# Introduction

**«** Depuis nos débuts il y a 25 ans, nous cherchons à vous proposer des produits innovants et à être à la pointe de la nutrition sportive. C’est ainsi que nous sommes devenus une marque internationale dans laquelle nombre de fans placent leur confiance pour leurs objectifs de santé.

Nous sommes persuadés que l’union fait la force. C’est pourquoi nous avons décidé en 2017 de nous associer à Glanbia Performance Nutrition (GPN), leader de l’industrie de la nutrition de performance, connu pour des marques de premier plan comme Optimum Nutrition, Slimfast, and BSN.

Que vous cherchiez à soutenir vos entraînements avec des boosters de performance de haute qualité ou à atteindre vos objectifs de poids avec les aliments, en-cas et compléments appropriés ou tout simplement à vous sentir en forme et en bonne santé, nos produits sont là pour vous y aider.

C'est cette raison qui a poussé la société **SportClub** à proposer à ses clients un site web pour faciliter l’accès à leurs produits en leur présentant un catalogue en ligne, un service de commande, un système de paiement sécurisé ainsi qu'une livraison rapide tout cela via un nouveau site d'e-commerce : <http://tfe-web.herokuapp.com>.

C’est dans ce cadre que la société **SportClub** m’a chargé de réaliser son site e-commerce. Ce travail a été réalisé en vue de l’obtention du diplôme de Candidat Webdeveloper.

# **Pourquoi un site d’e-commerce ?**

Les habitudes de consommation se sont modifiées ces dernières années. En plus de la possibilité de trouver les produits chez leurs commerçants, les consommateurs souhaitent désormais les trouver en ligne.

L’e-commerce, dès son arrivée, a amené une particularité qui lui est propre et a très vite changé les modes de consommation habituels. Dans le domaine du tourisme par exemple, il est très simple aujourd’hui d’acheter un voyage en ligne ; les outils techniques pour y parvenir se sont multipliés et les achats en magasin se font plus rares.

Internet a permis au consommateur de comparer plus facilement ou d’acheter des produits impossibles à trouver en magasins.

Le commerce traditionnel reste cependant un mode de consommation important.

Ainsi, les achats alimentaires se font encore majoritairement en grande surface. Les voitures aussi s’achètent directement dans les garages.

**Pourquoi ce maintien de la consommation en magasin ?**

Le fait d'avoir un spécialiste pour conseiller le consommateur peut le rassurer.[Text Wrapping Break]Le fait aussi de pouvoir disposer de son produit immédiatement.[Text Wrapping Break]Sur internet, postposer la jouissance de l’achat peut provoquer une certaine crainte quant à la fiabilité du produit. Ces deux éléments freinent le passage à l’achat par internet.

L’e-commerce a permis aux consommateurs d'acheter sans bouger de chez eux ; ils consomment ainsi moins d'essence. Ils ont également plus de choix. Ils peuvent mieux comparer les prix, les produits eux-mêmes. Enfin, l’achat est accessible à tout moment, quels que soient l’heure ou le jour.

On se rend compte que chaque mode de consommation amène à se positionner différemment pour vendre et livrer son produit.[Text Wrapping Break]Aucun mode de consommation n’est préférable à l’autre : **on dirait plutôt que l’ e-commerce et le commerce traditionnel sont complémentaires.**[Text Wrapping Break] L’e-commerce peut apporter au commerce traditionnel beaucoup d'avantages, mais aussi beaucoup d'inconvénients, ce qui explique que certains achats continuent à se faire traditionnellement.

Ces deux modes d'achat sont complémentaires et le resteront, si l'offre de l’e-commerce reste telle qu'elle est aujourd’hui.

# Description du site

## Thème et brève description

Le site d’e-commerce pour la société **SportClub** a été commandé dans l’*objectif* **d’augmenter le** **chiffre d’affaires** de la société et de permettre aux clients **l’achat en ligne** de leurs produits (compléments alimentaires, vêtements de sports, cosmétiques,…).

Les technologies utilisées pour créer le site de **sport Club** sont :

**Back End: Laravel** framework PHP, **Blade** le moteur d’affichage.

**Front End: Vuejs** un framework javascript, **jQuery**, **Javascript**, **REST API**, **WebPack** .

**Design:**  **Sass**, **Bootstrap4**, **CSS3**, **HTML5**, **Git**.

**Data Base:** Mysql.

**Editeur:** VS Code

Le site est déployé sur le serveur d’ **Heroku**.

Pour acheter sur le site de SPORTClub et découvrir ses produits en ligne, plusieurs approches sont possibles. Le client peut demander **une catégorie** spécifique et voir tous les produits, il peut aussi utiliser le **système de recherche avancé**, celui-ci permet aux clients de chercher un **produit** désiré.

En **sélectionnant** un produit, toutes les explications disponibles sur ce produit s’affichent d’une manière agréable et facile à comprendre.

Le site est responsive avec une très grande fluidité que ce soit sur Mobile, tablette et PC (voir photo dans les secteurs de site).

Les couleurs principales sont le vert et le noir avec des couleurs secondaires comme le rouge et le bleu.

Chaque page du site a son propre rôle et sa propre fonctionnalité.

Le site de la société **sport Club** se divise comme suit :

## Header et Footer

**Header** et Footer sont des sections qu’on peut trouver dans tout le site et auxquelles on peut accéder à tout instant.

L’accès au Header est simple ; on peut y trouver le logo de la société **SportClub** à gauche. En cliquant sur le logo, le site va se diriger vers la page **d’accueil**.

Le Header a été créé une seule fois et réutilisé dans tout le site via un *include* du moteur d’affichage BLADE; Pour le design, j’ai utilisé **Bootstrap4** (**framework** **css**) qui fonctionne avec **FlexBox**.

Le menu est divisé en deux parties : une partie, à droite, contient la connexion et l’inscription sur le site, ainsi qu’un menu déroulant pour sélectionner la langue ; une autre partie, au milieu du Header, contient un menu avec les pages d’accueil, de vente et de contact.

Le Header est différent selon le **niveau d’utilisateur** connecté au site.

1. Pour l’administrateur et l'auteur, ils auront ***l’espace administration*** pour gérer les différentes parties du site.
2. Pour l’utilisateur, il aura son ***propre espace*** via une connexion à son compte par un login et un mot de passe. Le header contiendra un menu de *dropdown* pour le profil et un autre pour le panier.

**Footer** ressemble beaucoup au Header. Il se décline en trois parties.

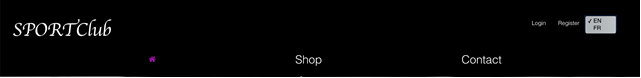
La première partie, à gauche, expose les droits réservés à la société **SportClub**.

La deuxième partie, au milieu, contient le menu. C’est le même que dans Header.

La troisième et dernière partie, à droite, permet de se diriger vers le haut du site en poussant sur un bouton.

[Text Wrapping Break]Plus loin dans ce travail, chaque partie du menu est décrite plus en détails.

Voir Photos :







## Page d’accueil

La page d’accueil est une **vue globale** du site. **Elle** contient un **slider** qui explique les avantages des produits en termes de bien-être pour le sportif.

Les photos du slider changent régulièrement en utilisant **jQuery** et **Bootstrap4**.

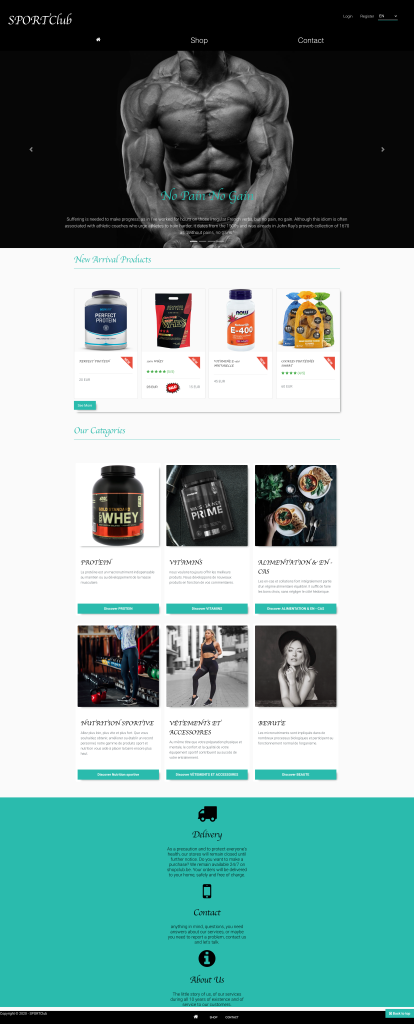
Une section présente les ***nouveautés du site*** et dans le magasin **SportClub**.[Text Wrapping Break] Les 4 derniers nouveaux produits sont ajoutés sur le site. Grâce à un clic sur un bouton (see more / voir plus), l’utilisateur va être redirigé vers la page shop pour voir plus de produits.

Nous trouvons, toujours dans la page d’accueil, une section qui présente 6 catégories de produits. Chaque **catégorie** contient une image et le nom de la catégorie, avec une brève description. Un bouton permet d’accéder aux différents produits de cette catégorie. L’administrateur peut lui, **ajouter,** **supprimer ou** **modifier** une catégorie, et c’est le même cas pour la liste de produits.

Juste avant le Footer, une section donne un aperçu général concernant les modes de livraison, comment contacter la société et une présentation de celle-ci.

Voici une capture d’écran pour donner une idée de la page d’accueil.

Voir Photo:



## Page de Shop

Cette page présente un **aperçu général** de tous les produits disponibles dans le magasin sous le lien **/** products .

Dans Shop, **il existe plusieurs parties ou sections**. L’une d’elle retient l’attention du client avec une photo suggestive ; une autre partie contient une annonce, une photo et une description à l’appui. En poussant sur un **bouton,** on peut ***en savoir plus***.

L'**annonce** est gérée par le **gestionnaire du site**.

Le plus important dans le Shop est la section des **produits recommandés**, qui se base sur la liste des produits les plus demandés par les clients. La section des produits recommandés est gérée en utilisant une requête Mysql avancée pour chercher les derniers produits ayant l’avis le plus positif.

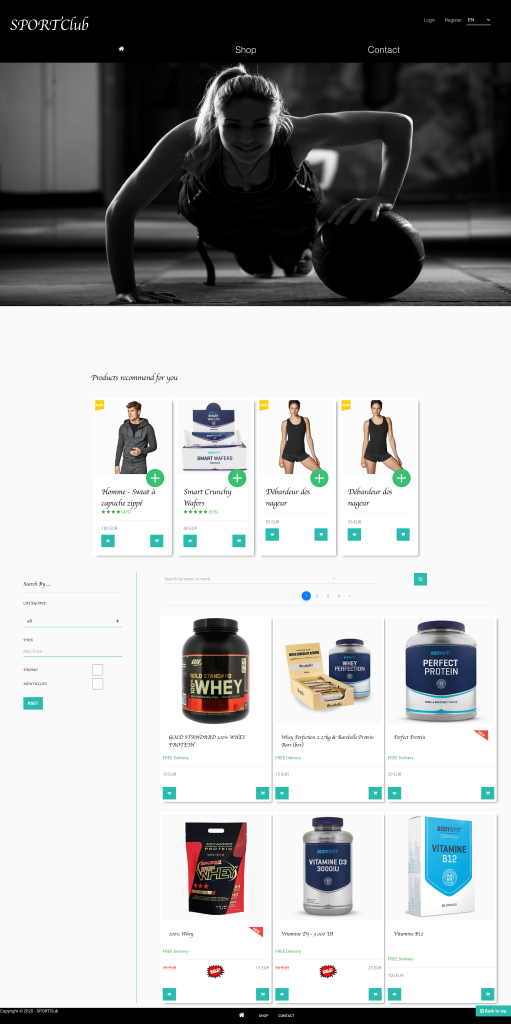
Avec une **recherche avancée**, nous pouvons découvrir très facilement les produits spécifiques selon le **filtre** choisi. Nous pouvons filtrer les produits selon les catégories à partir du menu déroulant, ou choisir le prix maximum grâce à **des checkbox** qui proposent les réductions, les nouveaux produits, et les produits les plus achetés. La recherche peut aussi être faite via un mot-clé sur le nom de la marque on utilise pour ce faire **VUEJS** **framework** côté client. L’important, en utilisant cette technologie, est d’envoyer une requête API qui va retourner sur le site avec des résultats **plus rapides sans** faire **RECHARGER** la page.

Pendant la recherche de résultat demandée, une animation de recharge va être effectuée pour montrer que le résultat est en train d’être traité.

L’**affichage de produit** est en ***pagination system***. Chaque page contient 9 catalogues de produits et cela peut être changé via la requête **MySQL** du côté back end ; chaque **catalogue** présente une photo et le nom du produit.[Text Wrapping Break]**Si** le produit est nouveau, un ***logo l’affiche*** comme **indiqué dans la base de données**. Pour **visualiser** **la réduction** sur le produit, dans une même case sont affichés l’ancien et le nouveau prix. Enfin, deux **boutons** permettent, l’un de **voir** dans le catalogue, l’autre de placer directement le produit au **panier**.

La **pagination** est ***userfriendly***, et réalisée via **VUEJS** pour chercher les résultats sans RECHARGER la page pour que ce soit plus rapide.

Voir photo:



## Page de Contact

Cette page présente sous le lien **/** contact, les différentes possibilités de **contact** avec la société **SportClub** : numéros de téléphone, adresse mail, ...

Une **carte** fait partie de la page contact pour connaître la localisation des différents magasins de la société **SportClub** via des **points de repérages**.[Text Wrapping Break]

La carte est modifiée en utilisant **Vue JS**, grâce au composant ***leaflet map*** avec les markers, il est facile maintenant de sortir avec une belle carte avec des explications sur le magasin et les heures de fermetures et l’adresse de chaque magasin.

Les différentes localisations des magasins sur la carte peuvent être changées selon les besoins : via l’administrateur, on peut **ajouter** ou **Supprimer** des localisations.

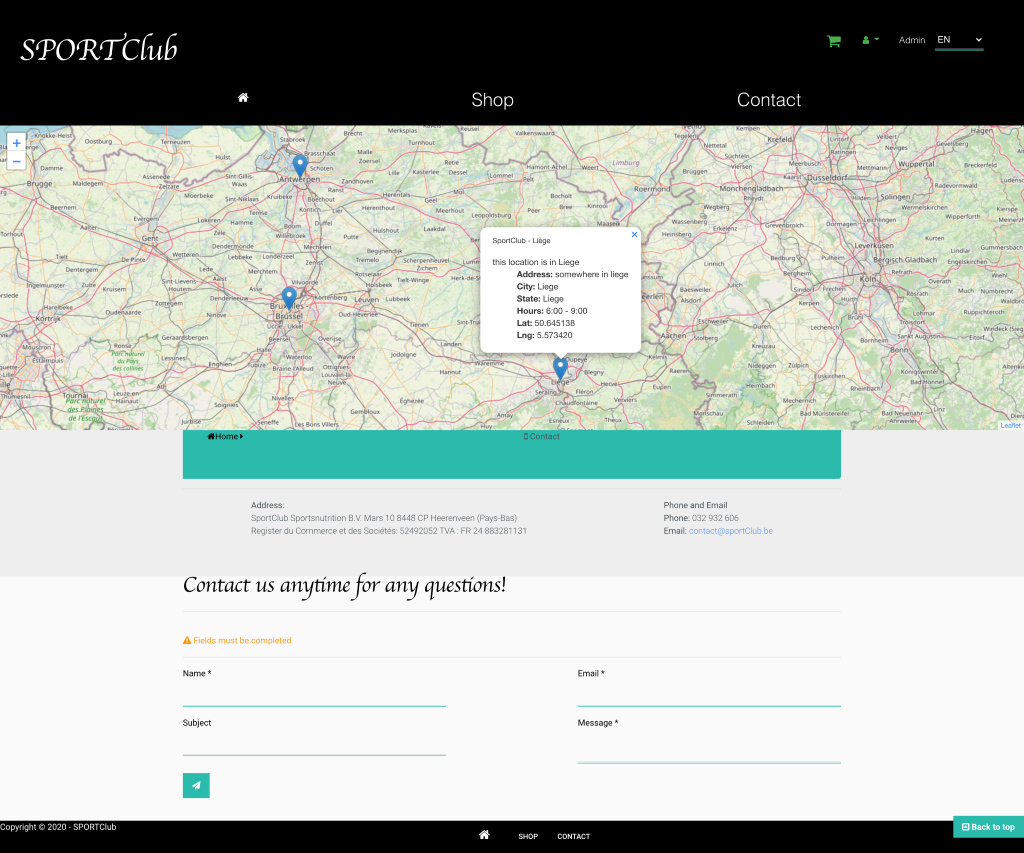
Un ***accès rapide au menu*** est possible dans la page contact, avec un **système de navigation aisé** entre les pages.

**Un formulaire de contact** avec des champs validés est disponible au cas où le client voudrait contacter la société.

La validation des champs est contrôlée via le côté backend de **Laravel** et le côté **frontend** de **Bootstrap**. Il faut que les champs obligatoires soient remplis pour envoyer le formulaire.

Les champs du formulaire reprennent le nom, l'email, le sujet du contact et le message.

Voir photo :



## Page d’inscription

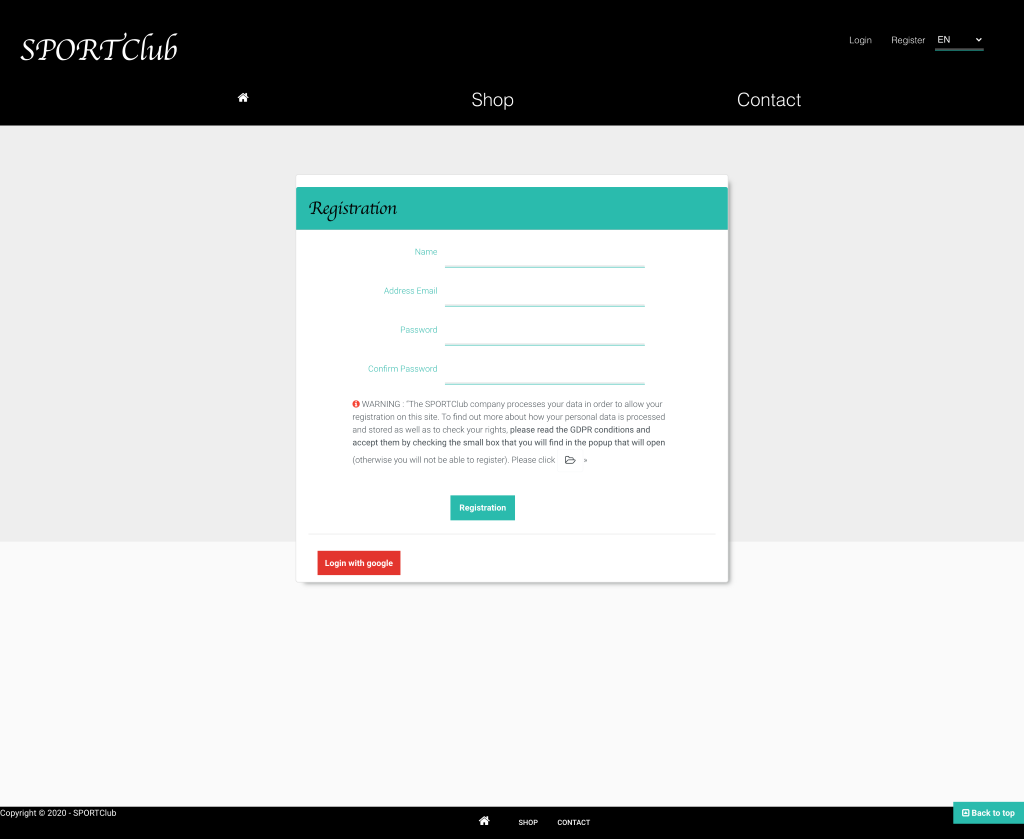
### **La page d'inscription utilise le lien / register. Elle ressemble beaucoup à la page** **de connexion : même message de validation, nom affiché dans le profil, adresse mail, mot de passe et confirmation de mot de passe.**

**RGBD** (Règlement européen sur la protection des données) est très important pour compléter l’inscription sur le site. Depuis le 25 mai 2018, la protection des données de la vie privée est régie par une loi. Le RGBD fait référence à celle-ci et ***explique à l’utilisateur*** quels usages l’administrateur du site pourra faire de ses données personnelles.

Pour que le **bouton d’inscription** soit activé, l’utilisateur doit ***ouvrir un pop-up*** pour lire la règle générale de la société, il les accepte en cliquant sur le bouton.

Pour **finaliser l’inscription** il lui faudra scroller, cocher et accepter les conditions du RGBD.

Voir Photo. :



## Page de connexion

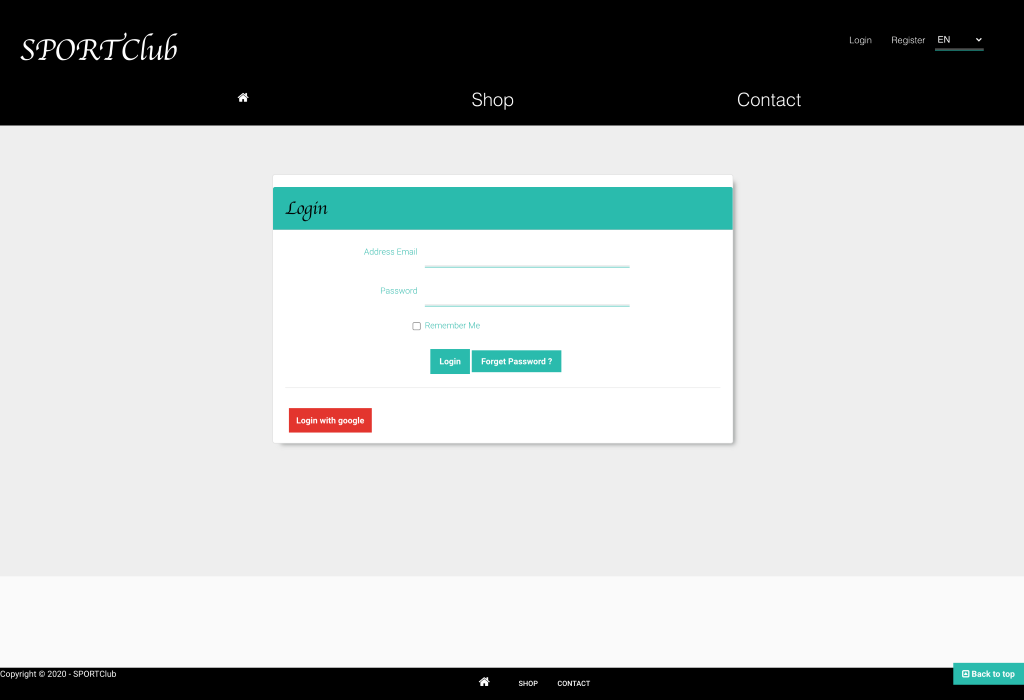
La **page de connexion** présente sous le lien **/** login un système de login classique, avec adresse mail et mots de passe, **“se souvenir de moi”,** et un bouton pour se connecter.

Les **champs** doivent être **remplis correctement** par l’utilisateur : **Si** une **adresse mail** par exemple ne figure pas dans notre *base de données*, un *message* s’affiche en expliquant l’*erreur en couleur rouge,* Une vérification du formulaire est prévue tant en Javascript côté client qu’en PHP côté serveur.

Un **bouton** permet de retrouver un **mot de passe oublié** ; en cliquant dessus, un email sera envoyé avec un lien pour réinitialiser un nouveau mot de passe.

Il y a la possibilité de **se connecter via Google**. Le login, via Google, fait la connexion rapidement et facilement. En cliquant sur ce **bouton**, un *pop-up s’affiche* pour choisir le compte Google désiré. Après que le client ait **vérifié sa manœuvre**, il pourra se connecter via Google et enregistrer les formations pour que le pop-up ne s’affiche plus la prochaine fois.

Voir photo:



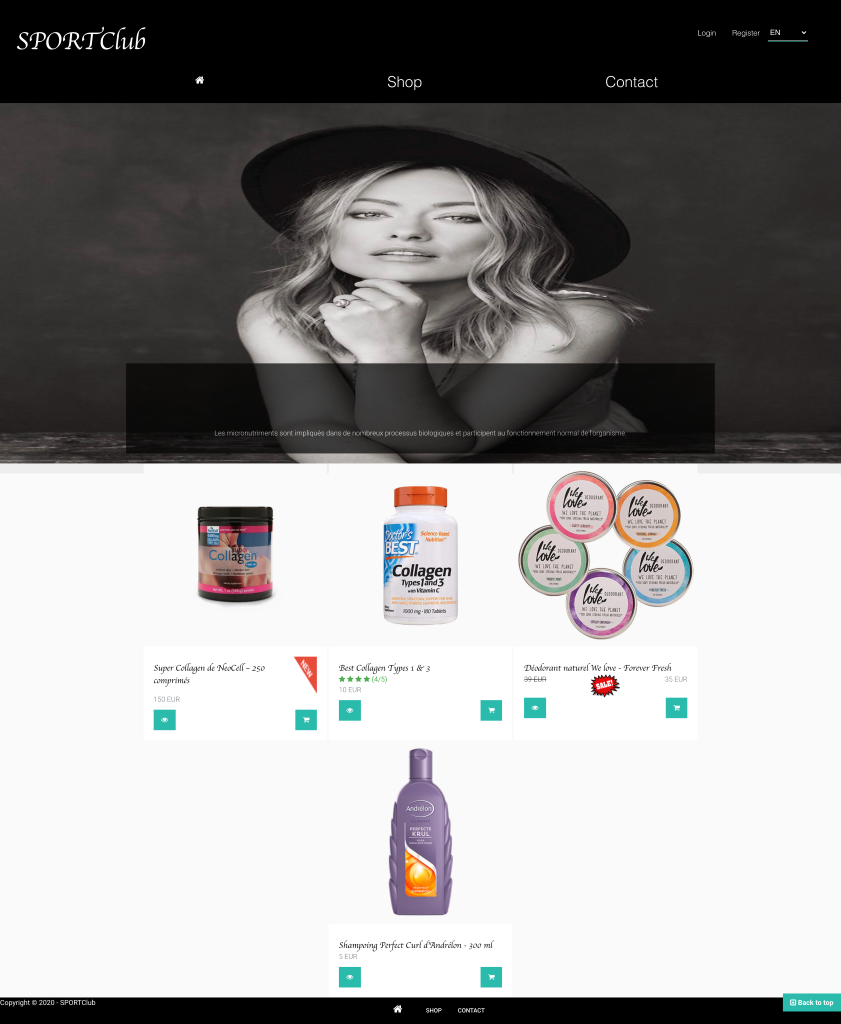
## Page de Single Catégorie

Dans le page **Single Catégorie** sous le lien **/** single-categorie **/** id, nous trouvons **tous les produits** de cette catégorie.

Les produits sont affichés dans un **catalogue**. Chaque catalogue contient une photo, des avis et le logo des nouveaux produits, avec un bouton pour passer au panier, ainsi qu’un autre pour **afficher les détails** sur le produit.

Il est possible d'effectuer une recherche sur les produits de cette catégorie.En cliquant sur telle catégorie, l’id est enregistré et utilisé dans un requête SQL pour chercher les produits qui ont la même catégorie.

Voir photo :



## Page de Single Produit

### **Single produit sous le lien / single-product/ id, présente une page avec un menu d'accès rapide et une couleur d’activation de menu pour indiquer la page ouverte.**

Un ***slider*** affiche plusieurs photos sur le produit, le nom, le logo des nouveaux produits, le logo de réduction s’il existe, avec l’ancien et le nouveau prix, la description du produit, l’avis des clients, les caractéristiques du produit comme sa couleur, sa marque, sa taille etc...

La section “**avis du client**” est un **formulaire** avec le nom, l’email, l’**avis** signalé sous **forme d’étoiles** et, enfin, le contenu du commentaire ( textarea). La **validation** sur le champ est **appliquée**, et **si** le client est déjà **connecté** sur le site, l’avis du client sera envoyé avec son nom et son email enregistrés.

**Si** l’utilisateur est **connecté** sur le site, une **salutation** avec son nom va s’afficher. Par contre, s’il s’agit d’un utilisateur anonyme, cela s'affiche différemment.[Text Wrapping Break]Le formulaire d’avis client est fait en **VUEJS** et le design est fait en utilisant le framework material Vue Js.

Des icônes sont utilisées partout dans le site, grâce à **fontawesome** et **material** **icônes**.

Une animation d’erreur s’affiche si les champs obligatoires ne sont pas bien remplis.

Si les champs sont correctement remplis une animation de succès s’affiche.

La partie d’affichage d’un avis contient tous les avis sur le produit avec la date et le nom de l’utilisateur, son image et son commentaire.

Si l’image n’est pas chargée sur le profil d’utilisateur ou si l’utilisateur est anonyme, une image d’avatar anonyme va être affichée.

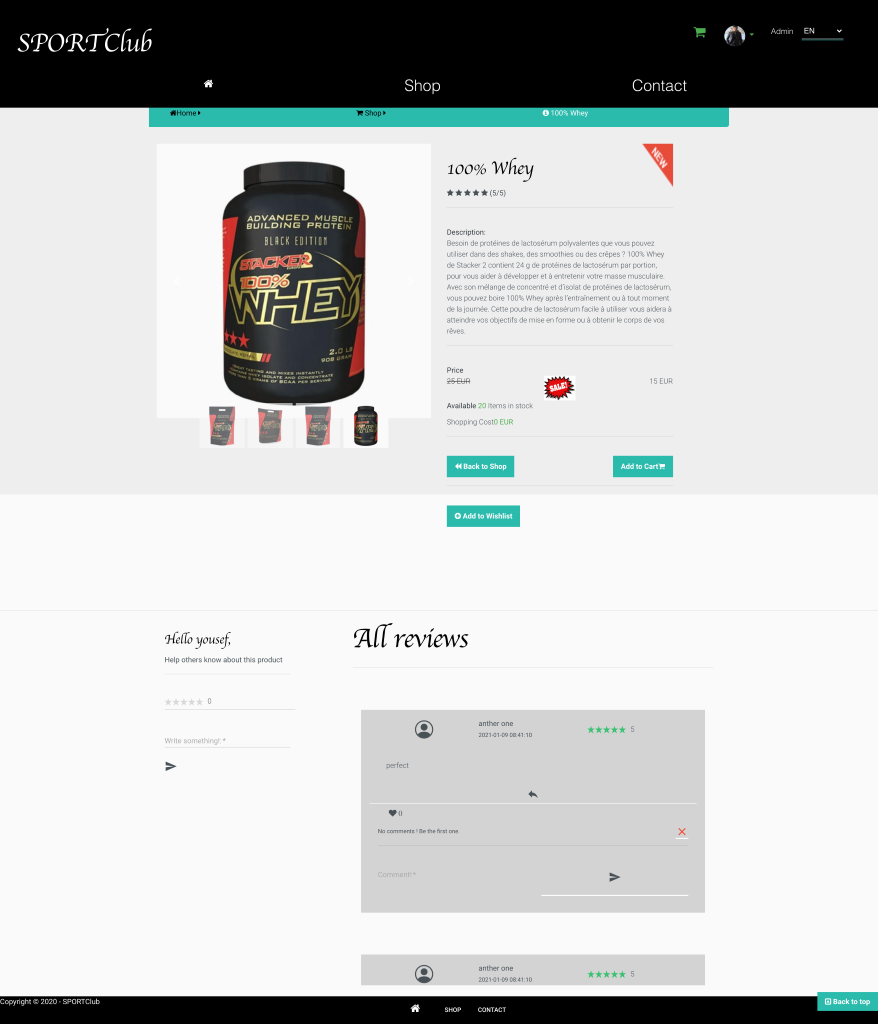
Cette partie est gérée via **vuejs ?,** des icônes de material **vuejs** et **fontawesome**.Si un utilisateur connecté sur le site dépose un avis, le « mote (you / vous) », va s’afficher pour indiquer que la personne a mis le commentaire ; pour les autres utilisateurs, son nom va aussi s’afficher.

Sur un commentaire, il y a la possibilité d’aimer (like), de répondre et de voir les autres réponses.

Pour répondre à un commentaire, il faut cliquer sur l’icône et un formulaire va s’afficher. Si l’utilisateur est connecté, son nom va s’enregistrer dans la base de données.

S’il n’y a aucune réponse, un « attention « va s’afficher pour indiquer qu’il n’y a aucun commentaire sur cet avis.

Voir photo :



## Page d’administrateur

*L'affichage sera différent selon le* ***niveau*** *de l’utilisateur connecté*. La page de l’administration s’appelle avec le lien **/** admin.

La partie administrative et toutes les fonctions sont réalisées en utilisant **VUEJS**, **Material** **vuejs**, **REST** **API** , **vue-router** pour gérer les différentes routes **sans** le rechargement de la page.

Comme cela sera expliqué en détail, chaque partie de l’administrateur est codé comme suit .

Un top nav est créé une fois, et utilisé partout dans l’administrateur. Ce top nav a une partie pour afficher un titre sur la page où on est, une icône de la boîte d’emails pour le message des clients et une icône pour le profil d’utilisateur connecté. En cliquant sur cette icône, on va sur le profil de l’utilisateur connecté.

Un menu sidebar, à gauche, propose les options disponibles, la gestion du profil connecté, la gestion des utilisateurs, la gestion des produits, des catégories, la carte, et des commandes.

Comment cela est fait?

Les composants sidebar, Top nav, footer sont créés une seule fois, et réutilisés partout dans l’interface administrateur via l’importation de ses composants.

La page de chaque partie de la gestion change selon la route demandée via le menu de sidebar. Une couleur du menu sélectionné est activée sur la page visitée.

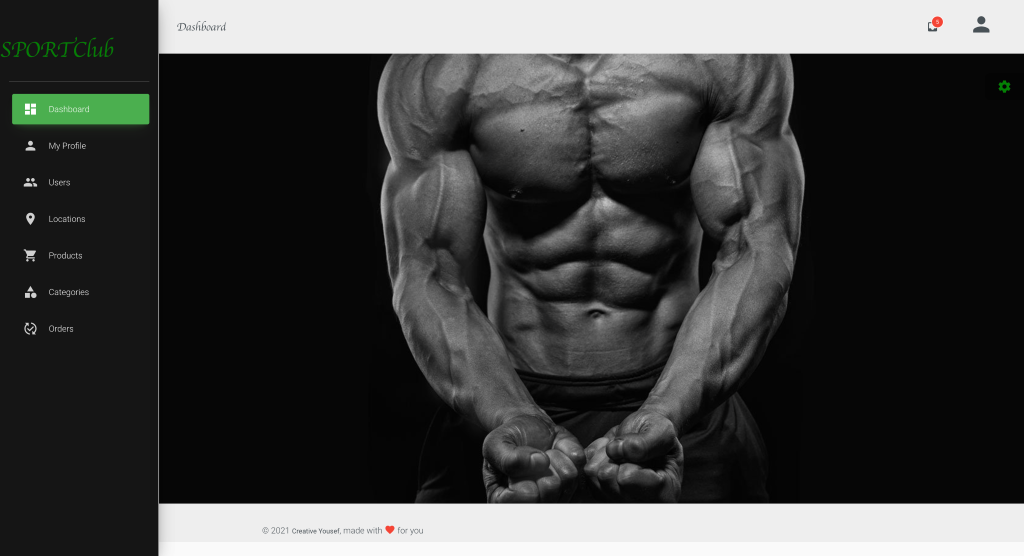
Pour se rendre sur une page, il faut une requête qui utilise les **hooks** (***lifecircles*** ) comme les **created**, **mounted**, **fonctions** pour envoyer une API et chercher les données nécessaires pour les afficher sur la page.

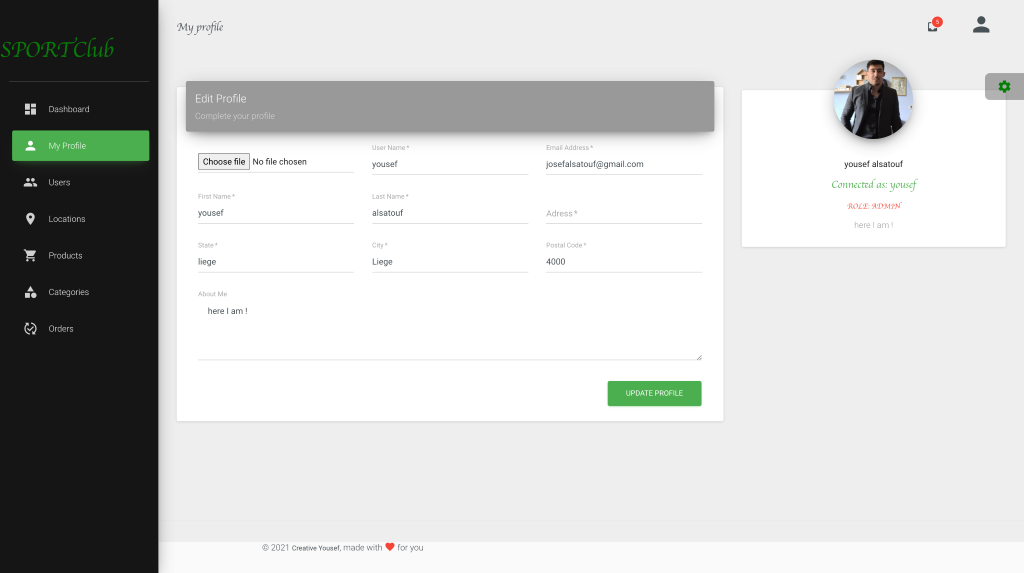
Ainsi, les données peuvent être partagées entre les composants grâce au **vuejs** et le **$emit**, une **fonctionnalité** qui permet de faire *un événement quand* les données changent, et ensuite de les changer partout où elles se trouvent.

La gestion de la partie administration est similaire partout pour la création, une modification, ou une suppression.

Tout est effectué via l’API et **axios package javascript** pour ajouter ou afficher les données de la base de données.

Voir photos :





### **Page de la boite des mails:**

En top nav se trouve une icône pour accéder à la boîte des messages. L’icône affiche le numéro de message reçu. L'icône de message est faite en utilisant des composant et des icônes material **vuejs** .Selon les niveaux d’utilisateurs, un administrateur et un acteur peuvent accéder à la boîte d’emails.

### **Page gestion des utilisateurs :**

L'administrateur, comme il est enregistré dans la base de données via le rôle admin, est le **seul niveau d’utilisation** qui peut **changer** le rôle d’un utilisateur.

Au niveau affichage, un utilisateur est décrit par son image, son nom, son prénom, son email, son rôle, etc…

Le rôle d’un utilisateur peut être modifié uniquement par l’administrateur.

Pour changer le rôle, une requête API est envoyée via AXIOS avec les données vers le backend et des animations de traitement et de réussite.

### **Page gestion des produits :**

Le formulaire compte une fonction pour valider une donnée et il existe des champs obligatoires à remplir.

Il est aussi possible de *changer* la galerie de photos d'un produit, de la sauvegarder. Il existe un autre formulaire distinct pour décrire les propriétés d’un produit.

Une **validation** est nécessaire pour le formulaire : elle permet d’envoyer des messages d’erreurs au cas où certains champs ne seraient pas remplis.

Pour créer un produit, le formulaire contient une liste de catégories : nom du produit, son prix, le nombre disponible en stock, la photo … .

Un message de réussite apparaît en cas de création, de modification ou de suppression.

On peut aussi ajouter des propriétés pour un produit, comme la couleur, la taille, la marque …

### **Page gestion des catégories :**

Le formulaire de la page gestion des catégories sert pour **créer** et **modifier** en validant **les messages de validation**, comme expliqué précédemment. Il en va de même pour les messages qui signalent la réussite, la modification et la suppression de la création, ainsi que la structure de la liste d’affichage.

La gestion d’une catégorie est similaire à la gestion d’un produit, tout est géré en **vuejs** et des requêtes API via **axios**.

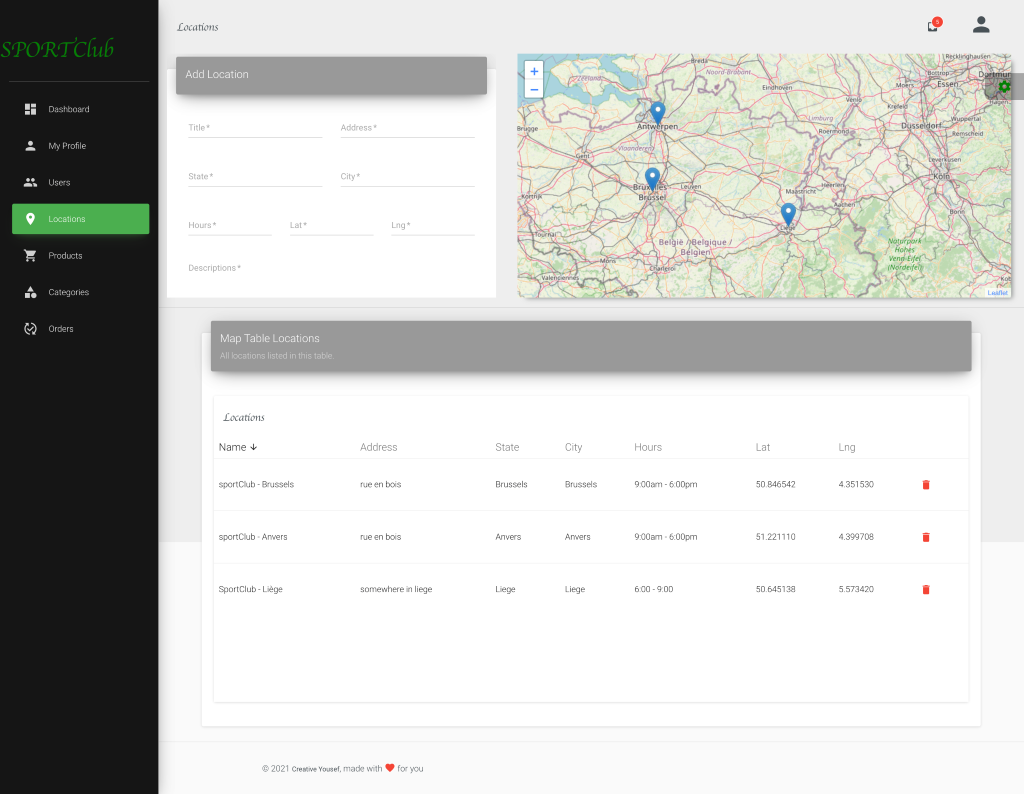
### **Page gestion des commandes :**

Pour **gérer les commandes**, l’éditeur et l’administrateur sont autorisés à changer l’état d’une commande, à la mettre en attente ou bien en route, à livrer…

Le lien vers admin**/**orders affiche la liste des commandes, qui contient le nom du client et vérifie s’il a bien payé. Les **boutons** “actions” permettent de modifier l’**état** d’une commande pour modifier l'état d’une commande.

Cela est fait via une requête API en vuejs qui permet de communiquer avec le backend, d’envoyer la requête et de recevoir la réponse sans charger la page.

### **Page gestion des localisations :**



## Page de profile d’utilisateur :

Avec le lien /user, on peut trouver le profil d’utilisateur qu’on salue, avec le nom de l’utilisateur connecté, ainsi qu’un sidebar avec les différents liens vers les pages auxquelles l’utilisateur peut accéder.

Le sidebar contient :

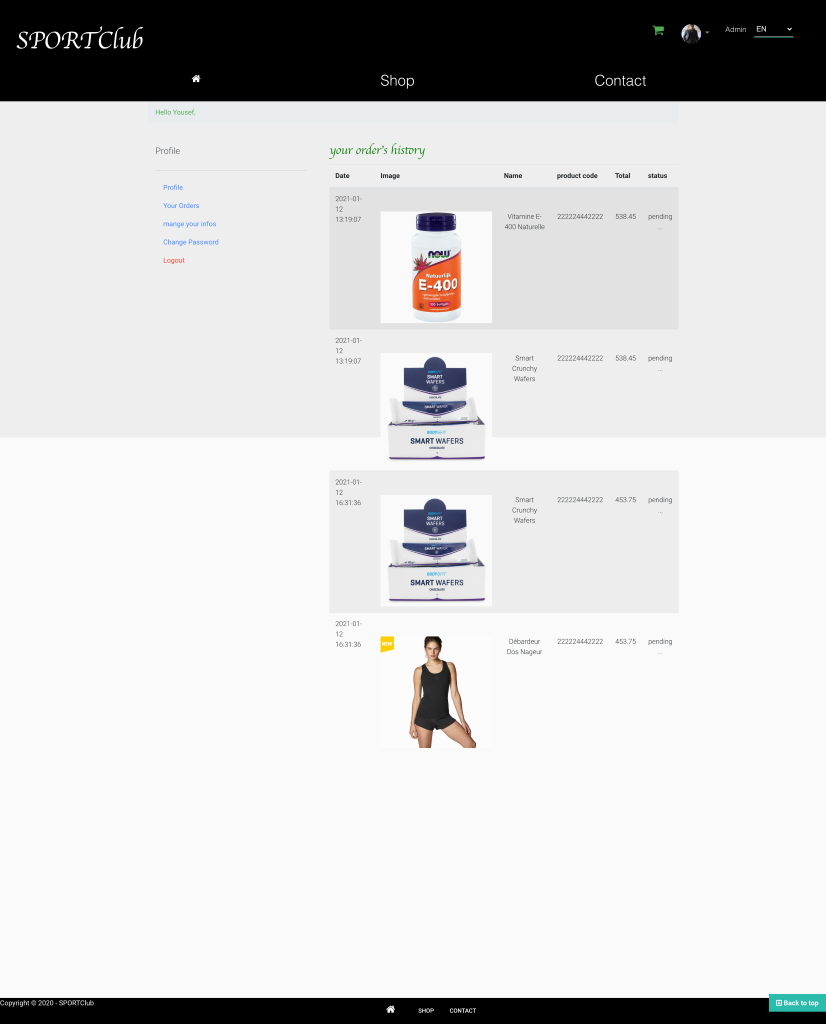
* La page de profil/user qui affiche toutes les infos sur l’utilisateur comme son nom, son adresse, son email, sa photo, ...
* La liste des commandes /orders qui affiche l’historique de toutes les commandes
* Le processus de gestion d’info/infos pour modifier ou ajouter des infos sur l’utilisateur ou pour changer la photo de profil.
* Un formulaire pour changer le mot de passe /password, ou pour en définir un autre.
* La déconnexion /logout pour l’utilisateur

## Page d’article commandé :

Grâce à /orders, un utilisateur peut voir l’historique de ses commandes.

La page contient un tableau avec la photo, la date, le code, le montant payé …

Voir photo :



## Page de participation aux newsletters :

Si un utilisateur n’a pas encore participé aux newsletters, le lien de participation va toujours s’afficher dans le menu de profil, en vérifiant dans la base de données s’il a participé ou pas.

En cliquant, l’utilisateur va se diriger vers la page newsletter pour pouvoir participer.

A l'aide d’un formulaire, l’utilisateur va entrer son adresse mail et recevoir un lien pour confirmer son adresse à personnelle.

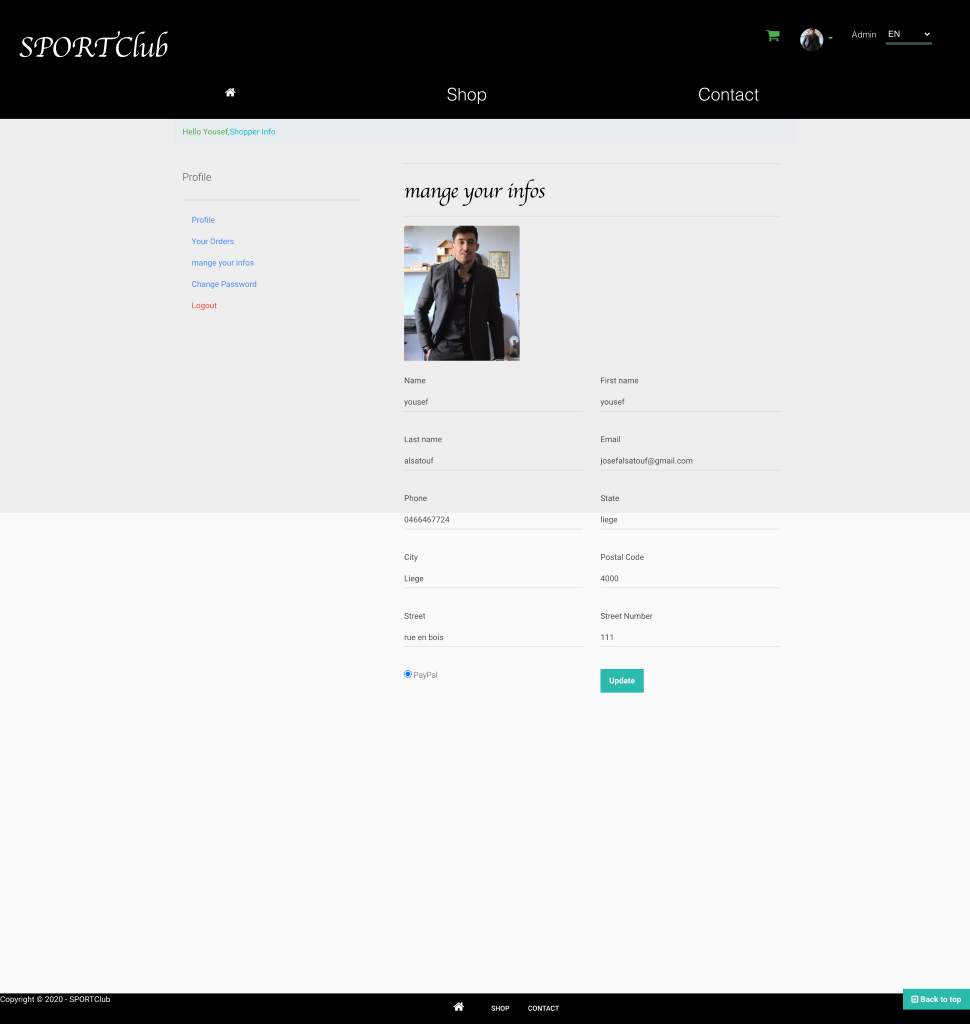
Un message de confirmation va s’afficher après la participation.

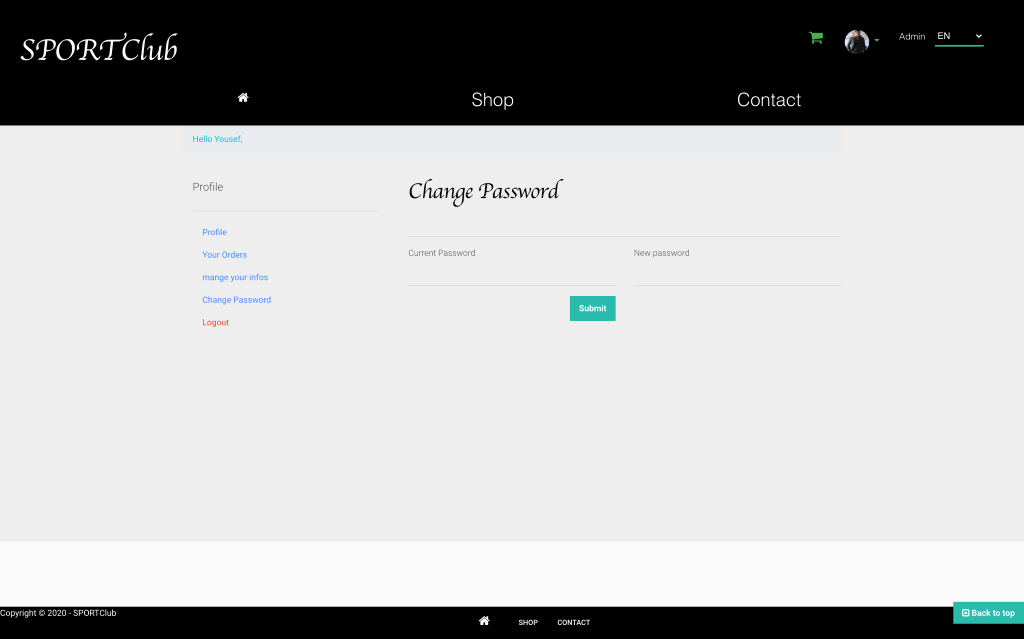
## Page de modification de profile d’utilisateur :

Pour gérer les informations d’utilisateur, un formulaire permet d’ajouter ou de modifier les informations existantes.

Le formulaire permet de valider et d’envoyer des messages pour prévenir des champs obligatoires et des messages de réussite, ou de choisir la photo de profil.

Voir photos :





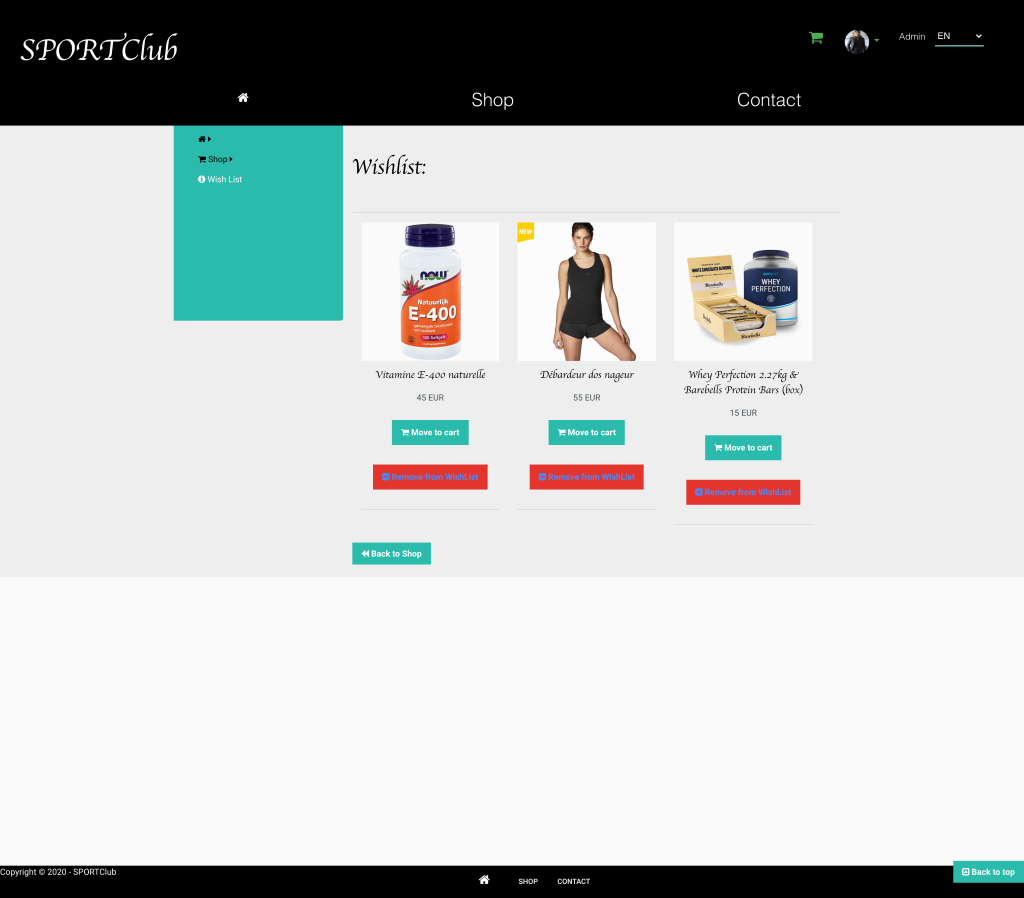
## Page de wishlist :

Parfois, le client voudrait placer un produit sur une liste d’attente mais pas au panier.

C’est le travail de **wishlist** (liste des produits souhaités) sous le lien **/**wishlist : le client peut ajouter des produits avec la possibilité de les retirer ou de les passer au panier.

J’ai pu le faire grâce à Laravel. Chaque client a sa *propre liste d’attente* qui est gérée avec la table **wishlist.** Quand le client ajoute un produit, Laravel va demander à l’utilisateur de bien se connecter, et d’utiliser son id pour créer son produit et l’ajouter en *wishlist*.

Voir photo :



## Page de panier :

**/**cart renvoie à la page panier.

Sur cette page, il existe un menu d'accès rapide avec activation à partir de la page actuelle. Ensuite, le nombre des éléments ajoutés au panier ainsi qu’un tableau avec la photo d’un produit, son nom, la disponibilité en stock, la quantité, le sous-total et les actions pour augmenter ou diminuer la quantité d’un produit, un bouton permet la suppression.

Le nombre d'articles commandés ne peut pas être supérieur au nombre enregistré dans la base de données.

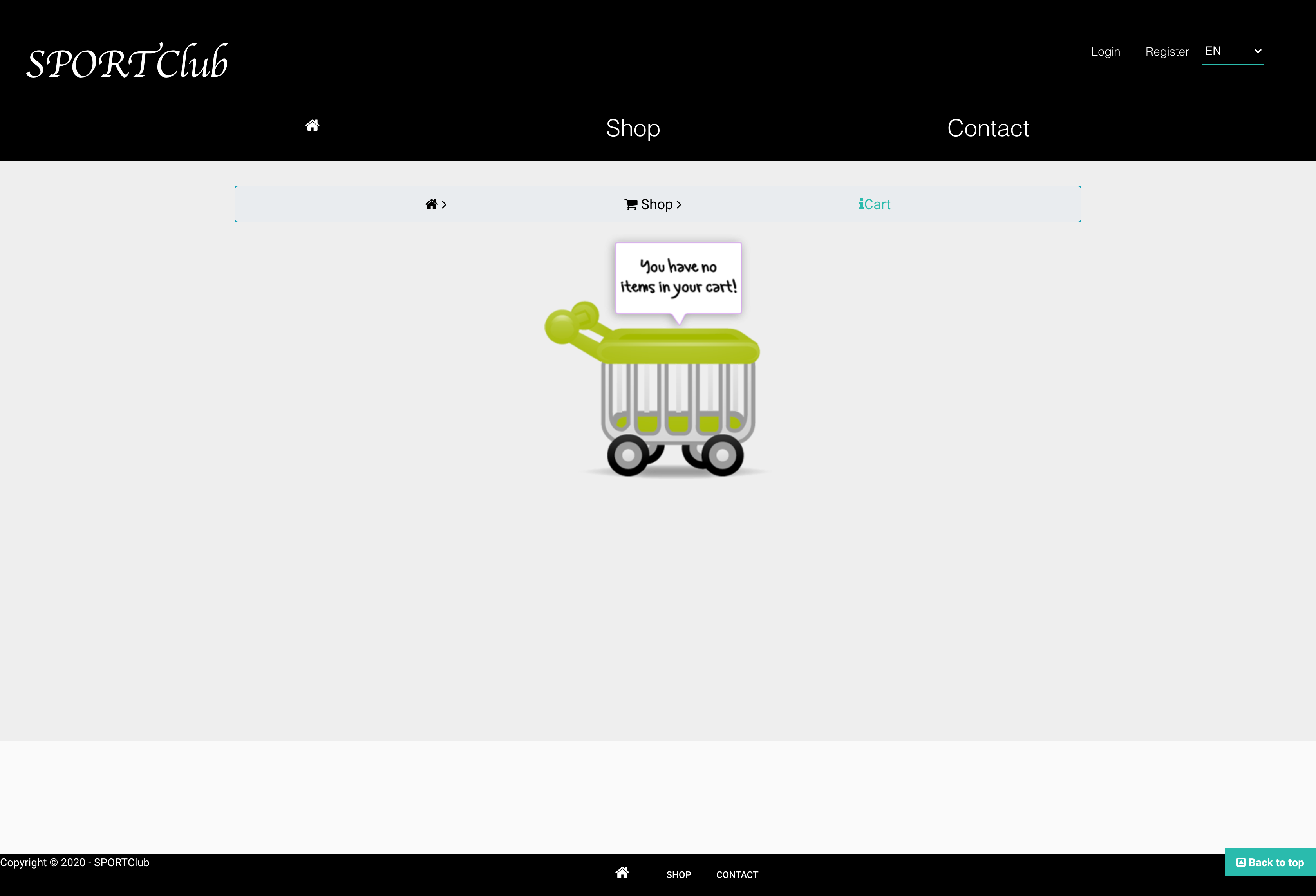
Tout cela est **contrôlé via javascript et vuejs** qui permet d’effectuer les différentes actions **sans** le rechargement de la page. En modifiant la quantité, le sous-total, le total et le stock, vont être modifiés selon la quantité enregistrée.

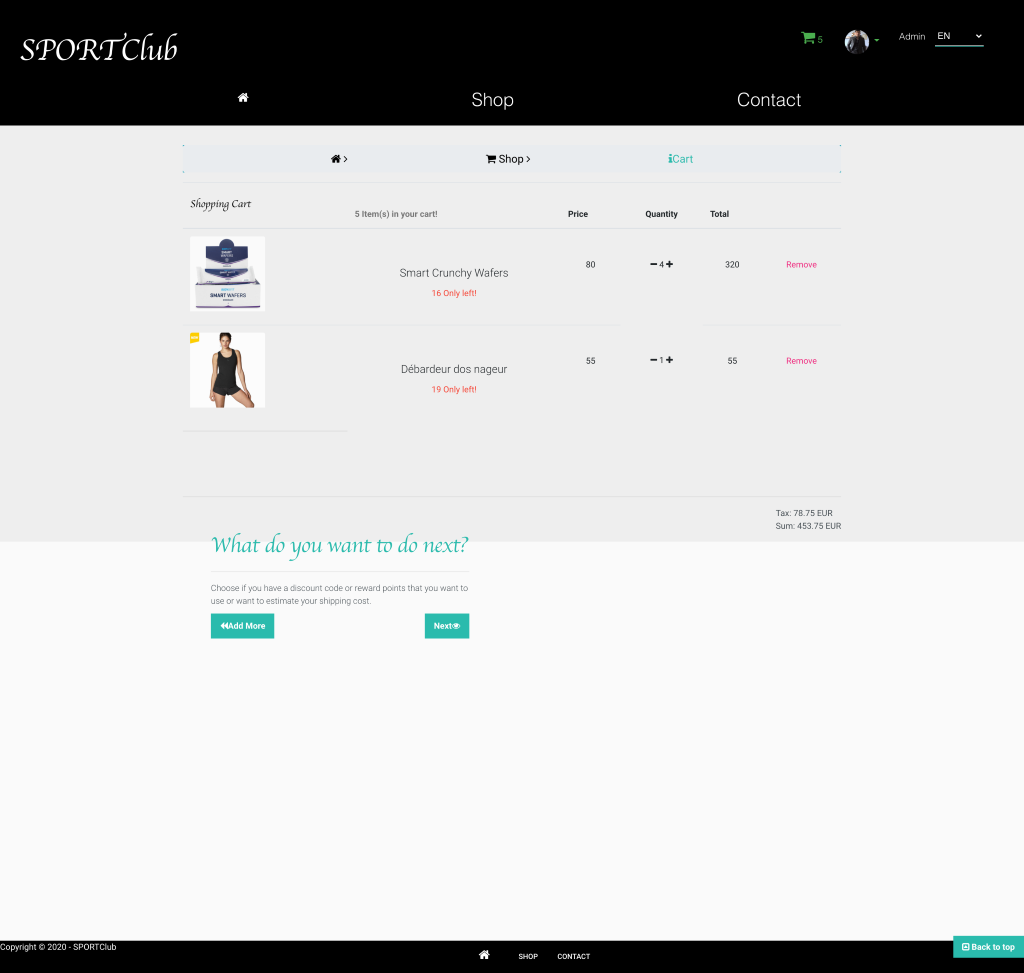
En bas, sur la page panier, nous trouvons la commande avec les taxes, les frais, le sous-total et le total.

La commande est mise à jour selon les éléments au panier.

Sur la page panier, le client a le choix : aller à la page suivante ou continuer d'acheter des produits.

Voir photos :





## Page de passation de commande :

Pour passer la commande, à la page suivante, on trouve un formulaire avec les informations du client ou les informations sur le client.

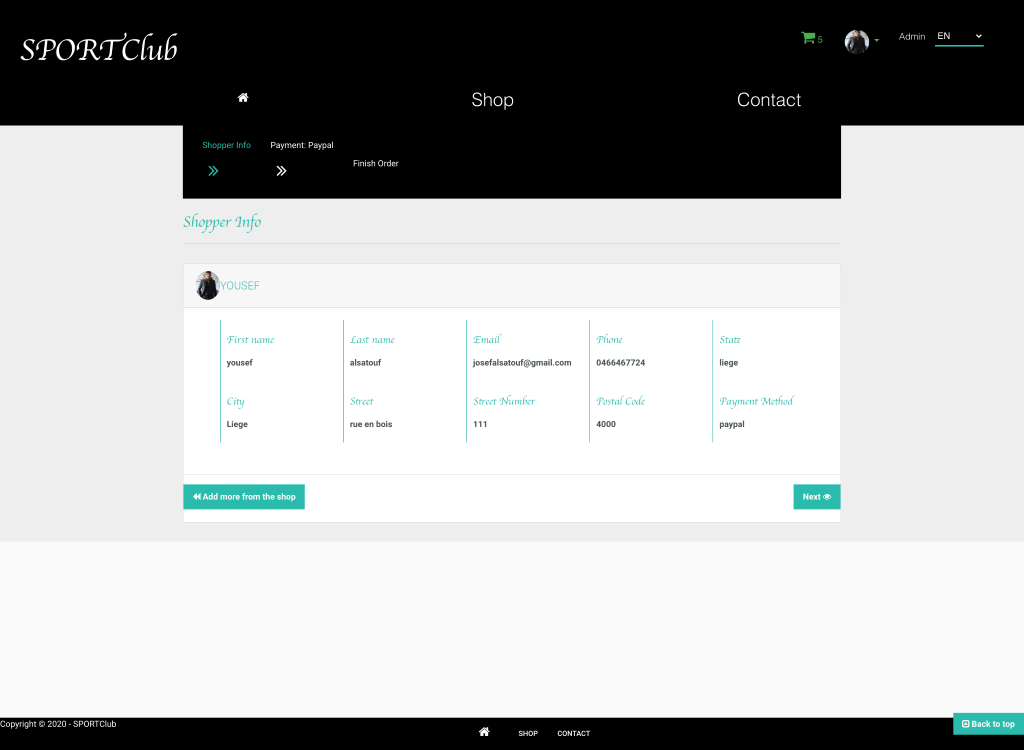
En cliquant sur la page suivante, si on rencontre un formulaire, cela signifie qu’il manque des informations sur le client (adresse ou numéro de téléphone), et il faut le compléter.

Pour passer la commande, le client doit compléter ses données, et remplir le formulaire.

Bien sûr, la validation de formulaire est appliquée et il faut insérer des vraies valeurs.

Si le client n’est pas connecté, grâce à la **fonction Auth::check()**, **Laravel** va lui demander de se connecter pour y accéder, sinon il va le rediriger vers la page de connexion.

Voir photo :



## Page de paiement :

Les méthodes de paiement utilisées sont PayPal, carte bancaire visa, etc. ....

La page contient un menu rapide avec la couleur d’activation sur la page actuelle.

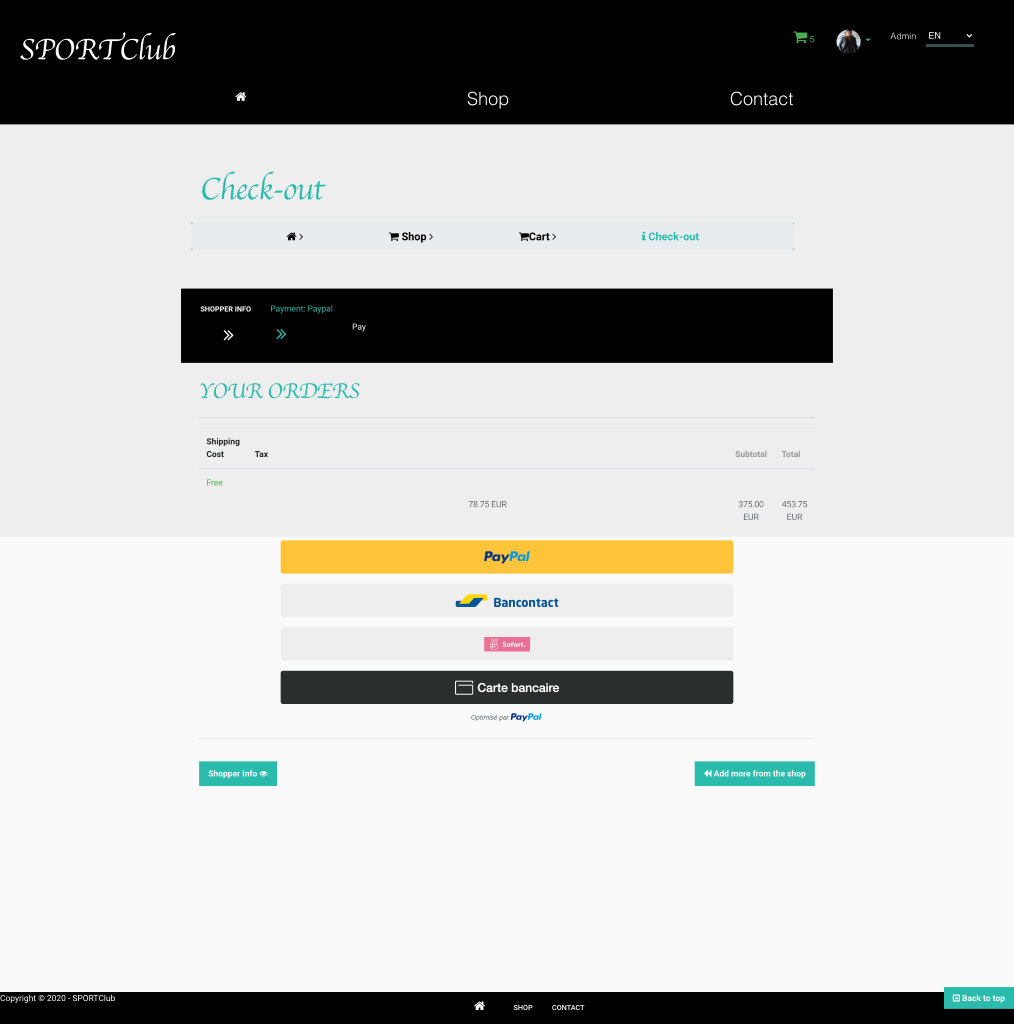
Nous trouvons aussi le détail de la commande passée et un bouton pour payer et effectuer le paiement. Une fois cliqué sur le bouton “pay”, un popup “PayPal” va s’afficher pour insérer l’email de PayPal. Après la validation et le paiement via PayPal, plusieurs actions vont se dérouler : la création de la commande, l’état de paiement va changer, la commande va être traitée et un email avec la facture va être envoyé vers le client en utilisant l’adresse mail enregistrée dans la base de données.

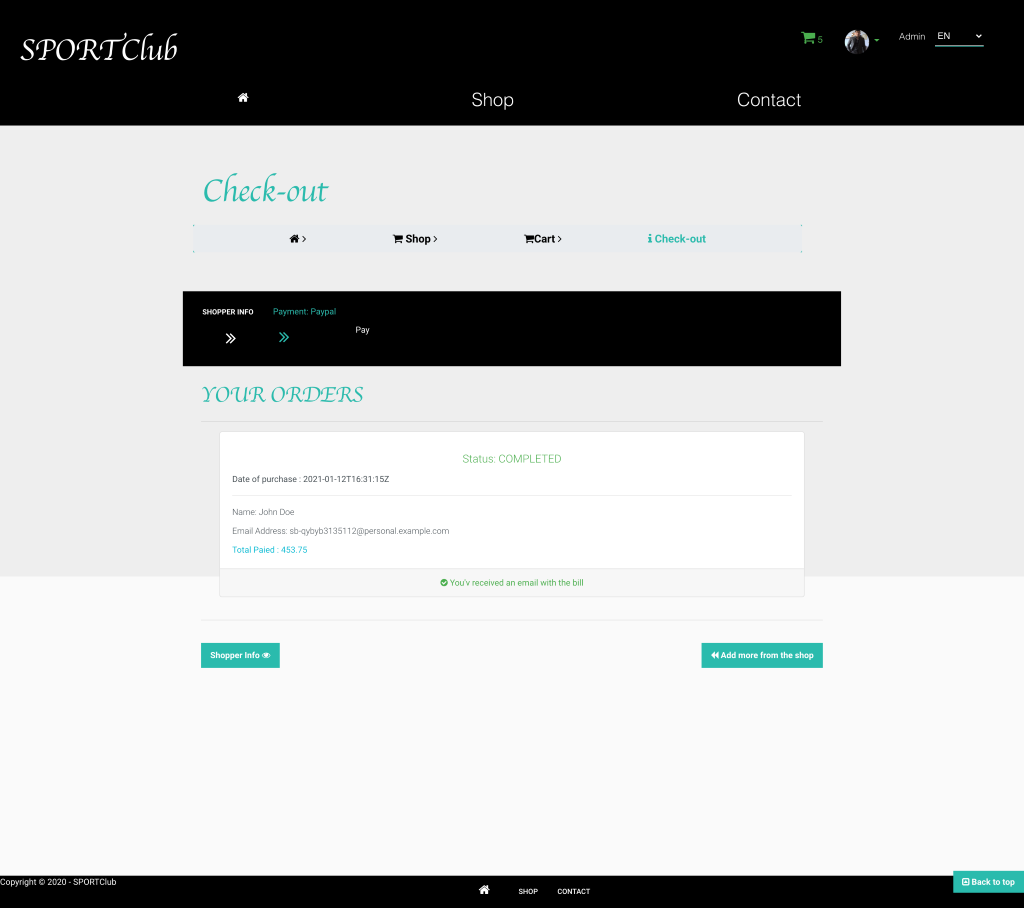
Pour que le client paye la commande à l'aide de **vuejs**, un composant de paypal est utilisé et le paiement va être traité sans recharger la page.

Enfin, le paiement sera effectué et l’animation va apparaître

Un message de réussite va s’afficher pour confirmer le paiement avec le détail de la commande.

Voir photos :





## Objectif

**Objectifs de ce site :**

* Un site internet est une source d’information sur la société.
* Les magasins sont situés hors agglomération, donc plus difficilement accessibles pour certains clients. La présentation en ligne est un atout.
* La commande et la livraison s’effectuent très facilement.
* Le paiement en ligne est plus simple.
* La notoriété du magasin est améliorée.
* La création d’un nouveau service aux clients pourra les fidéliser.
* La promotion des produits ou des services est simplifiée.
* L’envoi des emails et des factures pour les clients allège le travail administratif.

**Mon** objectif personnel **est, par l'élaboration de ce projet, de rassembler toutes les techniques apprises lors de mes trois années de formation tant d'un point de vue front-end que back-end en passant par la création de la DB. Dans ce site, j’ai utilisé notamment le PHP en génération de formulaire. Le cours de SGBD m’a servi pour la réalisation de la base de données J’ai utilisé Javascript pour rendre le site plus rapide, plus efficace et plus sécurisé. Les vérifications pour l’entrée des formulaires par exemple se sont faites tant en JavaScript côté client qu’en PHP côté serveur. J’ai utilisé Vue Js, un Framework JavaScript pour effectuer les requêtes sans devoir recharger la page. Pendant le traitement, une animation s’affiche pour montrer que la requête est en train d’être traitée.**

## Public cible

Le site a été réalisé pour une entreprise familiale qui vend des produits énergétiques ainsi que des vêtements de sports et des cosmétiques de marques renommées.[Text Wrapping Break]Son **public cible** est *le sportif* belge soucieux de son alimentation.[Text Wrapping Break]Le but est de pouvoir, en plus des magasins qui sont situés partout en Belgique, toucher les sportifs *qui souhaitent pouvoir acheter en ligne*.

La firme annonce : **«** Nous sommes fiers d'être une entreprise familiale. Nous mettons toute notre énergie pour promouvoir une nutrition sportive saine et performante.[Text Wrapping Break]Ce qui a commencé dans une cuisine est devenu une des sociétés leaders dans son domaine. Aujourd'hui, notre équipe est plus nombreuse qu'à ses débuts, mais l'esprit initial demeure.[Text Wrapping Break]Nous vous proposons la meilleure expérience nutritionnelle possible afin de répondre au mieux aux spécificités de votre pratique sportive et vous permettre de puiser des ressources inestimables au plus profond de vous-mêmes.[Text Wrapping Break]Chez **SportClub**, nous plaçons l’innovation au cœur de nos activités : de la recherche à la vente, en passant par la conception, le design, la production et la logistique. Les équipes de nos vingt marques Passion mettent en effet toute leur énergie pour concevoir des produits techniques, beaux et simples, toujours aux prix les plus bas possible.[Text Wrapping Break]Ces produits s’adressent à tous les passionnés, du débutant au sportif confirmé, et sont disponibles en exclusivité chez SportClub. **»**

# Exigences fonctionnelles

## Fonctionnalités

**1. Visiteur anonyme**

* Responsive-mobile first.
* Authentification par mail et Google.
* Voir la liste des produits.
* Voir le détail d’un produit.
* Voir les produits recommandés.
* Envoi des mails au vendeur.
* Ajout de produits au panier.
* Barre de recherche.
* Recherche avancée.
* Formulaire de contact.
* Intégrer une carte.
* Site multilingue en préparation.
* Possibilité d’émettre un commentaire sur un produit et de l’évaluer.
* Possibilité de répondre à un commentaire émis par le visiteur
* Réalisation d’une image anonyme dès que le commentaire ou la réponse est envoyé.
* Possibilité d'augmenter et de réduire la quantité au panier, de supprimer un produit du panier et aussi voir le détail d’un produit.

**2 Utilisateur**

**En plus de ce qui précède**

* Gestion de profil d’utilisateur : modifier ses infos, changer son mot de passe.
* Possibilité d’uploader une photo, de la changer ou de la supprimer.
* Possibilité de payer en ligne comme PayPal.
* Recevoir des factures par mail
* Recevoir des mails automatiques.
* Avoir la possibilité de participer aux Newsletters.
* Voir des messages en cas de modification, comme changer de mot de passe.
* Voir le détail d’une commande.
* Passer la commande aux prochaines étapes.
* Voir la liste des commandes passées.
* Déconnecter.
* Compter la quantité de produits ajoutés au panier.
* Voir l'état d'une commande.
* Accès rapide aux plusieurs parties du site via le menu.
* Ajouter des articles au wishlist.
* Réinitialiser un nouveau mot de passe si l’utilisateur oublie son mot de passe.

**3. Editeur**

**En plus de ce qui précède**

* Gestion d’interface d’administration.
* Gérer les produits (ajouter, modifier, supprimer).
* Gérer la catégorie des produits (ajouter, modifier, supprimer).
* Gérer les commandes (ajouter, modifier, supprimer).
* Gérer les couleurs du site (changer les couleurs du thème).

**4. Administrateur**

**En plus de ce qui précède**

* Gérer les utilisateurs (changer les rôles).
* Recevoir les messages de l’utilisateur et les lire, les supprimer.

## Règles métiers

* Le déblocage d'un compte se fait via l'envoi d'un email
* Un mot de passe contient au moins 7 caractères.
* Le formulaire doit être correctement complété pour que la commande passe aux prochaines étapes.
* Mots de passe oubliés ? Envoi d’un lien vers l’email entré via le formulaire de la page “mot de passe oublié”.
* Différents niveaux d'accès sur le site et possibilités différentes pour chaque utilisateur.
* Validation de formulaire partout sur le site.

# Exigences non fonctionnelles

## Conception

* Le layout est adapté aux desktops, tablettes et smartphones. Mobile first
* Flexbox et Bootstrap sont utilisés pour ajuster la position des éléments.
* Javascript / jQuery pour les animations comme les avis par étoiles
* Framework VueJs pour l’interface client et l’envoi des requêtes via l’API
* Les couleurs principales utilisées sont le vert et le noir.

## Acteurs

Un acteur représente un ensemble cohérent de rôles joués par des entités externes (utilisateur humain, matériel ou autre système) qui interagissent directement avec le système étudié et peuvent consulter ou modifier directement l'état du système en émettant et/ou recevant des messages susceptibles d'être porteurs de données.

Les **acteurs** de notre système sont :

- le **simple visiteur** du système : considéré comme une personne qui veut satisfaire sa curiosité.

- le **client** : il bénéficie des différents services offerts par l'application.

- **l'éditeur** : personne qui a pour rôle secondaire de gérer les tables de la base de données limitées selon son rôle.

- l**'administrateur** : personne qui a pour rôle principal de gérer toutes les tables de la base de données.

# Cas d'utilisation

Tout système peut être décrit par un certain nombre de cas d'utilisation correspondant aux besoins exprimés par l'ensemble des utilisateurs.

Un cas d'utilisation représente un ensemble de séquences d'actions réalisées par le système qui produisent un résultat observable intéressant pour un acteur. Il permet de décrire ce que le futur système devra faire, sans spécifier comment il le fera.

Les cas d'utilisation qu'on peut recenser sont :

Les cas d'utilisation d'un visiteur :

* Se connecter au site.
* Naviguer dans le site
* Accéder au Shop.
* Ajouter et accéder au Panier
* Voir les détails sur un produit, aimer un produit, donner un avis et évaluer.
* Écrire un nouvel avis.
* Effectuer une recherche avancée.
* Répondre à un avis existant.
* Découvrir les locations de magasins disponibles sur la carte.

Les cas d'utilisation d'un utilisateur :

* Accéder à l'interface d’utilisateur
* S'authentifier
* Accéder au shop
* Ecrire un nouvel avis
* Gérer son propre profil (photos, infos …)
* Passer une commande
* Répondre à un avis existant
* Évaluer un produit
* Effectuer une recherche avancée.
* Changer son propre mot de passe
* Mot de passe oublié
* Participer aux Newsletters.
* Paiement en ligne.
* Se connecter au site
* Se déconnecter

Les cas d'utilisation d'un éditeur:[Text Wrapping Break](en plus des accès utilisateur)

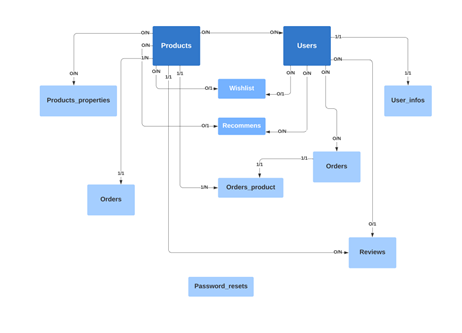
* Accéder à l'interface d’administration
* S'authentifier
* Gérer son propre profil (photos, infos …)
* Changer son propre mot de passe
* Déposer les produits et les modifier.
* Gérer les catégories et les modifier.
* Gérer les localisations et les modifier.
* Se connecter au site
* Se déconnecter

Les cas d'utilisation de l'administrateur:[Text Wrapping Break](en plus des accès utilisateur et éditeur)

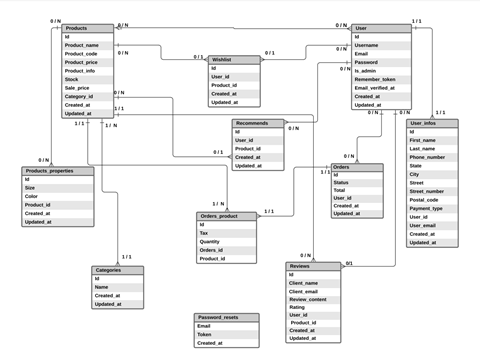
* Accéder à l'interface d’administration en ayant les pleins pouvoirs
* S'authentifier
* Gérer son propre profil (photos, infos …)
* Répondre à un avis existant
* Changer son propre mot de passe
* Gérer la base de données
* Consulter les tables de la base de données
* Modifier les tables de la base de données
* Ajouter les tables de la base de données
* Supprimer les tables de la base de données
* Changer de mot de passe
* Se connecter au site
* Se déconnecter

# Description de la base de données

## Schéma conceptuel



## Schéma physique



# Présentation des problèmes et solutions envisagées

Ma passion pour le code m’a motivé pour ce travail de fin d’études à utiliser des Frameworks et des technologies que je n’avais pas vues au cours de mes années d’études à Saint-Laurent. J'aime les nouveaux challenges, c’est pour cela que j’ai choisi de travailler ainsi. Ce fut pour moi le premier défi de ce travail.

**Laravel** est un framework php. Il m’a permis de découvrir beaucoup de fonctionnalités que je ne connaissais pas. J’ai passé beaucoup de temps sur la documentation de Laravel pour connaître sa capacité et ce que je pouvais réaliser en utilisant le framework.

Avec Laravel et le **Framework Vue Js** notamment, j’ai pu intégrer des nouvelles fonctionnalités sur des éléments comme la méthode de paiement, le système d’avis, le dashboard d’administration, les animations.

J’ai effectué aussi d’autres recherches sur la connexion via google, les requêtes mySql effectués, les sélections relationnelles entre multi tables ou la gestion d’une erreur 500, après le déploiement du site sur Heroku. Autant de challenges pour moi que j’ai mis du temps à découvrir.

Au niveau front-end, réaliser ce projet m’a appris aussi de nouvelles choses, comme connecter le frontend avec le back-end et aussi résoudre des challenges d**'API**.

Déboguer les erreurs m’a permis d’apprendre à réagir aux bugs et à résoudre ces problèmes rapidement.

J’ai aussi rencontré un problème d’intégration avec la carte. Je pensais utiliser Vue Js component pour intégrer la carte de Google Map. Je me suis rendu compte que, pour intégrer une carte, il fallait créer une API key et aussi créer un compte de facturation pour que la clé d'API fonctionne. La solution que j’ai trouvée pour google map est d’utiliser Iframe tag. Finalement, je n’ai pas gardé la solution avec Google Map mais je me suis dirigé vers le composant de Vue Js qui s’appelle **Leaflet Map** pour intégrer une carte avec des marques et des positions en latitude et longitude, chaque marque présente les infos nécessaires sur le magasin.

Un autre problème rencontré fut celui du rechargement de la page pour les changements sur le site (édition, modification, suppression sur le site). Ce problème a été résolu du côté client par l’utilisation d’une API de requêtes.

À l’aide de documentations des technologies comme Laravel.com, **Vue.Js** et MDN, mais aussi la communauté des développeurs, certains problèmes ont pu être résolus.

J'ai profité des sources et des conseils des autres développeurs pour bien prendre les problèmes un par un, tester et débugger, et ainsi gagner du temps.

# Conclusion

Au départ, j’avais pensé à l’informatique comme à un métier, un job pour gagner ma vie, mais maintenant je pense que programmer et travailler avec des technologies que j’aime est **mon mode de vie**.

Écrire un code et construire des projets actuels que les internautes utilisent chaque jour et partout dans le monde, c’est en effet quelque chose qui me donne du courage pour coder plus et partout, et cela me permet de rêver d’être toujours plus efficace et plus professionnel.

Pendant mes trois ans à l'école, j’ai reçu beaucoup de connaissances, j’ai compris que nous avons besoin d’apprendre et d'apprendre toujours, et le code est très amusant.

J'ai appris à bien réfléchir sur les problèmes rencontrés et à suivre des règles pour les résoudre. S’il n’est pas possible de les résoudre immédiatement, il ne faut pas perdre de temps et il faut les laisser pour un autre moment.

Après ce projet, je n’ai plus peur d’intégrer de nouvelles idées, d'utiliser et de découvrir de nouveaux frameworks.

J'aimerais bien aller encore plus loin que le web et découvrir et expérimenter tout ce que je peux et en savoir plus sur le monde de l’informatique.

**Table des matières**

[1. Introduction 1](#_Toc390282953)

[2. Description du site 2](#_Toc390282954)

[2.1. Thème et brève description 2](#_Toc390282955)

[2.2. Objectif 2](#_Toc390282956)

[2.3. Public cible 2](#_Toc390282957)

[3. Exigences fonctionnelles 3](#_Toc390282958)

[3.1. Fonctionnalités 3](#_Toc390282959)

[3.2. Règles métiers 3](#_Toc390282960)

[3.3. Interface 3](#_Toc390282961)

[4. Exigences non fonctionnelles 4](#_Toc390282962)

[4.1. Conception 4](#_Toc390282963)

[4.2. Acteurs 4](#_Toc390282964)

[5. Cas d'utilisation 5](#_Toc390282965)

[6. Description de la base de données 6](#_Toc390282966)

[6.1. Schéma conceptuel 6](#_Toc390282967)

[6.2. Schéma des entités 6](#_Toc390282968)

[7. Présentation des problèmes et solutions envisagées 7](#_Toc390282969)

[8. Conclusion 8](#_Toc390282970)

BIBLIOGRAPHIE

Sources:

|  |  |
| --- | --- |
| **Le Lien vers git repo** | <https://github.com/yousefalsatouf/TFE-ecomerce> |
| **Le lien vers le site en ligne** | <http://tfe-web.herokuapp.com/> |
| **Access** : | comme utilisateur  email: [user@sportclub.be](mailto:user@sportclub.be)  password: useruser |
|  | comme editeur  email: [actor@sportclub.be](mailto:actor@sportclub.be)  password: actoractor |
|  | - comme admin  email: [admin@sportclub.be](mailto:admin@sportclub.be)  password: adminadmin |
| **Documentations** : |  |
| **Laravel** : | laravel.com |
| **Vuejs**: | https://v3.vuejs.org/guide/introduction.html#getting-started |
| **Material UI** : | https://vuematerial.io/ |
| **Stackoverflow :** | **https://stackoverflow.com/** |
| **Github :** | https://github.com/ |
| **Mysql** : | https://www.mysql.com/ |
| **MDN :** | - https://developer.mozilla.org/en-US/ |
| **THANKS** . |  |