

Algobreizh Doc

PROJET PHP – BTS SIO

JEULAND Lucas
BTS SIO 2021

Table des matières

Algobreizh site e-commerce – Documentation	1
Contexte	1
Comptes.....	1
Fonctionnalités.....	2
Développement	5
Docker	7
Base de données	9
Git.....	10
Justification des choix	10

Algobreizh site e-commerce – Documentation



Figure 1 - Page d'accueil du site e-commerce

Contexte

AlgoBreizh est basée à Lampaul-Plouarzel face à la Mer d'Iroise, l'un des plus grands champs d'algues d'Europe. Les produits que nous fabriquons sont vendus sous la marque « Plein Ouest ». Contrairement à ce que l'on peut observer en Asie, en Europe le marché de l'algue alimentaire est un marché de niche. Les progressions de ces dernières années montrent que l'algue gagne du terrain. Depuis plus de 15 ans, toute l'équipe d'AlgoBreizh développe et fabrique une gamme originale de produits alimentaires à base d'algues pour le bonheur des chefs et des particuliers amateurs de nouvelles saveurs.

Comptes

Compte client exemple sur le site :

- Email : howling.wolf@gmail.com
- MDP : qsdfgh

Compte Admin sur le site :

- Email : prod@algobreizh.com
- MDP : administrateur

Fonctionnalités

Ce site e-commerce permet aux clients de l'entreprise Algobreizh de créer des commandes et de les visualiser dans leur espace client. Les commandes en attente sont visibles dans un espace « commandes », une fois expédiée par l'entreprise, elles sont disponibles dans l'espace « factures ».



Figure 2 - L'espace client

Un client peut créer un compte en remplissant un formulaire d'inscription en rentrant notamment son adresse mail et un mot de passe. Il peut ensuite se connecter à son espace client en utilisant les identifiants choisis.

Figure 3 - Formulaire d'inscription

Une fois connecté, le client peut visualiser ses commandes en attente de validation, ses factures et créer une nouvelle commande.

Commande n°5

Client Lucas Jeuland
Date de confirmation 2021-03-09 11:36:18

Facturation
M Lucas Jeuland
44 rue des digitales
35000 Rennes, France

Expédition
M Lucas Jeuland
44 rue des digitales
35000 Rennes, France

Produits

Type	Quantité	Prix (€ / kg)	Total
Algue verte	4	10	40€
Algue jaune	4	35	140€

Figure 4 - Un exemple de commande

La page du catalogue permet au client d'ajouter à son panier des produits et de choisir la quantité souhaitée. Le client peut ensuite visualiser ses produits en cliquant sur l'icône du panier dans la barre de navigation.

Nouvelle Commande

Algobreizh 2 Lucas

 Algue verte Prix : 10€ Qté : <input type="text"/> Ajouter	 Algue rouge Prix : 15€ Qté : <input type="text"/> 2 Ajouter	 Algue jaune Prix : 35€ Qté : <input type="text"/> Ajouter
---	---	---

Figure 5 - Page du catalogue de produits

Sur la page du panier, le client peut décider de supprimer des produits du panier, choisir son adresse de livraison, de facturation et valider sa commande.

The screenshot shows a shopping cart page for Algobreizh. At the top, there's a header with the brand name "Algobreizh" and a user icon showing "2" notifications. Below the header, the title "Votre panier" (Your cart) is displayed. A table lists items in the cart:

Type	Quantité	Prix (€ / kg)	Total
Algue rouge	3	15	45€
Algue jaune	4	35	140€
Total	-	-	185€

Next to each item in the table are two green "Supprimer" (Delete) buttons. Below the table, there are sections for "Expédition" (Delivery) and "Facturation" (Billing), both with dropdown menus set to "Rennes, 44 rue des digitales". A large green button at the bottom center says "Valider la commande" (Validate the order).

Figure 6 - La page du panier

Une fois validée, sa commande est visible dans son espace client.

Un compte administrateur permet aux équipes d'Algobreizh de visualiser toutes les commandes en attente et de les valider pour les expédier aux clients. Une fois la commande expédiée par Algobreizh, elle supprimée de l'espace « commandes » du client et ajoutée à l'espace « factures ».

The screenshot shows the Algobreizh admin dashboard. At the top, there's a header with the brand name "Algobreizh" and a user icon showing "0" notifications. Below the header, the title "Commandes à valider" (Pending orders) is displayed. To its right is a list of ten pending orders, each with a link:

- [Commande #2](#)
- [Commande #3](#)
- [Commande #5](#)
- [Commande #6](#)
- [Commande #7](#)
- [Commande #8](#)
- [Commande #10](#)

At the bottom of the page, a footer bar contains the text "© Algobreizh 2021".

Figure 7 - L'espace admin permet de visualiser toutes les commandes à valider

Facturation		Expédition	
M Lucas Jeuland		M Lucas Jeuland	
44 rue des digitales		44 rue des digitales	
35000 Rennes, France		35000 Rennes, France	

Produits

Type	Quantité	Prix (€ / kg)	Total
Algue verte	5	10	50€
Algue jaune	1	35	35€
Total	-	-	85€

Expédier

© Algobreizh 2021

Figure 8 - L'administrateur clique sur "Expédier" quand la commande est prête à être envoyée

Développement

Le site e-commerce est développé en PHP majoritairement pour le back-end et comporte également du JavaScript pour la partie front-end.

Le projet est organisé selon un framework MVC vu en cours. Il comporte du PHP 7 avec uniquement un plugin xdebug pour faciliter le debug.

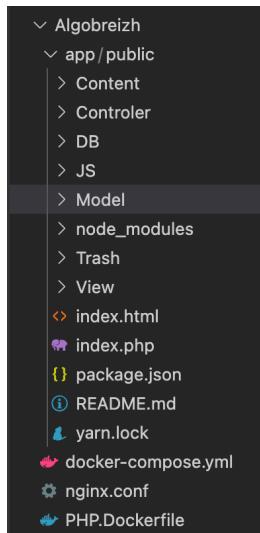


Figure 9 - L'arborescence du projet

- Content contient les différents fichiers CSS et autres ressources telles que les images et icônes utilisées.
- Les fichiers du dossier Model sont des objets comportant les méthodes permettant d'interagir avec la base de données.
- Dans le dossier Controler, les fichiers PHP traitent les données récupérées par les objets du dossier Model afin de les utiliser dans les différentes vues.
- View contient les fichiers PHP qui mettent en forme les données des « controlers ».
- Tous les scripts JS sont contenus dans le dossier JS.

Le projet comporte du JavaScript vanilla ainsi que les bibliothèques jQuery pour simplifier certaines syntaxes et Swup.js pour charger le contenu des pages en AJAX et ainsi ajouter des transitions entre les pages, améliorer les performances du site et la fluidité.

- Documentation jQuery : <https://learn.jquery.com/>
- Documentation Swup.js : <https://swup.js.org/getting-started>

Le site comprend un fichier « template.php » avec un header et un footer presque entièrement statique, la bibliothèque Swup charge en AJAX le contenu dynamique de chaque page dans la balise « Content » :

```
21 |     <body>
22 |         <header>
23 |             <nav>
24 |                 <div class="logo">
25 |                     <a href="index.php"><h2 id="logo-h2">Algobreizh</h2></a>
26 |                 </div>
27 |                 <div class="nav-bar">
28 |                     <?= $navBar ?>
29 |                 </div>
30 |             </nav>
31 |             <hr>
32 |
33 |         </header>
34 |         <div id="content" class="transition-fade">
35 |             <?= $content ?>
36 |         </div> <!-- #content -->
37 |
38 |         <footer>
39 |             <hr>
40 |             <h2 id="copyright">&copy; Algobreizh 2021</h2>
41 |         </footer>
```

Figure 10 - Extrait du fichier "View/template.php", on distingue le header, le footer et le contenu dynamique qui sera chargé dans la balise <div> avec l'id "content"

La variable \$content est définie dans les différents « controlers » et récupérée dans le fichier « View/View.php », elle contient un ou plusieurs arrays PHP qui contiennent eux-mêmes toutes les données nécessaires à la génération des différentes vues du site.

```
28
29     public function orderList($email) {
30         $vue = new View("orderList");
31         $orders = $this->order->getOrders[$this->user->getUserID($email)];
32         $vue->generate(array('orders' => $orders, 'orderType' => 'Commande'));
33     }
34
35     public function billList($email) {
36         $vue = new View("orderList");
37         $orders = $this->order->getBills[$this->user->getUserID($email)];
38         $vue->generate(array('orders' => $orders, 'orderType' => 'Facture'));
39     }
40 }
```

Figure 11 - Extrait du fichier "Controller/OrderController.php", dans les méthodes affichées, on définit le nom de la vue à afficher, puis on récupère les données à partir des modèles, enfin, la fonction generate() du fichier « View/View.php » permet tout simplement de générer la vue désirée.

Dans l'exemple de la figure 11, les deux méthodes permettent d'afficher la liste des commandes d'un client ou la liste de ses factures. Elles utilisent le modèle « Model/Order.php » pour récupérer des informations spécifiques aux commandes, puis génèrent la vue avec les données récupérées.

```
24     public function getOrders($user_id) {
25         $sql = "SELECT order_id, confirm_date FROM orders WHERE user_id = '{$user_id}' and status = 'pending'";
26         $orders = $this->executeRequest($sql);
27         return $orders;
28     }
```

Figure 12 - Extrait du fichier "Model/Order.php", la méthode getOrders(), effectue une requête SQL pour récupérer l'ensemble des commandes en attente d'un client en fonction de son ID utilisateur

Docker

La technologie de conteneurisation Docker est utilisée pour l'environnement de développement afin de simplifier la mise en production. Un docker compose a été utilisé avec des conteneurs individuels pour le serveur web Nginx, PHP, MariaDB, et PhpMyAdmin.

```
BlogMVC > docker-compose.yml
1  version: '3'
2  services:
3      web:
4          image: nginx:latest
5          ports:
6              - "80:80"
7          volumes:
8              - ./nginx.conf:/etc/nginx/conf.d/nginx.conf
9              - ./app:/app
10     php:
11         build:
12             context: .
13             dockerfile: PHP.Dockerfile
14         volumes:
15             - ./app:/app
16     mysql:
17         image: mariadb:latest
18         environment:
19             MYSQL_ROOT_PASSWORD: 'root'
20             MYSQL_USER: 'algobreizh'
21             MYSQL_PASSWORD: 'algobreizh'
22             MYSQL_DATABASE: 'algobreizh'
23         volumes:
24             - mysqldata:/var/lib/mysql
25         ports:
26             - 3306:3306
27     phpmyadmin:
28         links:
29             - mysql:mysql
30         image: phpmyadmin
31         container_name: phpmyadmin
32
```

Figure 13 - Extrait du fichier "docker-compose.yml"

Le fichier docker compose permet de gérer tous les conteneurs individuels et de les faire communiquer. Par exemple, le conteneur « mysql » est « linké » au conteneur « phpmyadmin », on pourra donc utiliser les utilisateurs définis dans le conteneur « mysql » dans PhpMyAdmin.

Le Dockerfile pour PHP récupère une variante fpm de l'image officielle de PHP et installe les extensions PDO et Xdebug.

La configuration du serveur Nginx est définie dans le fichier « nginx.conf » à la racine du projet.

Pour plus d'information sur Docker et son fonctionnement, la documentation officielle :
<https://docs.docker.com/>

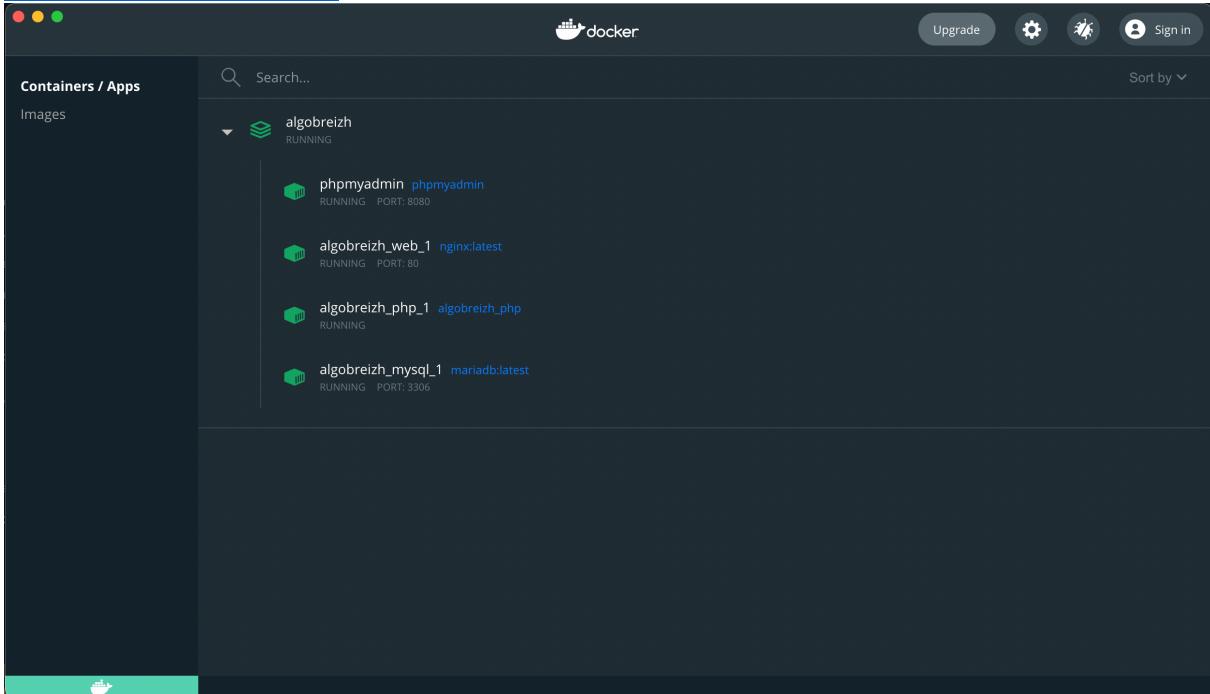


Figure 14 - Un aperçu du Dashboard Docker avec tous les conteneurs

Base de données

L'adresse pour se connecter à PhpMyAdmin : <https://mysql2.lwspanel.com/myadmin4>

Utilisateur : lucas1515870

MDP : 2pwwgz8dkk

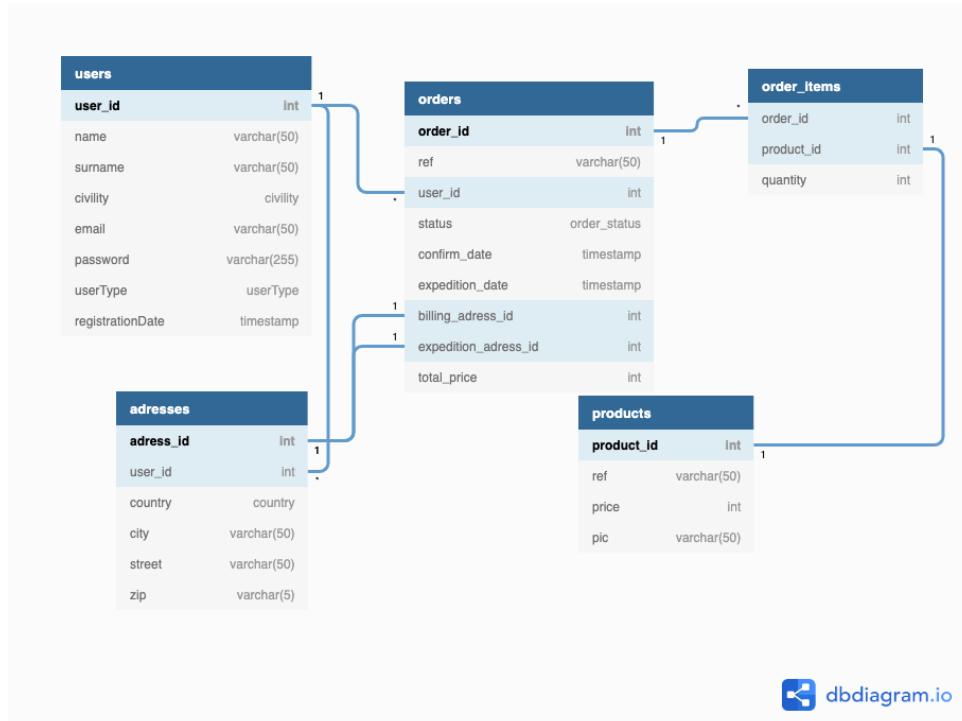


Figure 15 - Le MLD de la base de données algobreizh

```

Algobreizh > app > public > DB > Algobreizh DB (3).sql
1 CREATE TABLE `users` (
2     `user_id` int PRIMARY KEY AUTO_INCREMENT,
3     `name` varchar(50),
4     `surname` varchar(50),
5     `civility` ENUM ('M', 'Mme') DEFAULT "M",
6     `email` varchar(50),
7     `password` varchar(255),
8     `userType` ENUM ('customer', 'employee') DEFAULT "customer",
9     `registrationDate` timestamp DEFAULT (now())
10 );
11
12 CREATE TABLE `orders` (
13     `order_id` int PRIMARY KEY AUTO_INCREMENT,
14     `ref` varchar(50),
15     `user_id` int,
16     `status` ENUM ('pending', 'done') DEFAULT "pending",
17     `confirm_date` timestamp DEFAULT (now()),
18     `expedition_date` timestamp,
19     `billing_adress_id` int,
20     `expedition_adress_id` int,
21     `total_price` int
22 );
23
24 CREATE TABLE `order_items` (
25     `order_id` int,
26     `product_id` int,
27     `quantity` int
28 );
29
30 CREATE TABLE `products` (
31     `product_id` int PRIMARY KEY AUTO_INCREMENT,

```

Figure 16 - Extrait du fichier "DB/Algobreizh DB.sql" qui permet de créer la BDD

Git

Le projet utilise l'outil de versioning git et est hébergé publiquement sur Github : <https://github.com/Ljld/Algobreizh-ljld>

Justification des choix

Le langage PHP était imposé par le cahier des charges et l'envergure du projet nécessitait de définir une architecture structurée. J'ai choisi d'utiliser le MVC car j'y suis habitué, vu en cours, il est beaucoup utilisé en entreprise et il est à la base de nombreux frameworks très utilisés comme Symfony par exemple.

Docker et la conteneurisation permet de développer sur un environnement très proche d'un environnement de production et ainsi limiter les risques de bugs lors de la mise en production comme cela peut arriver en utilisant des environnements de développement comme WAMP par exemple. Il est également très utilisé en entreprise aujourd'hui car il permet de simplifier les tests et de faciliter l'intégration continue.

J'ai utilisé Nginx au lieu d'Apache qui est également fonctionnel. Nginx est plus moderne et à tendance à remplacer de plus en plus Apache dans l'industrie.

La bibliothèque Swup.js permet de fluidifier la navigation sur le site en chargeant le contenu plus rapidement et en ajoutant des transitions entre les pages. Elle permet de simuler une Single Page App, de plus en plus présente sur le marché.