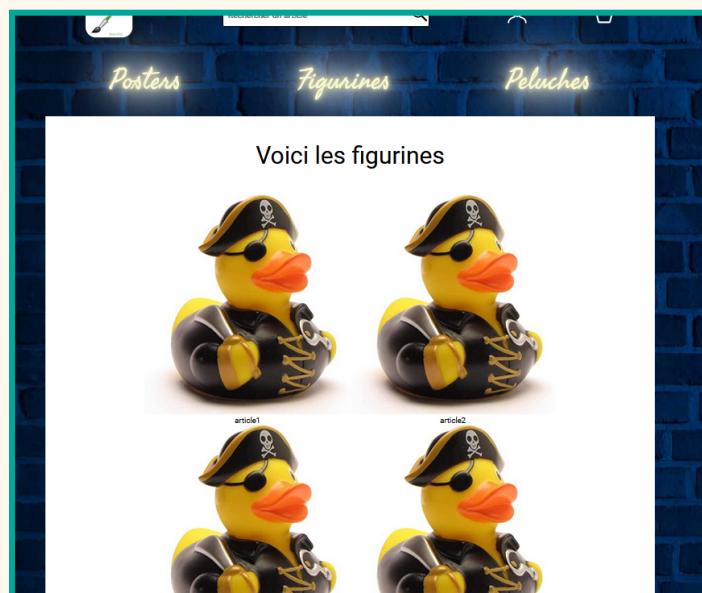


</br>
</a>
<div class="center">
<?php
    echo $article['NOMARTICLE'];
?>

```

Grâce à cela chaque produit est représenté par l'image “*test*” et chaque nom d'article est alors centré.

En voici le rendu (*site*) :



Pour le moment, un canard pirate prend la place des images des produits mais celui-ci sera remplacé par les bonnes images correspondantes lors du prochain sprint.

Par la suite, nous devons rendre cliquables les images et rediriger l'utilisateur vers la page produit qui correspond avec l'article interagit :

```
<a href="produit.php?nom=<?php echo $article['NOMARTICLE']?>">
```

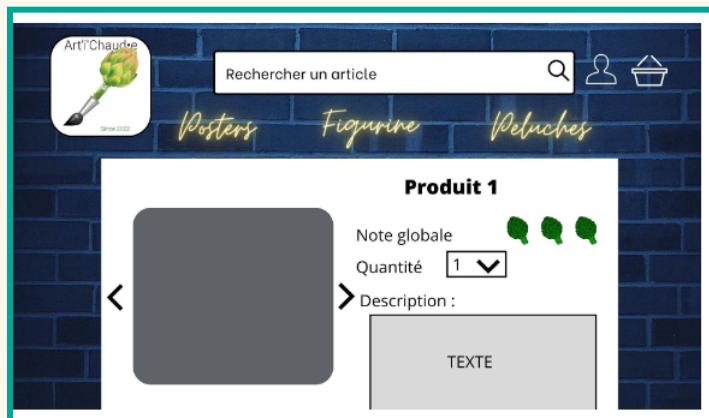
Pour cela, on redirige l'image cliquable vers “produit.php?nom=” puis le nom de l'article sur lequel nous avons cliqué. Sur la prochaine sous-partie, nous allons prendre l'exemple de l'article 1.

5. Page produit.php et requêtes

Une fois l'article 1 de cliqué, nous sommes redirigés vers “produit.php?nom=article1” :

```
193.54.227.164/~SAESYS10/produit.php?nom=article1
```

Pour la page “produit”, nous avons suivie la maquette que le client nous a donné :



On doit donc récupérer :

- Le nom du produit;
- La note globale du produit.

Le nom du produit étant transmis en GET dans l'URL, on peut donc le récupérer de cette façon :

```
<?php
$nom_produit = $_GET['nom'];
echo $nom_produit;
?>
```

Pour la note globale du produit, on effectue une requête dans la base de données qui elle-même effectue la moyenne de notes attribuées à un produit avec comme nom, le nom d'article dans l'URL :

```
require_once("connect.inc.php");

//Récupere la note du produit
$recherche = $_GET['nom'];
$reqeute = "SELECT AVG(AVIS.NOTE) AS NOTE FROM AVIS,ARTICLE WHERE AVIS.IDARTICLE = ARTICLE.IDARTICLE AND ARTICLE.NOMARTICLE = :nom" ;
$req1 = oci_parse($connect, $reqeute);
oci_bind_by_name($req1,':nom',$recherche); //article correspondant
$result= oci_execute($req1);
```

Afin d'illustrer la note globale du produit, on récupère l'image d'artichaut auprès du client pour ensuite, en fonction de la note globale du produit afficher de 1 jusqu'à 5 artichauts verts.

Voici le code correspondant :

```
div class= "text4">


Note globale :


<?php
while (($note = oci_fetch_assoc($req1))!=False) {
    echo $note['NOTE'];
    //calcul mieux faire pour notation
    echo "<br>";
    if($note['NOTE']<=2){
        >>
        
        
        
        <?php
    }else if($note['NOTE']<=4{
        >>
        
        
        
        <?php
    }else {
        >>
        
        
        
        <?php
    }
    echo "<br/>";
}
>>
</p>
</div>
```

Enfin, on ajoute un SELECT avec des options pour la quantité et une description de produit. A l'heure actuelle, la description est un texte aléatoire qui sera éventuellement modifiée par la suite :

```

<form>
<label for="quantite">Quantite:</label>
<select id="quantite" name="quantite">
    <option value="1">1</option>
    <option value="2">2</option>
    <option value="3">3</option>
    <option value="4">4</option>
    <option value="5">5</option>
</select>
</form>
<style>

</style>
<br>

<p>Description :<br>
<div class="texttt">  </br>
Le Lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression.
</div><br>
</p>

</div>
</div>

```

Et voici le résultat de la page produit.php :

