**Introduction générale**

Le commerce électronique ne se limite pas à la seule vente en ligne des produits ou articles à travers le réseau internet mais réellement, il englobe également la réalisation de vitrine en ligne, le conseil aux utilisateurs, la mise à disposition des catalogues électroniques.

Le commerce électronique est réalisé à partir de site internet donc il est important pour tout agent commercial de comprendre l’apport et l’intérêt du site pour gérer ses tâches d’une manière simple, efficace et rapide.

Les sites des boutiques en lignes n’ont cessé de se multiplier ces dernières années. Ils apparaissent aux yeux des utilisateurs comme le moyen de faire de bonnes affaires à moindres coûts.

Mon projet de fin d’étude s’inscrit dans le cadre de la création un site web dynamique de boutique en ligne qui donne la possibilité aux visiteurs de naviguer, de consulter les différents produits et éventuellement de faire des achats d’articles.

Le présent rapport, qui expose ce travail, est composé de chapitres structurés comme suite :

Le premier chapitre intitulé "Présentation générale" est consacré à la présentation de l’organisme d’accueil et du cadre du projet.

Le deuxième chapitre décrit l’étude de l’existant avec ses avantages, en cherchant aussi les objectifs à développer.

Le troisième chapitre décrit les besoins de notre projet, qu’ils soient fonctionnels ou non fonctionnels, et offre une modélisation de ses besoins par des diagrammes de cas d’utilisation suivant UNIFIED MODELIN LANGUAGE (UML).

Dans le quatrième chapitre, je vais présenter la conception de l’application.

Dans le cinquième chapitre, je vais présenter la phase de l’implémentation de l’application en justifiant les différents choix techniques et technologiques utilisés au cours de cette phase avant de projeter quelques captures écran de l’application finale réalisée. Je termine ce rapport par une conclusion générale et des perspectives futures.

**Chapitre 1 :**

**Présentation général et analyse du problème**

**Introduction**

Ayant essuyer des refus de stage par les entreprises contactées, je me suis résolu à travailler sur un sujet choisi en adéquation avec ma formation CDA en me basant essentiellement les connaissances acquises en cette période, et que je maîtrise le plus.

En conséquence, le projet à été effectué au sein de notre centre de formation AFPA Champs-sur marne.

Le but de mon travail est de mettre en pratique les enseignements (technologies web) appris durant ma formation Concepteur Développeur d’Applications.

**I.Présentation de l’organisme de formation :**

**Présentation :**

L'Afpa a été créée le 11 janvier 1949 sous l’appellation Association nationale interprofessionnelle pour la formation rationnelle de la main-d’œuvre (ANIFRMO).

L’**Agence nationale pour la formation professionnelle des adultes** (**Afpa**) est un [organisme](https://fr.wikipedia.org/wiki/Établissement_public_à_caractère_industriel_et_commercial_en_France) [français](https://fr.wikipedia.org/wiki/France) de [formation professionnelle](https://fr.wikipedia.org/wiki/Formation_professionnelle), au service des [régions](https://fr.wikipedia.org/wiki/Région_française), de l'[État](https://fr.wikipedia.org/wiki/État_en_France), des [branches professionnelles](https://fr.wikipedia.org/wiki/Branche_professionnelle) et des [entreprises](https://fr.wikipedia.org/wiki/Entreprise). Membre du [service public de l'emploi](https://fr.wikipedia.org/wiki/Service_public_de_l'emploi), l'Afpa, constituée en [association](https://fr.wikipedia.org/wiki/Association_loi_de_1901) avant de devenir en 2017 un [établissement public à caractère industriel et commercial](https://fr.wikipedia.org/wiki/Établissement_public_à_caractère_industriel_et_commercial_en_France) (EPIC), propose des formations professionnelles qualifiantes, sanctionnées par un titre professionnel du [ministère du Travail](https://fr.wikipedia.org/wiki/Ministère_du_Travail_(France)).

**II.Présentation du projet :**

**II.1. Cadre du projet :**

Ce projet s’inscrit dans le cadre de la réalisation d’un stage de projet de fin d’études au sein de l’AFPA et dont l’intitulé est « Réalisation d’une boutique en ligne de textile : e-shop».

Le but de ce travail est de développer une boutique en ligne qui permet de garantir plus de productivité pour les entreprises de commerce en fournissant un moyen de recherche et de communication plus efficace.

**II.2. La boutique en ligne de vente de textile :**

**La boutique en ligne**

* **Définition :**

Sur une boutique en ligne comme dans un magasin réel, on a la possibilité d’acheter différents articles et produits.

Pour acheter dans une boutique en ligne, il suffit de cliquer sur les produits ou articles sélectionnés puis de les ajouter un par un dans un panier d'achat virtuel.

L'acheteur ou le client peut ensuite remplir un bon de commande et payer à l'aide d'une carte bancaire ou d'un autre moyen de paiement autorisé par la boutique en ligne. La commande est formée en fonction du choix de l'internaute et selon les modalités définies par la boutique.

* **La vente en ligne :**

Le fait d'acheter de son domicile est devenu une pratique courante notamment grâce à la vente par correspondance proposée par des boutiques.

Mais cette pratique s'est encore plus développée avec l'explosion d'internet. Nombreux sont les magasins qui proposent sur leur site d'acheter toute sorte de choses (vêtements, meubles, matériel informatique, dvd, livres...).

Il suffit alors de payer en ligne l'article ainsi que les frais de livraison par carte bancaire, chèque, et la boutique se charge d'envoyer la marchandise commandée.

Rares sont maintenant les utilisateurs du web qui n'utilisent pas la vente en ligne alors qu'elle évite tout déplacement et où les prix sont souvent très compétitifs notamment sur les sites d'enchères ou d'occasions.

* **Le produit :**

Est un ensemble de vêtements, équipements et accessoires à personnaliser, qui possède de multiples modèles de formes et de coloris différents accessibles suivant les goûts de l’acheteur.

**III.Choix méthodologique de développement :**

**III.1. Processus de développement :**

La méthodologie UML (UNIFIED MODELIN LANGUAGE) étant avant tout un outil de modalisation, il ne définit pas un processus de développement particulier, il est développé à partir d’un ensemble de méthodes d’analyse et de conception orientée objet.

Néanmoins, il peut servir de notation pour différentes approches méthodologiques basées sur les objets.

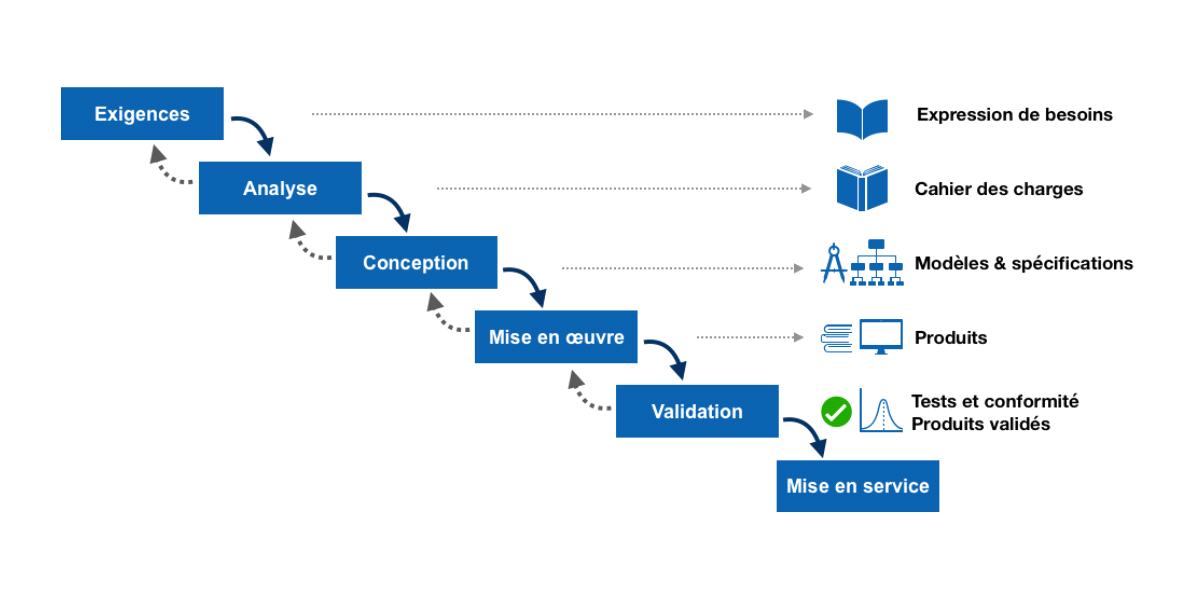
**III.2. Processus en cascade :**

J’ai choisi d’utiliser Le modèle en cascade de la méthode <<scrum>> qui comprend les phases et les livrables suivants :

* [Exigences](https://fr.wikipedia.org/wiki/Exigence_(ingénierie)) : les exigences font l'objet d'une expression des besoins ;
* [Analyse](https://fr.wikipedia.org/wiki/Analyse_des_exigences) : les exigences sont analysées pour établir un [cahier des charges fonctionnel](https://fr.wikipedia.org/wiki/Cahier_des_charges_fonctionnel) ;
* Conception : le produit est conçu et spécifié de sorte à pouvoir être réalisé ;
* [Mise en œuvre](https://fr.wikipedia.org/wiki/Mise_en_œuvre) : le produit est réalisé sur la base des spécifications ;
* Validation : le produit est testé et vérifié et sa conformité aux exigences est validée ;
* Mise en service : le produit est installé, les préparatifs pour sa mise en service sont organisés, puis le produit est utilisé.

Chaque phase ne commence qu'une fois les résultats de la phase précédente validés. Le point fort de cette approche est de garantir l'existence d'une documentation bien structurée.

Plusieurs variantes du modèle existent, dont l'ajout d'une phase de planification en amont, la réalisation préalable d'un prototype, la décomposition de la phase de validation, et le retour aux phases précédentes en cas de défauts découverts en aval.

***FIGURE : Modèle en cascade***

**Conclusion :**

Ce chapitre avait pour but d’expliquer le cadre général d’une définition de boutique en ligne avec ses avantages. J’ai précisé ensuite mon choix méthodologique de développement qu’il permettra de présenter le processus de développement.

**Chapitre 2 :**

**Etude de l’Art**

**Introduction**

Dans ce chapitre, je vais donner une analyse détaillée de l’existant et ses avantages, ainsi qu’un objectif pour améliorer et résoudre certains problèmes du domaine du commerce.

**I.Etude de l’existant :**

La présentation des produits et articles entre les clients et les fournisseurs est élémentaire. Elle se fait de différentes manières :

* La communication directe entre le grossiste et le détaillant se passe plus généralement par téléphone ou par messagerie.
* L’acheteur doit se déplacer en boutique et autre, et se présenter dans les temps d’ouverture pour acheter et commander ses besoins courants.
* Le fournisseur doit se déplacer au travail ou au domicile de ses clients pour présenter ses produits.

**II.Critique de l’existant :**

On trouvait d’une manière générale que l’absence des sites web dans les entreprises et surtout dans les entreprises commerciales produit beaucoup de difficultés dans le domaine de travail de ces mêmes entreprises.

D’après cette critique on peut citer les problèmes qui peuvent exister en l'absence de site commercial :

* Marché limité : niveau local seulement.
* Absence de marque au niveau mondial.
* Absence de promotion au niveau des ventes.
* Nombres des ventes limitées.
* Augmentation du budget publicitaire.
* La perte de temps dans la recherche de marchés.

**III.Avantages d'une boutique en ligne :**

* L’e-commerce, un marché en plein essor
* L'ouverture d'une boutique en ligne : un projet facilement finançable
* Faire du monde entier son lieu de travail potentiel
* Qui n'a jamais rêvé de devenir son propre patron ?
* Toucher des clients partout dans le monde grâce au e-commerce
* Votre boutique reste ouverte 24h/24 et 7j/7
* Optimiser ses ventes en mettant facilement en avant ses produits stars
* Une pléthore d'outils marketing pour accroître ses ventes
* La possibilité de varier les formes de contenus
* Une boutique facilement adaptative aux tendances
* L’e-commerce : un réel confort d'achat pour le client
* Un style de vente moins intrusif pour le client

Une immense source d'informations pour enrichir votre fichier clients.

**IV.Objectif à développer :**

L’objectif de notre projet est de réaliser une boutique en ligne de vente d’articles textiles pour les utilisateurs et qui dispose de divers choix d’articles et produits.

D’où l’explosion des sites web qui permettent d’assurer plus de productivité pour les entreprises de marchandises en fournissant un moyen de recherche et de communication plus efficace. L’application permettra d’assurer les objectifs suivants :

* Recherche rapide et approfondie des produits et articles.
* Possibilité de faire des ventes directes.
* La vitesse : les délais sont plus rapides par rapport à l’achat classique.
* Rassurer les clients : donner plus de détails sur le produit ou l'article, produire des garanties et sécuriser le paiement.
* Découverte de nouveaux marchés : un avantage pour les petites et moyennes entreprises.
* Diminuer le temps de traitement des commandes.

**Conclusion :**

Ce deuxième chapitre a présenté une vue sur la boutique en ligne. Au troisième chapitre, on va passer à l’étape d’analyse et de spécification des besoins.

**Chapitre 3 :**

**Spécification des besoins**

**Introduction**

La spécification des besoins représente la première phase du cycle de réalisation d'une application. Elle doit décrire sans ambiguïté le site à développer.

Dans ce chapitre j’ai spécifié l'ensemble des besoins fonctionnels et non fonctionnels liés à notre application et qui découlent directement du cahier des charges.

Ensuite je dois modéliser les spécifications des besoins à l'aide des diagrammes de cas d’utilisation.

**I.CAHIER DES CHARGES**

1. **Présentation d’ensemble du projet**

**1.1 Présentation de l’entreprise :**

"e-shop" est une entreprise de vente de vêtements en ligne.

**1.2 Les objectifs du site :**

- Le site de "e-shop" sera la seule plateforme de vente de l'entreprise, et doit être en mesure de proposer une expérience de qualité aux visiteurs du site. Le back-office du site doit également permettre une gestion quotidienne des activités e-commerce (suivi des commandes, mise à jour du catalogue produit).

**1.3 La cible adressée par le site :**

- "e-shop" cible les particuliers, consommateurs réguliers de vêtements à bas prix. Le discours marketing, les modes de tarification pratiqués ou l'image de marque qui sera développée ciblera principalement une clientèle relativement jeune (25-35 ans) utilisant les supports digitaux de manière intensive.

**1.4 Objectifs quantitatifs après 1 an :**

- Nombre de vêtements vendues : 3 000

- Visiteurs par jour : 1 500

- Taux de conversion : supérieur à 1,2%

- Nombre de comptes clients : 1 000

**1.5 Périmètre du projet :**

- L'activité vise uniquement la France métropolitaine.

- Le site sera disponible en français uniquement.

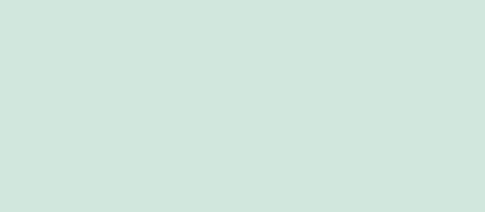
- Le site sera intégralement "Responsive Design".

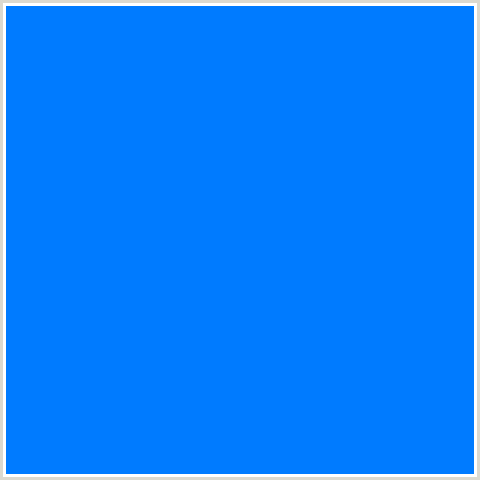
- L'ensemble des fonctionnalités détaillées dans ce document seront accessibles depuis un mobile.

2. **Description graphique et ergonomique**

**2.1 Charte graphique :**

- La couleur dominante du site sera le blanc : #ffffff

- La couleur secondaire, utilisée pour les fonds de titres, pied de page et autres éléments de navigation sera le vert pistache : #d1e7dd 

- La troisième couleur, utilisée pour les boutons entre autre sera le bleu : #007bff 

**2.2 Inspirations**

Inspirations graphiques métiers

- <http://halleauxvetements.com/>

- <http://www.kiabi.com/fr/>

**2.3 Logo**

"e-shop" se réserve le droit de fournir directement le logo à utiliser.

3. **Description fonctionnelle et technique**

**3.1 Arborescence du site :**

Le site se décompose en 6 pages principales, toutes accessibles depuis le menu du site, lui-même présent à l'identique sur l'ensemble des pages du site. Ces pages principales sont :

- Home (<http://e-shop>)

- Nos fringues (<http://e-shop/>)

- Notre catalogue (<http://e-shop.fr/vitrine.php>)

- Panier (<http://e-shop/panier.php>)

- Produit ? (<Http://e-shop/gestion_commande>)

- Espace Client (<http://e-shop/profil.php>)

En plus de ces pages principales, les pages suivantes doivent être accessibles depuis le FOOTER :

- Contact (<http://e-shop/feedback.php>)

- Liens réseaux sociaux

**3.2 Description fonctionnelle du site :**

**Boutique en ligne n°1 : Nos Produits :**

Le site de "e-shop" doit bien évidemment comporter une boutique en ligne, accessible depuis l'onglet "Notre Catalogue". Cette boutique en ligne doit par défaut afficher l'ensemble des produits mis en ligne depuis le back-office, et disponibles en stocks.

Depuis la liste de chaque produit, les utilisateurs doivent pouvoir :

- Ajouter le produit au panier

- Consulter la fiche du produit

Inspiration : <https://www.lahalle.com/>

Fiches produits

Chaque fiche produit est composée des informations structurées suivantes :

- Nom du produit

- Photo du produit

- Description du produit

- Prix du produit

L'utilisateur pourra depuis ces pages produits effectuer un ajout panier.

**Espace Client :**

Un utilisateur sera dans l'obligation de créer un compte client pour compléter un achat. Afin de créer un compte, l'utilisateur devra seulement fournir son adresse e-mail et définir un mot de passe.

La création d'un compte est indispensable afin de pouvoir finaliser une commande.

**3.3 Description fonctionnelle du Back-office :**

Depuis le back-office, l'équipe de "e-shop" doit être en mesure d'effectuer en autonomie l'ensemble des tâches quotidiennes nécessaires au bon fonctionnement du site e-commerce.

Fonctionnalités standards

- Publication d'articles : Mise en ligne de nouveaux articles en toute autonomie. Possibilité d'utiliser du HTML aussi bien qu'un éditeur simplifié.

- Création de page : Elle doit être facilement réalisable. Le positionnement d'une page dans l'arborescence du site doit être facilement éditable.

**Fonctionnalités E-Commerce**

1. **Gestion du front**

- Gestion du catalogue de produits : Ajout et retrait des produits affichés dans chacune des boutiques.

- Gestion des promotions et codes promos : Création et suppression de codes promo et offres temporaires sur des produits ou groupes de produits.

- Mise à jour de la page d'accueil : Possibilité de mettre en avant des produits, promotions et articles

2. **Gestion du Back-office**

- Suivi des commandes : Affichage et modification de l’état (en cours de livraison, etc.)

- Gestion des stocks : Manuelle dans un premier temps. Il doit être possible de suivre depuis le Back-office le niveau des stocks pour chacun des produits.

**3.4 Informations relatives aux contenus :**

- Type de contenus : Le site utilisera des types de contenus différents : images, ressources téléchargeables, qu'il doit être possible de télécharger facilement sur le site.

**3.5 Contraintes techniques :**

- Le site doit être compatible avec l'ensemble des navigateurs standards : Google Chrome, Android, Mozilla Firefox, Internet Explorer, Safari & Opéra. Les versions compatibles doivent être définies explicitement par le prestataire).

- Le développement du site doit en tenir compte et rendre des intégrations futures facilement réalisables.

**4. Prestations attendues :**

- Design : Réalisation de maquettes pour chacune des pages principales listées dans l'arborescence du site

- Intégration : Intégration de ces maquettes après validation

- Développement du site et du back-office

- Création et paramétrage de la base de données du site

**4.2 Planning :**

Le projet sera découpé en 3 phases :

1. Phase de prototypage : Construction des maquettes, validation des choix d'architectures et de technologies.

2. Phase de développement du pilote : Développement de l'ensemble des fonctionnalités, intégration des maquettes

3. Phase d'industrialisation : Intégration du catalogue de produits et des contenus.

L'objectif est donc un passage en production (livraison finale) le 06 Septembre 2021.

**4.3 Méthodologie de suivi :**

L'ensemble du projet sera basé sur une méthodologie agile type "**scrum**".

**Organisation du projet :**

- Construction du back log. Pour chacune des 3 phases, un Product back log découpant l'ensemble des fonctionnalités et éléments à réaliser sera définit.

- Sprints : Le projet sera organisé sur la forme de sprints hebdomadaires.

**Suivi du projet**

L'ensemble du product Backlog et des sprints seront organisés et suivis sur un board Trello, afin de garantir une bonne compréhension et lisibilité de l'avancement du projet.

**II.Analyse fonctionnelle :**

L’analyse fonctionnelle est une démarche qui consiste à chercher, ajouter, ordonner, expliquer, hiérarchiser les fonctions d’un produit ou d'un article, en tenant compte des idées imposées par les besoins des clients ou acheteurs.

Les besoins ont été partagés en deux parties fonctionnelles et non fonctionnelles.

**II.1. Présentation du langage de modélisation UML :**

* **Définition :**

Nous avons utilisé, pour la spécification et la conception de ce travail, la méthodologie UML (UNIFIED MODELIN LANGUAGE), qui est un langage de modélisation qui permet de décrire les besoins, de documenter les systèmes et d’esquisser les architectures logicielles.

Les diagrammes sont dépendants hiérarchiquement et se complètent, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie. Il en existe quatorze depuis UML 2.3.

Toutefois, je ne présenterai que quelques diagrammes que j’ai jugés utiles et suffisants pour comprendre le projet à savoir les diagrammes de cas d’utilisations, les diagrammes de classes et les diagrammes de séquences.

**Avantage de l’UML :**

UML est un langage formel et normalisé. Il est précis et sans ambigüité avec des notations graphiques simples et compréhensibles.

**II.1.1. Besoins fonctionnels :**

Cette section a pour objet de présenter les besoins fonctionnels auxquels doit répondre notre site web :

**BF1 Gérer produit :**

Cette méthode se démarque par les trois actions : ajouter, modifier ou supprimer un produit et article. Donc l’administrateur après l’authentification, demande d’ajouter un produit ou article de modifier ou supprimer un produit et article. Méthode CRUD (Création, Lecture, Mise à jour et Suppression).

**BF2 Gérer promotion :**

Il est possible à tout moment d’ajouter, modifier, supprimer un produit ou article en promotion avec un prix barré.

**BF3 Gérer son compte :**

L’administrateur doit modifier fréquemment son authentification (login et mot de passe) pour sécuriser leur site.

**BF4 Inscrire :**

Le visiteur pourra s’inscrire au site.

**BF5 Authentifier :**

Le client doit saisir son login et son mot de passe pour accèder au site.

**BF6 Recherche de produit ou article :**

La recherche s’effectue par mots clés.

**BF7 Passer la commande :**

Le client peut choisir le mode de paiement, la quantité d’article et le type de livraison.

**II.1.2. Spécification des besoins non fonctionnels :**

Les besoins non fonctionnels spécifient les propriétés du système tel que les contraintes d'environnement et d'implémentation, la performance, la maintenance, l'extensibilité et la flexibilité.

Pour ceci, le site à réaliser doit satisfaire les exigences de la totalité des clients.

**BF1- Sécurité :**

Grâce à une connexion sécurisée, nos clients feront leurs achats dans notre boutique en toute confiance. Mots de passe contrôlés à chaque connexion avec un e-mail unique.

**BF2- La performance :**

Un site doit être avant tout performant c'est à-dire à travers ses fonctionnalités, répondre à toutes les exigences des utilisateurs d'une manière optimale.

**BF3-Faciliter l’utilisation des données du site :**

Il s’agit d’ajouter, rechercher, supprimer et de modifier les données du site.

**BF4- Disponibilité :**

Les commandes sont instantanées, le site de vente en ligne doit répondre à tout moment.

**BF5-Ergonomie :**

Elle est composée par charte graphique, Couleur web, Images, Animations

Cette méthode est importante pour retenir les visiteurs dans ce site et pour attirer l’attention du client sur ma boutique en ligne.

**III.Modélisation des besoins :**

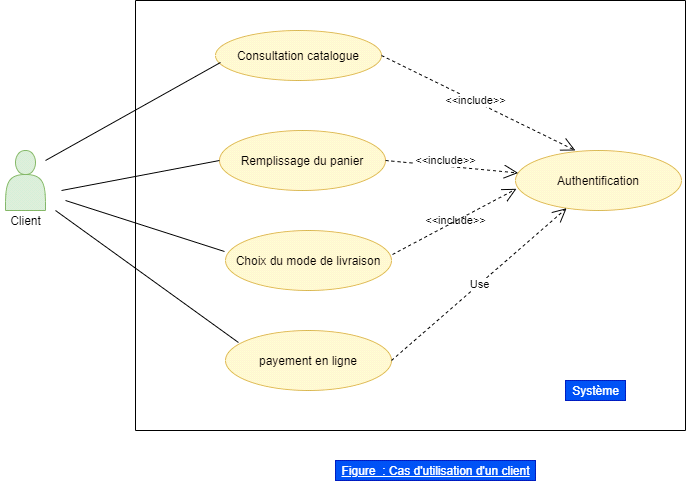
Le diagramme de cas d’utilisation est un modèle de communication, il permet d’analyser et d’organiser les besoins, du point de vue de l’administrateur et du client, il définit les limites précises du système en utilisant une notation simple et compréhensible par tous.

**III.1. Identification des acteurs :**

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié.

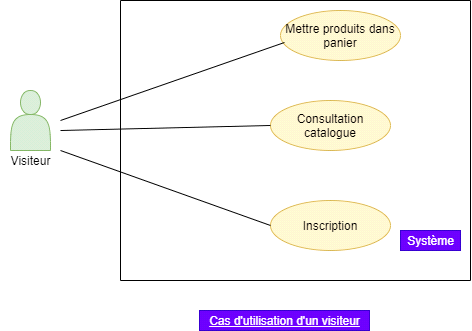
* **Client :**

C'est un acteur externe, Il peut se connecter au site, il cherche des produits et articles, passe des commandes et choisit le mode de paiement. Il a la main pour modifier son profil.

***FIGURE : Cas d'utilisation client« e-shop»***

* **Visiteur :**

C'est un acteur externe, Il peut s’inscrire sur le site, chercher des produits et articles et les mettre dans son panier.

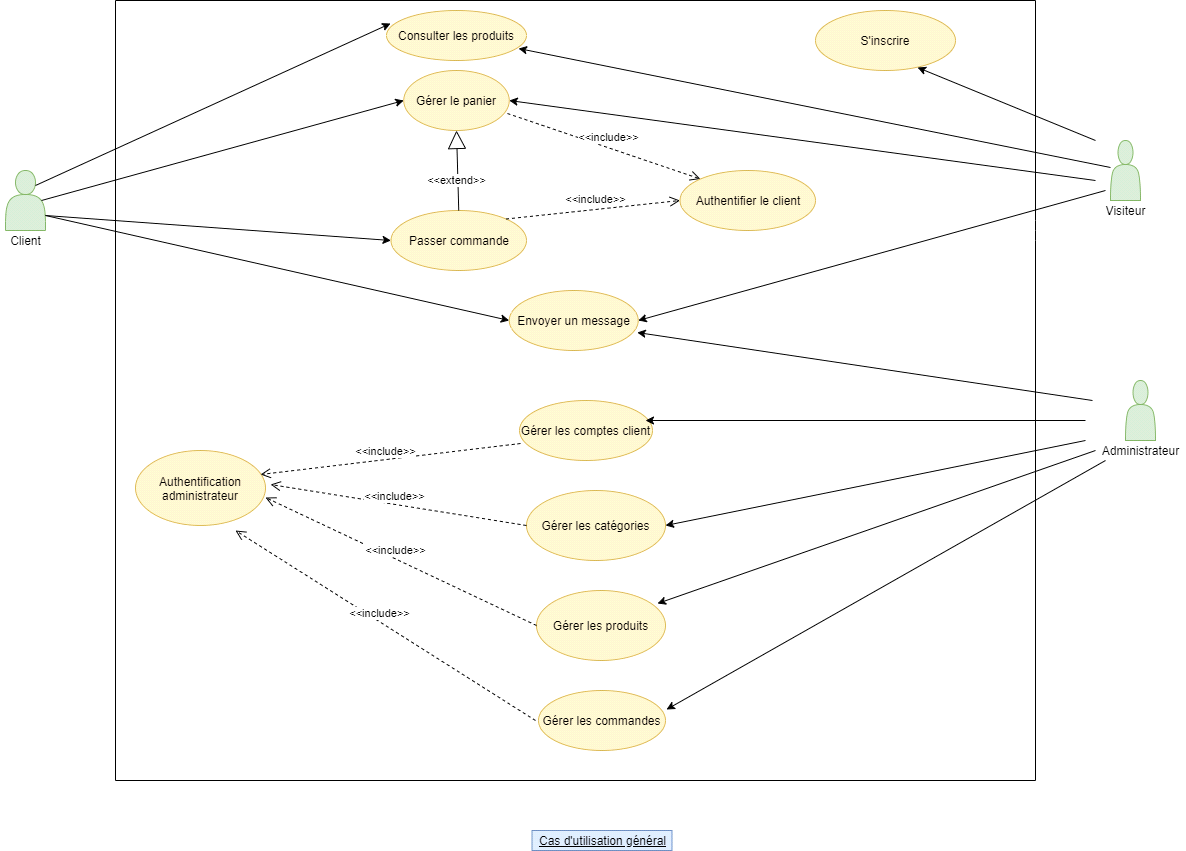
***FIGURE : Cas d'utilisation visiteur« e-shop»***

* **Administrateur :**

C’est un acteur actif, il vérifie toutes les actions effectuées dans ce site. C’est lui qui va gérer les catégories, gérer les produits, gérer son compte, gérer les clients, gérer les promotions et les commandes.

**III.1.1. Diagramme de cas d'utilisation global « e-shop ».**

La figure suivante désigne le diagramme de cas d'utilisation global « boutique en ligne » dans lequel nous allons mettre en évidence les services offerts par l'application.

***FIGURE : Cas d'utilisation global « Vente en ligne »***

**III.1.2. Explication du diagramme de cas d’utilisation :**

***Tableau 1 : Explication de diagramme de cas d’utilisation global.***

|  |  |
| --- | --- |
| **Diagrammes** | **Explications** |
| Diagramme de cas d’utilisation | Ce diagramme de cas d’utilisation présente 3 types d’acteurs : visiteur, client et administrateur :  **Visiteur :**  Rechercher article ou produit : Le visiteur doit chercher le produit soit par catégorie ou par moteur de recherche. S'inscrire : remplir un formulaire.  **Le Client :**  Passer Commande : pour passer les commandes le client doit choisir le mode de paiement ou de livraison, et il peut choisir la quantité d’article.  **Administrateur :**  S'authentifier : saisir son login et son mot de passe.  Gérer les produits/articles : pour ajouter ou modifier et supprimer un article ou produit.  Gérer les catégories : pour ajouter ou supprimer ou modifier une catégorie.  Gérer les promotions : pour ajouter ou supprimer et modifier une promotion. Gérer son compte : remplir l’authentification. |

**Conclusion :**

Dans ce chapitre on a dégagé les besoins fonctionnels et non fonctionnels auxquels doit répondre l’application. Ensuite on a pu présenter les besoins en spécifiant plus en détails les fonctionnalités attendues à travers des diagrammes de cas d’utilisation. On peut ainsi passer au dernier chapitre qui est la conception.

**Chapitre 4 :**

**Conception**

**Introduction :**

Ce chapitre est consacré à la conception de modèles d’architectures et des différents modules du projet, conformément à la technique de modélisation UML.

On doit aussi présenter le diagramme de classes qui représente une vue statique de l’application, et par la suite expliciter une vue dynamique moyennant un diagramme de séquences par la suite on évoquera le diagramme d’activité du site.

**I.Modèles d’architectures :**

Dans cette partie, j’ai présenté quelques architectures associées à notre système, et qui doit expliquer cette application.

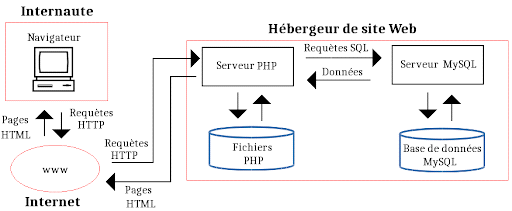
**I.1. Architecture 2tiers :**

C’est une architecture qui est composée d’un client (navigateur.) et d’un serveur (web + base de données).

**I.1.1. Fonctionnement d'un système client/serveur :**

Le client émet une requête vers le serveur grâce à son adresse IP et le port qui désigne un service particulier du serveur.

Le serveur reçoit la demande et répond à l'aide de l'adresse de la machine cliente et son port.



***FIGURE : Système client/serveur.***

**I.1.2 Types de clients de l’architecture 2 tiers :**

**Client léger :**

Le poste client accède à une application située sur un ordinateur dit « serveur » via une interface et un navigateur Web.

L'application fonctionne entièrement sur le serveur, le poste client reçoit la réponse « toute faite » à la demande (requête) qu'il a formulée.

* **Un serveur de bases de données :**

Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table.

Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations définies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête.

Le SQL dans ``MySQL'' signifie ``STRUCTURED QUERY LANGUAGE'' : le langage standard pour les traitements de bases de données.

* **Une base de données :**

C'est un ensemble organisé de données. Cela peut aller d'une simple liste de courses au supermarché à une galerie de photos, ou encore les grands systèmes d'informations des multinationales.

Pour ajouter, lire et traiter des données dans une base de données, vous avez besoin d'un système de gestion de bases de données tel que le serveur MySQL.

Comme les ordinateurs sont très bons à manipuler de grandes quantités de données, le système de gestion de bases de données joue un rôle central en informatique, aussi bien en tant qu'application à part entière, qu'intégré dans d'autres logiciels.

**II. Conception détaillée :**

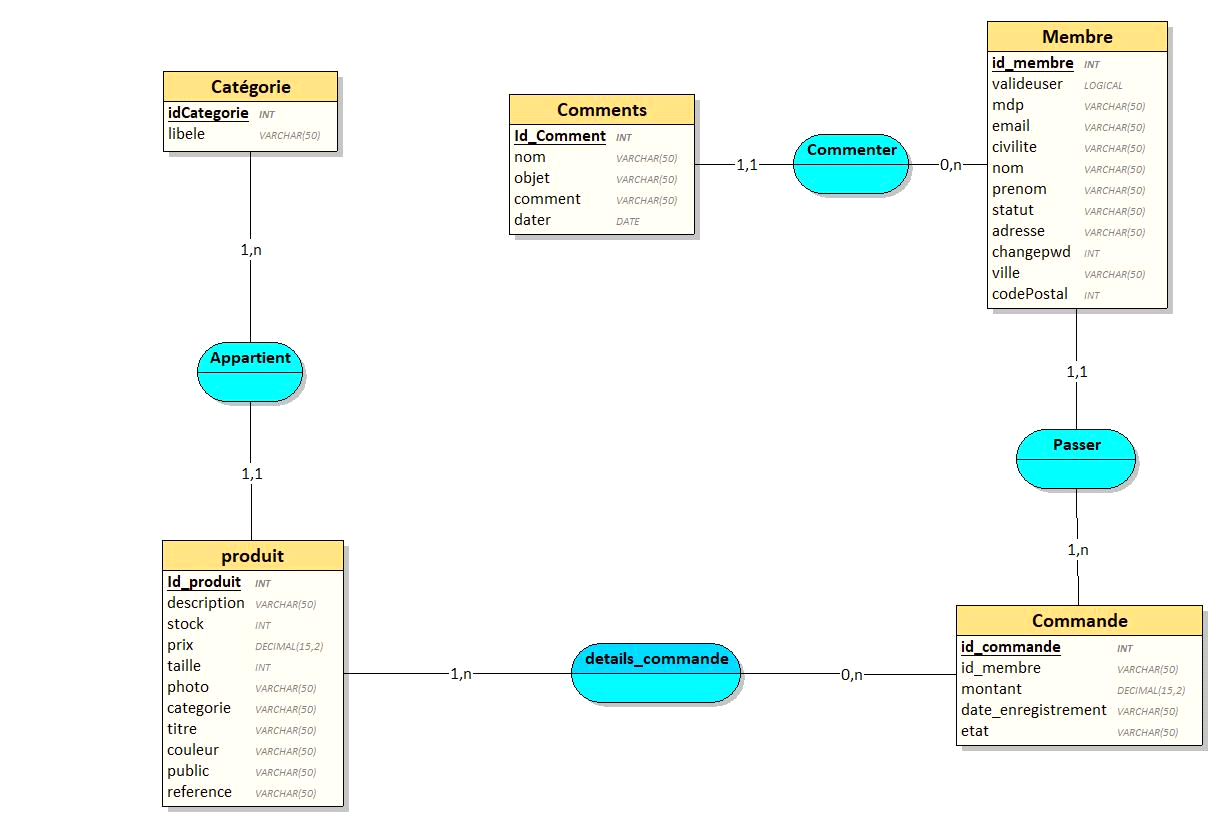
Après avoir étudié les différentes parties de notre application (partie client et partie serveur) d’une façon globale on doit détailler dans cette phase ces parties en présentant les diagrammes de classes, les diagrammes de séquences et les diagrammes d’activités.

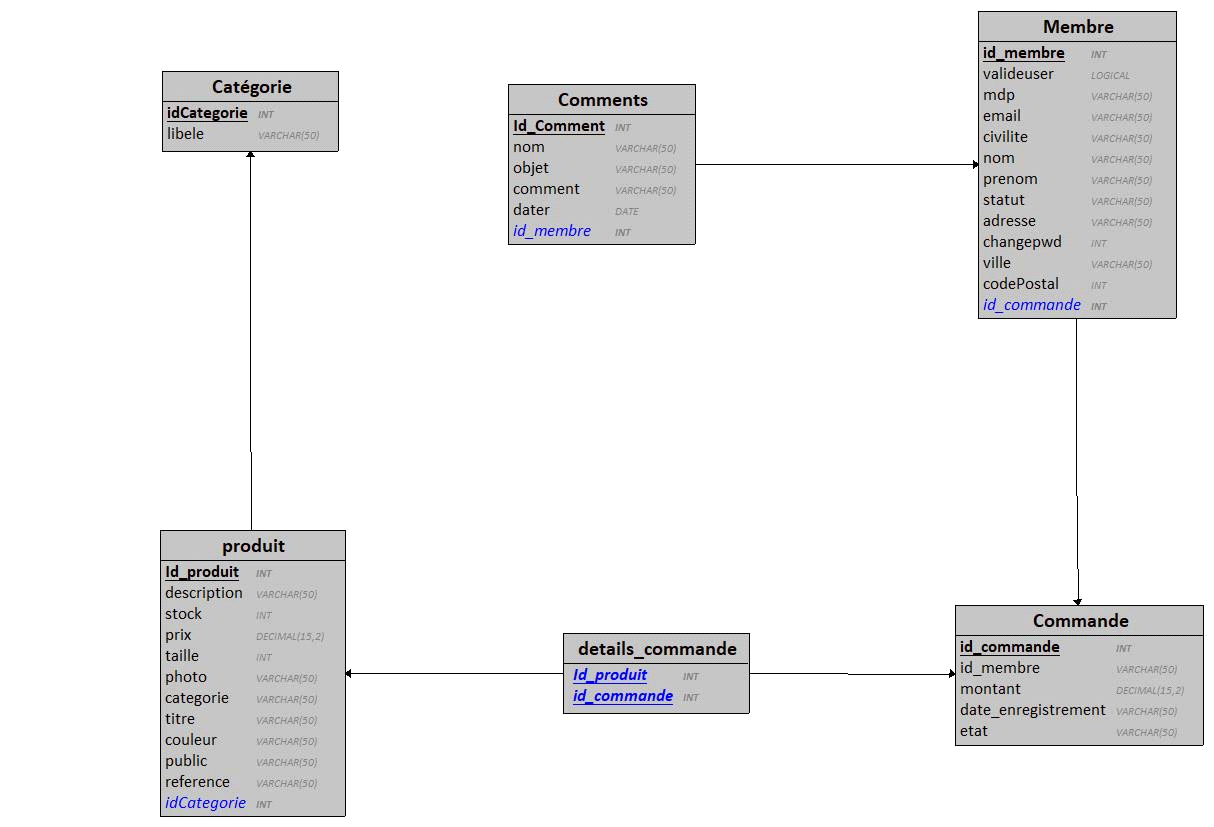
**II.1. Diagramme des classes :**

Le diagramme de classes est considéré comme le plus important de la modélisation orienté objet. Il s'agit d'une vue statique du fait qu'on ne tient pas compte du facteur temporel dans le comportement du système, et qui présente les classes et les interfaces des systèmes .

MERISE est une méthode informatique dédiée à la MODÉLISATION qui analyse la structure à informatiser en terme de systèmes. Le gros avantage de cette méthode est qu’elle permet de cadrer le projet informatique et de « discuter » en se comprenant entre utilisateurs et informaticiens.

**II.1.1 Diagramme de classe global : « « e-shop » »**

***FIGURE : Diagramme MCD« « boutique en ligne :  e-shop» »***

***FIGURE : Diagramme MPD« « boutique en ligne : e-shop » »***

Catégorie = (idCategorie INT, libele VARCHAR(50));

produit = (Id\_produit INT, description VARCHAR(50), stock INT, prix DECIMAL(15,2), taille INT, photo VARCHAR(50), categorie VARCHAR(50), titre VARCHAR(50), couleur VARCHAR(50), public VARCHAR(50), reference VARCHAR(50), #idCategorie);

Commande = (id\_commande INT, id\_membre VARCHAR(50), montant DECIMAL(15,2),date\_enregistrement VARCHAR(50), etat VARCHAR(50));

Comments = (Id\_Comment INT, nom VARCHAR(50), objet VARCHAR(50), comment VARCHAR(50), dater DATE);

Membre = (id\_membre INT, valideuser LOGICAL, mdp VARCHAR(50), email VARCHAR(50), civilite VARCHAR(50), nom VARCHAR(50), prenom VARCHAR(50), statut VARCHAR(50), adresse VARCHAR(50), changepwd INT, ville VARCHAR(50), codePostal INT, #id\_commande);

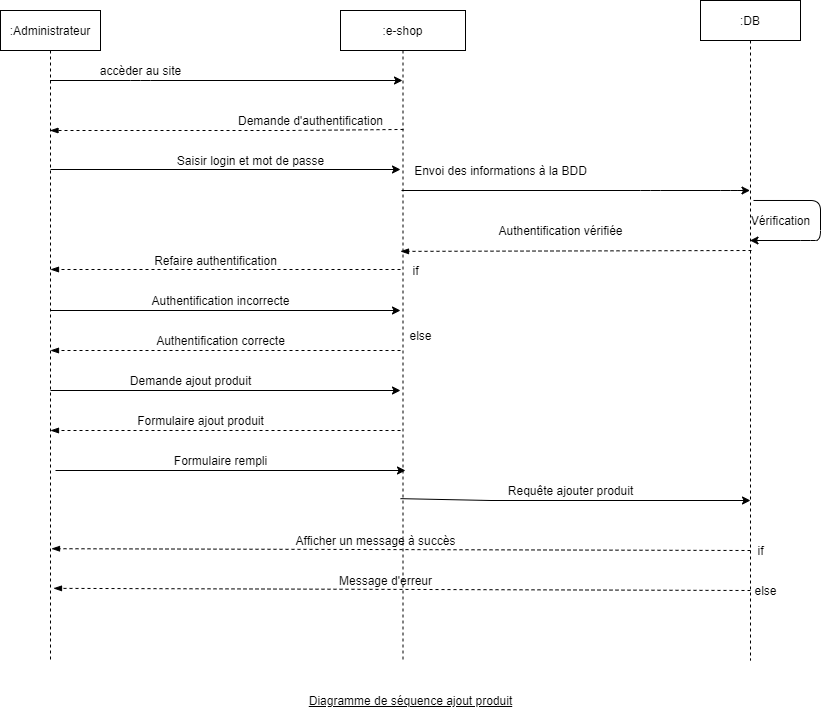
details\_commande = (#Id\_produit, #id\_commande,designation,quantite,prix,photo);

***FIGURE : Diagramme MPD« « boutique en ligne : e-shop » »***

**II.2 Diagramme de séquence :**

La description de la vue dynamique de notre application est réalisée à travers les différents diagrammes de séquence. Ce dernier exprime la séquence d’interaction entre objet des systèmes selon un point de vue temporaire pour une relation. Le cas d’utilisation est un diagramme de séquence.

**II.2.1 Diagramme de séquence d’ajout d'un produit/article :**

***FIGURE : Diagramme de séquence « « Ajouter d'un produit/article » »***

**Explication du diagramme de séquence :**

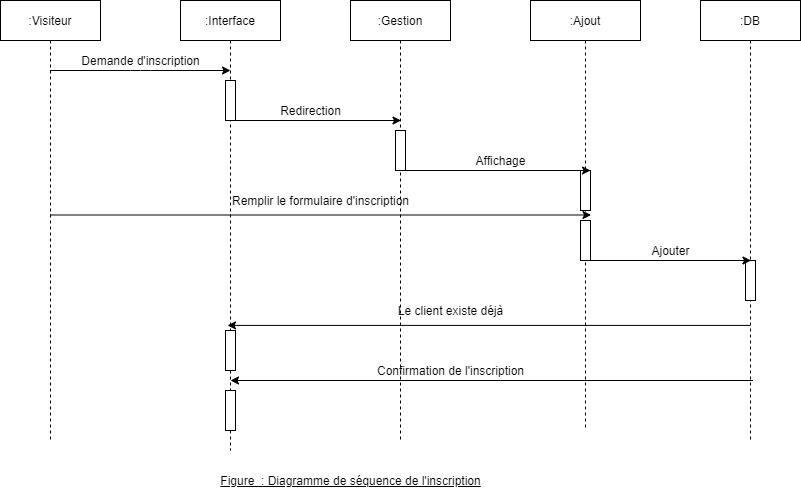
|  |  |
| --- | --- |
| **Diagramme** | **Explications** |
| Diagramme de séquence pour administrateur | **L’Administrateur :**  Accéder au site : accéder au site de boutique en ligne.  Demande d’authentification (login et mot de passe) : l’administrateur entre son login et son mot de passe.  Saisir login et mot de passe : saisir son login et son mot de passe.  Envoi des informations à la BDD : rechercher l’authentification qui a été saisi par l’administrateur dans la table de BDD.  -Vérification : vérifié que le login et mot de passe sont enregistrés dans la table de la base de données.  Authentification vérifiée.  -Authentification incorrecte : l’administrateur doit refaire une autre fois son login et son mot de passe.  -Authentification correcte : le compte de l’administrateur serra ouvert.  -Demande d'ajout d'un produit/article : l’administrateur doit ajouter un nouveau produit/article à la liste de produits/articles existants.  -Formulaire d'ajout un produit/article : affichage d’un formulaire d’ajout d’un produit ou article pour l’administrateur.  -Formulaire rempli : saisir le nom, description, marque d’article ou produit.  -Requête d’ajouter un produit/article : les informations que l’utilisateur doit saisir sont envoyées à la BDD.  -Afficher un message à succès : produit ou article serra ajouter et enregistrer dans la BDD.  -Sinon message d'erreur : le formulaire d’ajout pas bien rempli. |

***Tableau : Explication du diagramme de séquence « Ajouter un produit/article ».***

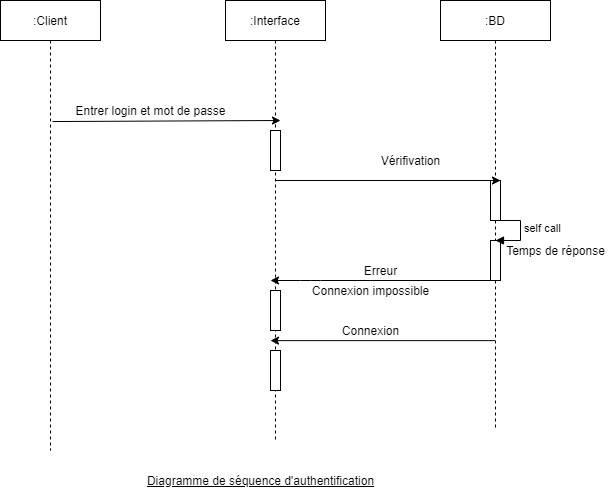
**II.2.2 Diagramme de séquence de modification d'un produit/article :**

Même fonctionnement que l’ajout d’un produit !!!

**II. 2.3 Diagramme de séquence d’inscription :**



***FIGURE : Diagramme de séquence « « inscription » »***

***FIGURE : Diagramme de séquence « « Authentification» »***

**Explication de diagramme de cas séquence :**

***Tableau 4: Explication du diagramme de séquence «inscription».***

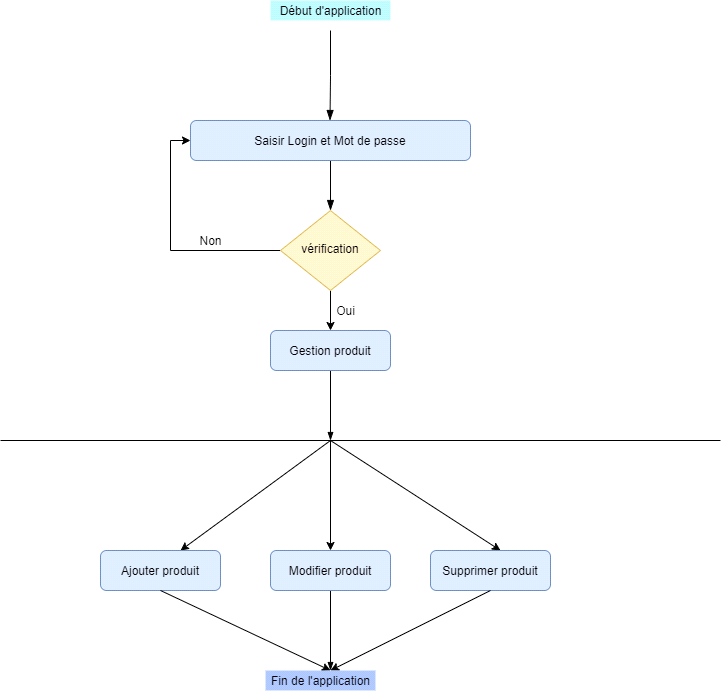
|  |  |
| --- | --- |
| **Diagramme** | **Explications** |
| Diagramme de séquence | Visiteur : - Accéder au site : accéder au site de boutique en ligne.  -Afficher le menu : le système lui affiche le menu d’inscription au client.  -Choisir l’option inscription : le client doit se repérer sur la menue inscription.  -Affichage du formulaire : le système lui affiche le formulaire d’inscription qu'il va remplir.  -Remplissage du formulaire : le client remplit le formulaire.  -Vérification : le système le vérifie que le formulaire est bien rempli.  -Insertion dans la base : effectue l’insertion des données dans la base de donnés.  -Enregistrement : enregistrer l’insertion des données.  -message d’inscription effectuée avec succès : toutes les donnée serront ajoutées avec succès.  -message d'erreur : le formulaire est incorrect.  -Répéter l'action : remplir une autre fois le formulaire. |

**II.3 Diagramme d’activité :**

Dans la phase de conception, les diagrammes d’activités sont particulièrement adaptés à la description des cas d’utilisations.

Donc le diagramme d’activité représente comment l’administrateur doit gérer les produits/articles. Les diagrammes d’activités ont pour but de représenter l’état de création d’une application.

**III.3.1Diagramme d’activité de Gestion produit/article :**

***FIGURE : Diagramme d’activité « « Gestion produit/article » »***

**Explication du diagramme d’activité :**

Après son authentification correcte, l’administrateur doit gérer les produits ou articles.

Il gére les produit ou articles par 3 actions :

L’administrateur doit ajouter de nouveaux produits et il doit ajouter aussi de nouvelles promotions.

Il doit modifier produit ou article.

L’administrateur a la possibilité de supprimer un produit et une promotion.

**Conclusion :**

Les différents diagrammes exposés dans ce chapitre expliquent bien le fonctionnement global du site.

Cette modélisation nous donne une idée claire sur les principes et les objectifs de l’application¸ et on peut facilement par la suite atteindre la phase de réalisation.

**Chapitre 5 :**

**Réalisation**

**Prérequis**

- Maîtrise des langages PHP , HTML et CSS.

- Bonnes connaissances du langage SQL.

**Objectifs**

- Créer un site web « 2en1 » avec une partie cliente (front) et une interface de gestion (back).

- Développement des fonctionnalités. (CRUD orienté e-commerce).

**Introduction**

Dans ce chapitre je présente la description des environnements matériels et logiciels qui ont permis la réalisation de notre projet.

Ensuite je détaille les différents outils qu’on utilisera pour réaliser notre application, et enfin je donne un aperçu sur le travail réalisé.

**I. Environnement de développement :**

Je présente dans ce qui suit les choix techniques qui ont aidé à organiser notre application, le matériel utilisé lors de la réalisation de l’application et les environnements logiciels avec lesquels j’ai assuré le développement.

**I.1. Environnement matériel :**

* Ordinateur dont les caractéristiques sont comme suite :
* Marque : LENOVO
* Mémoire : 16,0 Go de RAM
* Processeur Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz, 2808 MHz, 6 cœur(s), 6 processeur(s) logique(s)
* Système d’exploitation : Microsoft Windows 10 Professionnel

**I.2. Environnement logiciel :**

Les logiciels utilisés dans cette application sont :

* Drawio : diagrammes cas d’utilisation et séquence
* Figma : maquetage
* Looping : digrammes UML
* VS code : éditeur de code
* XAMPP : plateforme hébergement<<Apache>> et base de donnée<<Mysql>>
* GitHub : versionning et partage

Microsoft Word : pour la rédaction du rapport.

**II. Outils de développement :**

Dans cette partie je détaille les différents outils utilisés pour réaliser notre application :

**II.1. Choix de Xampp:**

Xampp est une plateforme de développement permettant de faire fonctionner localement (sans se connecter à un serveur externe) des scripts PHP.

Xampp n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.

La grande nouveauté de Xampp réside dans la possibilité d'y installer et d'utiliser n'importe quelle version de PHP, Apache ou MySQL en un clic. Ainsi, chaque développeur peut reproduire fidèlement son serveur de production sur sa machine locale.

Les rôles des composants de Xampp sont les suivants :

* Apache est le serveur web « frontal » : il est « devant » tous les autres et répond directement aux requêtes du client web (navigateur).
* Le langage de script PHP sert la logique et permet des traitements (calculs, vérification, test, etc.…).
* MySQL stocke toutes les données de l'application (c'est une base de données).

**II.2. Choix du langage PHP :**

PHP (Hypertext Processor) Signifiait (Personnal Homme Page) avant, est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au HTML.

C'est un langage de scripts libre principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété.

* Le PHP est disponible sur la grande majorité des hébergements grand public.
* Il permet à n'importe quel bricoleur d'ajouter rapidement quelques  
  pages dynamiques dans un site statique.

**II.3Visual Studio Code (VS code) :**

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS2. 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.

**II.4. Choix de CSS 3 :**

Les feuilles de style en cascade1, généralement appelées CSS de l'anglais Cascading Style Sheets, forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C). CSS est couramment utilisé dans la conception de sites web et bien pris en charge par les navigateurs web.

**II.5. Choix de HTML5 :**

Le HyperText Markup Language, généralement abrégé HTML ou dans sa dernière version HTML5, est le langage de balisage conçu pour représenter les pages web.

Ce langage permet :

- d’écrire de l’hypertexte, d’où son nom,

- de structurer sémantiquement la page,

- de mettre en forme le contenu,

- de créer des formulaires de saisie,

- d’inclure des ressources multimédias dont des images, des vidéos, et des programmes informatiques,

- de créer des documents interopérables avec des équipements très variés de manière conforme aux exigences de l’accessibilité du web.

**II.6. Choix de Figma :**

Figma est un éditeur de graphiques vectoriels et un outil de prototypage. Il est principalement basé sur le web, avec des fonctionnalités hors ligne supplémentaires activées par des applications de bureau pour macOS et Windows. L'ensemble des fonctionnalités de Figma est axé sur l'utilisation dans la conception de l'interface utilisateur et de l'expérience utilisateur, en mettant l'accent sur la collaboration en temps réel.

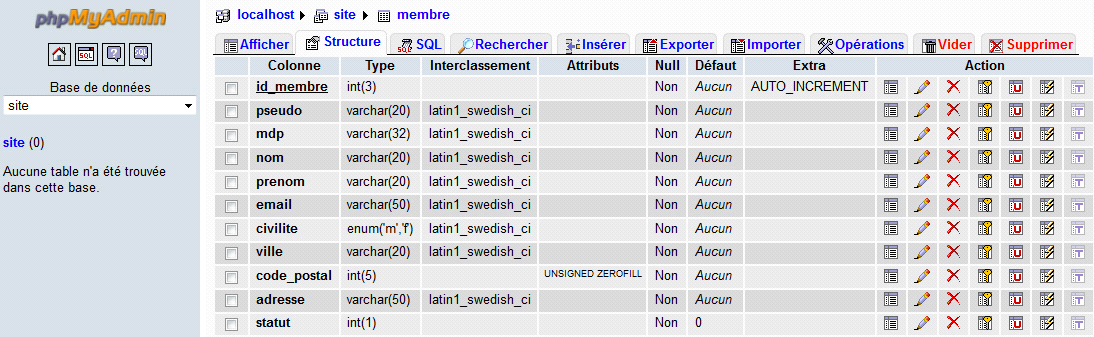
**III.Réalisation du site Etape par Etape :**

**Etape 1 : Modélisation et création de la base de données, table et champs.**

**La table membre**  
  
Base de données : **db\_e\_shop**   
Table : **membre** 

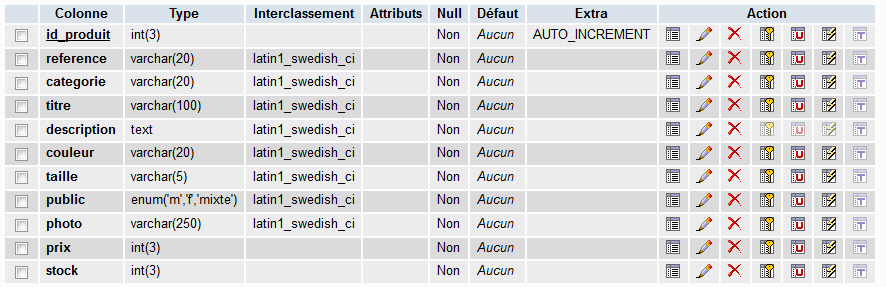
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Champ** | **Type** | **Taille** | **Spécificité** | **Description** |
| id\_membre | INT | 3 | Clé primaire (PK - Primary Key), AUTO\_INCREMENT (AI) | Ce champ correspond au numéro du membre qui sera auto-généré et incrémenté |
| mdp | VARCHAR | 32 | - | Ce champ correspond au mot de passe du membre. La taille fait 32 caractères car il sera crypté par la suite. |
| nom | VARCHAR | 20 | - | Ce champ correspond au nom de famille du membre. |
| prenom | VARCHAR | 20 | - | Ce champ correspond au prénom du membre. |
| email | VARCHAR | 50 | - | Ce champ correspond à l'email du membre. |
| civilite | ENUM | 'm','f' | - | Ce champ correspond à la civilité du membre. Le membre sera soit Homme (M) soit Femme (F). Logiquement il n'y a que 2 choix possibles :p |
| ville | VARCHAR | 20 | - | Ce champ correspond à la ville du membre. |
| code\_postal | INT | 5 | UNSIGNED ZEROFILL | Ce champ correspond au code postal du membre. |
| adresse | VARCHAR | 50 | - | Ce champ correspond à l'adresse du membre. |
| statut | INT | 1 | DEFAULT 0 | Ce champ correspond au statut du membre. Par défaut il sera à zéro (ce qui correspondra à 1 membre). Nous pourrons mettre le chiffre 1 pour donner des droits d'administration à certains membres (1 admin est aussi 1 membre). |

**Structure de la table :**



**La table produit**  
  
Base de données : **site**   
Table : **produit** 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Champ** | **Type** | **Taille** | **Spécificité** | **Description** |
| id\_produit | INT | 3 | Clé primaire (PK - Primary Key), AUTO\_INCREMENT (AI) | Ce champ correspond au numéro du produit qui sera auto-généré et incrémenté |
| reference | VARCHAR | 20 | UNIQUE | Ce champ correspond à la référence du produit. Il sera unique et par conséquent 2 produits ne pourront pas avoir la même référence. |
| categorie | VARCHAR | 20 | - | Ce champ correspond a la catégorie du produit |
| titre | VARCHAR | 100 | - | Ce champ correspond au titre du produit. |
| description | TEXT | - | - | Ce champ correspond a la description du produit. |
|  |  |  |  |  |
| couleur | VARCHAR | 20 | - | Ce champ correspond à la couleur du produit. |
| taille | VARCHAR | 5 | - | Ce champ correspond à la taille du produit. |
| public | ENUM | 'm','f', 'mixte' | - | Ce champ permettra de determiner à quel public est destiné ce produit. Les choix possibles sont Homme (M), soit Femme (F) ou mixte (mixte). |
| photo | VARCHAR | 250 | - | Ce champ correspond au chemin de la photo qui sera enregistré pour représenté le produit. Ce ne sera pas le fichier image directement mais bien son chemin qui sera enregistré. |
| prix | INT | 3 | - | Ce champ correspond au prix du produit. |
| stock | INT | 3 | - | Ce champ correspond au stock restant du produit. |

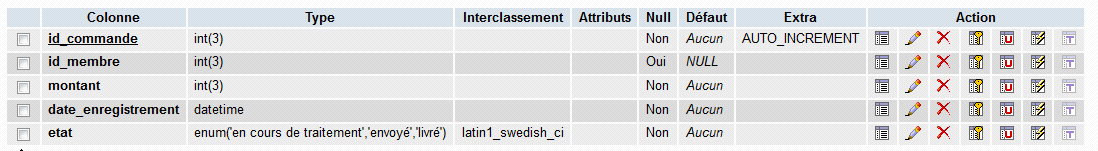
**Création de la structure de la table (champs/colonnes) :  
  
Structure de la table:**  


**La table commande**

Base de données : **site** 

Table : **commande** 

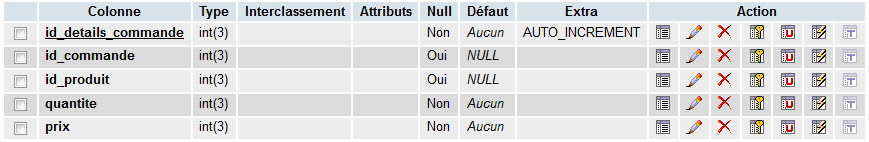
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Champ** | **Type** | **Taille** | **Spécificité** | **Description** |
| id\_commande | INT | 3 | Clé primaire (PK - Primary Key), AUTO\_INCREMENT (AI) | Ce champ correspond au numéro de commande qui sera auto-généré et incrémenté |
| id\_membre | INT | 3 | Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL | Ce champ correspond à l'id\_membre qui aura commandé. |
| montant | INT | 3 | - | Ce champ correspond au montant total (en euros) de la commande |
| date\_enregistrement | DATETIME | - | - | Ce champ correspond à la date et heure d'enregistrement de la commande. |
| etat | Enum | 'en cours de traitement','envoyé','livré' | DEFAULT 'en cours de traitement' | Ce champ correspond a l'état de commande. |

Structure de la table (relecture) :  


**La table details\_commande**  
  
Base de données : **site**   
Table : **details\_commande** 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Champ** | **Type** | **Taille** | **Spécificité** | **Description** |
| id\_details\_commande | INT | 3 | Clé primaire (PK - Primary Key), AUTO\_INCREMENT (AI) | Ce champ correspond au numéro du détails de la commande qui sera auto-généré et incrémentée |
| id\_commande | INT | 3 | Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL | Ce champ correspond à l'id\_commande a laquelle le détail est rattaché. |
| id\_produit | INT | 3 | Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL | Ce champ correspond à l'id\_produit qui aura été commandé. |
| quantite | INT | 3 | - | Ce champ correspond à la quantité demandée par produit. |
| prix | INT | 3 | - | Ce champ correspond au prix du produit |

**Structure de la table :**



Avec cette table nous pourrons avoir 1 commande (dans la table commande) comprenant par exemple 5 produits (dans la table details\_commande, pour cette même commande).

**Etape 2. Création d'une aborescence**

Nous allons créer notre arborescence de site web :

|  |
| --- |
| -  /e-shop/ |
| -------  /conception/ ------- -------  /db/ |
| ------- ------- -------  db.sql -------  /photo/ -------  /admin/ ------- -------  gestion\_boutique.php ------- -------  gestion\_membre.php -------  /ctrl/ ------- -------  /img/ ------- -------  /js/ ------- -------  /css/ ------- ------- -------  style.css ------- -------  init.ctrl.php ------- -------  fonction.ctrl.php ------- -------  header.php ------- -------  footer.php |
| ------- -------  /Model/ ------- -------  commande.php ------- -------  details\_commande.php ------- -------  membre.php ------- -------  produit.php |
| -------  /Vue/ ------- -------  inscription.php ------- -------  connexion.php |
| ------- -------  gestion\_commande.php ------- -------  profil.php ------- -------  boutique.php ------- -------  fiche\_produit.php ------- -------  panier.php |
| ------- -------  feedback.php |
| ------- -------  facture.php |
| ------- -------  powlost.php |
| ------- -------  recherch.php |
| ------- -------  vitrine.php |
| ------- -------  paiement.php |
| ------- -------  membres.php -------  .htaccess |
| -------  .index.php |

Notre site web se trouvera à l'intérieur du dossier  /e-shop/.  
  
1 dossier  /photo/ sera présent pour contenir les photos de nos produits.  
  
1 dossier  /admin/ sera présent pour contenir les pages d'administration (BackOffice).  
  
1 dossier  /ctrl/ sera présent pour contenir les fichiers n'étant pas des pages web a part entière (bien souvent il s'agit de fichiers inclus dans des pages web).  
  
les pages web du site web côté FRONT :

*  inscription.php : page d'inscription pour les visiteurs
*  connexion.php : page de connexion
*  profil.php : page de profil pour connectés
*  vitrine.php : catalogue des différents produits séparés par catégorie
*  fiche\_produit.php : fiche d'un produit en particulier
*  panier.php : panier
*  panier.php : panier
*  commentaire.php : nous contacter
*  .htaccess : fichier permettant de faire des réglages, notamment sur les URLS pour améliorer le référencement/li>

**Etape 3. Ecriture des fichiers en inclusion**

**Ecriture du fichier ctrl/init.ctrl.php**

Le fichier init.ctrl.php (destiné à l'inclusion et non pas à l'affichage), va nous permettre d'initialiser plusieurs choses sur notre site web.  
  
Le fichier init.ctrl.php sera donc inclut par toutes nos pages web afin de profiter de l'initialisation.  
  
**Quelques explications du code PHP**  
  
**Mysqli** est une classe prédéfinie en PHP me permettant de me connecter à la base de données.

Pour cela il est nécessaire de lui annoncer le nom du serveur, le email, le mot de passe, et la base de données à laquelle nous souhaitons nous connecter.  
  
1 condition est présente **$mysqli->connect\_error** pour afficher un message d'erreur en Français si jamais la connexion ne peut pas se faire.  
  
**$mysqli->set\_charset("utf8");** permet de régler l'encodage de la base de données.  
  
  
**session\_start()** permet de créer (ou lire) 1 fichier de session sur le serveur. Sans cette ligne, nous ne pourrons pas connecter d'internautes à leurs espaces membres plus tard.  
Session\_start() permettra en effet de maintenir (et ne pas perdre) l'internaute connecté au site web même s'il navigue de page en page.  
  
**define("RACINE\_SITE","//");** permettra de gérer notre site web en chemin absolu et non pas relatif. Et pour éviter tout problème, nous pourrons modifier cette constante une fois en ligne pour que cela ait une repercution immédiate partout où elle sera appelée..  
  
**$contenu = '';** est une variable initialisée à vide pour éviter d'avoir des erreurs undefined si jamais nous tentons de l'afficher.  
Nous nous servirons de cette variable pour retenir des messages que nous devrions adresser à l'internaute, cela nous permettra de faire 1 affichage global de tous nos éventuels messages à un endroit précis.  
  
**require\_once("fonction.ctrl.php");** nous allons inclure notre fichier de fonction avec nous. Du coup, lorsque nous appellerons le fichier init.ctrl.php, cela aura aussi pour effet d'inclure le fichier fonction.ctrl.php.  
  
**Ecriture du fichier ctrl/header.php**

Ce fichier nous permet de déclarer le Header avec le menu comprenant quelques liens (anticipés, puisque les pages ne sont pas encore créées).  
Les balises ne sont pas toutes fermées, c'est volontaire puisque nous allons mettre du contenu entre le fichier header.php et footer.php.  
  
**Ecriture du fichier ctrl/footer.php**

Nous allons fermer les balises indispensables de notre site web au format HTML.

**Ecriture du fichier ctrl/css/style.css**

Nous allons écrire un code permettant de faire une mise en forme minimale.

Durant le projet, nous avons eu l'occasion de revenir dans ce fichier pour l'alimenter d'avantages

**Ecriture de notre fichier ctrl/fonction.ctrl.php**

Nous allons créer 2 fichiers pour nous aider dans la conception du site web.

* Une fonction que nous nommerons **executeRequete** pour exécuter des requêtes SQL .  
  En effet, échanger des informations avec la base est une action que nous devrons faire certainement plusieurs fois par page web, autant nous faciliter un peu la vie avec une fonction déjà prête.
* Une fonction que nous nommerons **debug** pour nous debugger !  
  En PHP, il est souvent nécessaire d'effectuer des var\_dump ou print\_r pour voir le contenu d'un tableau array, d'un objet ou de variables, nous allons donc prévoir un code en conséquence afin de gagner du temps.

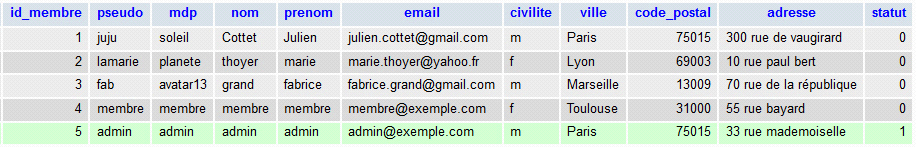
**La fonction executeRequete**  
  
**function executeRequete($req)** La fonction sera destinée à recevoir 1 argument entrant (la requête SQL arrivera dans la variable de reception **$req** prévue à cet effet).  
  
**global $mysqli;** permet d'avoir accès à la variable $mysqli définie dans le fichier **init.ctrl.php** (espace global) à l'intérieur de notre fonction (espace local).  
  
**$resultat = $mysqli->query($req);** on exécute la requête reçue en argument et on gardera les résultats dans la variable **$resultat**.  
  
**if(!$resultat)** si la variable **$resultat** renvoie false, c'est qu'il y a une erreur de requête SQL.  
  
**die("Erreur sur la requete sql.<br>Message : " . $mysqli->error . "<br>Code: " . $req);** Dans le cas où la requête échoue, on lui demande d'adresser 1 message et d'arreter l'exécution du code avec l'utilisation de die.  
  
**return $resultat;** en cas d'une requête de SELECTion, on retournera un objet issu de la classe mysqli\_result. Sinon (pour INSERT/UPDATE/DELETE), nous retournerons un boolean TRUE (1).

**Etape 4. Ecriture de la page inscription.php**

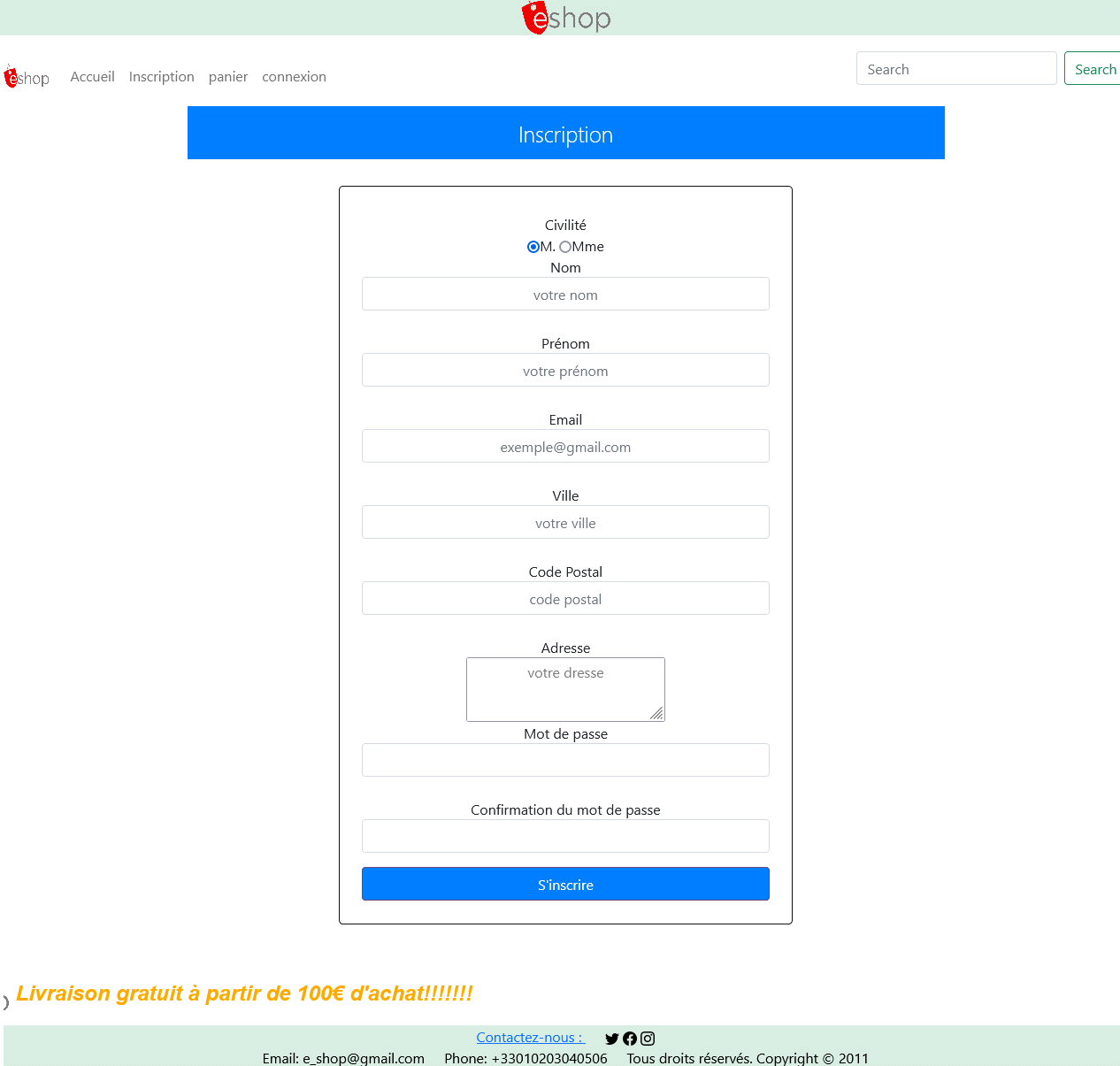
Nous allons commencer notre site web par l'espace membre et donc la page d'inscription.  
Pour cela, commençons par le formulaire HTML :

Nous incluons le fichier init.ctrl.php, le Header, le Footer, et entre le haut et le bas nous mettons notre formulaire html afin que nos futurs internautes puissent s'inscrire.  
  
Il est très important que les attributs name du formulaire soient prévus afin de pouvoir récupérer et exploiter les saisies en PHP. De préférence, nous pouvons garder les mêmes name que le nom de nos champs dans notre base de données.

Il faut aussi prévoir la partie traitement en PHP.   
  
**Quelques Explications du code PHP**  
  
**if($\_POST)** Cette condition IF permet de detecter si l'internaute à cliquer sur le bouton submit pour s'inscrire.  
  
**$membre = executeRequete("SELECT \* FROM membre WHERE email='$\_POST[email]'");** Nous allons utiliser notre fonction executeRequete pour allez voir si le email que l'internaute tente de prendre n'est pas déjà attribué à un autre membre.  
  
**if($membre->num\_rows > 0)** Si la requête renvoie plus de 0 résultat (donc au moins 1), c'est que le email est déjà attribué à quelqu'un d'autre.  
  
**$contenu .= "<div class='erreur'>Email indisponible. Veuillez en choisir un autre svp.</div>";** Nous invitons le membre à choisir un autre email si celui qu'il convoite est déjà attribué.  
  
**else** Sinon, on lance l'inscription.  
  
**foreach($\_POST as $indice => $valeur){ $\_POST[$indice] = htmlEntities(addSlashes($valeur));}** nous bouclons sur toutes les saisies afin de les passer dans les fonctions prédéfinies PHP htmlEntities et addSlashes.  
  
**executeRequete("INSERT INTO membre (email, mdp, nom, prenom, email, civilite, ville, code\_postal, adresse) VALUES ('$\_POST[email]', '$\_POST[mdp]', '$\_POST[nom]', '$\_POST[prenom]', '$\_POST[email]', '$\_POST[civilite]', '$\_POST[ville]', '$\_POST[code\_postal]', '$\_POST[adresse]')");** Cette requête permet d'insérer le membre dans la base ! C'est à ce moment-là que l'enregistrement se fait.  
  
**$contenu .= "<div class='validation'>Vous êtes inscrit à notre site web. <a href=\"connexion.php\"><u>Cliquez ici pour vous connecter</u></a></div>";** Nous l'informons de son inscription et lui proposons de se connecter.



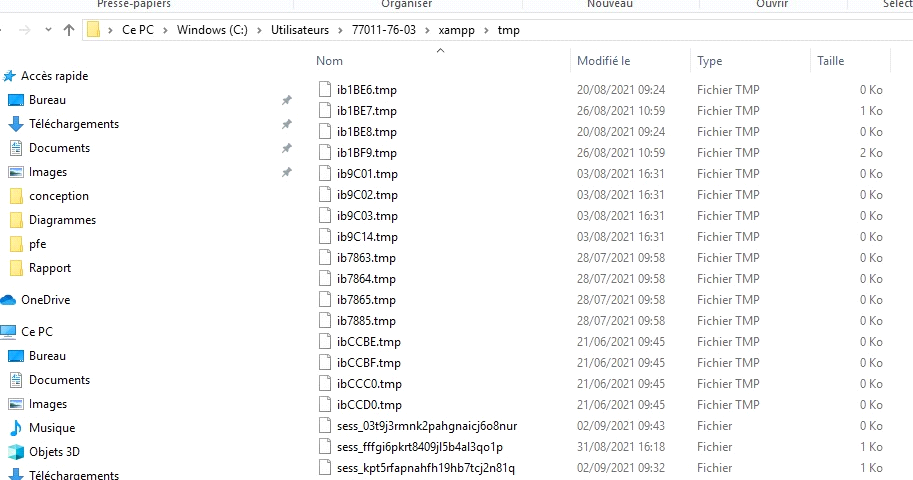
**Résultat - inscription.php**



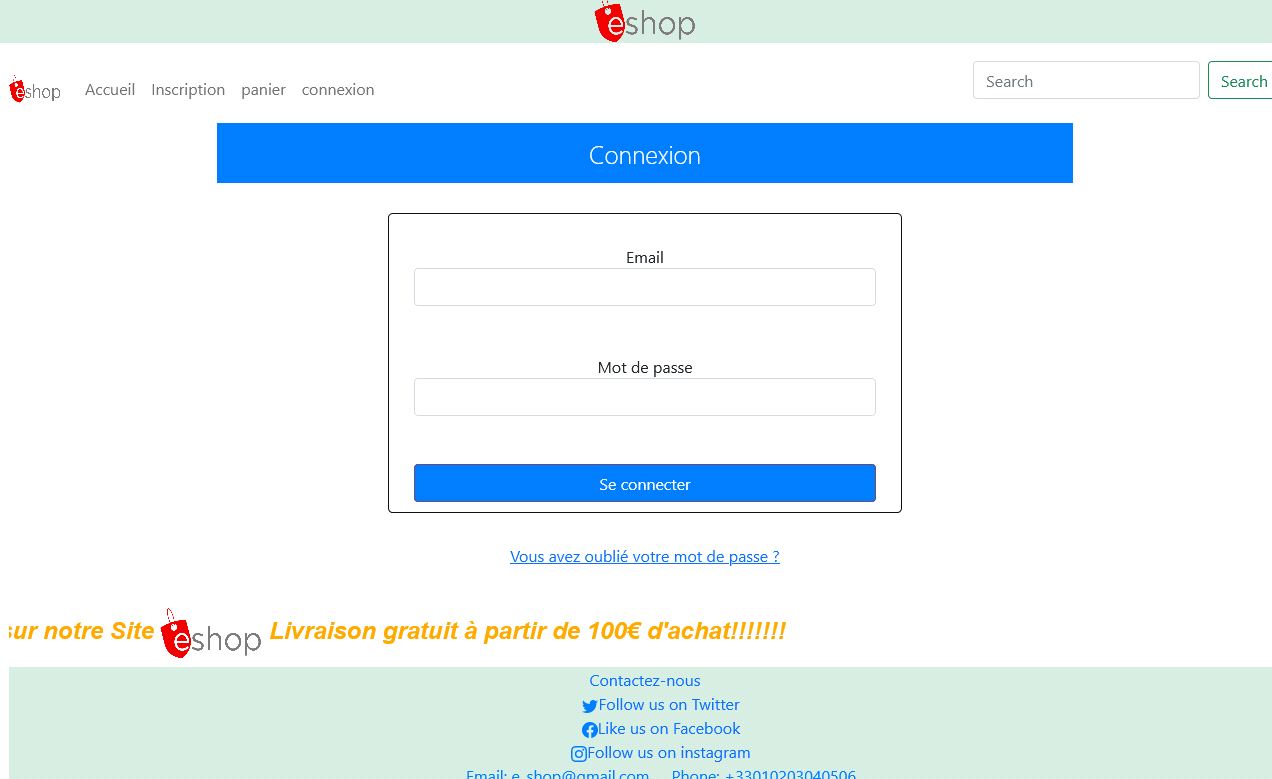
***FIGURE  : Formulaire d'inscription.***

**Etape 5. Ecriture de la page connexion.php**

Maintenant que nous avons une page d'inscription fonctionnelle et plusieurs inscrits, nous allons attaquer la page de connexion !  
  
Pour cela, commençons par le formulaire HTML :

Nous incluons le fichier init.ctrl.php, le Header, le Footer, et entre le haut et le bas nous mettons notre formulaire html afin que nos futurs internautes puissent se connecter.  
  
Il est très important que les attributs name du formulaire soit prévue afin de pouvoir récupérer et exploiter les saisies en PHP. De préférence, pour une meilleure cohérence, nous pouvons garder les mêmes name que le nom de nos champs dans notre base de données.  
  
**Quelques Explications du code PHP**  
  
**if($\_POST)** Cette condition IF permet de detecter si l'internaute a cliqué sur le bouton submit pour se connecter.  
  
**$resultat = executeRequete("SELECT \* FROM membre WHERE email='$\_POST[email]'");** Nous allons utiliser notre fonction executeRequete pour allez consulter la base afin de savoir si le email avec lequel l'internaute tente de se connecter correspond bien à 1 compte réel sur notre site web. Y'a t'il un enregistrement correspondant dans notre base ?  
  
**if($resultat->num\_rows != 0)** Si le nombre de retours est différent de 0 (donc 1 logiquement :p), c'est que le email est connu et que le compte existe, on peut avancer...  
  
**else** Sinon, nous informons l'internaute qu'il y a une erreur sur son email...  
  
**$membre = $resultat->fetch\_assoc();** Revenons sur le cas du email valide, nous devons absolument traiter (fetch\_assoc) pour connaitre les informations récupérées en base. En effet, nous devons savoir si le membre a le bon email mais aussi s'il possède le bon mot de passe associé.  
  
**if($membre['mdp'] == $\_POST['mdp'])** On compare le mdp posté (dans le formulaire de connexion) avec le mdp du membre (récupéré dans la base de données), s'il s'agit du même mdp dans les deux cas, on crée à l'internaute une session et on la remplit avec certaines informations.  
  
**else** Sinon, le mdp est mauvais et nous informons l'internaute.  
**foreach($membre as $indice => $element){if($indice != 'mdp'){ $\_SESSION['membre'][$indice] = $element; } }** Nous créons une session avec les éléments de la base de données. La boucle foreach évite d'écrire les lignes suivantes : $\_SESSION['membre']['id\_membre'] = $membre['id\_membre']; $\_SESSION['membre']['email'] = $membre['email']; etc.. Par sécurité et comme nous n'en n'aurons pas besoin, nous ne conserverons pas le mdp dans la session (condition IF).  
Pour rappel, sans le session\_start() placée dans le fichier init.ctrl.php, nous n'aurions pas pu se servir du système de session en PHP.  
  
**header("location:profil.php");** Si l'accouplement email/mot de passe est bon, nous redirigeons l'internaute vers sa page de profil (puisqu'il est maintenant connecté !)  
  
Quoi qu'il en soit "être connecté à un site web" ne signifie pas seulement avoir le bon email et le bon mot de passe.  
  
Pour maintenir une connexion il faut avoir un fichier de session (sur le serveur du site) et un fichier cookie sur votre ordinateur pour assurer la liaison.  
  
**Session dans le dossier /tmp/ du serveur :**  
  
  
*Ce fichier a été créé par session\_start() (placé dans le fichier ctrl/init.ctrl.php) automatiquement.  
Ce même fichier a été rempli par la superglobale $\_SESSION lors de la connexion.*

**Résultat - connexion.php**

***FIGURE  : Interface d’authentification***

**Etape 6. Ecriture de la page profil.php**

A présent, nos visiteurs peuvent s'inscrire et aussi se connecter !  
C'est un bon début mais si les internautes font la démarche de s'inscrire et de se connecter c'est pour arriver quelque part et ainsi profiter d'un espace reservé aux membres sur le site web.  
  
Nous allons donc prévoir une page de profil avec quelques informations !  
  
Pour récupérer les informations, nous pourrons nous servir du fichier de session *(par l'intermédiaire de la superglobale $\_SESSION)* que nous avons créé au moment de la connexion. (C'est pratique car ces informations de session seront accessibles sur tout le site web).  
  
Avant de rentrer dans l'espace de profil nous devons nous assurer que l'internaute qui tente d'y accèder a les droits nécessaires (c'est à dire, être passé par la page de connexion avec un bon email et un bon mot de passe. En gros avoir un fichier de session).  
  
**Explications du code PHP**  
La fonction **internauteEstConnecte()** va nous permettre de savoir si l'internaute est connecté par une simple condition :  
  
   **if(!isset($\_SESSION['membre'])) return false;** si la session membre n'existe pas, c'est que l'internaute n'est pas passé par la page de connexion (ou alors qu'il a été déconnecté depuis). on retournera la valeur "FALSE" pour dire "Faux l'internaute n'est pas connecté".  
  
   **else** sinon, c'est que la session membre existe et donc que l'internaute est connecté (avec 1 fichier de session + cookie). on retournera la valeur "TRUE" pour dire "Vrai l'internaute est connecté".  
  
L'avantage d'avoir mis ce code dans une fonction **internauteEstConnecte()** et non pas directement dans la page web c'est qu'il sera plus facile de s'en reservir sur les autres pages web (plutôt que re-trimbaler un même morceau code en copier/coller d'un fichier à l'autre). Cela sera pratique de savoir si l'internaute est connecté à divers endroits du site web.  
  
La fonction **internauteEstConnecteEtEstAdmin()** va nous permettre de savoir si l'internaute est connecté en tant qu'administrateur (statut a 1) ou en tant que membre (statut a 0) :  
  
   **if(internauteEstConnecte() && $\_SESSION['membre']['statut'] == 1** si la fonction internauteEstConnecte() renvoie "TRUE", l'internaute est connecté (avec 1 fichier de session + cookie), on vérifie donc si son statut est a 1. Si oui, nous renverrons TRUE pour dire "Vrai, cet internaute est connecté et est admin".  
  
   **else** sinon, c'est soit que l'internaute n'est pas connecté ou soit que l'internaute est connecté mais sans avoir les droits d'administration, nous renverrons donc "FALSE" pour dire "Faux cet internaute n'est pas administrateur".  
  
L'avantage d'avoir mis ce code dans une fonction **internauteEstConnecte()** et non pas directement dans la page web c'est qu'il sera plus facile de s'en reservir sur les autres pages web (plutôt que re-trimbaler un même morceau code en copier/coller d'un fichier à l'autre). Cela sera pratique de savoir si l'internaute est connecté à divers endroits du site web.  
  
Maintenant que nous avons 2 fonctions qui vont pouvoir nous aider, nous allons créer la page de profil :  
  
**Explications du code PHP**  
Nous verifions **if(!internauteEstConnecte())** si l'internaute (!) N'EST PAS connecté (le point d'exclamation demande si la fonction renvoie false, donc si l'internaute n'est pas connecté).  
  
Si l'internaute n'est pas connecté, il n'a rien à faire sur la page de profil, nous le renvoyons vers la page de connexion **header("location:connexion.php");**.  
  
Pour construire la page de profil, nous piochons dans le fichier session (dans le dossier /tmp/ sur le serveur, par l'intermédiaire de la superglobale $\_SESSION) afin d'afficher les informations de l'internaute connecté.

**Résultat - profil.php**  
  
  
**Etape 6. Un menu évolutif en fonction de notre statut**

Au sein de notre site web, il y a plusieurs statuts :

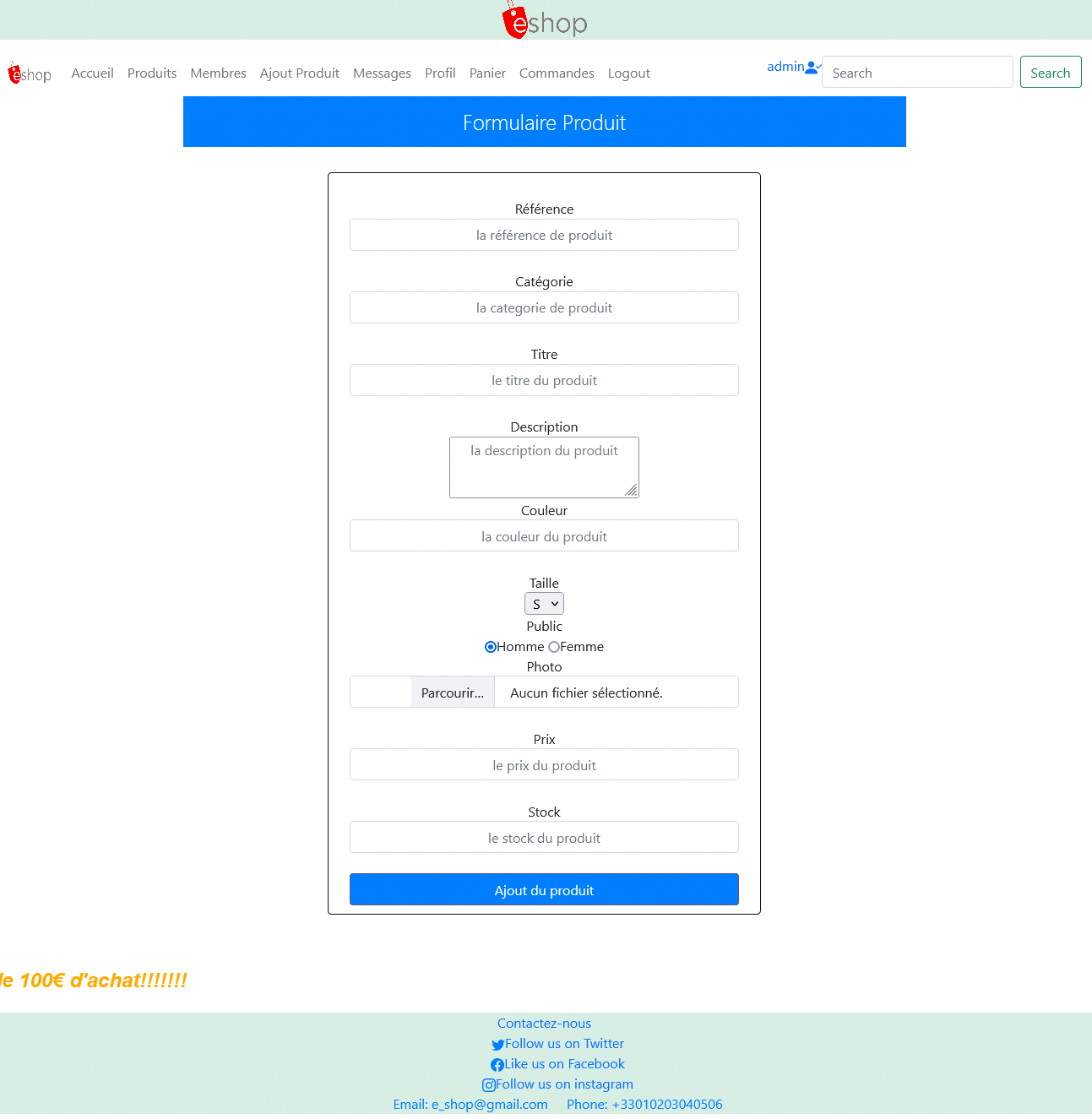
* **Visiteur** : correspond à un internaute non connecté
* **Membre** : correspond à un internaute connecté
* **Admin** : correspond à un internaute connecté avec des droits d'administration

Il est donc normal que selon son statut nous n'ayons pas accès aux mêmes droits sur le site web.  
  
  
**Explications du code PHP**  
Si l'internaute est **Administrateur** **if(internauteEstConnecteEtEstAdmin())** nous lui proposerons des liens de gestion (backOffice) pour gérer ses produits, ses commandes, ses membres, etc.  
  
Si l'internaute est **Membre** **if(internauteEstConnecte())** nous lui proposerons plusieurs liens dont son espace de profil  
  
Si l'internaute est **Visiteur** **else** nous lui proposerons d'autres liens notamment l'inscription et la connexion  
  
En conclusion :  
 1 visiteur aura donc accès à 4 liens  
 1 membre aura également 4 liens affichés  
 1 administrateur verra 7 liens sur la page web (1 administrateur est aussi 1 membre).  
  
Au passage, pour gérer la deconnexion, nous enverrons dans l'url une information ?action=deconnexion afin de pouvoir détecter que l'internaute ai bien cliqué sur le lien "se déconnecter".  
  
Il nous faut une information dans l'url que nous pourrons aller chercher plus tard avec la superglobale $\_GET afin d'associer le traitement permettant de déconnecter l'internaute.  
  
Nous gérerons la deconnexion sur la page de connexion.  
  
**Etape 6. La deconnexion et la fin de l'espace membre**

Que l'on puisse se connecter, c'est super, mais il faut aussi pouvoir se déconnecter.  
  
**Quelques Explications du code PHP**  
  
**if(isset($\_GET['action']) && $\_GET['action'] == "deconnexion")** Si l'internaute clic sur le lien deconnexion, nous arriverons sur la page connexion.php avec l'information suivante dans l'url ?action=deconnexion. C'est la raison pour laquelle nous utilisons la superglobale $\_GET afin de detecter cette action et de déconnecter l'internaute via session\_destroy();.  
  
**Etape 7. Le BackOffice - Gestion Boutique - Ajout de produits**

Le BackOffice d'un site web est une interface de gestion pour les réglages, reservée seulement à l'administateur (ou aux administrateurs).  
  
Nous avons prévu une partie (dans le dossier **/admin/**) reservée à l'administrateur pour la gestion de son site web (produits, commandes, membres).  
  
Cette partie ne doit être accessible que par un internaute connecté ayant son statut fixé à 1 (autrement dis : un administrateur).  
  
Pour cela, nous allons utiliser la fonction utilisateurEstConnecteEtEstAdmin puisque nous avions prévu un code permettant de renvoyer TRUE (vrai tu est admin) ou FALSE (faux tu n'es pas admin).  
  
Nous prévoirons également un formulaire d'ajout de produit avec récupération des données en POST et requête d'enregistrement (INSERT) dans la base de données :

Résultat :

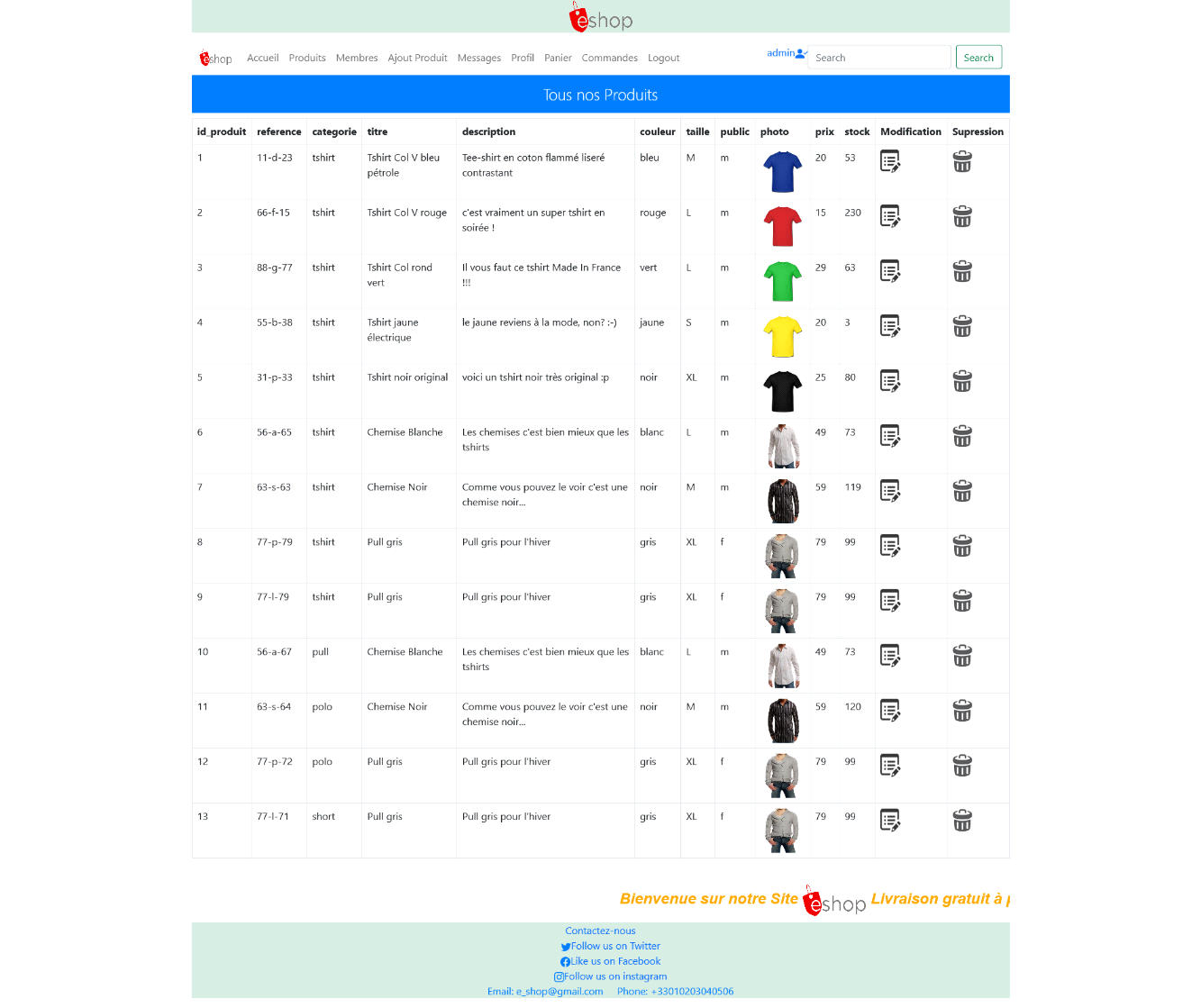


**FIGURE : Interface d’ajout produit.**

**Quelques Explications du code PHP**  
  
**Le contrôle admin**  
Nous demandons dès les premières lignes **if(!internauteEstConnecteEtEstAdmin())** si l'internaute n'est pas admin, alors nous effectuons une redirection vers la page de connexion.  
  
Nous en profitons pour mettre un exit() de manière à ce que l'interpréteur ne décode pas la suite du code et que la redirection se fasse immédiatement.  
**Le formulaire d'ajout de produit**  
La balise <form> comporte l'attribut enctype qui est indispensable pour permettre l'upload de fichier au sein de ce formulaire.  
  
Les tailles de produit ont été mises en lettre, mais cela peut être changé.  
  
  
**Le traitement PHP pour l'ajout de produit**  
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 nous 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 nous envoyer des .exe ou des fichiers d'attaques) (même si effectivement l'administrateur n'est pas censé pirater son propre site)
* *etc.*

**Le BackOffice - Gestion Boutique - Affichage de produits**

Maintenant que nous avons pu ajouter des produits, nous allons les afficher sur la page web.  
  
En effet, cela évitera d'aller dans PhpMyAdmin et surtout le commerçant pourra voir sa liste de produits en cliquant sur un lien prévu à cet effet.  
  


**Quelques Explications du code PHP**  
  
Nous avons prévue 2 liens permettant soit d'afficher les produits, soit d'ajouter 1 produit.  
  
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 detecter l'action à déclencher.  
L'administrateur peut donc choisir l'action qu'il souhaite mettre en oeuvre : ajout ou affichage.  
  
L'affichage des produits se fait dans 1 table (1 tableau).  
  
Nous ajoutons 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.

**Le BackOffice - Gestion Boutique - Suppression des produits**

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.  
  
**Quelques Explications du code PHP**  
  
Nous detectons l'action suppression dans l'url if(isset($\_GET['action']) && $\_GET['action'] == "suppression").  
  
Nous allons d'abord selectionner toutes les informations sur ce produit dans la base, 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 et nous rebasculons vers l'action d'affichage pour observer tous les produits.

**Le BackOffice - Gestion Boutique - modification des 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.  
  
Seule différence, lorsque l'on modifie il nous faudra les champs de formulaire pré-saisis avec les informations actuelles.  
  
**Quelques Explications du code PHP**

Nous demandons à ce que le formulaire produit s'affiche également en cas de modification.  
  
Nous récupérons 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).  
  
Nous utilisons l'action se trouvant dans l'url ($\_GET) pour définir le texte du bouton submit.  
  
Nous créons 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, nous récupérons 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.  
  
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  
 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).

**Résultat :**

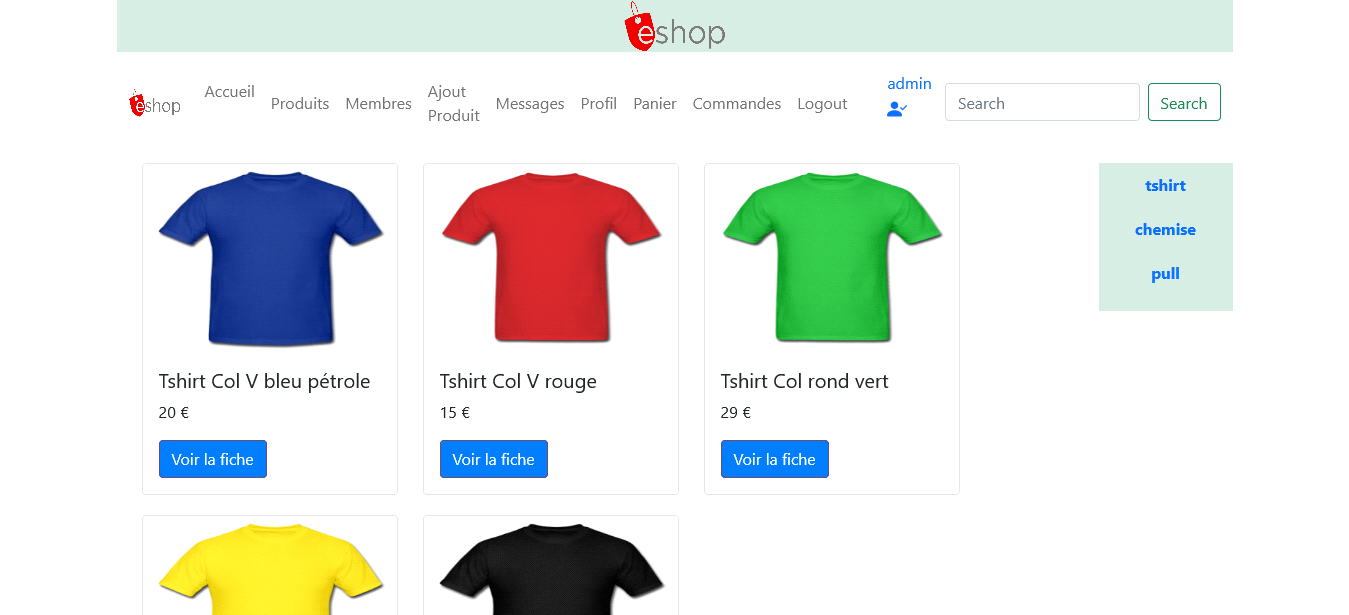


**Etape8. Le FrontOffice - La boutique**

Maintenant que nous avons des produits en boutique, il faut que l'on puisse les exposer aux internautes dans la partie FRONT.  
  
Pour créer rééllement notre boutique, nous allons travailler dans les fichier : index.php et vitrine.php.

**Quelques Explications du code PHP**  
  
Nous commençons par selectionner les catégories de produits différents (DISTINCT) afin de les afficher dans une liste de liens.  
  
Si l'internaute clique sur l'une des catégories que nous proposons, nous passerons l'information dans l'url, par exemple ?categorie=tshirt.  
  
Lorsque l'url change, la page se recharge et le code va détecter qu'une information est passée dans l'url if(isset($\_GET['categorie'])).  
  
Nous ferons donc une 2e requête SQL consistant à récupérer tous les produits relatifs à une catégorie dans la base (par exemple : tous les tshirts).  
  
Nous prévoyons également des liens sur les produits afin de proposer l'affichage d'un produit particulier dans une fiche produit dédiée.

**Résultat :**

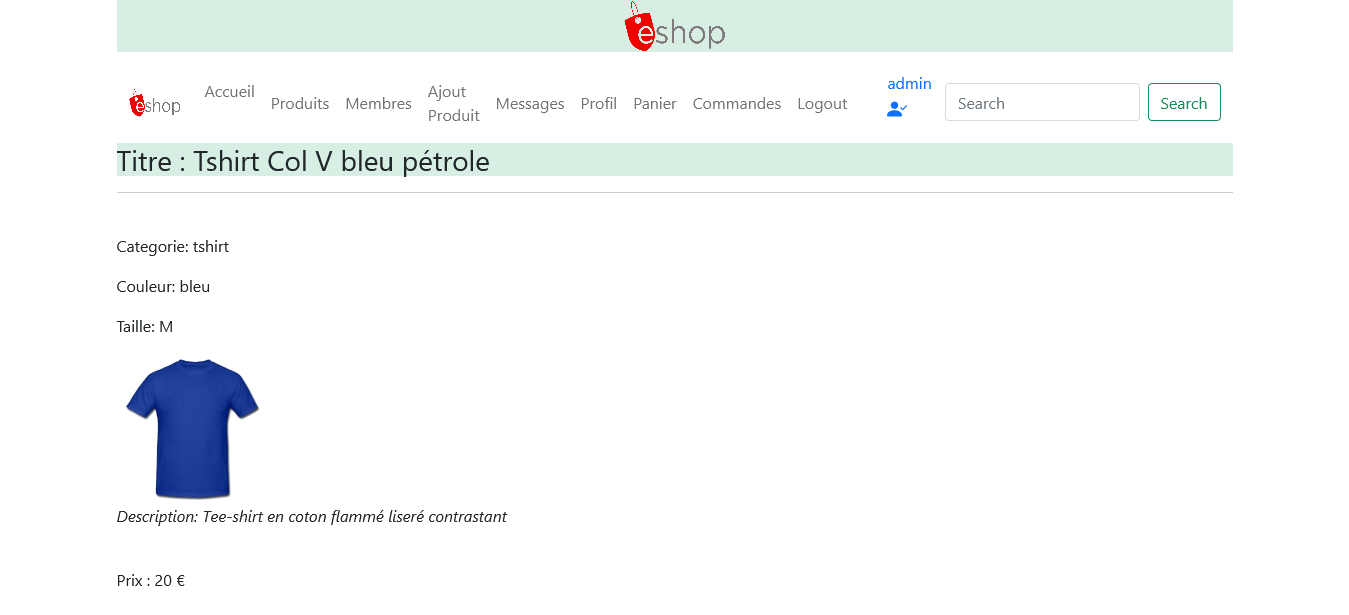


**FIGURE : Interface catalogue produits**

**Boutique - La fiche produit**

Avec la fiche produit, nous allons créer une page qui permettra de visualiser plus en détail un produit.  
**Quelques Explications du code PHP**  
  
Si l'id d'un produit se trouve dans l'url if(isset($\_GET['id\_produit'])), nous le récupérons via une requête SQL.  
  
Si nous ne récupérons pas de produit dans la base if($resultat->num\_rows <= 0), cela indique que le produit n'existe pas ou plus, nous redirigeons l'internaute vers la page boutique.  
  
Si l'interpréteur continue d'exécuter la suite des instructions, c'est que nous devons afficher le produit. Pour cela nous réalisons un traitement $produit = $resultat->fetch\_assoc(); afin de rendre le résultat exploitable sous forme de tableau ARRAY. Nous ne faisons pas de boucle dans la mesure où nous récupérons 1 seul enregistrement de la base de données.  
  
Nous affichons les informations du produit, toujours par l'intermédiaire de la variable $contenu, afin que cela n'apparaisse pas avant les premières balises type doctype, html, body, etc.  
  
Nous vérifions s'il y a du stock disponible afin de proposer la selection d'une quantité et l'ajout au panier.  
  
Si nous avons un stock disponible de 230 exemplaires pour un même produit, nous proposerons de prendre une quantité 5 par 5 (sauf si le stock est inférieur à 5, nous proposerons le stock restant).  
  
La boucle for intéroge la variable $stock et le 5 à chaque tour de boucle pour savoir si on ne le dépasse pas. cela prend donc le plus petit des deux nombres pour délimiter la boucle.  
  
En cliquant sur le bouton "ajout au panier" l'internaute sera envoyé vers la page panier.php.  
  
Si le produit n'a plus de stock disponible, nous informons l'internaute de la rupture.  
  
Pour terminer, si l'internaute ne souhaite pas acheter ce produit, nous proposons un lien permettant de le relancer dans notre boutique directement dans la categorie du produit courant.

Résultat :

  
**FIGURE : Affichage produit.**

**Etape 10. Boutique - Le panier**

**Boutique - Le panier - Ajout de produit**

Maintenant que notre boutique existe, nous allons créer une fonctionnalité panier avec une simulation de paiement.  
  
Techniquement, nous allons créer un panier dans notre fichier de session.  
Nous gérons le panier dans un fichier de session afin que les informations soient gardées de page en page durant la navigation.  
Nous ne gérons pas cela en base de données car d'une part ces informations ne sont pas destinées à rester très longtemps, et d'autre part, beaucoup de paniers sont abandonnés (il est donc préférable de ne pas polluer la base avec des données temporaires, ni alourdir le site web avec des requêtes SQL trop gourmandes).

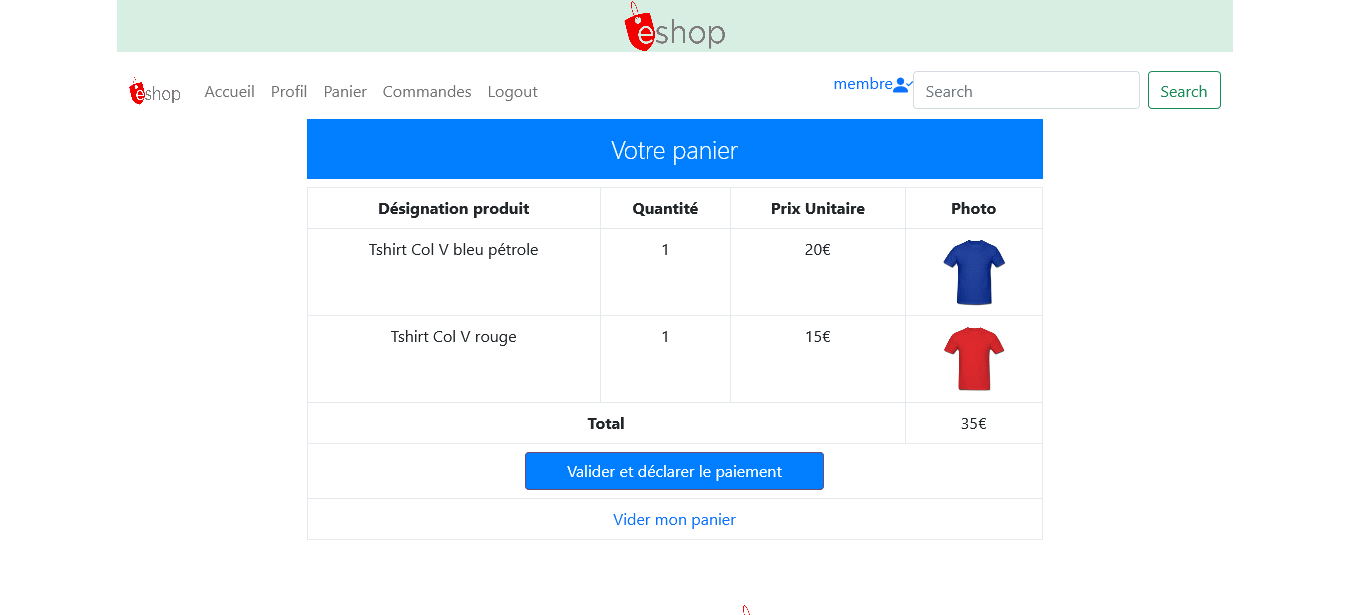
De cette manière, nous savons que le produit en position 1 dans chaque sous tableau array correspond à une chemise blanche, il s'agit de l'id\_produit n°6, avec une quantité de 1 et un prix de 49€.  
  
Avant de créer notre page panier.php, nous allons prévoir quelques fonctions afin de préparer au mieux cette fonctionnalité.  
  
**Quelques Explications du code PHP**  
  
**Creation du panier**  
  
Nous créons une première fonction creationDuPanier() afin de créer l'espace nécessaire dans le fichier de session pour accueillir les données des futurs produits.  
Nous vérifierons si la session existe déjà, et seulement dans le cas ou elle n'existe pas nous la créerons : if(!isset($\_SESSION['panier']))  
  
**Ajout d'un produit dans le panier**  
  
Nous exécutons d'abord creationDuPanier() afin que l'espace permettant d'accueillir les données produits soit créé.  
  
Notre fonction ajouterProduitDansPanier() est destinée à recevoir 4 arguments : le titre, l'id du produit, la quantité desirée et le prix *(ce sont les informations que nous garderons sur un produit dans notre panier).*  
  
Avec la fonction prédéfinie array\_search, nous allons d'abord procéder à une vérification. Avant de l'ajouter, est-ce que le produit est déjà présent dans le panier ?  
Par exemple, cela pourrait arriver si l'internaute ajoute le produit 2 (tshirt rouge) dans le panier, qu'il repart de la page panier, qu'il rerentre dans la fiche produit 2 et qu'il ajoute à nouveau un autre tshirt rouge (donc toujours le produit 2).  
Pour éviter d'avoir 2 lignes de produits différentes dans notre panier (comme s'il s'agissait de 2 produits différents), nous mettrons à jour uniquement la quantité du produit en question (déjà présente dans le panier).  
La fonction prédéfinie array\_search est pratique car elle nous renvoie précisément la position du produit dans le tableau ARRAY et nous savons donc à quel endroit intervenir pour adapter la quantité.  
Le symbole += permet de ne pas perdre la quantité précédente. S'il y avait 1 en quantité, et que l'internaute ajoute 2 autres tshirt rouges (toujours le produit n°2) et bien nous ne voulons pas remplacer le chiffre 1 par 2 mais faire l'opération suivante : 1 + 2 = 3.  
  
Dans le cas contraire, si l'id du produit n'est pas connue dans le panier, nous l'ajouterons normalement dans notre fichier session.  
  
**Montant Total**  
  
La fonction montantTotal() va nous permettre de multiplier chaque prix par chaque quantité et donc de faire la somme des prix contenus dans le panier.  
for($i = 0; $i < count($\_SESSION['panier']['id\_produit']); $i++) tant que $i est inférieur au nombre de produits contenus dans la session panier, nous allons parcourir les éléments.  
  
$total += $\_SESSION['panier']['quantite'][$i] \* $\_SESSION['panier']['prix'][$i]; on multiplie la quantite par le prix; ex 1\*10€ ou 3\*10€ sans remplacer pour autant la derniere valeur contenue dans la variable $total (+=). Addition et multiplication.  
  
return round($total,2); nous retournons le prix total pour tous les produits (avec round nous demandons à avoir 2 chiffres après la virgule)  
**RetirerProduitDuPanier**  
  
Cette fonction permet de retirer un produit du panier.  
La fonction prédéfinie Array\_Search retourne un chiffre afin de savoir à quel indice se trouve le produit à supprimer du panier. Lorsqu'un produit est retiré du panier, son indice dans les sous tableaux array titre, id\_produit, quantite et prix vont être vidés.  
Pour éviter de garder des indices vides dans nos tableaux ARRAY, nous nous servons de la fonction prédéfinie array\_splice pour faire glisser les produits des indices supérieurs vers des indices numériques du tableau inférieur.

**Quelques Explications du code PHP**  
  
Si le POST a été réalisé if(isset($\_POST['ajout\_panier'])), les informations proviennent de la page fiche\_produit.php.  
  
Nous réalisons une requête SQL pour récupérer des informations sur le produit.  
  
Nous nous servons de la fonction ajouterProduitDansPanier qui entraine l'exécution de la fonction creationDuPanier.  
  
Le prix et le titre viennent de la base de données par l'intermédiaire de la variable $produit, tandis que l'id\_produit et la quantite proviennent de la fiche\_produit.php.  
  
Un peu plus bas dans le code, nous regardons si le produit est vide ou non et adressons un message à l'internaute.

**Boutique - Le panier - Affichage des produits et enregistrement de la commande**

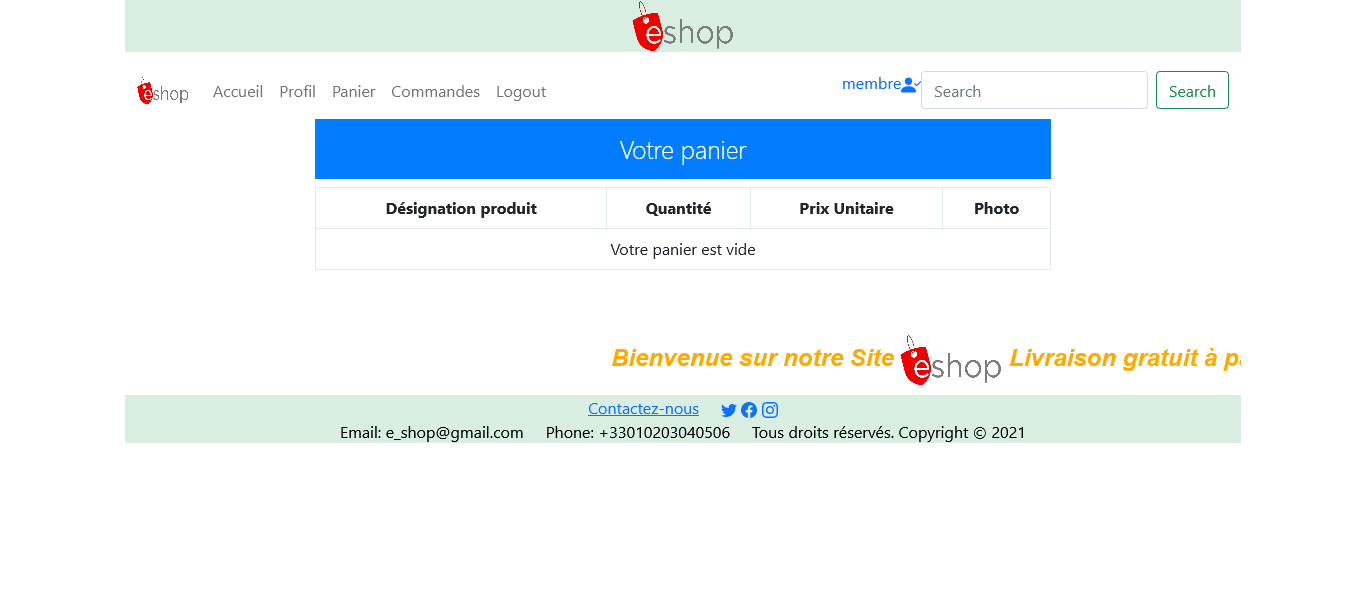
Si nous avons réussi à rentrer des produits dans notre panier, nous allons maintenant les afficher afin que l'internaute puisse les observer sur la page web.  
**Quelques Explications du code PHP**  
  
Nous faisons plusieurs tours avec la boucle for afin de parcourir tous les produits présents dans notre panier.  
  
L'affichage se fait par l'intermédiaire de la superglobale $\_SESSION (puisque toutes les informations sont enregistrées dans le fichier de session).

Résultat :

**FIGURE : Interface produits dans panier.**

Nous prévoyons un formulaire permettant de déclarer le paiement avec un bouton submit "Valider et déclarer le paiement".  
1 autre lien est disponible pour vider l'ensemble du panier.

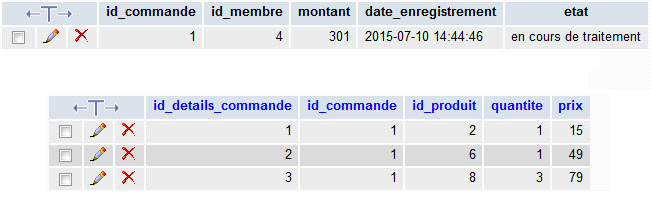
Résultat :

**Quelques Explications du code PHP**  
  
Pour tous les produits dans le panier, nous allons observer si les quantités demandées sont encore en stock.  
Pour cela, nous devons formuler une requête SQL qui va chercher en base les informations sur 1 produit, nous devons faire ça pour tous les produits c'est la raison pour laquelle la requête SQL se trouve dans une boucle afin d'entamer une répétition sur le traitement.  
  
Voici 3 situations qui peuvent arriver :

* Si la quantité demandée est couverte par le stock, il n'y a pas de soucis.
* Si la quantité demandée est inférieure au stock mais qu'il reste du stock, nous diminuons la quantité demandée par le stock restant (afin de vendre les derniers exemplaires).  
  l'internaute demande 6 tshirts, il en reste 2. Si le stock est supérieur à zéro, on lui remplace le 6 par un 2.
* Si la quantité demandée est inférieure au stock et qu'il n'y a plus de stock, nous retirons totalement le produit du panier.  
  sinon, nous sommes en rupture de stock, on lui retire carrèment le produit.

Nous aurions pu faire une boucle à l'envers au niveau du code $i--;. Nous avons choisi d'ajouter un retour dans la boucle, voici l'explication :  
si le produit en indice zéro pose problème pour cause de rupture de stock, il est retiré.  
Du coup le produit en indice 1 passe en indice 0 (réordonner via array splice de la fonction retirerproduitDuPanier) et ne sera pas vérifié (car le tour de boucle 0 sera passé, nous serons à 1).  
Il faut donc que je reparcours l'indice 0 pour me préoccuper du produit qui vient de se décaler à l'indice 0 et qui n'est pas encore testé.  
Ceci explique la présence du code $i--;.  
  
Quoi qu'il en soit (stock réduit ou rupture totale de stock), s'il y a eu un souci dans le traitement, nous mettrons la variable $erreur à TRUE $erreur = true.

Pour traiter et enregistrer la commande, la variable $erreur ne doit pas exister if(!isset($erreur)) (sinon cela voudrait dire que nous sommes passés dans le stock réduit ou dans la rupture de stock totale).  
  
Nous enregistrons la commande dans la table commande : l'id du membre est récupéré dans le fichier de session (membre actuellement connecté), le prix est calculé par notre fonction montantTotal(), la date d'enregistrement est générée via la fonction SQL NOW()  
  
Pour une commande il peut y avoir plusieurs produits et donc plusieurs lignes d'informations (dans le cas où il s'agit d'une commande avec plusieurs produits différents).  
Nous récupérons donc l'id de la commande (qui a été générée en AUTO\_CTRLREMENT) : $id\_commande = $mysqli->insert\_id;  
Il y a 1 seule commande mais plusieurs détails de commande à enregistrer, nous refaisons donc une boucle pour tous les enregistrer en les reliant au numéro de commande principale.  
  
Une fois que l'internaute à déclaré le paiement, nous vidons son panier car tous les produits sont censés être payés !  
  
Nous envoyons un email de confirmation au membre qui vient de réaliser la commande.  
  
Au sein de ce fichier, nous mélangeons à la fois des instructions d'affichage echo et la variable $contenu.  
La variable $contenu correspond à des affichages que nous devons faire lors des traitements mais que l'on doit retarder pour garder une structure viable.

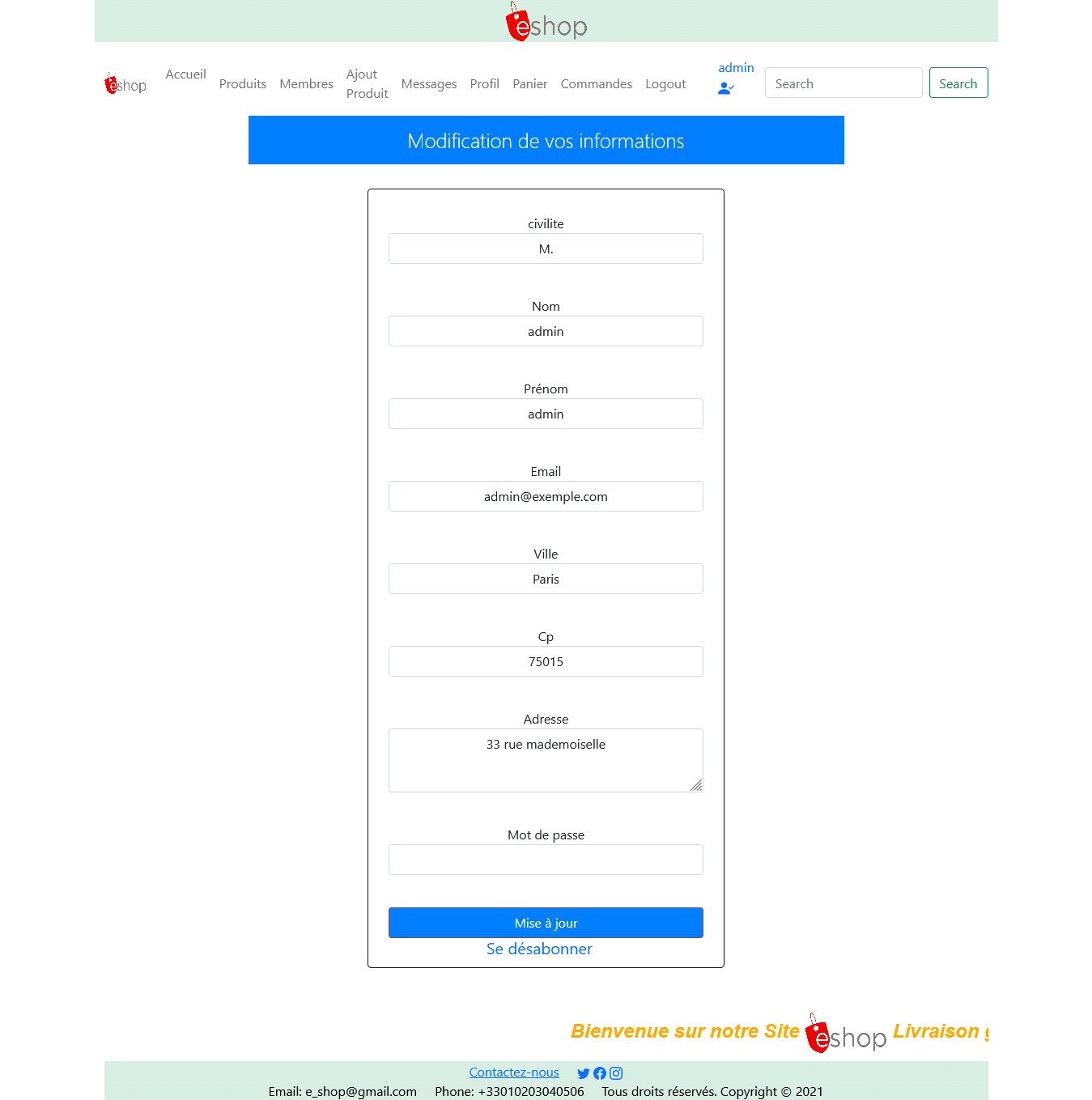
**Résultat**  
  
  
*il y a bien 1 commande principale et 3 produits rattachés à cette commande*.

**III.Aperçu du travail réalisé :**

**III.1.*Interface Modification informations Administrateur***

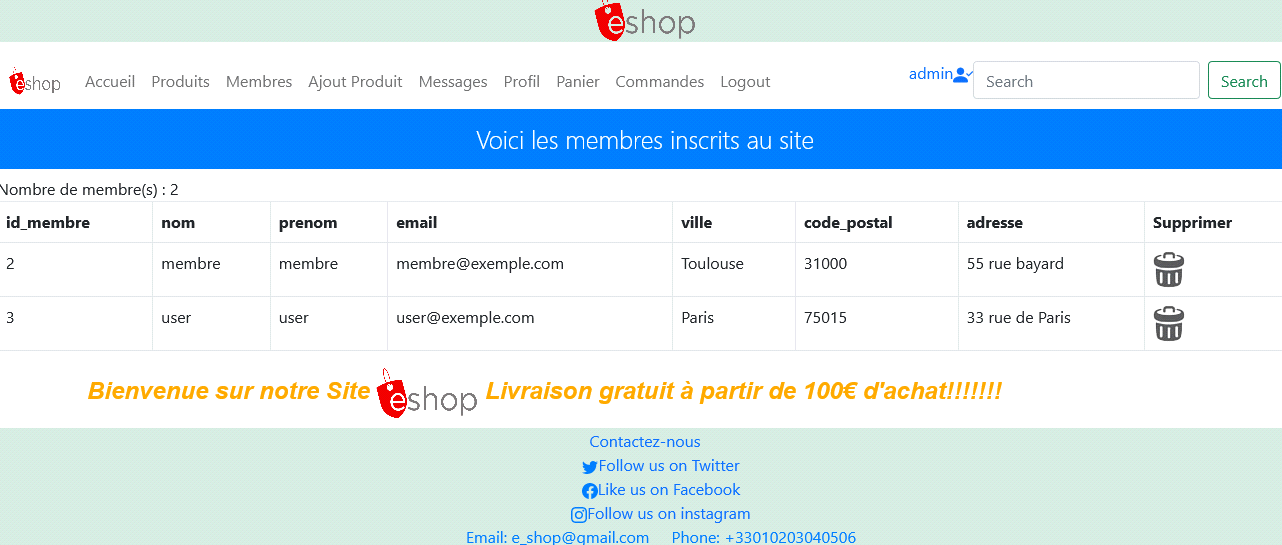
J’ai essayée dans cette partie de présenter les interfaces les plus significatives de notre application boutique en ligne.

L’administrateur à la possibilité de gérer son compte, il doit pouvoir changer ses informations.

***FIGURE : Interface Modification informations Administrateur***

**III.1.4. Liste des clients :**

L’administrateur peut afficher la liste de clients. Il peut supprimer un client dans cette liste.



***FIGURE : Interface de liste des clients.***

**III.1.5. Les news :**

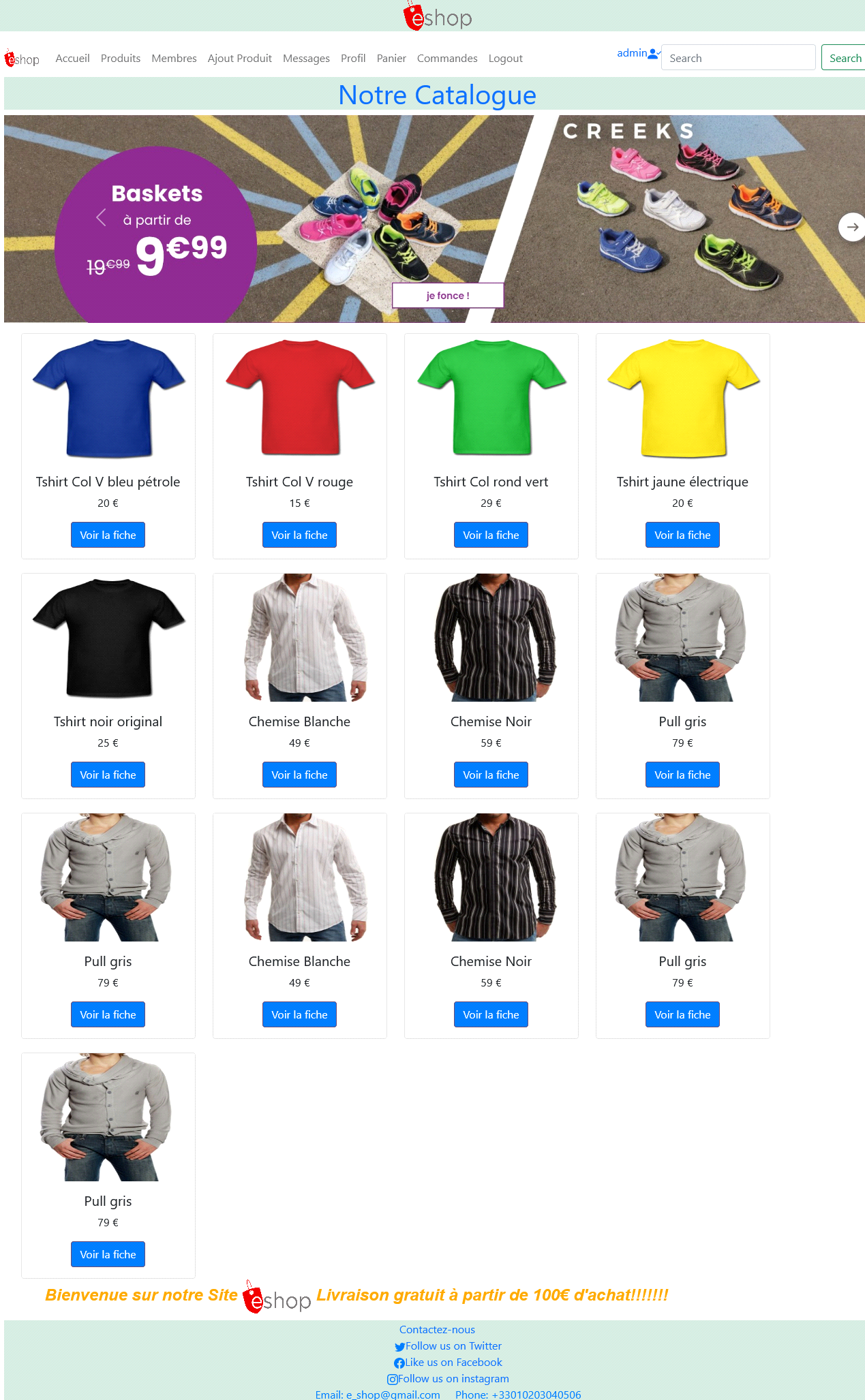
L’administrateur a la possibilité de modifier la page d’accueil ou page de présentation.

***FIGURE  : Interface de news.***

**III.2. Les interface de l’application :**

**III.2.1. Interface d’accueil**

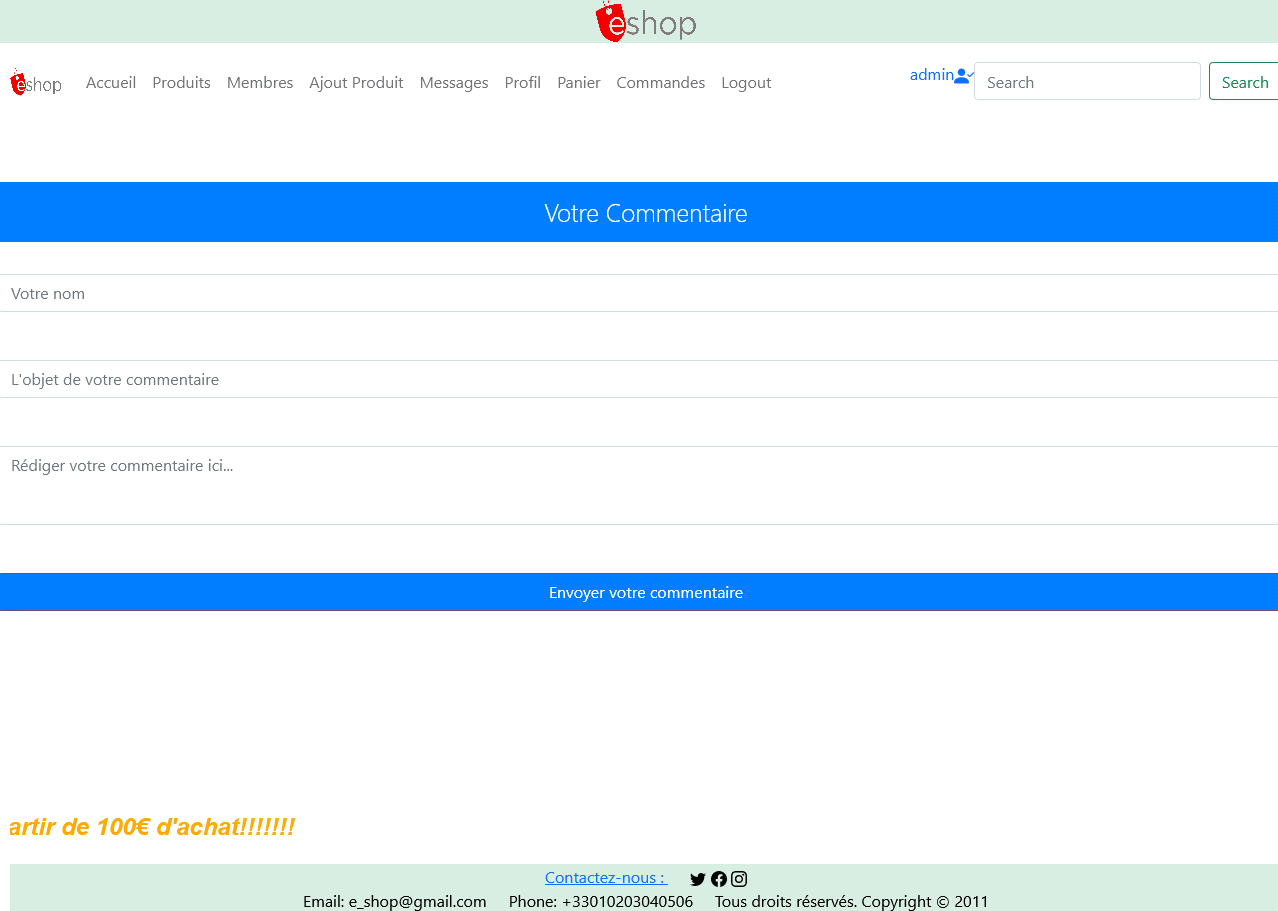
Cette interface possède toutes les informations nécessaires du site.



***FIGURE : Page d’accueil.***

**III.2.3. Interface contact :**

Dans cette partie le visiteur à la possibilité de contacter est envoie un message à l’administrateur.

***FIGURE  : Page contact.***

**Conclusion :**

Dans ce chapitre, j’ai cité la description des environnements matériels et logiciels de mon application web avec les différents outils de développement, et je donne un aperçu sur le travail réalisé.

**Conclusion générale**

Ce projet de fin d'étude a pour objectif de réaliser une application web représentant une boutique en ligne de vente d’articles sportifs.

J’ai commencé mon rapport par une présentation générale assez approfondie sur la boutique en ligne, et l’étude des solutions existantes, ce qui a été le thème du deuxième chapitre de ce rapport.

J’ai consacré la partie suivante à la spécification des besoins du client pour bien concevoir les fonctionnalités attendues de notre application. Cette étude a été couronnée par une conception détaillée de la solution faisant sujet du quatrième chapitre et se basant sur les diagrammes UML, est on passe à la réalisation de la boutique en ligne qui est le dernier chapitre.

Ce projet a été pour moi une expérience intéressante et riche, qui m’a permis d'améliorer mes connaissances du langage PHP et la création d’une application web e-commerce.

Ainsi à travers cette application j’ai pu offrir à cette structure une chance de s'imposer et d'entrer de plein pied dans le monde compétitif du E-Commerce pour matérialiser sa consultation par des milliers d'internaute et donc augmenter le nombre de ses clients partout dans le monde.

**Perspectives :**

- Utiliser un Framework come codeIgniter

- Mettre en place un vrai paiement en ligne

- Retravailler le DESIGN du site

***Bibliographie***

[1] : <https://www.php.net/manual/fr/index.php>

[2] : <https://developer.mozilla.org/fr/docs/Web/HTML>

[3] : <https://www.w3.org/Style/CSS/>

[4] :[www.wikipedia.fr](http://www.wikipedia.fr/)

[5] : <https://app.diagrams.net/>

[6] : <https://trello.com/>

[7] : <https://www.looping-mcd.fr/>

[8] : <https://www.figma.com/>

[8] : [https://www.lahalleauxvetements.com/](https://www.figma.com/)

[8] : <https://www.kiabi.com/>

**Glossaire**

**PHP : Hypertext Preprocessor**

**HTML** : HyperText Mark up Langages,

**Http :** HyperText Transfer Protocol

**SQL** : Structured Query Language (le langage standard pour les traitements de bases de donnée)

**CMS :** Content Management Système (système de gestion de contenu) c’est un site web modifiable disposant de fonctionnalités de publication.

**SCRUM** : framework lié aux méthodes agiles de gestion de projet, utilisées notamment en développement logiciel ;

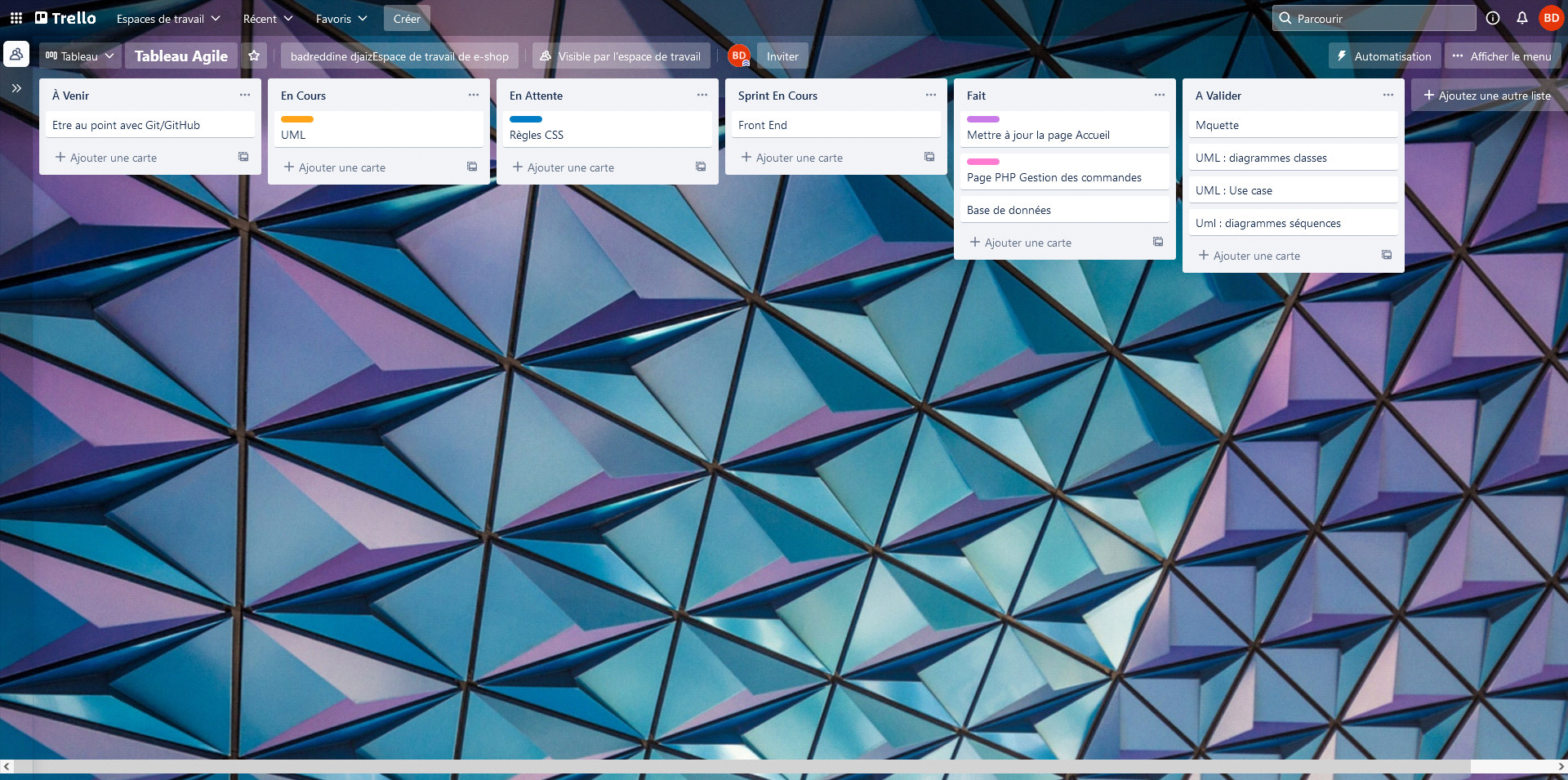
**Trello :** Un outil de gestion de projet en ligne. Il repose sur une organisation des projets en planches listant des cartes, chacune représentant des tâches.

**SCRUM :** Scrum est un framework ou cadre de développement de produits complexes. Cadre de travail holistique itératif qui se concentre sur les buts communs en livrant de manière productive et créative des produits de la plus grande valeur possible.

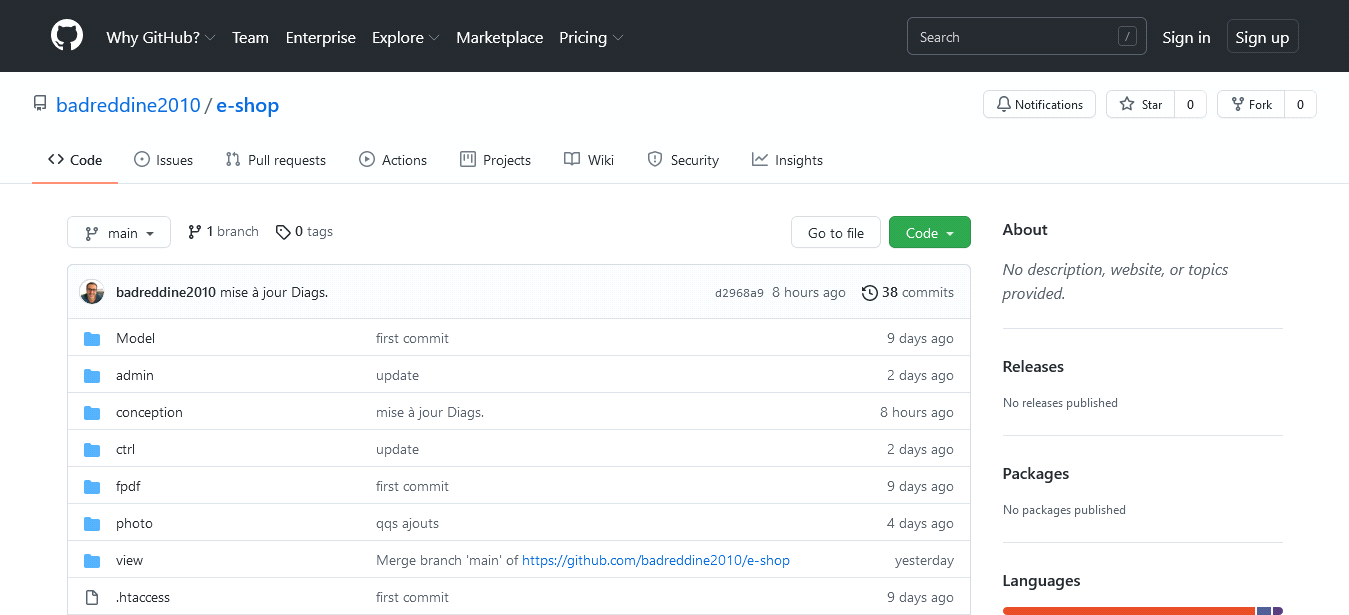
**GitHub** : exploité sous le nom de GitHub, Inc.) est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Ce site est développé en Ruby on Rails et Erlang par Chris Wanstrath, PJ Hyett et Tom Preston-Werner. GitHub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de logiciels libres. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches.

**Annexes**

**TRELLO**



**GitHub**



***La table membre***

|  |
| --- |
| ***CREATEDATABASEdb\_e\_shop ;***  ***USE db\_e\_shop ;***  ***CREATETABLEmembre (***  ***id\_membre INT(3) NOTNULLAUTO\_INCREMENT PRIMARYKEY,***  ***pseudo VARCHAR(20) NOTNULL,***  ***mdp VARCHAR(32) NOTNULL,***  ***nom VARCHAR(20) NOTNULL,***  ***prenom VARCHAR(20) NOTNULL,***  ***email VARCHAR(50) NOTNULL,***  ***civilite ENUM('m', 'f') NOTNULL,***  ***ville VARCHAR(20) NOTNULL,***  ***code\_postal INT(5) UNSIGNED ZEROFILL NOTNULL,***  ***adresse VARCHAR(50) NOTNULL,***  ***statut INT(1) NOTNULLDEFAULT0,***  ***UNIQUE(pseudo)***  ***) ENGINE = InnoDB;*** |

**La table produit**

|  |
| --- |
| ***CREATETABLEproduit (***  ***id\_produit INT(3) NOTNULLAUTO\_INCREMENT PRIMARYKEY,***  ***reference VARCHAR(20) NOTNULL,***  ***categorie VARCHAR(20) NOTNULL,***  ***titre VARCHAR(100) NOTNULL,***  ***description TEXT NOTNULL,***  ***couleur VARCHAR(20) NOTNULL,***  ***taille VARCHAR(5) NOTNULL,***  ***publicENUM('m', 'f', 'mixte') NOTNULL,***  ***photo VARCHAR(250) NOTNULL,***  ***prix INT(3) NOTNULL,***  ***stock INT(3) NOTNULL,***  ***UNIQUE(reference)***  ***) ENGINE = InnoDB;*** |

**La table commande**

|  |
| --- |
| ***CREATETABLEcommande (***  ***id\_commande INT(3) NOTNULLAUTO\_INCREMENT PRIMARYKEY,***  ***id\_membre INT(3) NULLDEFAULTNULL,***  ***montant INT(3) NOTNULL,***  ***date\_enregistrement DATETIME NOTNULL,***  ***etat ENUM('en cours de traitement', 'envoyé', 'livré') NOTNULL***  ***) ENGINE = InnoDB;*** |

**La table details\_commande**

|  |
| --- |
| ***CREATETABLEdetails\_commande (***  ***id\_details\_commande INT(3) NOTNULLAUTO\_INCREMENT PRIMARYKEY,***  ***id\_commande INT(3) NULLDEFAULTNULL,***  ***id\_produit INT(3) NULLDEFAULTNULL,***  ***quantite INT(3) NOTNULL,***  ***prix INT(3) NOTNULL***  ***) ENGINE = InnoDB;*** |

**Code PHP : init.ctrl.php**

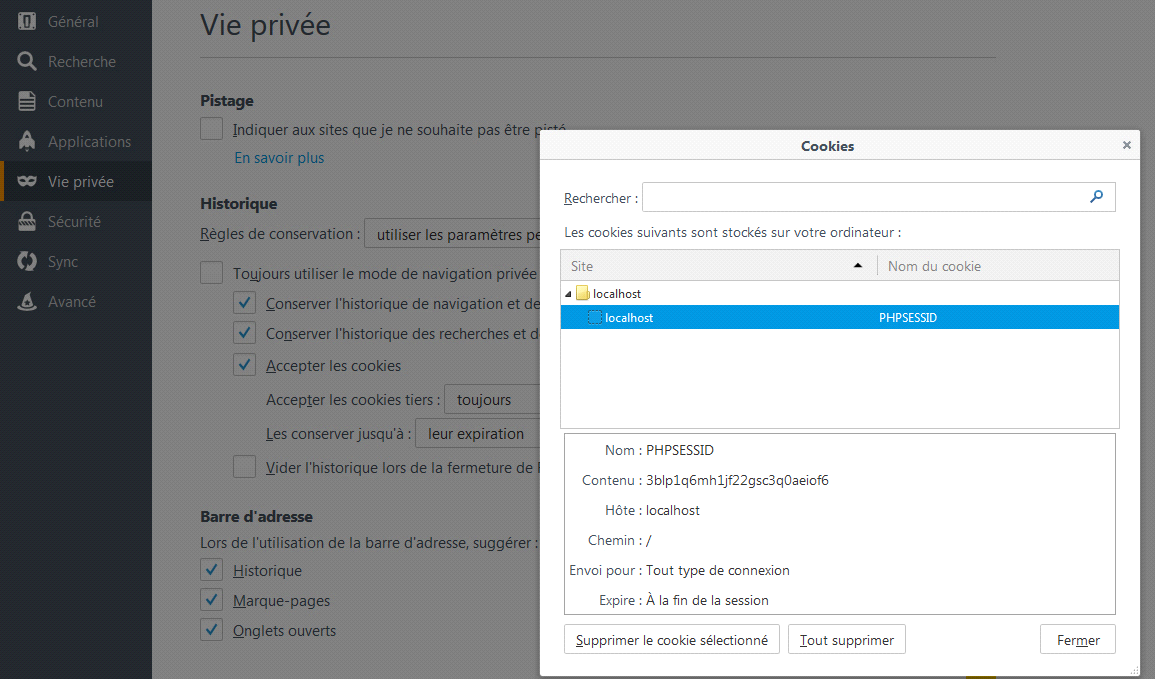
|  |
| --- |
| ***<?php***  ***//--------- BDD***  ***$mysqli= newmysqli("localhost", "root", "", "site");***  ***if($mysqli->connect\_error) die('Un problème est survenu lors de la tentative de connexion à la BDD : '. $mysqli->connect\_error);***  ***// $mysqli->set\_charset("utf8");***  ***//--------- SESSION***  ***session\_start();***  ***//--------- CHEMIN***  ***define("RACINE\_SITE","/http://localhost/e-shop/");***  ***//--------- VARIABLES***  ***$contenu= '';***  ***//--------- AUTRES INCLUSIONS***  ***require\_once("fonction.ctrl.php");*** |

**Code PHP : footer.php**

|  |
| --- |
| ***</div>***  ***</section>***  ***<footer>***  ***<divclass="conteneur">***  ***<?phpechodate('Y'); ?>- Tous droits reservés - MonNom MonPrenom.***  ***</div>***  ***</footer>***  ***</body>***  ***</html>*** |

**Code PHP : inscription.php**

|  |
| --- |
| ***<?phprequire\_once("ctrl/init.inc.php");***  ***//--------------------------------- TRAITEMENTS PHP ---------------------------------//***  ***if($\_POST)***  ***{debug($\_POST);***  ***$verif\_caractere= preg\_match('#^[a-zA-Z0-9.\_-]+$#', $\_POST['pseudo']);***  ***if(!$verif\_caractere&&(strlen($\_POST['pseudo']) <1 || strlen($\_POST['pseudo']) >20) ) //***  ***{$contenu.= "<div class='erreur'>Le pseudo doit contenir entre 1 et 20 caractères. <br>Caractère accepté :***  ***Lettre de A à Z et chiffre de 0 à 9</div>";***  ***}else***  ***{ $membre= executeRequete("SELECT \* FROM membre WHERE pseudo='$\_POST[pseudo]'");***  ***if($membre->num\_rows >0)***  ***{   $contenu.= "<div class='erreur'>Pseudo indisponible. Veuillez en choisir un autre svp.</div>";***  ***}  else***  ***{        // $\_POST['mdp'] = md5($\_POST['mdp']);***  ***foreach($\_POSTas$indice=>$valeur)***  ***{     $\_POST[$indice] = htmlEntities(addSlashes($valeur));***  ***}       executeRequete("INSERT INTO membre (pseudo, mdp, nom, prenom, email, civilite, ville, code\_postal, adresse)***  ***VALUES ('$\_POST[pseudo]', '$\_POST[mdp]', '$\_POST[nom]', '$\_POST[prenom]', '$\_POST[email]', '$\_POST[civilite]', '$\_POST[ville]',***  ***'$\_POST[code\_postal]', '$\_POST[adresse]')");***  ***$contenu.= "<div class='validation'>Vous êtes inscrit à notre site web. <a href=\"connexion.php\"><u>***  ***Cliquez ici pour vous connecter</u></a></div>";***  ***}   }}***  ***//--------------------------------- AFFICHAGE HTML ---------------------------------//***  ***?>***  ***<?phprequire\_once("inc/haut.inc.php"); ?>***  ***<?phpecho$contenu; ?>***  ***<formmethod="post"action="">***  ***<labelfor="pseudo">Pseudo</label><br>***  ***<inputtype="text"id="pseudo"name="pseudo"maxlength="20"placeholder="votre pseudo"pattern="[a-zA-Z0-9-\_.]{1,20}"***  ***title="caractères acceptés : a-zA-Z0-9-\_."required="required"><br><br>***  ***<labelfor="mdp">Mot de passe</label><br>***  ***<inputtype="password"id="mdp"name="mdp"required="required"><br><br>***  ***<labelfor="nom">Nom</label><br>***  ***<inputtype="text"id="nom"name="nom"placeholder="votre nom"><br><br>***  ***<labelfor="prenom">Prénom</label><br>***  ***<inputtype="text"id="prenom"name="prenom"placeholder="votre prénom"><br><br>***  ***<labelfor="email">Email</label><br>***  ***<inputtype="email"id="email"name="email"placeholder="exemple@gmail.com"><br><br>***  ***<labelfor="civilite">Civilité</label><br>***  ***<inputname="civilite"value="m"checked=""type="radio">Homme***  ***<inputname="civilite"value="f"type="radio">Femme<br><br>***  ***<labelfor="ville">Ville</label><br>***  ***<inputtype="text"id="ville"name="ville"placeholder="votre ville"pattern="[a-zA-Z0-9-\_.]{5,15}"title="caractères acceptés : a-zA-Z0-9-\_.">***  ***<br><br>***  ***<labelfor="cp">Code Postal</label><br>***  ***<inputtype="text"id="code\_postal"name="code\_postal"placeholder="code postal"pattern="[0-9]{5}"title="5 chiffres requis : 0-9"><br><br>***  ***<labelfor="adresse">Adresse</label><br>***  ***<textareaid="adresse"name="adresse"placeholder="votre dresse"pattern="[a-zA-Z0-9-\_.]{5,15}"title="caractères acceptés :  a-zA-Z0-9-\_.">***  ***</textarea><br><br>***  ***<inputtype="submit"name="inscription"value="S'inscrire">***  ***</form>***  ***<?phprequire\_once("ctrl/bas.ctrl.php"); ?>*** |

**Cookie sur l'ordinateur de l'internaute :  
  
  
Code à insérer : fonction.ctrl.php**

|  |
| --- |
| ***functionexecuteRequete($req)***  ***{ global$mysqli;***  ***$resultat= $mysqli->query($req);***  ***if(!$resultat)***  ***{ die("Erreur sur la requete sql.<br>Message : ". $mysqli->error . "<br>Code: ". $req);***  ***}  return$resultat;***  ***}functiondebug($var, $mode= 1)***  ***{ echo'<div style="background: orange; padding: 5px; float: right; clear: both; ">'***  ***$trace= debug\_backtrace();***  ***$trace= array\_shift($trace);***  ***echo'Debug demandé dans le fichier : $trace[file] à la ligne $trace[line].';***  ***if($mode=== 1)***  ***{print'<pre>'; print\_r($var); print'</pre>'; }***  ***else***  ***{ print'<pre>'; var\_dump($var); print'</pre>'; }***  ***echo'</div>';}***  ***//------------------------------------***  ***functioninternauteEstConnecte()***  ***{   if(!isset($\_SESSION['membre'])) returnfalse;***  ***elsereturntrue;}***  ***//------------------------------------***  ***functioninternauteEstConnecteEtEstAdmin()***  ***{if(internauteEstConnecte() &&$\_SESSION['membre']['statut'] == 1) returntrue;***  ***elsereturnfalse;}*** |

**Code PHP : profil.php**

|  |
| --- |
| ***<?php***  ***require\_once("ctrl/init.ctrl.php");***  ***//--------------------------------- TRAITEMENTS PHP ---------------------------------//***  ***if(!internauteEstConnecte()) header("location:connexion.php");***  ***// debug($\_SESSION);***  ***$contenu.= '<p class="centre">Bonjour <strong>'. $\_SESSION['membre']['pseudo'] . '</strong></p>';***  ***$contenu.= '<div class="cadre"><h2>Voici vos informations </h2>';***  ***$contenu.= '<p>votre email est: '. $\_SESSION['membre']['email'] . '<br>';***  ***$contenu.= 'votre ville est: '. $\_SESSION['membre']['ville'] . '<br>';***  ***$contenu.= 'votre cp est: '. $\_SESSION['membre']['code\_postal'] . '<br>';***  ***$contenu.= 'votre adresse est: '. $\_SESSION['membre']['adresse'] . '</p></div><br><br>';***  ***//--------------------------------- AFFICHAGE HTML ---------------------------------//***  ***require\_once("ctrl/haut.ctrl.php");***  ***echo$contenu;***  ***require\_once("ctrl/bas.ctrl.php");*** |

**Code PHP : header.php**

|  |
| --- |
| ***<!Doctype html>***  ***<html>***  ***<head>***  ***<title>Mon Site</title>***  ***<linkrel="stylesheet"href="<?php echoRACINE\_SITE; ?>ctrl/css/style.css">***  ***</head>***  ***<body>***  ***<header>***  ***<divclass="conteneur">***  ***<div>***  ***<ahref=""title="Mon Site">MonSite.com</a>***  ***</div>***  ***<nav>***  ***<?php***  ***if(internauteEstConnecteEtEstAdmin())***  ***{                     echo'<a href="'. RACINE\_SITE . 'admin/gestion\_membre.php">Gestion des membres</a>';***  ***echo'<a href="'. RACINE\_SITE . 'admin/gestion\_commande.php">Gestion des commandes</a>';***  ***echo'<a href="'. RACINE\_SITE . 'admin/gestion\_boutique.php">Gestion de la boutique</a>';***  ***}             if(internauteEstConnecte())***  ***{                   echo'<a href="'. RACINE\_SITE . 'profil.php">Voir votre profil</a>';***  ***echo'<a href="'. RACINE\_SITE . 'boutique.php">Accès à la boutique</a>';***  ***echo'<a href="'. RACINE\_SITE . 'panier.php">Voir votre panier</a>';***  ***echo'<a href="'. RACINE\_SITE . 'connexion.php?action=deconnexion">Se déconnecter</a>';***  ***}         else***  ***{                    echo'<a href="'. RACINE\_SITE . 'inscription.php">Inscription</a>';***  ***echo'<a href="'. RACINE\_SITE . 'connexion.php">Connexion</a>';***  ***echo'<a href="'. RACINE\_SITE . 'boutique.php">Accès à la boutique</a>';***  ***echo'<a href="'. RACINE\_SITE . 'panier.php">Voir votre panier</a>';***  ***}             ?>***  ***</nav>***  ***</div>***  ***</header>***  ***<section>***  ***<divclass="conteneur">*** |

**Code PHP : connexion.php**

|  |
| --- |
| ***<?phprequire\_once("ctrl/init.ctrl.php");***  ***//--------------------------------- TRAITEMENTS PHP ---------------------------------//***  ***if(isset($\_GET['action']) &&$\_GET['action'] == "deconnexion")***  ***{session\_destroy();***  ***}if(internauteEstConnecte())***  ***{header("location:profil.php");***  ***}if($\_POST)***  ***{// $contenu .=  "pseudo : ". $\_POST['pseudo'] . "<br>mdp : ".  $\_POST['mdp'] . "";***  ***$resultat= executeRequete("SELECT \* FROM membre WHERE pseudo='$\_POST[pseudo]'");***  ***if($resultat->num\_rows != 0)***  ***{     // $contenu .=  '<div style="background:green">pseudo connu!</div>';***  ***$membre= $resultat->fetch\_assoc();***  ***if($membre['mdp'] == $\_POST['mdp'])***  ***{ //$contenu .= '<div class="validation">mdp connu!</div>';***  ***foreach($membreas$indice=>$element)***  ***{ if($indice!= 'mdp')***  ***{$\_SESSION['membre'][$indice] = $element;***  ***}}header("location:profil.php");***  ***}Else***  ***{$contenu.= '<div class="erreur">Erreur de MDP</div>';***  ***}       } else***  ***{$contenu.= '<div class="erreur">Erreur de pseudo</div>';***  ***}}***  ***//--------------------------------- AFFICHAGE HTML ---------------------------------//***  ***?>***  ***<?phprequire\_once("ctrl/haut.ctrl.php"); ?>***  ***<?phpecho$contenu; ?>***  ***<formmethod="post"action="">***  ***<labelfor="pseudo">Pseudo</label><br>***  ***<inputtype="text"id="pseudo"name="pseudo"><br><br>***  ***<labelfor="mdp">Mot de passe</label><br>***  ***<inputtype="text"id="mdp"name="mdp"><br><br>***  ***<inputtype="submit"value="Se connecter">***  ***</form>***  ***<?phprequire\_once("ctrl/bas.ctrl.php"); ?>*** |

**Code PHP : gestion\_boutique.php**

|  |
| --- |
| ***<?php***  ***require\_once("../ctrl/init.ctrl.php");***  ***//--------------------------------- TRAITEMENTS PHP ---------------------------------//***  ***//--- VERIFICATION ADMIN ---//***  ***if(!internauteEstConnecteEtEstAdmin())***  ***{header("location:../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($\_POSTas$indice=>$valeur)***  ***{$\_POST[$indice] = htmlEntities(addSlashes($valeur));***  ***} executeRequete("REPLACE INTO produit (id\_produit, reference, categorie, titre, description, couleur, taille, public, photo, prix, stock)***  ***values ('$\_POST[id\_produit]', '$\_POST[reference]', '$\_POST[categorie]', '$\_POST[titre]', '$\_POST[description]',***  ***'$\_POST[couleur]', '$\_POST[taille]', '$\_POST[public]',  '$photo\_bdd',  '$\_POST[prix]',  '$\_POST[stock]')");***  ***$contenu.= '<div class="validation">Le produit a été ajouté</div>';***  ***$\_GET['action'] = 'affichage';***  ***}***  ***//--- LIENS PRODUITS ---//***  ***$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($ligneas$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='. $ligne['id\_produit'] .'"><img src="../ctrl/img/edit.png"></a></td>';***  ***$contenu.= '<td><a href="?action=suppression&id\_produit='. $ligne['id\_produit'] .'"***  ***OnClick="return(confirm(\'En êtes vous certain ?\'));"><img src="../ctrl/img/delete.png"></a></td>';***  ***$contenu.= '</tr>';***  ***} $contenu.= '</table><br><hr><br>';***  ***}***  ***//--------------------------------- AFFICHAGE HTML ---------------------------------//***  ***require\_once("../ctrl/haut.ctrl.php");***  ***echo$contenu;***  ***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= $resultat->fetch\_assoc();***  ***}echo'***  ***<h1>Formulaire Produits </h1>***  ***<formmethod="post"enctype="multipart/form-data"action="">***  ***<inputtype="hidden"id="id\_produit"name="id\_produit"value="'; if(isset($produit\_actuel['id\_produit']))***  ***echo $produit\_actuel['id\_produit']; echo '">***  ***<labelfor="reference">reference</label><br>***  ***<inputtype="text"id="reference"name="reference"placeholder="la référence de produit"value="'; if(isset($produit\_actuel['reference']))***  ***echo $produit\_actuel['reference']; echo '"><br><br>***  ***<labelfor="categorie">categorie</label><br>***  ***<inputtype="text"id="categorie"name="categorie"placeholder="la categorie de produit"value="'; if(isset($produit\_actuel['categorie']))***  ***echo $produit\_actuel['categorie']; echo '"><br><br>***  ***<labelfor="titre">titre</label><br>***  ***<inputtype="text"id="titre"name="titre"placeholder="le titre du produit"value="'; if(isset($produit\_actuel['titre']))***  ***echo $produit\_actuel['titre']; echo '"><br><br>***  ***<labelfor="description">description</label><br>***  ***<textareaname="description"id="description"placeholder="la description du produit">'; if(isset($produit\_actuel['description']))***  ***echo $produit\_actuel['description']; echo '</textarea><br><br>***  ***<labelfor="couleur">couleur</label><br>***  ***<inputtype="text"id="couleur"name="couleur"placeholder="la couleur du produit"  value="'; if(isset($produit\_actuel['couleur']))***  ***echo $produit\_actuel['couleur']; echo '"><br><br>***  ***<labelfor="taille">Taille</label><br>***  ***<selectname="taille">***  ***<optionvalue="S"'; if(isset($produit\_actuel) &&$produit\_actuel['taille'] == 'S') echo 'selected '; echo '>S</option>***  ***<optionvalue="M"'; if(isset($produit\_actuel) &&$produit\_actuel['taille'] == 'M') echo 'selected '; echo '>M</option>***  ***<optionvalue="L"'; if(isset($produit\_actuel) &&$produit\_actuel['taille'] == 'L') echo 'selected '; echo '>L</option>***  ***<optionvalue="XL"'; if(isset($produit\_actuel) &&$produit\_actuel['taille'] == 'XL') echo 'selected '; echo '>XL</option>***  ***</select><br><br>***  ***<labelfor="public">public</label><br>***  ***<inputtype="radio"name="public"value="m"'; if(isset($produit\_actuel) &&$produit\_actuel['public'] == 'm') echo 'checked ';***  ***elseif(!isset($produit\_actuel) &&!isset($\_POST['public'])) echo 'checked'; echo '>Homme***  ***<inputtype="radio"name="public"value="f"'; if(isset($produit\_actuel) &&$produit\_actuel['public'] == 'f') echo 'checked ';***  ***echo '>Femme<br><br>***  ***<labelfor="photo">photo</label><br>***  ***<inputtype="file"id="photo"name="photo"><br><br>';***  ***if(isset($produit\_actuel))***  ***{echo'<i>Vous pouvez uplaoder une nouvelle photo si vous souhaitez la changer</i><br>';***  ***echo'<img src="'. $produit\_actuel['photo'] . '"  ="90"height="90"><br>';***  ***echo'<input type="hidden"name="photo\_actuelle"value="'. $produit\_actuel['photo'] . '"><br>';***  ***}echo'***  ***<labelfor="prix">prix</label><br>***  ***<inputtype="text"id="prix"name="prix"placeholder="le prix du produit"  value="'; if(isset($produit\_actuel['prix']))***  ***echo $produit\_actuel['prix']; echo '"><br><br>***  ***<labelfor="stock">stock</label><br>***  ***<inputtype="text"id="stock"name="stock"placeholder="le stock du produit"  value="'; if(isset($produit\_actuel['stock']))***  ***echo $produit\_actuel['stock']; echo '"><br><br>***  ***<inputtype="submit"value="'; echo ucfirst($\_GET['action']) . 'du produit">***  ***</form>';***  ***}require\_once("../ctrl/bas.ctrl.php"); ?>*** |

**Code PHP : style.css**

|  |
| --- |
| ***\*{ margin: 0; }***  ***a{ text-decoration: none; color: #000; }***  ***.conteneur{ margin: 0auto; max-: 1170px; }***  ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HAUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***  ***header { background: #000000; padding: 5px; text-align: center; }***  ***header  span{ color: #fff; font-weight: bold; text-transform: uppercase; margin-right: 5%; }***  ***header  nav{ display: inline; }***  ***header  a{ color: #ffffff; text-decoration: none; padding: 5px; }***  ***header  nav  a:hover{ background: #04baf6; }***  ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* MILIEU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***  ***section{ padding: 30px; min-height: 800px; }***  ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BAS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***  ***footer{ background: #000; color: white; text-align: center; padding: 7px0;}***  ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GENERAL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***  ***.erreur{ background: #ff0000; padding: 5px; margin: 5px; }***  ***.validation{ background: #669933; padding: 5px; margin: 5px; }***  ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BOUTIQUE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/***  ***.boutique-gauche{ float: left; : 30%; background: #f2f2f2; min-height: 500px; margin-right: 10%; padding-top: 10px; text-align: center; }***  ***.boutique-gauche  ul{ list-style: none; padding: 0; }***  ***.boutique-gauche  ul  li  a{ display: block; padding: 10px; }***  ***.boutique-gauche  ul  li  a:hover{ background: #04baf6; color: #fff; }***  ***.boutique-droite{float: left; : 60%; }***  ***.boutique-produit{ float: left; : 30%; text-align: center; padding: 5%; border-bottom: 1pxsolid#c0c0c0; }***  ***.boutique-produit:hover{ background: #c0c0c0; }*** |

**Code PHP : fiche\_produit.php**

|  |
| --- |
| ***<?php***  ***require\_once("ctrl/init.ctrl.php");***  ***//--------------------------------- TRAITEMENTS PHP ---------------------------------//***  ***if(isset($\_GET['id\_produit']))  { $resultat= executeRequete("SELECT \* FROM produit WHERE id\_produit = '$\_GET[id\_produit]'"); }***  ***if($resultat->num\_rows <= 0) { header("location:boutique.php"); exit(); }***  ***$produit= $resultat->fetch\_assoc();***  ***$contenu.= "<h2>Titre : $produit[titre]</h2><hr><br>";***  ***$contenu.= "<p>Categorie: $produit[categorie]</p>";***  ***$contenu.= "<p>Couleur: $produit[couleur]</p>";***  ***$contenu.= "<p>Taille: $produit[taille]</p>";***  ***$contenu.= "<img src='$produit[photo]'='150'height='150'>";***  ***$contenu.= "<p><i>Description: $produit[description]</i></p><br>";***  ***$contenu.= "<p>Prix : $produit[prix] €</p><br>";***  ***if($produit['stock'] >0)***  ***{$contenu.= "<i>Nombre d'produit(s) disponible : $produit[stock] </i><br><br>";***  ***$contenu.= '<form method="post"action="panier.php">';***  ***$contenu.= "<input type='hidden'name='id\_produit'value='$produit[id\_produit]'>";***  ***$contenu.= '<label for="quantite">Quantité : </label>';***  ***$contenu.= '<select id="quantite"name="quantite">';***  ***for($i= 1; $i<= $produit['stock'] &&$i<= 5; $i++)***  ***{$contenu.= "<option>$i</option>";***  ***}$contenu.= '</select>';***  ***$contenu.= '<input type="submit"name="ajout\_panier"value="ajout au panier">';***  ***$contenu.= '</form>';***  ***}else{***  ***$contenu.= 'Rupture de stock !';***  ***}$contenu.= "<br><a href='boutique.php?categorie=". $produit['categorie'] . "'>Retour vers la séléction de ". $produit['categorie'] . "</a>";***  ***//--------------------------------- AFFICHAGE HTML ---------------------------------//***  ***require\_once("ctrl/haut.ctrl.php");***  ***echo$contenu;***  ***require\_once("ctrl/bas.ctrl.php"); ?>*** |

**Code PHP : vitrine.php**

|  |
| --- |
| ***functionexecuteRequete($req)***  ***{global$mysqli;***  ***$resultat= $mysqli->query($req);***  ***if(!$resultat)***  ***{die("Erreur sur la requete sql.<br>Message : ". $mysqli->error . "<br>Code: ". $req);***  ***}  return$resultat;***  ***}//------------------------------------***  ***functiondebug($var, $mode= 1)***  ***{ echo'<div style="background: orange; padding: 5px; float: right; clear: both; ">';***  ***$trace= debug\_backtrace();***  ***$trace= array\_shift($trace);***  ***echo'Debug demandé dans le fichier : $trace[file] à la ligne $trace[line].';***  ***if($mode=== 1)***  ***{print'<pre>'; print\_r($var); print'</pre>';***  ***} else***  ***{print'<pre>'; var\_dump($var); print'</pre>';***  ***}  echo'</div>';***  ***}***  ***//------------------------------------***  ***functioninternauteEstConnecte()***  ***{  if(!isset($\_SESSION['membre'])) returnfalse;***  ***elsereturntrue;***  ***}***  ***//------------------------------------***  ***functioninternauteEstConnecteEtEstAdmin()***  ***{ if(internauteEstConnecte() &&$\_SESSION['membre']['statut'] == 1) returntrue;***  ***elsereturnfalse;***  ***}***  ***//------------------------------------***  ***functioncreationDuPanier()***  ***{ if(!isset($\_SESSION['panier']))***  ***{$\_SESSION['panier'] = array();***  ***$\_SESSION['panier']['titre'] = array();***  ***$\_SESSION['panier']['id\_produit'] = array();***  ***$\_SESSION['panier']['quantite'] = array();***  ***$\_SESSION['panier']['prix'] = array();***  ***}}***  ***//------------------------------------***  ***functionajouterProduitDansPanier($titre, $id\_produit, $quantite, $prix)***  ***{creationDuPanier();***  ***$position\_produit= array\_search($id\_produit,  $\_SESSION['panier']['id\_produit']);***  ***if($position\_produit!== false)***  ***{       $\_SESSION['panier']['quantite'][$position\_produit] += $quantite;***  ***} else***  ***{$\_SESSION['panier']['titre'][] = $titre;***  ***$\_SESSION['panier']['id\_produit'][] = $id\_produit;***  ***$\_SESSION['panier']['quantite'][] = $quantite;***  ***$\_SESSION['panier']['prix'][] = $prix;***  ***}}***  ***//------------------------------------***  ***functionmontantTotal()***  ***{$total=0;***  ***for($i= 0; $i<count($\_SESSION['panier']['id\_produit']); $i++)***  ***{ $total+= $\_SESSION['panier']['quantite'][$i] \* $\_SESSION['panier']['prix'][$i];***  ***}returnround($total,2);***  ***}//------------------------------------***  ***functionretirerProduitDuPanier($id\_produit\_a\_supprimer)***  ***{$position\_produit= array\_search($id\_produit\_a\_supprimer,  $\_SESSION['panier']['id\_produit']);***  ***if($position\_produit!== false)***  ***{array\_splice($\_SESSION['panier']['titre'], $position\_produit, 1);***  ***array\_splice($\_SESSION['panier']['id\_produit'], $position\_produit, 1);***  ***array\_splice($\_SESSION['panier']['quantite'], $position\_produit, 1);***  ***array\_splice($\_SESSION['panier']['prix'], $position\_produit, 1);***  ***}}*** |

**Code PHP : panier.php**

|  |
| --- |
| **<?php**  **require\_once("ctrl/init.ctrl.php");**  **//--------------------------------- TRAITEMENTS PHP ---------------------------------//**  **//--- AJOUT PANIER ---//**  **if(isset($\_POST['ajout\_panier']))**  **{   // debug($\_POST);**  **$resultat= executeRequete("SELECT \* FROM produit WHERE id\_produit='$\_POST[id\_produit]'");**  **$produit= $resultat->fetch\_assoc();**  **ajouterProduitDansPanier($produit['titre'],$\_POST['id\_produit'],$\_POST['quantite'],$produit['prix']);**  **}//--- VIDER PANIER ---//**  **if(isset($\_GET['action']) &&$\_GET['action'] == "vider")**  **{ unset($\_SESSION['panier']);**  **}//--- PAIEMENT ---//**  **if(isset($\_POST['payer']))**  **{for($i=0 ;$i<count($\_SESSION['panier']['id\_produit']) ; $i++)**  **{$resultat= executeRequete("SELECT \* FROM produit WHERE id\_produit=". $\_SESSION['panier']['id\_produit'][$i]);**  **$produit= $resultat->fetch\_assoc();**  **if($produit['stock'] <$\_SESSION['panier']['quantite'][$i])**  **{$contenu.= '<hr><div class="erreur">Stock Restant: '. $produit['stock'] . '</div>';**  **$contenu.= '<div class="erreur">Quantité demandée: '. $\_SESSION['panier']['quantite'][$i] . '</div>';**  **if($produit['stock'] >0)**  **{ $contenu.= '<div class="erreur">la quantité de l\'produit '. $\_SESSION['panier']['id\_produit'][$i] .**  **'à été réduite car notre stock était insuffisant, veuillez vérifier vos achats.</div>';**  **$\_SESSION['panier']['quantite'][$i] = $produit['stock'];**  **} else{**  **$contenu.= '<div class="erreur">l\'produit '. $\_SESSION['panier']['id\_produit'][$i] . '**  **à été retiré de votre panier car nous sommes en rupture de stock, veuillez vérifier vos achats.</div>';**  **retirerProduitDuPanier($\_SESSION['panier']['id\_produit'][$i]);**  **$i--;**  **}$erreur= true;**  **}} if(!isset($erreur))**  **{ executeRequete("INSERT INTO commande (id\_membre, montant, date\_enregistrement)**  **VALUES (". $\_SESSION['membre']['id\_membre'] . ",". montantTotal() . ", NOW())");**  **$id\_commande= $mysqli->insert\_id;**  **for($i= 0; $i<count($\_SESSION['panier']['id\_produit']); $i++)**  **{executeRequete("INSERT INTO details\_commande (id\_commande, id\_produit, quantite, prix)**  **VALUES ($id\_commande, ". $\_SESSION['panier']['id\_produit'][$i] . ",". $\_SESSION['panier']['quantite'][$i] . ",".**  **$\_SESSION['panier']['prix'][$i] . ")");**  **}unset($\_SESSION['panier']);**  **mail($\_SESSION['membre']['email'], "confirmation de la commande", "Merci votre n° de suivi est le $id\_commande",**  **"From:vendeur@dp\_site.com");**  **$contenu.= "<div class='validation'>Merci pour votre commande. votre n° de suivi est le $id\_commande</div>";**  **}}  //--------------------------------- AFFICHAGE HTML ---------------------------------//**  **include("ctrl/header.php");**  **echo$contenu;**  **echo"<table border='1'style='border-collapse: collapse'cellpadding='7'>";**  **echo"<tr><td colspan='5'>Panier</td></tr>";**  **echo"<tr><th>Titre</th><th>Produit</th><th>Quantité</th><th>Prix Unitaire</th></tr>";**  **if(empty($\_SESSION['panier']['id\_produit'])) // panier vide**  **{echo"<tr><td colspan='5'>Votre panier est vide</td></tr>";**  **}else{for($i= 0; $i<count($\_SESSION['panier']['id\_produit']); $i++)**  **{echo"<tr>";**  **echo"<td>". $\_SESSION['panier']['titre'][$i] . "</td>";**  **echo"<td>". $\_SESSION['panier']['id\_produit'][$i] . "</td>";**  **echo"<td>". $\_SESSION['panier']['quantite'][$i] . "</td>";**  **echo"<td>". $\_SESSION['panier']['prix'][$i] . "</td>";**  **echo"</tr>";**  **}echo"<tr><th colspan='3'>Total</th><td colspan='2'>". montantTotal() . "euros</td></tr>";**  **if(internauteEstConnecte())**  **{echo'<form method="post"action="">';**  **echo'<tr><td colspan="5"><input type="submit"name="payer"value="Valider et déclarer le paiement"></td></tr>';**  **echo'</form>';**  **}  else{ echo'<tr><td colspan="3">Veuillez vous <a href="inscription.php">inscrire</a>ou vous <a href="connexion.php">**  **connecter</a>afin de pouvoir payer</td></tr>';**  **}echo"<tr><td colspan='5'><a href='?action=vider'>Vider mon panier</a></td></tr>";**  **}echo"</table><br>";**  **echo"<i>Valider votre paiement</i><br>";**  **// echo "<hr>session panier:<br>"; debug($\_SESSION);**  **include("ctrl/bas.ctrl.php");?>** |