

Nans LAFITTE
Adam GHAJDAOUI
Nicolas WEBER

Client Léger de Fast Sushi - Click & Collect

Projet en groupe pour BTS SIO SLAM
Année Scolaire : 2024-2025

Sommaire

Cahier des charges FastSushi.....	3
Spécification.....	4
Client léger.....	4
1. Objectif du site.....	4
2. Navigation (Header).....	4
3. Page d'accueil.....	4
4. Page des Plats Fixes.....	5
5. Page des Plats Composés.....	5
6. Panier.....	6
7. Gestion de compte.....	7
Partie Technique.....	8
Planification & Organisation.....	8
Base de données.....	8
Environnement de travail local avec Docker.....	8
Architecture MVC avec Laravel.....	8
Annexes.....	9
A1. Use Case - Client Léger.....	9
A2. Planning - Client Léger.....	10
A3. Modélisation BDD (MCD + MLD).....	11
A4. Arborescence.....	12
A6. produitsController.php.....	18
A7. produits.php.....	19
A8. web.php.....	20
A9. create_table_produits.php (migration + insert).....	21
A10. produitsCompose.blade.php.....	23
A11. produitsCompose.js.....	27
A12. Ingredients.php.....	29

Cahier des charges FastSushi

L'entreprise FastSushi est spécialisée dans la restauration rapide, et principalement dans les produits alimentaires d'inspiration japonaise tels que les sushis, les makis, les sashimis, les springs rolls, ... La société qui a vu le jour en 2008, et se spécialise pour faire découvrir à sa clientèle la vraie cuisine japonaise à base de produits locaux et de qualités. Le restaurant garantit que tous ses poissons sont traités entiers et sélectionnés suivant les arrivages pour conserver un maximum de fraîcheur. Les produits sont certifiés AB et Label rouge. Le gérant de l'entreprise est M. Makytori

Expressions des besoins :

La société FastSushi veut réaliser un site pour faire du click & collect auprès de sa clientèle. De plus son offre permet la possibilité de composer soit même les makis, california rolls et springs rolls (à partir des listes d'ingrédients fournies), ces recettes spéciales sont appelées recettes composables, elles peuvent être constituées de 2, 3 ou 4 ingrédients. Afin de simplifier la réalisation et le traitement, il vous ai demandé de réaliser deux applications qui permettent la création et le suivi des recettes, ventes, et collecte des produits. L'application backoffice sera utilisée par les administrateurs et ses préparateurs, tandis que le frontoffice sera utilisé par les clients pour réaliser leur panier et passer leur commande de produits. Vous réaliserez pour répondre à ce besoin deux applications (client lourd , client léger) :

Application FastSushi FrontOffice - Client léger :

- Les utilisateurs peuvent créer leurs comptes eux-mêmes sur le site avant de commander (nom, téléphone, adresse email, mot de passe).
- L'utilisateur peut naviguer grâce à un menu vers les différentes catégories.
- Lorsqu'un utilisateur choisit une recette composable, une fenêtre s'ouvre pour lui permettre de choisir la composition.
- Chaque fois qu'un produit est ajouté (via un bouton acheté), il est ajouté dans le panier de l'utilisateur qui recense les différents composants de sa commande.
- L'utilisateur peut consulter son panier via un bouton en forme de panier en haut de la page, cet écran recense donc le contenu du panier avec le prix total à payer en bas de la page.
- Sur l'écran du panier, un bouton permet de valider la commande. C'est l'appui sur ce bouton qui passe la commande en statut à préparer par FastSushi.
- Il n'est pas nécessaire de gérer les moyens de paiement, ceux-ci se faisant au retrait en boutique.

Application FastSushi BackOffice - Client Lourd :

- Les administrateurs/préparateur disposent d'un identifiant et d'un mot de passe pour se connecter à l'application qui sera installée sur son poste de travail et communique avec une base de données distantes (le plus sécurisé possible).
- Les administrateurs/préparateur s'occupent d'administrer la disponibilité des produits.
- Les préparateurs disposent d'un écran facilement accessible qui leur indique les commandes en cours qu'il faut préparer
- A partir de l'écran précédent, ils peuvent valider la remise de la commande au client quand celui-ci l'a récupéré en boutique.
- Les administrateurs ont la possibilité d'accéder aux historiques des commandes.

Spécification

Client léger

1. Objectif du site

Ce site vise à permettre aux clients de commander facilement des plats japonais (sushis, soupes, brochettes, etc.) via une interface intuitive, avec retrait en Click & Collect au restaurant.

2. Navigation (Header)

Un en-tête fixé en haut de chaque page proposera les pages ci-dessous:

- Accueil (Home)
- Plat Fixe
- Plat Composé
- Gestion de compte



3. Page d'accueil

La page d'accueil affichera en premier des photos avec des phrases donnant envie de commander ainsi que deux boutons principaux permettant d'accéder directement aux sections de commande :

- Accéder aux Plats Fixes
- Composer mon plat

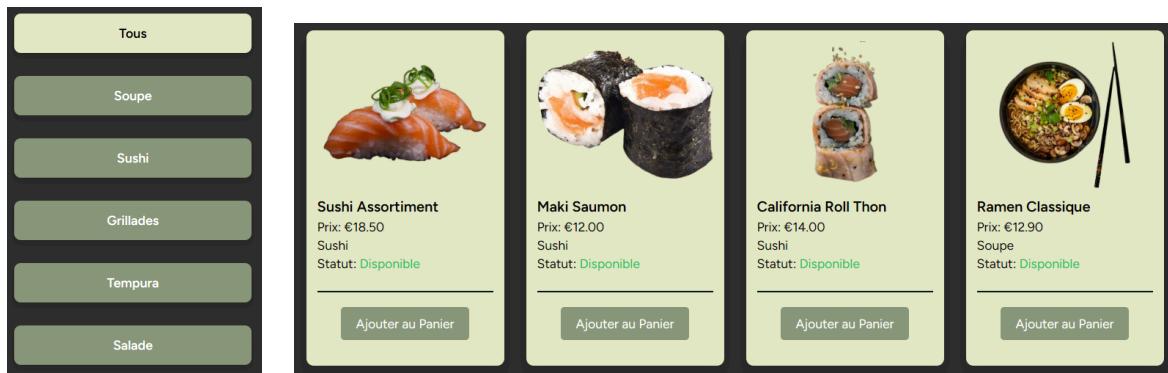


4. Page des Plats Fixes

Cette page permet aux utilisateurs de choisir rapidement parmi les plats pré-définis du restaurant en accédant soit dans l'en-tête avec un bouton "Notre carte" ou avec un bouton de la page d'accueil.

Fonctionnalités principales :

- Liste des produits affichée sous forme de cartes comprenant photo, nom ,description et prix.
- Possibilité d'ajouter au panier pour chaque produit via le bouton "Ajouter au Panier"
- Système de filtrage par catégorie Soupe, Sushis, Grillades, Tempura et Salade.

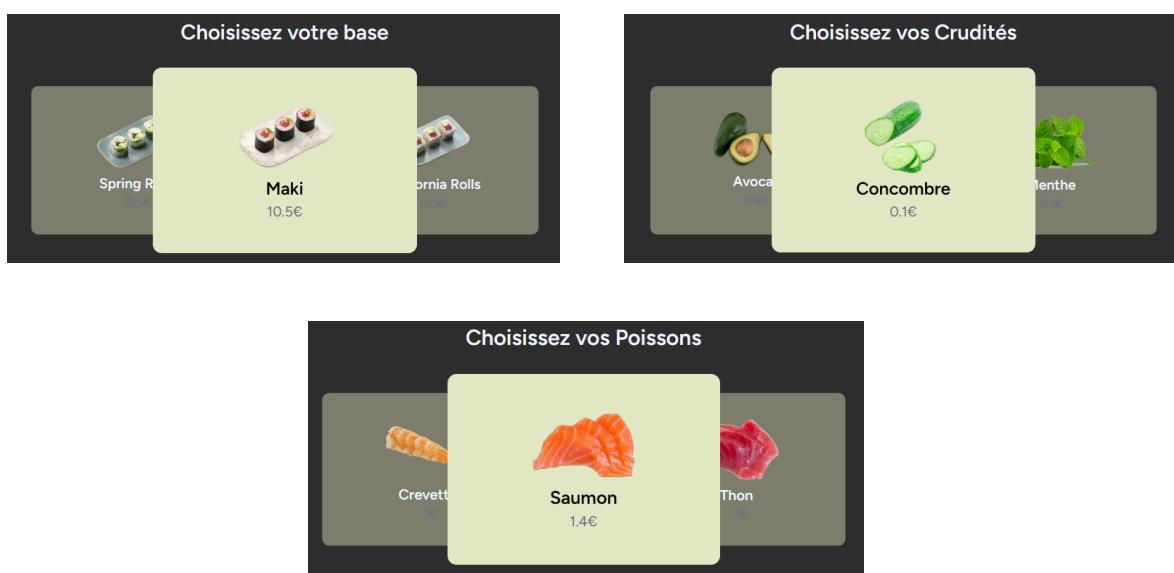


5. Page des Plats Composés

Cette page permet aux utilisateurs de composer leur propre plat à partir de différents ingrédients en accédant soit dans l'en-tête avec un bouton "Votre plat à composer" ou avec un bouton de la page d'accueil.

Structure en carrousel par étapes successives :

- **Choix de la base** : Maki, Spring Rolls, California Rolls
- **Choix des crudités** : Concombre, Menthe, Coriandre, Chou rouge, Salade, Poivron, Radis, Carotte, Mangue, Avocat
- **Choix des Poissons** : Daurade, Crevettes, Saumon, Thon, Anguille, Crabe, Surimi



Proposition des suppléments ayant la possibilité d'ajouter en cliquant sur le bouton + : Boursin, Nori, Fromage, Mayonnaise japonaise, Sésame, Ciboulette, Feuille de riz



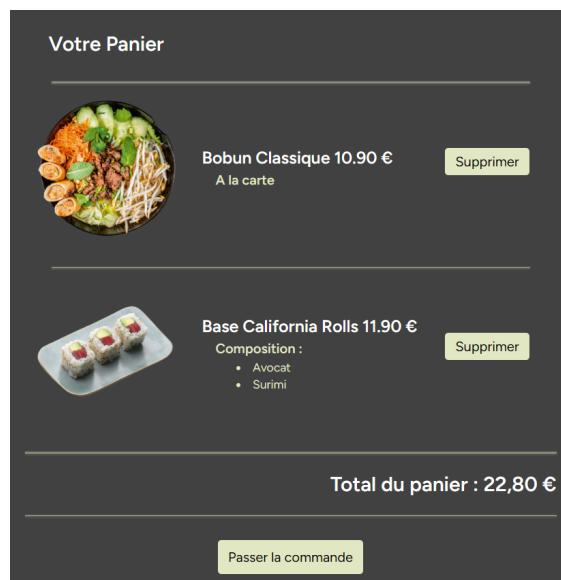
Chaque ingrédient affichera son nom, une photo et un prix. Un bouton "Ajouter au panier" permettra de finaliser la composition.

6. Panier

Page de récapitulatif de la commande.

Contenu du panier :

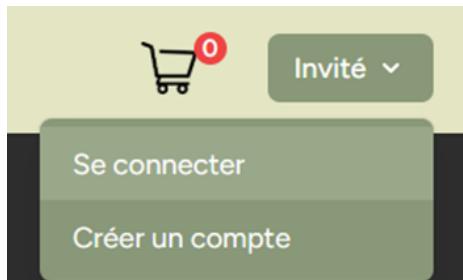
- Liste des produits (nom, image, prix)
- Possibilité pour supprimer un ou plusieurs produits
- Affichage du prix total
- Bouton "Passer la commande" redirigeant vers la page de validation de commande



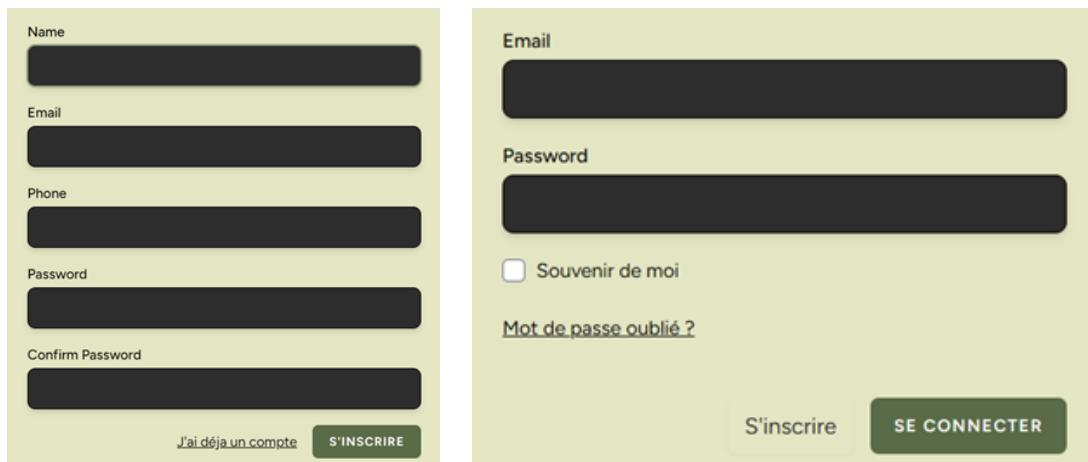
7. Gestion de compte

Fonctionnalités disponibles :

- Inscription / Connexion



- Formulaire d'inscription et de connexion



Name
Email
Phone
Password
Confirm Password

J'ai déjà un compte S'INSCRIRE

Email
Password

Souvenir de moi
[Mot de passe oublié ?](#)

S'inscrire SE CONNECTER

- Voir l'historique de ses commandes



John ▾

Mes Commandes
Profil
Déconnexion

Vos Commandes

Commande	Date	Statut	ID
Commande du 01/04/2025 - Cloturé - 31.6€	01/04/2025	Cloturé	3Q1Y8NZSZ9 ▾
Commande du 10/04/2025 - en cours - 30.8€	10/04/2025	en cours	NGF7HIX1XF ▾
Commande du 12/04/2025 - En attente de récupération - 18.5€	12/04/2025	En attente de récupération	UAR7KMUVRT ▾

Partie Technique

Planification & Organisation

- Utilisation de **Excel** pour planifier les tâches, suivre l'avancement, et attribuer les responsabilités.
 - Utilisation de **Discord** pour la communication entre les collaborateurs.
 - Versioning avec **GitHub**, permettant le travail collaboratif et la gestion de branches.
-

Base de données

- [Planification du schéma](#) de la base de données : tables `users`, `produits`, `commandes`, `panier`, etc.
 - Mise en place des **migrations Laravel** pour créer les tables.
 - phpMyAdmin une manipulation facilitée.
-

Environnement de travail local avec Docker

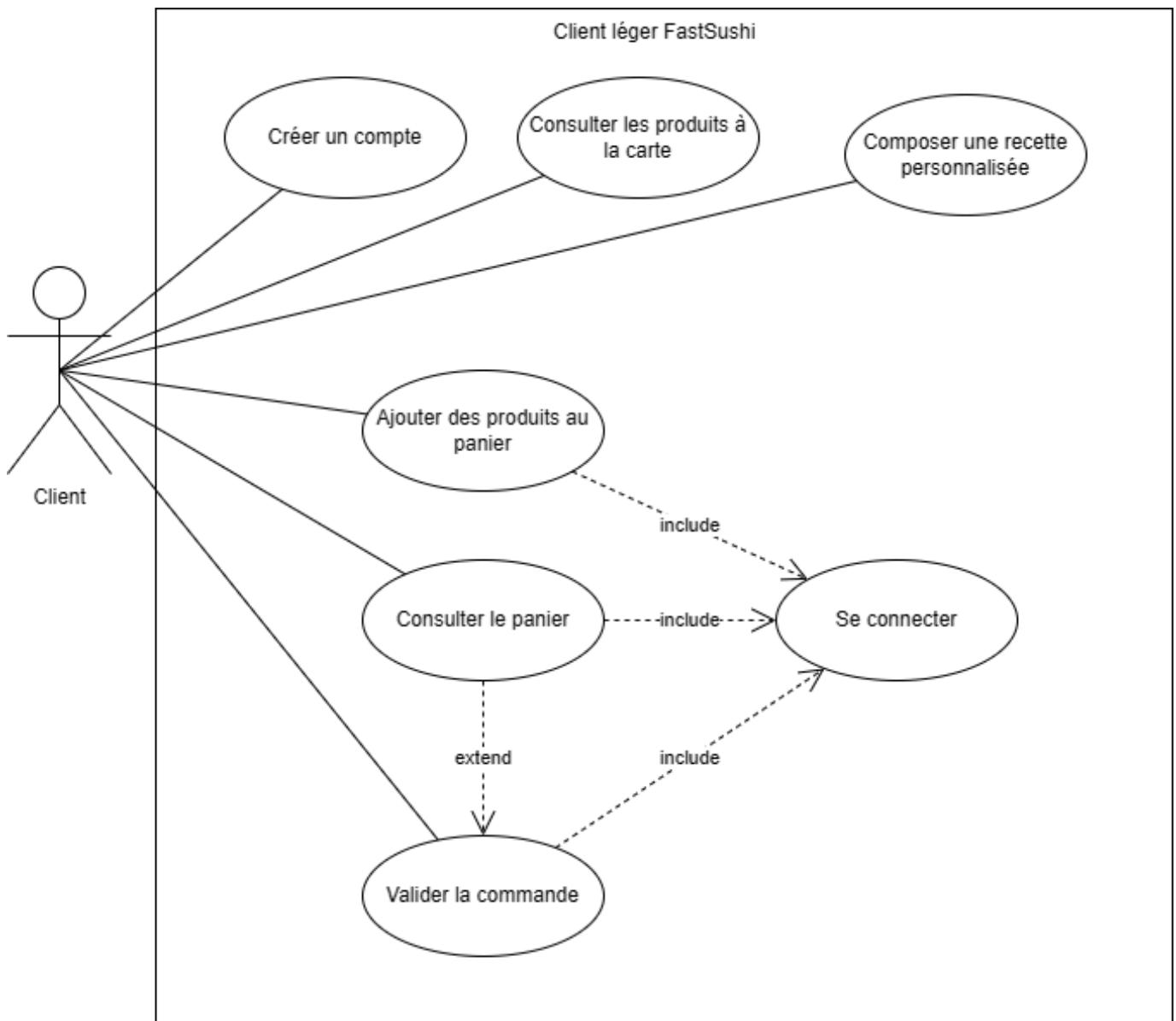
- Installation de Laravel dans un dossier monté sur un conteneur de serveur web utilisant une image PHP-FPM (port 8000).
 - Base de données gérée par un conteneur MySQL (port 3306).
 - phpMyAdmin accessible via un conteneur (port 8080).
-

Architecture MVC avec Laravel

- Création de routes web pour le client léger (site web).
- Modèles (Models) PHP représentant les tables de la base de données.
- Conception des Vues Blade pour afficher dynamiquement les informations aux utilisateurs.
- Développement des contrôleurs (Controllers) pour gérer l'injection et la manipulation des données (plats, panier, compte, etc.).

Annexes

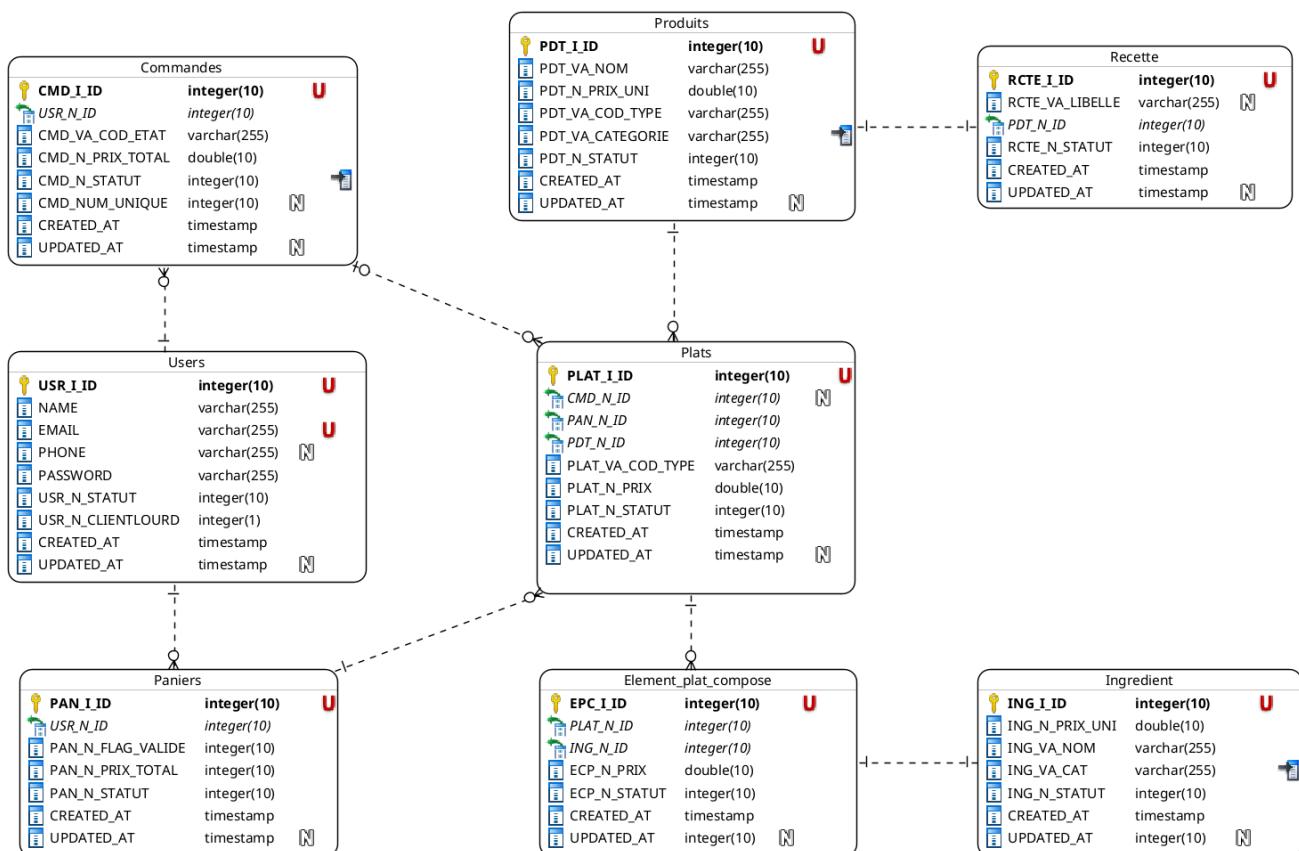
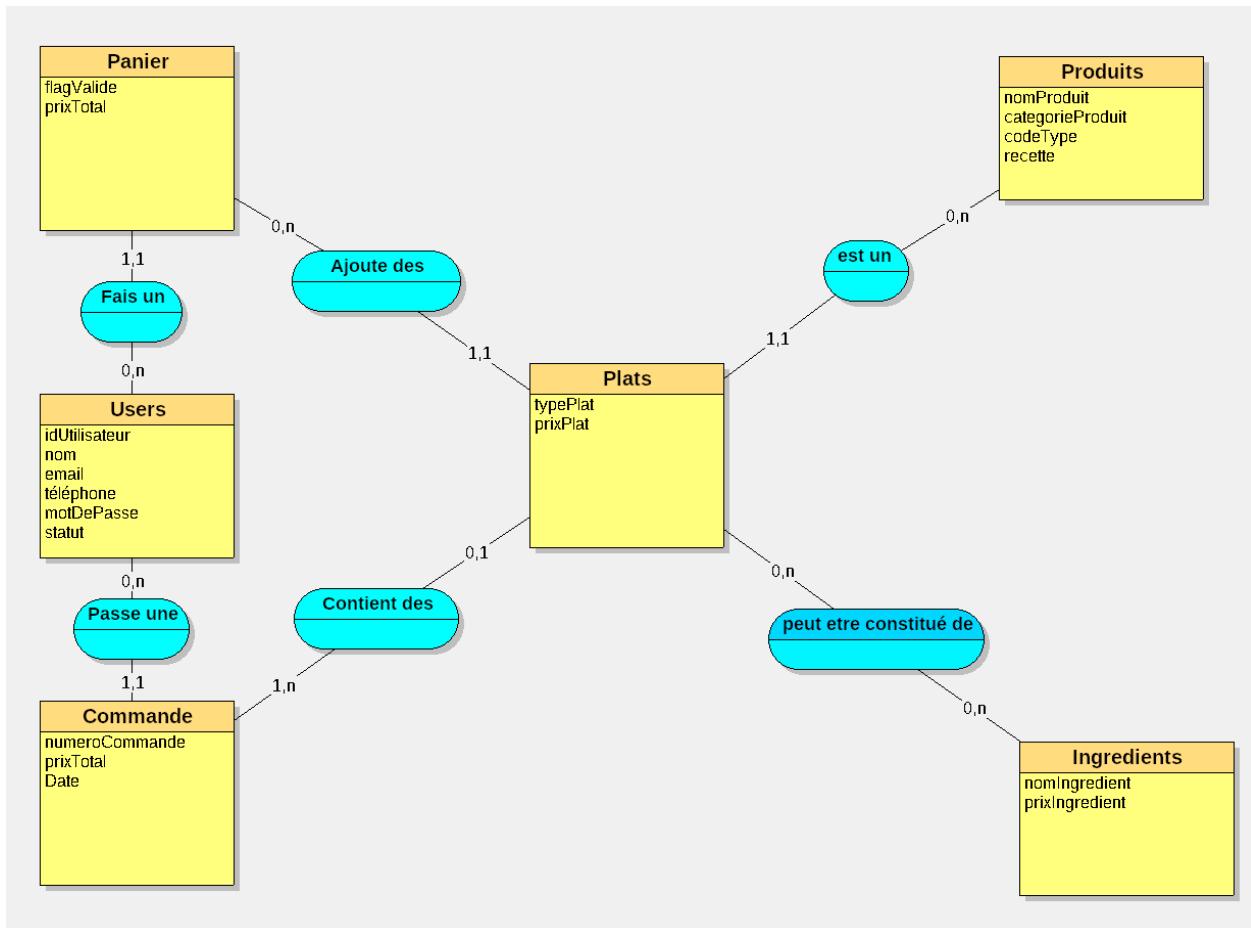
A1. Use Case - Client Léger



A2. Planning - Client Léger

Tâche	Sous-tâches	Description	Durée estimée
Docker	Créer fichier docker-compose.yml + Dockerfile	Containers : - serveur WEB (image PHP FPM + serveur vite pour CSS, JS etc...) - mySQL (SGBD choisi) - PHPmyAdmin (Facilité la manipulation de la BDD)	0,5 J
GitHub	Créer Repository distant : CLIENT LEGER	Créer repo, Ajouter collaborateurs, Créer les branches	0,5 J
	Lien avec le dossier local	Lien avec le dossier local	0,5 J
Modélisation	UseCase	Conception de la vision globale du système	0,5 J
	MCD / MLD	Structure BDD	1 J
BDD	Migration Laravel	Création des tables, insertion des valeurs, déclaration des relations	3 J
	Faire les models	Pour chaque table, créer un model adapté	1 J
Gestion du compte	Breeze	Gestion des utilisateurs, connexion, création de compte	0,5 J
Header	Barre de navigation	Panier, Page produitFixe, Page Produit Composer, logo	2 J
	Gestion de compte	Créer compte, Connecter, modifier compte	2 J
Produit fixe	Vue Produit	Affichages des produits fixes, Système de filtrage, Ajout de panier	4 J
	Controller	Récupération des données (pdt_va_cod_type = PF)	0,5 J
Produit Composer	Vue Produit	Carrousel, affichage des ingrédients et suppléments	4 J
	Controller	Récupération des données (pdt_va_cod_type = PC)	0,5 J
Accueil	Vue	Carrousel, Bouton de redirection	4 J
	Controller	Afficher la vue "home.blade.php"	0,5 J
Panier	Vue	Liste des produits (fixe ou composer) ajouté, bouton de paiement	3 J
	Controller + services	Gestion du panier	3 J
Commande	Vue	Lister l'historique des commandes et voir leurs détails	2 J
	Controller	Récupération des commandes du client	0,5 J

A3. Modélisation BDD (MCD + MLD)



A4. Arborescence

```
app
  └── Http
      ├── Controllers
      │   ├── Auth
      │   │   ├── AuthController.php
      │   │   ├── CommandesController.php
      │   │   ├── Controller.php
      │   │   ├── DisplayController.php
      │   │   ├── FormController.php
      │   │   ├── HardRequestController.php
      │   │   ├── HelloController.php
      │   │   ├── HomeController.php
      │   │   ├── PaniersController.php
      │   │   ├── ProduitsController.php
      │   │   └── ProfileController.php
      │   ...
      └── Models
          ├── Commandes.php
          ├── Elements_plat_compose.php
          ├── Ingredients.php
          ├── Paniers.php
          ├── Plats.php
          ├── Produits.php
          ├── User.php
          └── UserData.php
      └── Providers
          ├── AppServiceProvider.php
          ├── AuthServiceProvider.php
          ├── BroadcastServiceProvider.php
          ├── EventServiceProvider.php
          └── RouteServiceProvider.php
      └── Services
          └── PanierService.php
      ...
  ...
  └── database
      ├── ...
      └── migrations
          ├── 2014_10_12_000000_create_users_table.php
          ├── 2014_10_12_100000_create_password_reset_tokens_table.php
          ├── 2019_08_19_000000_create_failed_jobs_table.php
          ├── 2019_12_14_000001_create_personal_access_tokens_table.php
          ├── 2025_01_14_084726_add_phone_to_user_model.php
          ├── 2025_01_17_133351_create_table_commandes.php
          ├── 2025_01_17_150640_create_table_produits.php
          └── 2025_01_17_151204_create_table_recettes.php
```

```
    └── 2025_01_17_151527_create_table__plats.php
    └── 2025_01_28_170636_create_table_ingredients.php
    └── 2025_01_28_170640_create_table_element_plat_compose.php
    └── 2025_01_28_172928_insert_data_produits.php
    └── 2025_01_29_083335_create_table__paniers.php
    └── 2025_01_29_151943_insert_data_ingredient.php
    └── 2025_04_15_081356_insert_data_recettes.php
    ...
...
    └── resources
        ├── css
        │   ├── app.css
        │   ├── home.css
        │   ├── produitsCompose.css
        │   ├── produitsFixe.css
        │   ├── style.css
        │   └── test.scss
        ├── js
        │   ├── app.js
        │   ├── bootstrap.js
        │   ├── home.js
        │   ├── notifAjoutPanier.js
        │   ├── produitsCompose.js
        │   └── produitsFixe.js
        └── views
            ├── CommandeSuccess.blade.php
            ├── Commandes.blade.php
            ├── HardRequest.blade.php
            ├── auth
            ├── components
            ├── dashboard.blade.php
            ├── display.blade.php
            ├── displayAll.blade.php
            ├── form.blade.php
            ├── hello.blade.php
            ├── home.blade.php
            ├── layouts
            ├── panier.blade.php
            ├── produitsCompose.blade.php
            ├── produitsFixe.blade.php
            ├── profile
            └── welcome.blade.php
            ...
    └── routes
        ├── api.php
        ├── auth.php
        ├── channels.php
        ├── console.php
        └── web.php
```

A5. produitFixe.blade.php

```
<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Sushi Click & Collect</title>

    @vite(['resources/js/produitsFixe.js', 'resources/css/produitsFixe.css'])

    <script src="https://cdn.tailwindcss.com"></script>

</head>

<x-app-layout>

    <main class="container mx-auto px-4">
        <!-- Conteneur principal avec Flexbox -->
        <div class="flex flex justify-center">
            <!-- Filtres à gauche -->
            <div class="w-1/4 bg-black-700 p-9 ml-16 mt-16 rounded-lg">
                <div class="flex flex-col gap-7">
                    <button
                        class="delete-article filter-btn bg-[#899878] text-white px-5 py-3 rounded-lg font-semibold shadow-md transition-all duration-300 hover:bg-[#9AA88A] active:scale-95"
                        data-filter="all">
                        Tous
                    </button>
                    <button
                        class="filter-btn bg-[#899878] text-white px-5 py-3 rounded-lg font-semibold shadow-md transition-all duration-300 hover:background-[#9AA88A] active:scale-95"
                        data-filter="soupe">
                        Soupe
                    </button>
                    <button
                        class="filter-btn bg-[#899878] text-white px-5 py-3 rounded-lg font-semibold shadow-md transition-all duration-300 hover:background-[#9AA88A] active:scale-95"
                        data-filter="sushi">
                        Sushi
                    </button>
                </div>
            </div>
        </div>
    </main>
</x-app-layout>
```

```

        <button
            class="filter-btn bg-[#899878] text-white px-5 py-3
rounded-lg font-semibold shadow-md transition-all duration-300
hover: hover: bg-[#9AA88A] active: scale-95"
            data-filter="grillades">
            Grillades
        </button>
        <button
            class="filter-btn bg-[#899878] text-white px-5 py-3
rounded-lg font-semibold shadow-md transition-all duration-300
hover: hover: bg-[#9AA88A] active: scale-95"
            data-filter="tempura">
            Tempura
        </button>
        <button
            class="filter-btn bg-[#899878] text-white px-5 py-3
rounded-lg font-semibold shadow-md transition-all duration-300
hover: hover: bg-[#9AA88A] active: scale-95"
            data-filter="salade">
            Salade
        </button>
    </div>
</div>

<!-- Produits à droite -->

<div id="produitsContainer"
    class="mt-[3vw] flex flex-wrap gap-6 w-3/4 flex justify-center
bg-black-500 rounded max-h-[80vh] overflow-auto ">

    @foreach($produits as $categorie => $items)

        @foreach($items as $produit)

            <div class="product-card hidden bg-[#E4E6C3] shadow-lg
rounded-lg p-3 flex-2 my-8 w-[216px]" style="height: fit-content;" data-category="{{ strtolower($categorie) }}>

                <div class="w-48 h-40 overflow-hidden">
                    
                </div>

```

```

        <h5 class="text-md font-semibold mt-2">{{ $produit->pdt_va_nom }}</h5>

        <p class="text-white-700 text-sm">Prix: €{{ number_format($produit->pdt_n_prix_uni, 2) }}</p>

        <p class="text-white-500 text-sm">{{ $categorie }}</p>

        <!-- <p class="text-white-500 text-sm ">{{ $produit->rcte_va_libelle }}</p>-->

        <p class="text-white-500 text-sm">Statut:<br>
            <span
                class="{{ $produit->pdt_n_statut == 1 ? 'text-green-500' : 'text-red-500' }}>
                    {{ $produit->pdt_n_statut == 1 ? 'Disponible' : 'Indisponible' }}
                </span>
            </p>
            <hr class="mb-2 mt-5 border border-[#021417]">
            <div class="flex justify-center align-center">
                @if (Auth::check())
                    <form action="{{ route('panier.ajouterProduitFixeAuPanier') }}" method="POST">
                        @csrf
                        <input type="hidden" name="idProduit" value="{{ $produit->pdt_i_id }}>
                        <button type="submit"
                            class="mt-2 px-4 py-2 rounded text-sm
                                {{ $produit->pdt_n_statut == 1 ? 'bg-[#899987] text-white
                                hover:bg-[#9AA88A] cursor-pointer' :
                                'bg-gray-400 text-white
                                cursor-not-allowed' }}>
                            {{ $produit->pdt_n_statut == 1 ? '' :
                            'disabled' }}>
                            {{ $produit->pdt_n_statut == 1 ?
                            'Ajouter au Panier' : 'Indisponible' }}>
                        </button>
                    </form>
                </div>
            </div>
        </div>
    </div>

```

```
        @else

            <a href="{{ route('login') }}" 
                class="mt-2 bg-[#899878] text-white px-4
py-2 rounded hover:bg-[#9AA88A] text-sm">
                Ajouter au Panier
            </a>

        @endif

    </div>
</div>

        @endforeach
    @endforeach

    </div>
</div>
@if(session('success'))
<div id="customAlert" class="custom-alert">
    <div class="custom-alert-content">
        <h3>Succès !</h3>
        <p>{{ session('success') }}</p>
    </div>
</div>
@vite(['resources/js/notifAjoutPanier.js'])
@endif
</main>
</x-app-layout>
```

A6. produitsController.php

```
<?php

namespace App\Http\Controllers;
use App\Models\Produits;
use App\Models\Ingredients;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class ProduitsController extends Controller
{
    public function afficheProduitsFixe()
    {
        // Requête brute
        $resultats = DB::select("
            SELECT Produits.*, Recettes.rcte_va_libelle
            FROM Produits
            JOIN Recettes ON Produits.pdt_i_id = Recettes.pdt_n_id
            WHERE Produits.pdt_va_cod_type = 'PF'
        ");

        // Transformation en tableau associatif groupé par catégorie
        $produits = [];
        foreach ($resultats as $produit) {
            $categorie = $produit->pdt_va_categorie ?? 'Inconnu';
            $produits[$categorie][] = $produit;
        }

        return view('produitsFixe', compact('produits'));
    }

    public function afficheProduitsCompose()
    {
        // Récupération des bases
        $bases = Produits::where('pdt_va_cod_type', 'PC')->get();

        $ingredients = Ingredients::orderBy('ing_va_cat') // Trie par catégorie
        ->get()
        ->groupBy('ing_va_cat');

        //dd($ingredients);
        return view('produitsCompose', compact('bases', 'ingredients'));
    }
}
```

A7. produits.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Produits extends Model
{
    use HasFactory;

    /**
     * Le nom de la table associée au modèle.
     *
     * @var string
     */
    protected $table = 'Produits';

    /**
     * Indique le nom de la clé primaire.
     *
     * @return string
     */
    public function getKeyName()
    {
        return 'pdt_i_id';
    }

    /**
     * Les attributs qui peuvent être assignés en masse.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'pdt_va_nom',
        'pdt_va_categorie',
        'pdt_va_cod_type',
        'pdt_n_prix_uni',
        'pdt_n_statut',
    ];
}
```

A8. web.php

```
<?php

use App\Http\Controllers\ProfileController;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\ProduitsController;
use App\Http\Controllers\HomeController;
use App\Http\Controllers\PaniersController;
use App\Http\Controllers\CommandesController;

Route::get('/', function () {
    return view('home');
});

Route::get('/home', [HomeController::class, 'afficheHome'])->name('home');
Route::get('/panier', [PaniersController::class,
'AffichePanier'])->name('panier');

Route::post('/panier/supprimer/{platId}', [PaniersController::class,
'supprimerPlat'])->name('panier.supprimer');
Route::post('/ajouter-pf-au-panier', [PaniersController::class,
'ajouterProduitFixeAuPanier'])->name('panier.ajouterProduitFixeAuPanier');
Route::post('/ajouter-pc-au-panier', [PaniersController::class,
'ajouterProduitComposeAuPanier'])->name('panier.ajouterProduitComposeAuPanier');

Route::get('/produits-a-la-carte', [ProduitsController::class,
'afficheProduitsFixe'])->name('produits.afficheProduitsFixe');
Route::get('/produits-a-composer', [ProduitsController::class,
'afficheProduitsCompose'])->name('produits.afficheProduitsCompose');

Route::get('/mes-commandes', [CommandesController::class,
'afficheCommandes])->name('commandes.afficheCommandes');
Route::post('/panier/creerCommande/{panierId}', [CommandesController::class,
'creerCommande'])->name('commandes.creerCommande');

require __DIR__.'/auth.php';

use App\Http\Controllers\HardRequestController;

Route::get('/requete-en-dur', [HardRequestController::class,
'getHardRequestController']);
```

A9. create_table_produits.php (migration + insert)

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('Produits', function (Blueprint $table) {
            $table->id('pdt_i_id');
            $table->string('pdt_va_nom');
            $table->string('pdt_va_categorie');
            $table->string('pdt_va_cod_type');
            $table->float('pdt_n_prix_uni', 8, 2);
            $table->unsignedInteger('pdt_n_statut')->default(1);
            $table->timestamps();
        });
    }

    DB::table('Produits')->insert([
        [
            'pdt_va_nom' => 'Sushi Assortiment',
            'pdt_va_categorie' => 'Sushi',
            'pdt_va_cod_type' => 'PF',
            'pdt_n_prix_uni' => 18.50,
            'pdt_n_statut' => 1,
            'created_at' => now(),
            'updated_at' => now(),
        ]
    ]);
}

}
```

```

$ingredients = [
    'Fromage', 'Concombre', 'Avocat', 'Thon', 'Saumon', 'Crevettes',
'Daurade', 'Mangue', 'Boursin',
    'Carotte', 'Radis', 'Poivron', 'Salade', 'Chou rouge', 'Coriandre',
'Menthe',
    'Anguille', 'Crabe', 'Surimi',
    'Mayonnaise japonaise', 'Sésame', 'Ciboulette', 'Feuille de riz',
'Nori'
];

$prix = [
    '0.20', '0.10', '0.40', '1.70', '1.40', '1.00', '1.50', '0.20',
'0.20',
    '0.10', '0.10', '0.15', '0.10', '0.20', '0.10', '0.10',
    '2.00', '1.80', '1.00',
    '0.30', '0.20', '0.15', '0.25', '0.50'
];

$categories = [
    'Suppléments', 'Crudités', 'Crudités', 'Poissons', 'Poissons',
'Poissons', 'Poissons', 'Crudités', 'Suppléments',
    'Crudités', 'Crudités', 'Crudités', 'Crudités', 'Crudités',
'Crudités', 'Crudités',
    'Poissons', 'Poissons', 'Poissons',
    'Suppléments', 'Suppléments', 'Suppléments', 'Suppléments',
'Suppléments'
];

foreach ($ingredients as $index => $ingredient) {
    DB::table('Ingredients')->insert([
        'ing_i_id' => $index + 1,
        'ing_va_nom' => $ingredient,
        'ing_n_prix_uni' => $prix[$index],
        'ing_va_cat' => $categories[$index],
        'ing_n_statut' => 1, // 1 pour actif, ajustez si nécessaire
        'created_at' => Carbon::now(),
        'updated_at' => Carbon::now(),
    ]);
}

public function down(): void
{
    Schema::dropIfExists('Produits');
}
};

```

A10. produitsCompose.blade.php

```
@vite(['resources/js/produitsCompose.js'])
@vite(['resources/css/produitsCompose.css'])
@vite(['resources/js/notifAjoutPanier.js'])

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Sushi Click & Collect</title>

    <script src="https://cdn.tailwindcss.com"></script>
</head>

<x-app-layout>
    <div class="py-2">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
            <div id="support" class="overflow-hidden sm:rounded-lg p-6">
                <form method="POST" action="{{ route('panier.ajouterProduitComposeAuPanier') }}>
                    @csrf

                    <!-- Choix de la base avec carrousel -->
                    <div class="categorie-container">
                        <h2 class="text-2xl font-bold mb-6">Choisissez votre
base</h2>
                        <div class="base-carousel">

                            @foreach ($bases as $base)
                                <label class="carte-element" >
                                    <!--
                                    

                                    <input type="radio" name="idBase" value="{{ $base->pdt_i_id }}" required class="hidden">
                                <div class="text-center">
```

```

                <span class="block text-xl
font-semibold">{{ $base->pdt_va_nom }}</span>
                <span class="block text-gray-500">{{
$base->pdt_n_prix_uni }}€</span>
            </div>
        </label>
    @endforeach

    </div>
</div>

@foreach ($ingredients as $categorie => $listeIngredients)
<hr class="elegant-line">

@if($categorie != "Suppléments")
    <div class="categorie-container">
        <h2 class="text-2xl font-bold mt-4
mb-6">Choisissez vos {{ $categorie }}</h2>
        <div class="base-carousel" data-category="{{
Str::slug($categorie) }}">
            @foreach ($listeIngredients as $ingredient)

                <label class="carte-element">
                    <!-- 
                    
                    <input type="hidden" name="id{{ ucfirst(Str::slug($categorie)) }}" value="{{ $ingredient->ing_i_id }}">
                    <div class="text-center">
                        <span class="block text-xl
font-semibold">{{ $ingredient->ing_va_nom }}</span>
                        <span class="block
text-gray-500">{{ $ingredient->ing_n_prix_uni }}€</span>
                    </div>
                </label>
            @endforeach
        </div>
    @else
        <div class="categorie-container
supplement-container">
            <!-- titre -->

```

```

        <h2 class="text-2xl font-bold mt-8
mb-2">Choisissez vos {{ $categorie }} </h2>
        <!-- icones selection -->
        <div class="selection-supplements"
data-base-url="{{ asset('images/') }}"></div>
        <!-- carroussel -->
        <div class="base-carousel mt-3" >
            @foreach ($listeIngredients as $ingredient)
                <label class="carte-element
carte-supplement">
                    <!-- 
alt="imgnondispo"-->
                    
                    <div class="text-center">
                        <span class="block text-xl
font-semibold">{{ $ingredient->ing_va_nom }}</span>
                        <span class="block
text-gray-500">{{ $ingredient->ing_n_prix_uni }}€</span>
                    </div>
                    <div
onclick="selectIngredient('{{ $ingredient->ing_i_id }}', '{{
$ingredient->ing_va_nom }}') " class="absolute bottom-2 right-50 cursor-pointer
">
                        
                    </div>
                </label>
            @endforeach
        </div>
    @endif
    <!-- <input type="radio" name="id{{ ucfirst(Str::slug($categorie)) }}" value="{{ $ingredient->ing_i_id }}">-->
    @endforeach

    <!-- Bouton d'ajout au panier -->
    <hr class="elegant-line">
    @if (\Auth::check())
        <div class="submit-container">

```

```
        <button type="submit" class="bg-[#e4e6c3] text-black px-4 py-2 rounded-lg shadow-md hover:bg-[#b8ba9c] transition-colors duration-300">Ajouter au panier</button>
    </div>
    @else

        <div class="submit-container">
            <a href="{{ route('login') }}" class="bg-[#e4e6c3] text-black px-4 py-2 rounded-lg shadow-md hover:bg-[#b8ba9c] transition-colors duration-300">

                Ajouter au Panier
            </a>
        </div>
    @endif
</form>
</div>
</div>
@if(session('success'))
<div id="customAlert" class="custom-alert">
    <div class="custom-alert-content">
        <h3>Succès !</h3>
        <p>{{ session('success') }}</p>
    </div>
</div>
@endif
</x-app-layout>
```

A11. produitsCompose.js

```
let nbrSupplementSelect = 0;

window.selectIngredient = function(id, nom) {

    if(nbrSupplementSelect < 5) {
        nbrSupplementSelect+=1;
        // initialisation de la div parente et enfant
        const DivParente = document.querySelector('.selection-supplements');
        const DivIcon = document.createElement('div');
        DivIcon.classList.add('DivIcon');

        // url image
        const baseUrl = DivParente.getAttribute('data-base-url');
        const imageName = nom.replace(/\ /g, "_");
        const imageUrl = `${baseUrl}/ProduitsComposeImage/${imageName}.png`;

        // ajout de l'image
        DivIcon.innerHTML = `
            
            <input type="hidden" name="idSupplement[]" value="${id}">
            
        `;

        // Initialisation du bouton suppression
        const boutonSupp = DivIcon.querySelector(".bouton_supp_invisible");

        DivIcon.addEventListener('click', function() {
            DivIcon.remove();
            nbrSupplementSelect--;
        });

        // Apparition du bouton suppression au survol
        DivIcon.addEventListener('mouseover', function() {
            boutonSupp.classList.remove("bouton_supp_invisible");
            boutonSupp.classList.add("bouton_supp_visible");
        });

        DivIcon.addEventListener('mouseleave', function() {
            boutonSupp.classList.remove("bouton_supp_visible");
        });
    }
}
```

```

        boutonSupp.classList.add("bouton_supp_invisible");
    });

    // Ajouter l'élément à la div de sélection
    DivParente.appendChild(DivIcon);
}

//else ici pour adam la notif

}

document.addEventListener("DOMContentLoaded", function () {

    document.querySelectorAll(".base-carousel").forEach((carousel) => {
        const items = carousel.querySelectorAll(".carte-element");
        let currentIndex = 0;

        function updateCarousel() {
            items.forEach((item, index) => {
                item.classList.remove("carte-active", "carte-gauche",
"carte-droite", "carte-invisible");

                if (index === currentIndex) {
                    item.classList.add("carte-active");
                    // Sélectionne le bon input (désactive les autres)
                    const input = item.querySelector("input");
                    if (input) {
                        // Désélectionner tous les autres inputs de la même
catégorie
                        carousel.querySelectorAll("input").forEach(i => i.checked
= false);
                        input.checked = true;
                    }
                } else if (index === (currentIndex - 1 + items.length) %
items.length) {
                    item.classList.add("carte-gauche");
                } else if (index === (currentIndex + 1) % items.length) {
                    item.classList.add("carte-droite");
                } else {
                    item.classList.add("carte-invisible");
                }
            });
        }

        items.forEach((item, index) => {
            item.addEventListener("click", function () {
                currentIndex = index;
            });
        });
    });
});

```

```

        updateCarousel();
    } );
}

updateCarousel(); // Initialisation
} );
}

```

A12. Ingredients.php

```

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Ingredients extends Model
{
    use HasFactory;

    protected $table = 'Ingredients';
    public $timestamps = true;

    public function getKeyName()
    {
        return 'ing_i_id';
    }

    // Propriétés qui peuvent être assignées en masse (mass assignment)

    protected $fillable = [
        'ing_va_nom',
        'ing_n_prix_uni',
        'ing_va_cat',
        'ing_n_statut'
    ];
}

```