

JunkFood

Création de plats et commande en ligne

Dossier Projet

DWWM

Présenté par:

François Pazery

Afpa - Dwwm



Sommaire

Sommaire

Introduction

Liste des compétences couvertes par le projet

I. Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

- A. Maquetter une application
- B. Réaliser une interface utilisateur web statique et adaptable
- C. Développer une interface utilisateur web dynamique
- D. Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce.

II. Développer la partie back-end d'une application web ou web mobile intégrant les recommandations de sécurité :

- A. Créer une base de données
- B. Développer les composants d'accès aux données.
- C. Développer la partie back-end d'une application web ou web mobile.
- D. Élaborer et mettre en oeuvre des composants dans une application de gestion de contenu ou e-commerce.

Résumé du projet

Cahier des charges

I. Besoin et objectifs de l'application

A. Besoins

B. Objectifs

C Cibles

II. Users Stories

III. Arborescence

IV. MVP

V. fonctionnalités détaillées des pages

VI. Wireframe

VIII. Charte graphique

IX. Exemples de maquettes

Specifications techniques

I. Technologies

II. Création de la Base de données

Vulnérabilité de sécurité et veille

I. Veille technologique

II. Veille de sécurité

Conclusion

Annexe

Introduction

Ayant grandi dans les années 90 avec des jeux vidéo et le début d'internet J'ai toujours entretenu un lien particulier avec le monde de l'informatique qui me passionnait, plus tard c'est dans le domaine des transmissions militaires que j'ai découvert d'autres usages et aspect de l'informatique (combat collaboratif, gestion en temps réel d'unités par de la transmission de données radio etc..)

Après 11 ans d'infanterie j'avais besoin de découvrir de nouveaux horizons et de me former à un métier. J'ai donc suivi une reconversion à l'Afpa afin de découvrir un nouveau métier et de changer de vie. Ayant un Bac L l'algorithmie m'a posé de nombreux défis, après ma formation ayant été très absent ces dix dernières années j'ai décidé de prendre une année pour être auprès de mes parents.

Ne me sentant pas prêt à rentrer sur le marché du travail j'ai repris une formation au Wagon à Bordeaux afin de compléter mes savoir faire et de me former sur d'autres technologies (Ruby on Rails)

J'ai restitué dans ce projet ce qui m'a été enseigné lors de ma première formation à l'Afpa de Marseille et je doit dire que si je devais aujourd'hui le reprendre "*from scratch*" il changerait du tout au tout ! Néanmoins ce projet réalisé sur une courte période m'a permis de me familiariser avec des méthodes de travail et des outils particulièrement utiles (méthode Agile, Trello, Canva, Figma)

J'ai donc acquis de nouvelles compétences en travaillant sur ce projet de part l'utilisation d'outils, le processus de mise au point du projet et les problématiques techniques sur lesquelles j'ai travaillé. Mon seul regret est de ne pas avoir pu le faire en équipe pour partager des solutions et des problématiques et envisager d'autres méthodes.

Liste des compétences couvertes par le projet

I.Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

A. Maquetter une application

Avant la réalisation à de l'application, j'ai dans un premier temps travaillé sur la réalisation des documents de conception. Dans ce but, j'ai donc réalisé en premier lieu des users stories pour définir ce que pourront faire les utilisateurs sur notre application. Puis j'ai réalisé les wireframes du projet, aux formats mobile et desktop pour la structuration des pages, enfin, j'ai établit une charte graphique et une maquette avec Figma pour définir ce à quoi le site allait ressembler.

B. Réaliser une interface utilisateur web statique et adaptable

Ce projet étant conçu comme un outil pour aider un commerçant, il était nécessaire d'avoir une interface simple et facile d'utilisation, peu importe le niveau de compétence en informatique, côté client comme côté administration.

De plus, la majorité des personnes accédant à ce type de services via leur téléphone portable, il fallait que l'application soit compatible avec le format mobile.

C. Développer une interface utilisateur web dynamique

Ce projet étant conçu en procédural la partie dynamique sera assurée en javascript par notre affichage de panneau d'administration il rendra cette table HTML plus dynamique en utilisant le plugin jQuery DataTables. cela améliorera la fonctionnalité du tableaux HTML en ajoutant des fonctionnalités de tri, de recherche et de pagination.

D. Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce.

Pour réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce, je suis parti de zéro (from scratch) en suivant mes maquettes, j'ai opté pour des solutions simples basiques avec un affichage des produits et un panier facilement compréhensible pour tous les clients. Cette interface s'adapte aux mobiles et reste claire et lisible.

II. Développer la partie back-end d'une application web ou web mobile intégrant les recommandations de sécurité :

A. Créer une base de données

Après le maquettage j'ai organisé la manière dont allait être structuré ma base de données. Après avoir réfléchis aux données intégrées au site pour être en accord avec le cahier des charges j'ai créé un MCD. J'ai opté pour MySql que j'avais pratiqué durant ma formation.

B. Développer les composants d'accès aux données.

Pour l'accès aux données dans mon application, j'ai utilisé des requêtes SQL préparées via Php, exécutées avec des paramètres, les résultats sont récupérés à partir de la base de données à l'aide des méthodes fournies par l'objet de connexion

C. Développer la partie back-end d'une application web ou web mobile.

Pour réaliser le back-end de mon site, j'ai utilisé php et des requêtes Sql. Je n'ai pas utilisé de frameworks.

D. Élaborer et mettre en oeuvre des composants dans une application de gestion de contenu ou e-commerce.

Pour élaborer les composants de mon application, j'ai séparé les fonctionnalités telles que la connexion, les sessions, l'inscription ou la modification du profil je les ai codés en modules. par exemple le header est appelé et s'afficher sur toute la partie utilisateur en provenance d'un fichier à l'aide d'un include.

Résumé du projet

Un restaurateur souhaite vendre ses produits au travers un site internet afin de ne pas perdre de temps avec des commandes téléphoniques, il a peu de budget et souhaite une interface basique et facilement compréhensible, côté administration il utilise un desktop, mais côté utilisateur il souhaite une application responsive pour que ses clients commandent via leur smartphone.

Ce site est donc pensé pour permettre au restaurateur de gérer facilement ses plats proposés à la vente pour ses clients.

Pour ce projet, je me suis concentré sur les bases du code en Php.

J'ai choisi de présenter un site de e-commerce "from scratch" c'est à dire en partant de zéro sans utiliser un Framework ou un CMS.

Un Framework est un ensemble cohérent de composants logiciels structurels qui sert à créer les fondations ainsi que les grandes lignes de tout ou partie d'un logiciel, c'est-à-dire une architecture. En parallèle d'un CMS qui est une solution "no code" ou tout peut être réalisé via des modules, parfois payants ce qui permet de réaliser un site sans les services d'un développeur.

La méthode utilisé sera procédurale qui s'oppose à la méthode orientée objet. En effet, par l'approche procédurale le code est écrit séquentiellement, alors que l'approche orientée objet le code est encapsulé dans des méthodes de classes et fonctionne via l'interaction d'objet. Cette dernière étant privilégiée en entreprise car elle facilite le travail en groupe permettant de répartir les tâche entre les différentes personnes qui vont coder des modules indépendants les uns des autres.

Pour ce projet, l'administrateur pourra modifier les données du site facilement.

Un système d'inscription de connexion et de panier va être mis en place.

La maquette est réalisée à l'aide de l'outil Figma.

Je me suis aidé de Bootstrap qui est une bibliothèque d'outils HTML CSS contenant du Javascript.

J'ai utilisé de l'Ajax pour des messages d'alerte.

Cahier des charges

I. Besoin et objectifs de l'application

A. Besoins

Pour notre projet nous avons besoin de deux espaces distincts: Un espace utilisateur et un espace d'administration.

Espace utilisateur :

- Un espace d'inscription,
- un espace de connexion
- une page profil pour éditer ses coordonnées
- une page produits
- une page panier

Espace administration :

- un accès à la boutique, pour visualiser les produits comme un client
- une page gestion avec les interactions du CRUD
- une page profil

B. Objectifs

les informations importantes sont:

Le client doit pouvoir créer/modifier un compte.

Le client doit pouvoir ajouter un produit au panier et le supprimer.

L'administrateur doit visualiser tous les produits proposés.

en ajouter, les modifier ou les supprimer, facilement via un panneau de contrôle lisible

C. Cibles



Emma Bensaïd 17 ans, étudiante

Équipement informatique : iphone 14

Ce qu'elle recherche : Commander facilement ses plats préférés sans se déplacer

Ce qu'elle attend : Accéder à toutes les fonctionnalités sur son téléphone

Prérequis: le site doit être optimisé pour mobiles



Stéphane Medel 42 ans, cadre

Équipement informatique : iphone 14, desktop windows

Ce qu'il recherche : Savourer des plats délicieux

Ce qu'il attend : Une interface réactive et claire pour commander

Prérequis : Le site doit avoir une bonne expérience utilisateur



Memeth Osgur 64 ans, propriétaire du restaurant

Équipement informatique : desktop windows

Ce qu'il recherche : Gagner du temps pour être avec sa famille

Ce qu'il attend : Accéder rapidement aux informations de son restaurant

Prérequis : L'interface doit être simple il n'est pas à l'aise avec l'informatique

II. Users Stories

visiteur-> créer un compte

visiteur-> se connecter

visiteur-> contacter le restaurant

membre> consulter les plats -> en ajouter à son panier -> modifier son panier

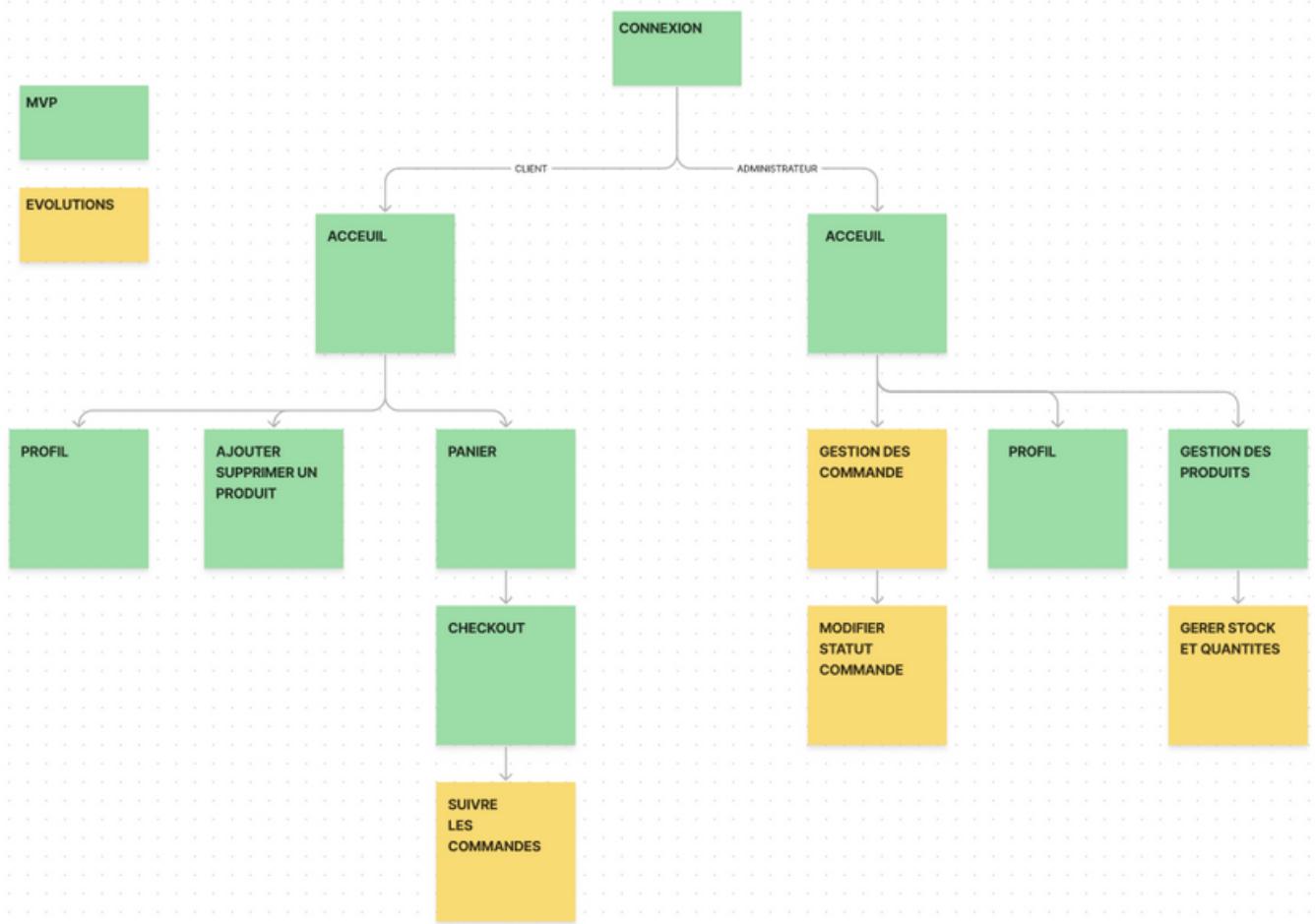
administrateur > se connecter -> gérer les plats proposés -> créer modifier mettre à jour

MVP

Evolutions potentielles

En Tant que	Je souhaite	Afin de
visiteur	me connecter au site	accéder aux différents services
visiteur	me déconnecter du site	quitter le site
utilisateur connecté	accéder à mon profil	modifier mes info personnelles
utilisateur connecté	suivre ma commande	gagner du temps
utilisateur connecté	ajouter un produit au panier	commander en ligne
utilisateur connecté	payer ma commande	gagner du temps
administrateur	ajouter/modifier un plat	le proposer à mes clients
administrateur	supprimer un plat	changer ma carte
administrateur	modifier le statut d'une commande	avoir des clients satisfait
administrateur	gérer le stock	rester cohérent

III. Arborescence



IV. MVP

Le *Produit Minimum Viable* est construit autour des plats proposées à la commande. Une inscription et une connexion sont essentielles, ainsi que les fonctionnalités liées à l'administrateur qui permettent que les plats soient disponibles et mis à jour. Voici donc les fonctionnalités présentes dans notre MVP :

- l'utilisateur (client ou administrateur) doit pouvoir s'authentifier avec un email et un mot de passe

- l'utilisateur doit pouvoir consulter les mentions légales et la politique de confidentialité du site

A. Client

- arrive sur la page accueil : il doit pouvoir voir les plats, ou accéder à son profil

- pour commander un plat, il doit pouvoir voir la liste des plats.

- il doit pouvoir cliquer sur un plat pour ajouter la quantité désirée

- sur la fiche du plat, il doit pouvoir consulter les informations (caractéristiques) et valider l'ajout au panier par un bouton

- une fois le bouton “ajouter au panier” cliqué, il est redirigé vers la page panier

- Après tous ses ajouts il peut, dans la page panier cliquer sur “checkout” qui va résumer et confirmer sa commande, il peut également supprimer une ligne ou vider le panier.

- il doit pouvoir accéder à son compte et modifier ses informations dans la rubrique "profil".

B. Administrateur

- arrive sur la page d'administration, il doit pouvoir :

- consulter la liste des plats(comme pour l'utilisateur)

- créer une nouvelle fiche plat

- éditer une fiche plats

- supprimer une fiche plats

- accéder à son profil (comme pour l'utilisateur)

V. fonctionnalités détaillées des pages

Les pages/routes sont sécurisées et donc accessibles seulement à partir du moment où l'utilisateur est connecté. Seule la page d'accueil de connexion et d'inscription sont accessibles par un visiteur.

Au préalable, le client doit s'inscrire. Il pourra modifier ses informations dans la page profil

Page Inscription :

La page d'inscription est un formulaire. Elle tient compte des informations nécessaires pour une éventuelle livraison. le visiteur doit compléter les champs pseudo, nom, prénom, email, adresse, code postal, ville et password, il est une fois enregistré assigné au rôle de client par défaut

Page connexion :

Pour le moment l'utilisateur n'a pas accès au panier. Il doit compléter les champs :

- email
- mot de passe

Il valide son authentification grâce à un bouton. Il accède à la page d'accueil si c'est un bénévole ou un admin. les Navbar présentent des fonctionnalités adaptés à chaque rôle

Page accueil :

Cette page présente les produits sous forme de cartes.

Dans la partie navigation, on peut accéder au menu et à la page profil

l'administrateur peut accéder à la gestion des plats, l'utilisateur au panier et au formulaire de contact

Page Profil:

Cette page récapitule les informations de profil de l'utilisateur connecté:

Infos de profil

- Prénom :
- Nom :
- Pseudo :
- Email :
- Statut : Admin/client
- Adresse de livraison

Un bouton retour accueil est présent, ainsi qu'un lien vers la modification des informations.

Page Mes Informations:

Cette page récapitule les informations de profil de l'utilisateur connecté sur un formulaire et en permet donc l'édition. les champs suivant y sont présents:

- Prénom :
- Nom :
- Pseudo :
- Email :
- Statut : Admin/client
- Adresse de livraison

Page Panneau d'administration :

Cette page présente les produits sous forme de liste.
l'administrateur accède à la gestion des plats,

Un filtre lui permet de classer les plats selon les critères suivants :

-Id -Nom -catégorie -Prix

Une barre de recherche lui permet d'effectuer une requête précise, il peut sélectionner le nombre de plats qu'il affiche sur une page

Des boutons lui permettent d'ajouter une recette, d'en afficher les détails, d'en effacer ou d'en éditer.

Page fiche plat:

Cette page présente les produits sous forme de carte.
on y voit l'id le nom la catégorie la préparation et le prix
un bouton de retour au panneau d'administration est également présent

Page Update fiche plat:

La page Mise à jour produit affiche un formulaire avec les champs:
Nom - Recette - Prix - une image a uploader - une sélection (entrée, plat, dessert) de catégorie
un bouton de mise à jour est présent
un bouton de retour au panneau d'administration est également présent

Bouton de suppression “Effacer”

Ce bouton affiche une modale de confirmation d'effacement de la ligne.

Page Mon panier:

Cette page présente les produits ajoutés sous forme de liste.
le client accède à la gestion de son panier, il visualise les plats ajoutés et leur quantité, les sous totaux ainsi que le prix total,
plusieurs boutons sont à sa disposition pour soit supprimer une ligne de produits soit vider le panier
Un bouton checkout permet d'afficher la page checkout
Un bouton Continue shopping lui permet de retourner sur la page d'accueil

Page Checkout:

Cette page résume les éléments de la commande et le total en affichant un message.

VI. Wireframe

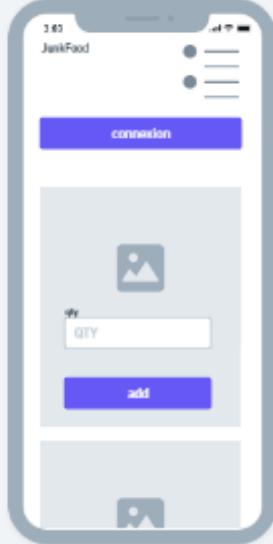
Pour faire le wireframe j'ai utilisé whimsical, j'ai détaillé l'arrivée d'un visiteur son inscription puis sa connexion.

le téléphone portable étant très courant chez les utilisateurs il convient en fonction du cahier des charges de commencer par penser son application (ou site) pour mobile, on appelle cela le “mobile first”.

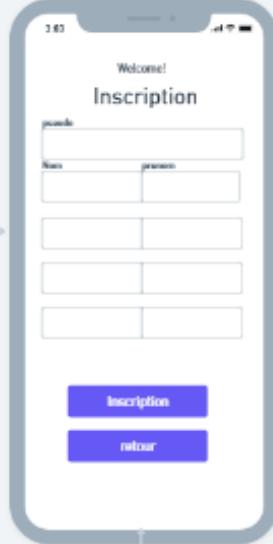
Inscription et connexion

Description...

page accueil



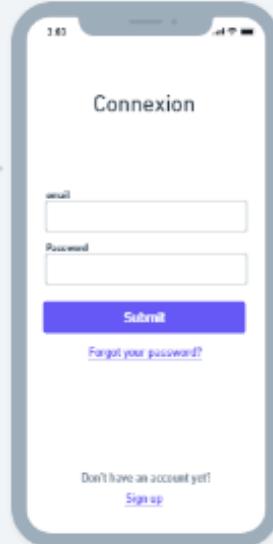
Create Account



Has account?

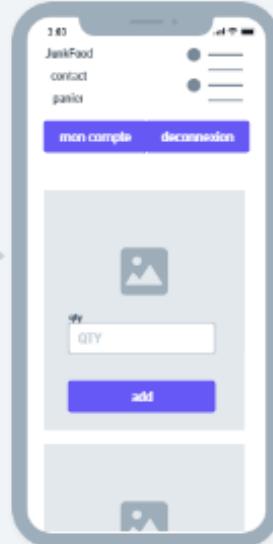
Yes

Sign In



Successful submission

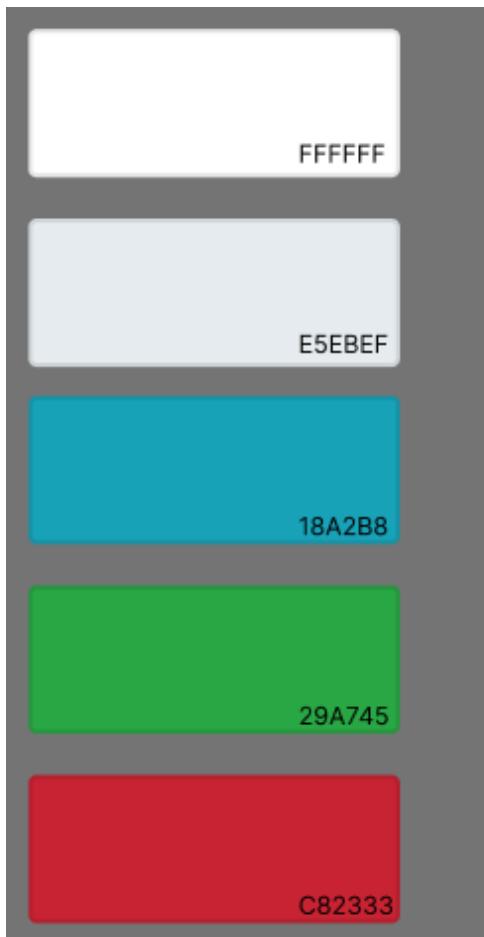
page accueil



Etant seul sur le projet et donc par souci d'économie de temps et de répétition des tâches, après cette partie basique sur mobile j'ai décidé de passer directement au maquettage du site afin d'y dérouler toutes les situations

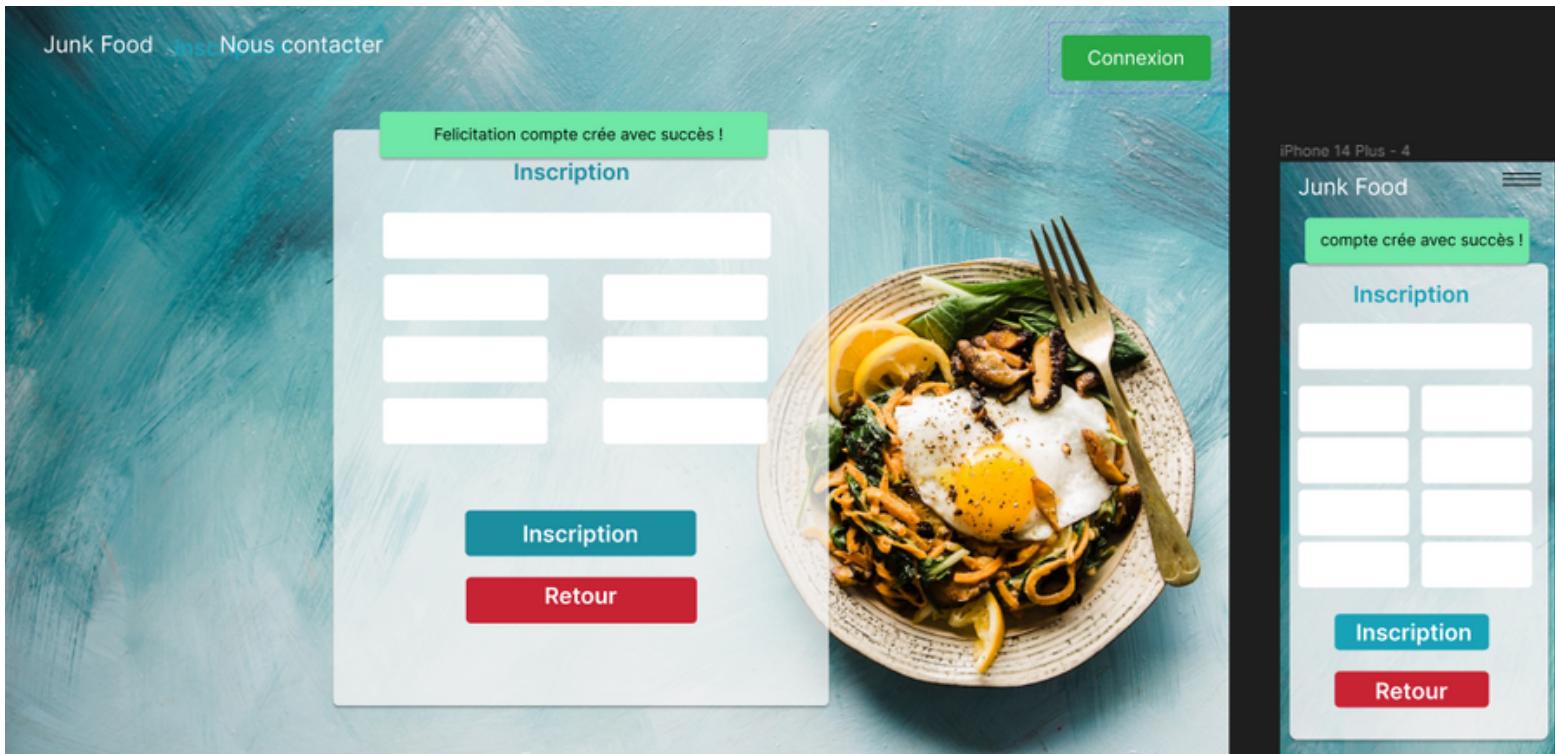
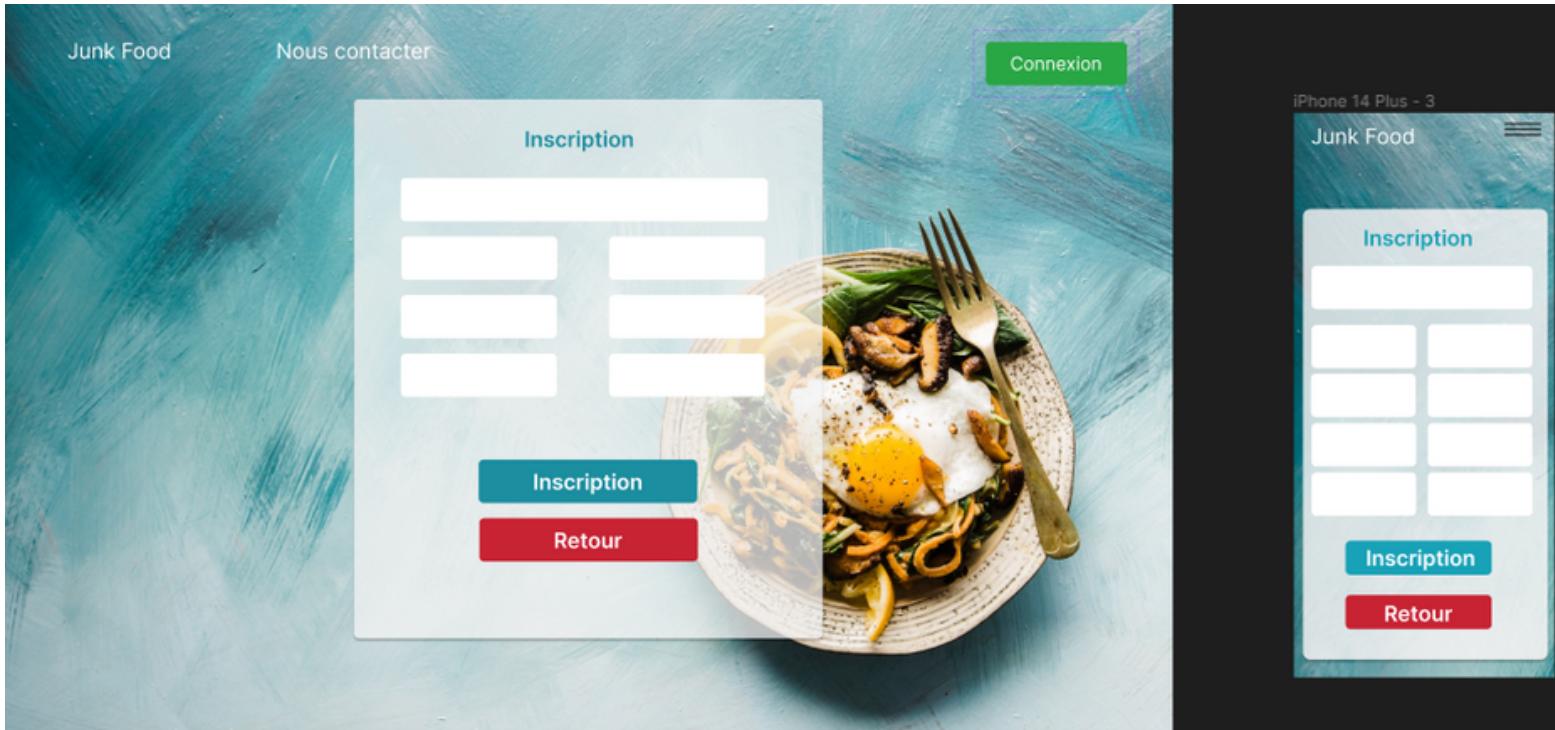
VIII. Charte graphique

Le site étant à destination de la restauration l'objectif était de proposer une charte graphique rassurante pour le client sans parti pris trop fort sur des couleurs vives, la transparence et le bleu rappelant la fraîcheur des produits proposés. le vert et le rouge sont la uniquement pour les boutons



IX. Exemples de maquettes

inscription



connexion

Junk Food Nous contacter

Bouton connexion

Connexion

Me Connecter

Connexion

Inscription



iPhone 14 Plus - 2

Junk Food

Me Connecter

Connexion

Inscription

Accueil connecté

↳ NavBar2

Junk Food Nous contacter Mon panier

Mon Compte

Deconnexion

Divers plats
10\$

Quantité:

Ajouter au panier

iPhone 14 Plus - 5

Junk Food
contact
Panier

Mon compte

Deconnexion

Divers plats
10\$

Quantité:

Ajouter au panier

Divers plats
10\$

Quantité:

Ajouter au panier

Infos profil “mon compte”

Junk Food

Nous contacter

Mon compte

Déconnexion

Infos de profil

Prénom: Jean Michel
Nom: Exemple
pseudo: Exemple
Email: Exemple@example.net
Statut: Client

adresse de livraison

Jean Michel exemple
14 route du curé
13000 Marseille

Retour accueil

Modifier mes infos

Nom utilisateur **éditer mes informations**



iPhone 14 Plus - 3

Junk Food

Infos de profil

Prénom: Jean Michel
Nom: Exemple
pseudo: Exemple
Email: Exemple@example.net
Statut: Client

adresse de livraison

Jean Michel exemple
14 route du curé
13000 Marseille

Retour accueil

Modifier mes infos

Nom **éditer informations**

edit “mon compte”

Junk Food

Nous contacter

Mon compte

Déconnexion

Editor mes infos

Mettre à jour

Retour accueil



iPhone 14 Plus - 6

Junk Food

Editor mes infos

Mettre à jour

Retour accueil

Panier

The screenshot shows the 'Panier' (Cart) page for a food delivery service named 'Junk Food'. The desktop view features a header with navigation links: 'Junk Food', 'Nous contacter', 'Mon panier' (with a shopping cart icon), 'Mon Compte' (green button), and 'Deconnexion' (red button). Below the header is a section titled 'Produits dans le panier' (Products in the cart) showing three items: 'prod 1' (a bowl of soup), 'prod 1' (another bowl of soup), and 'prod 1' (a third bowl of soup). Each item has details: 'Prix unitaire 5\$', 'quantité 1', and 'sous total 5\$'. Below each item is a red 'Retirer du panier' (Remove from cart) button. A large call-to-action box at the bottom right contains: 'Total panier 10\$', 'vider panier' (empty cart) in red, 'Checkout' in blue, and 'Continue shopping' in green. The mobile view on an iPhone 14 Plus shows a similar layout but includes a 'Total panier 10\$' summary at the top and a 'vider panier' button above the 'Checkout' button.

checkout

The screenshot shows the 'Recapitulatif commande' (Order Summary) page. The desktop view has a header with 'Junk Food', 'Nous contacter', 'Mon panier' (with a shopping cart icon), 'Mon Compte' (green button), and 'Deconnexion' (red button). Below is a section titled 'Recapitulatif commande' showing two items: 'prod 1' (Component 24) and 'prod 1' (Component 25). Each item has details: 'Prix unitaire 5\$', 'quantité 1', and 'sous total 5\$'. A 'Total panier 10\$' summary is at the bottom. The mobile view on an iPhone 14 Plus shows a summary: 'commande enregistrée' (order registered), 'Recapitulatif commande', and a list of items: 'prod 1' (Component 24) and 'prod 1' (Component 25), each with its details and a 'total commande' (total command) summary.

Gestion Produits

Ajouter un plat



Produit	Prix unitaire	quantité	prix	détails	Editer	Effacer
prod 1	5\$	1	5\$	détails	Editer	Effacer
prod 2	5\$	1	5\$	détails	Editer	Effacer
prod 3	5\$	1	5\$	détails	Editer	Effacer



Ajouter un produit:

Creation fiche produit

 Nom Recette Prix de vente TTC fichier[Ajouter le plat](#)[Retour aux produits](#)

planification

Pour planifier mon projet je me suis aidé de la **méthode agile**.

J'ai également utilisé l'outil **Trello** pour m'aider dans ma tâche et dans les multiples étapes que j'ai suivi:

The screenshot shows a Trello board titled "JunkFood". The board has four main columns: "À faire", "Sprint", "Improvements", and "Terminé".

- À faire:** Contains cards for defining functionalities, reviewing code with comments, and writing the project report.
- Sprint:** Contains cards for displaying the cart and the back-end of the cart.
- Improvements:** Contains a card for hashing passwords.
- Terminé:** Contains cards for realizing the MCD, wireframe, mobile and desktop mockups with Figma, HTML/CSS for mobile and desktop, creating the Bdd, creating the front and back page, inscription/connection front and back, back-end crud - SQL requests and display in the admin panel, contact, and tests. A card for tests is due on October 20th.

A "Ajouter une autre liste" button is visible in the top right corner.

La méthode **Agile** recommande de se fixer des objectifs à court terme.

Le projet est donc divisé en plusieurs sous-projets.

Une fois l'objectif atteint, on passe au suivant, et ce jusqu'à l'accomplissement de l'objectif final. Cette approche est plus flexible. Puisqu'il est impossible de tout prévoir et de tout anticiper, elle laisse la place aux imprévus et aux changements.

La méthode Agile repose sur une relation privilégiée entre le client et l'équipe projet.

Sa satisfaction étant la priorité, l'implication totale de l'équipe et sa réactivité face aux changements s'imposent. Le dialogue est privilégié. C'est le client qui valide chaque étape du projet. Il convient donc de prendre en compte l'évolution de ses besoins. Des ajustements sont effectués en temps réel afin de répondre à ses attentes.

La plus célèbre des méthodologies de gestion de projets déclinées de la méthode Agile relève de la « Scrum » autrement dit la “mêlée” dans le langage rugby.

Le responsable de projet s'appelle ainsi le “SCRUM Master”.

Cette approche s'organise autour de cycles courts, qu'on appelle les itérations. En langage Scrum, une itération se nomme un “sprint”. À chaque nouveau sprint, l'équipe projet se rassemble pour lister les tâches à exécuter. Cette liste s'appelle le “sprint backlog”.

Spécifications techniques

Pour réaliser les spécifications techniques, j'ai réfléchi aux technologies que je souhaitais utiliser pour réaliser ce projet.

I. Technologies

Pour réaliser ce projet je me suis orienté sur **Php** avec **Bootstrap** pour le code avec des ajouts en **javascript** et en **ajax** pour certaines fonctionnalités comme les messages d'alerte, **phpmyadmin** pour la gestion de la base de données et **Laragon** pour développer en local.

II. Création de la Base de données

Pour créer une base de données cohérentes sur notre site il faut créer un schéma qui regroupe tout nos besoins.

J'ai utilisé la méthode **merise** pour nous permettre de nous donner les règles de construction du schéma et les dépendances fonctionnelles de notre base de données.

Merise est une méthode de conception de systèmes d'information. Il faut Analyser et avant d'écrire notre programme/site.

Il faut donc établir les relations entre nos champs et tables de bdd.

ici cart est une table de jointure avec deux clé étrangères.

Une "cart" belongs to one "user" et has many "plats"

Un user a un panier "cart" et plusieurs "plats" trough "cart"

On a ainsi pu réaliser notre modèle conceptuel des données (MCD).

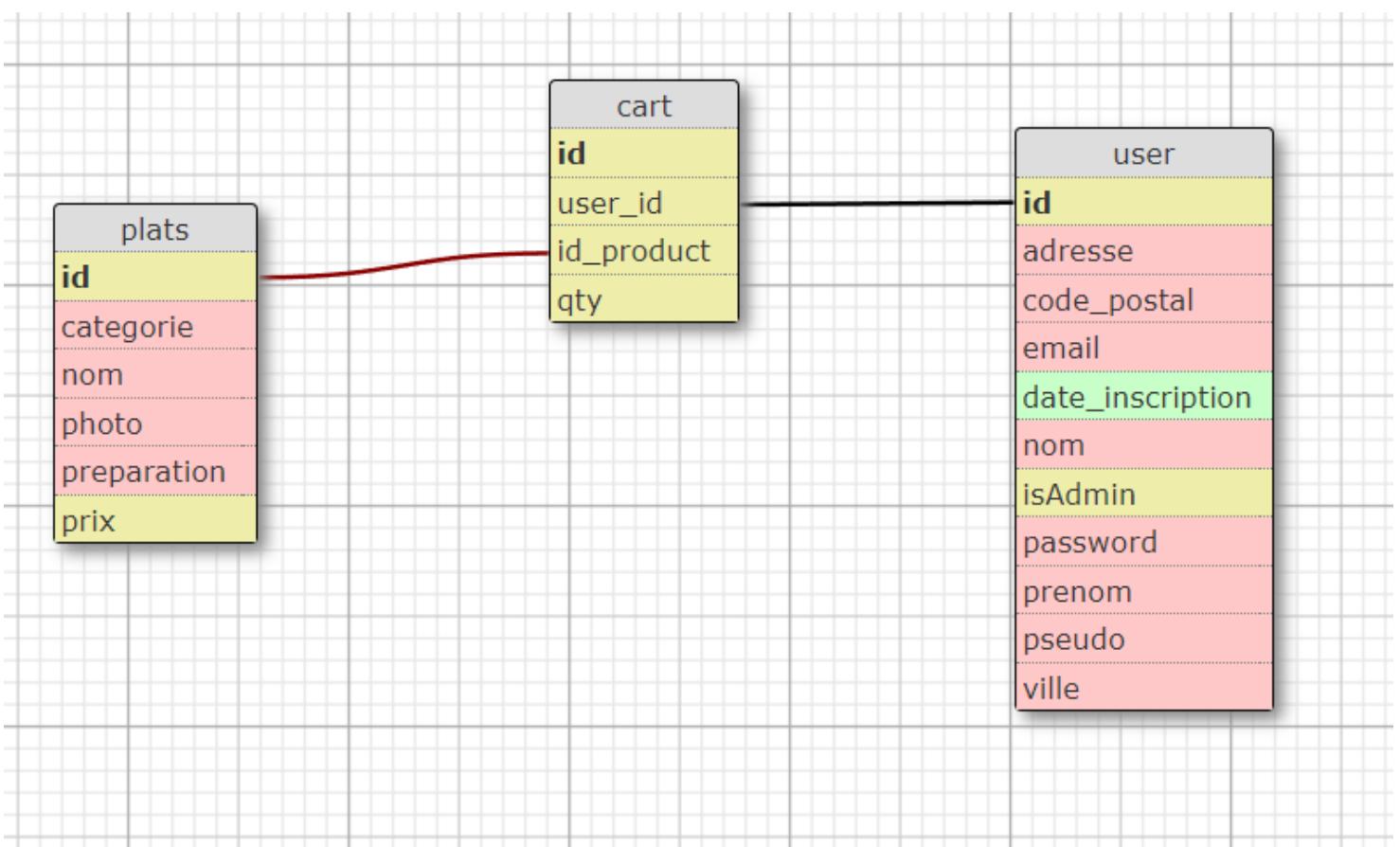
Dans un MCD, les entités sont comme des objets. Par exemple, l'entité membre a des attributs tels que pseudo, mot de passe... Ces attributs possèdent des types standardisés pour les décrire, que ce soit une date, un booléen (renvoyant vrai ou faux), un entier ou un nombre.

Sur le MCD figurent des associations qui illustrent le lien entre les entités, appelées relations sémantiques.

Par exemple, un membre peut avoir plusieurs produits dans son panier, mais ce panier ne peut être créé que par un seul membre. Ces jeux de relations sont appelés **cardinalités**.

- Un membre peut commander de 0 à n plats.
- Une commande peut être créée que par une seul user
- Un produit peut être contenu dans au moins un panier, sans maximum de produits.

Ensuite, on réalise un MPD (modélisation physique des données), qui absorbe souvent le MLD (modélisation logique des données) pour affiner le MLD en vue d'un SGBD (système de gestion de base de données).



Maintenant que notre arborescence est mise en place et que l'on sait ce que l'on veut créer, nous nous rendons dans phpMyAdmin pour créer la base de données, soit de manière mécanique soit en tapant des requêtes SQL.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
cart	Parcourir Structure Rechercher Insérer Vider Supprimer	6	InnoDB	utf8mb3_general_ci	48,0 kio	-
orders	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb3_general_ci	16,0 kio	-
plats	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	latin1_swedish_ci	16,0 kio	-
product	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb3_general_ci	16,0 kio	-
user	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8mb3_general_ci	16,0 kio	-
5 tables	Somme				112,0 kio	0

Ainsi, la base de données créée, nous allons établir un lien avec le **BackOffice** de notre site.

Le BackOffice d'un site web constitue une interface de gestion des réglages, réservée uniquement à l'administrateur (ou aux administrateurs).

Nous avons prévu une section (dans le dossier /admin/) réservée à l'administrateur pour la gestion de son site web (produits). Cette section ne doit être accessible que par un internaute connecté dont le statut est fixé à 1 (autrement dit : un administrateur).

Pour garantir cela, nous allons utiliser la **fonction AdminConnected()** comme prévu dans notre code permettant de renvoyer TRUE (vrai, tu es admin) ou FALSE (faux, pas admin).

Nous prévoirons également un formulaire d'ajout de produit avec récupération des données en POST et requête d'enregistrement (INSERT) dans la base de données. Nous avons prévu trois liens permettant soit d'afficher les produits, soit d'ajouter un produit soit de les supprimer. Techniquement, cela permet de passer dans l'URL ?action=affichage ou ?action=ajout, et donc en récupérant l'information véhiculée dans l'URL (via \$_GET), le site peut détecter l'action à déclencher.

L'administrateur peut ainsi choisir l'action qu'il souhaite mettre en œuvre : ajout suppression ou affichage. L'affichage des produits se fait dans une table (un tableau).

Nous ajoutons trois liens (représentés par des boutons) afin de proposer la modification et la suppression de produits pour l'administrateur (le commerçant).

Pour la modification et la suppression, nous passerons par l'action dans l'URL ainsi que l'ID du produit correspondant.

L'action nous permettra de savoir que nous devons supprimer ou modifier un produit, et l'ID nous permettra de savoir duquel il s'agit.

Gestion du produit Nous devons également proposer la modification de produits. Pour cela, nous aurons besoin d'un formulaire permettant d'accueillir les données en vue d'une modification.

Pour factoriser le code (c'est-à-dire réduire le code en le réutilisant pour plusieurs fonctions différentes), nous aurions pu utiliser un seul formulaire pour l'ajout et la modification. En effet, que l'on ajoute ou que l'on modifie, il s'agit des mêmes champs. La seule différence réside dans le fait que, lorsqu'on modifie, il nous faudra pré-remplir les champs du formulaire avec les informations actuelles.

Nous détectons l'action de suppression dans l'URL avec la condition if(isset(\$_GET['action']) && \$_GET['action'] == "suppression"). Ensuite, nous sélectionnons toutes les informations sur ce produit dans la base, notamment le chemin vers la photo afin de la supprimer de notre serveur.

Nous formulons une deuxième requête pour supprimer effectivement le produit de notre base, puis nous redirigeons vers l'action d'affichage pour observer tous les produits.

Le CRUD

```

1 <?php
2 require_once '../config.php'; // On inclut la Connexion à la Bdd
3 require_once '../fonction.php'; //Include des fonctions pour vérification isAdmin
4
5 $update = false;
6 $id = "";
7 $nom = "";
8 $preparation = "";
9 $prix = "";
10 $photo = "";
11 $categorie = "";
12
13
14 // ***** CRUD *****
15 // Si les champs sont remplis:
16
17 if (isset($_POST['add'])) {
18     $nom = $_POST['nom'];
19     $preparation = $_POST['preparation'];
20     $prix = $_POST['prix'];
21     $photo = $_FILES['image']['name'];
22     $upload = "uploads/" . $photo;
23     $categorie = $_POST['categorie'];
24
25
26
27 // ***** AJOUTER *****
28 // On prépare la requête SQL
29 $stmt = $bdd->prepare("INSERT INTO plats (nom,preparation,prix,photo,categorie)VALUES(:nom,:preparation,:prix,:photo,:categorie)");
30 // On execute la requête SQL
31 $stmt->execute(['nom' => $nom, "preparation" => $preparation, "prix" => $prix, "photo" => $upload, "categorie" => $categorie]);
32 // La partie upload du fichier image via un dossier temporaire
33 move_uploaded_file($_FILES['image']['tmp_name'], $upload);
34 // On renvoie l'utilisateur à la page CRUD + message de confirmation
35 header('location:product.php');
36 $_SESSION['response'] = "Insertion réussie en base de donnée !";
37 $_SESSION['res_type'] = "success";
38
39
40 // ***** EFFACER *****
41
42 } else if (isset($_GET['delete'])) {
43     $id = $_GET['delete'];
44     // On sélectionne ce qui va être effacé
45     $stmt = $bdd->prepare("SELECT photo FROM plats WHERE id=:id");
46     // On efface l'entrée sélectionnée
47     $stmt->execute(["id" => $id]);
48     $row = $stmt->fetch();
49     $imagepath = $row['photo'];
50     unlink($imagepath);
51     $stmt = $bdd->prepare("DELETE FROM plats WHERE id=:id");
52     $stmt->execute(["id" => $id]);
53     // On renvoie l'utilisateur à la page CRUD + message de confirmation
54     header('location:product.php');
55     $_SESSION['response'] = "Champ effacé de la base de donnée !";
56     $_SESSION['res_type'] = "danger";
57 }
58
59
60 // ***** EDITION MAJ *****
61
62 if (isset($_GET['edit'])) { // Editer une entrée de la Bdd
63     $id = $_GET['edit'];
64
65     $stmt = $bdd->prepare("SELECT * FROM plats WHERE id=:id");
66     $stmt->execute(["id" => $id]);
67     $row = $stmt->fetch();
68
69     $id = $row['id'];
70     $nom = $row['nom'];
71     $preparation = $row['preparation'];
72     $prix = $row['prix'];
73     $photo = $row['photo'];
74     $categorie = $row['categorie'];
75     $update = true;
76

```

```

75     $update = true;
76 }
77 ✓ if (isset($_POST['update'])) {
78     $id = $_POST['id'];
79     $nom = $_POST['nom'];
80     $preparation = $_POST['preparation'];
81     $prix = $_POST['prix'];
82     $oldimage = $_POST['oldimage'];
83     $categorie = $_POST['categorie'];
84
85 ✓ if (isset($_FILES['image']) && ($_FILES['image']['name'] != "")) {
86     $newimage = "uploads/" . $_FILES['image']['name']; //timestamper et garder l'ext du fichier (recup nom fichier + traitement recuperation changer le nom avec un tsmp)
87     unlink($oldimage);
88     move_uploaded_file($_FILES['image']['tmp_name'], $newimage);
89 } else {
90     $newimage = $oldimage;
91 }
92
93 $stmt = $bdd->prepare("UPDATE plats SET nom=:nom, preparation=:preparation, prix=:prix, categorie=:categorie, photo=:photo WHERE id=:id");
94 $stmt->execute(["nom" => $nom, "categorie" => $categorie, "preparation" => $preparation, "photo" => $newimage, "id" => $id, "prix" => $prix,]);
95 $_SESSION['response'] = "Mise à jour effectuée !";
96 $_SESSION['res_type'] = "primary";
97 header('location:product.php');
98 }
99
100
101
102
103 // ***** CRUD DETAILS AFFICHAGE *****
104 ✓ if (isset($_GET['details'])) {
105     $id = $_GET['details'];
106     $stmt = $bdd->prepare("SELECT * FROM plats WHERE id=:id");
107     $stmt->execute(["id" => $id]);
108     $row = $stmt->fetch();
109

```

Includes et connexion à la Bdd

Parlons des **includes** et de la **connection** à la Bdd: “config.php”

J'ai choisi la méthode *new Pdo* à la place de *msqli* pour mon site, je crée ma connexion à la base de données dans un fichier unique que je viendrais appeler dans mes différents fichiers via des includes:

config.php

```

1  <?php
2      //Connexion à la Bdd en localhost via l'objet PDO
3      session_start();
4
5      try {
6          $bdd = new PDO('mysql:host=localhost;dbname=cruddb;charset=utf8mb4', 'root', '');
7      } catch (PDOException $e) {
8          die('Erreur : ' . $e->getMessage());
9      }

```

form_co.php

```
1 <?php
2 //Affichage du header et Co à la Bdd
3 require_once "config.php";
4 require_once "header.php";
5 ?>
6
```

Ainsi, le header est inclus dans tous les affichages il contient l'ouverture de la balise <body> le footer quand a lui contient la balise fermante: il utilise la fonction ClientConnected() pour moduler son affichage

```
</head>
```

```
<body>
```

```
<header> <!-- Navbar Bootstrap modifiée Transparente - Collapse set sur "expand-sm" -->
<nav class="navbar navbar-expand-sm navbar-dark">
    <a class="navbar-brand" href="index.php">Junk Food</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation"><span class="navbar-toggler-icon"></span>
        <?php if (!AdminConnected()) {
            //Affichage de 'nous contacter' si pas Admin
            { ?>
                <li class="nav-item">
                    <a class="nav-link" href="./contact.php">Nous contacter<span class="sr-only" style="font-size: small;">(current)</span></a>
                </li>
            <?php } ?>
        <?php
        if (AdminConnected())
            //Fonction d'affichage du lien vers le CRUD si Admin.
        { ?>
            <li class="nav-item">
                <a class="nav-link" href="./admin/product.php">Panneau d'administration</a>
            </li>
        <?php } ?>
        <?php if (ClientConnected()) {
            //Fonction PHP pour afficher l'icône 'Mon panier' si l'utilisateur est connecté: $_SESSION['connexion'] = true; si le mdp e
        ?>
    </button>
</nav>
```

footer.php

```
1 <!--JavaScript executé par navigateur (plugin jQuery
2 améliore la fonctionnalité des tableaux HTML en ajoutant des fonctionnalités de tri,
3 de recherche et de pagination)-->
4 <script type="text/javascript">
5     $(document).ready(function() {
6         $('#data-table').DataTable({
7             paging: true
8         });
9     });
10 </script>
11 </body>
12
13 </html>
```

La sécurité

Pour le système de connexion des utilisateurs, j'ai mis en place une vérification savoir si l'utilisateur était bien présent en Bdd:

```
// On check si l'utilisateur est inscrit dans la table user:  
$stmt = $bdd->prepare('SELECT * FROM user WHERE email = :email');  
$stmt->execute(array("email" => $_POST['email']));  
$row = $stmt->rowCount();
```

Puis une création de variables de session pour accéder aux données plus tard dans mon code.

```
// Si > a 0 alors l'utilisateur existe  
if ($row > 0) {  
    $data = $stmt->fetch();  
    // Si le mail est bon niveau format  
    if (filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)) {  
        // Si le mot de passe est le bon  
        if (password_verify($_POST['password'], $data['password'])) {  
            // On crée la session et on redirige sur landing.php  
            // on crée un booléen pour le bouton connexion déconnexion sur la navbar  
            $_SESSION['connexion'] = true; // Utilisateur connecté  
            $_SESSION['isAdmin'] = (int)$data['isAdmin']; // déclare isAdmin  
            $_SESSION['pseudo'] = $data['pseudo']; // Stockage du pseudo  
            $_SESSION['id'] = $data['id']; // Stockage de l'id  
            $_SESSION['nom'] = $data['nom'];  
            $_SESSION['prenom'] = $data['prenom'];  
            $_SESSION['email'] = $data['email'];  
            $_SESSION['adresse'] = $data['adresse'];  
            $_SESSION['code_postal'] = $data['code_postal'];  
            $_SESSION['ville'] = $data['ville'];  
            //$_SESSION['user'] = $data['token'];
```

Pour les inscriptions j'ai mis en place un formulaire, afin de limiter les risques d'attaques XSS (injection de code), on peut utiliser htmlspecialchars: il convertit les caractères spéciaux pour empêcher l'utilisation de balises script avant un enregistrement en Bdd

```
$regex = "#^[a-zA-Z0-9][a-zA-Z0-9]*[-]?[a-zA-Z0-9]*[a-zA-Z0-9]+#$";  
  
// Si les champs du formulaire ont été remplis :  
if (!empty($_POST['pseudo']) && !empty($_POST['nom']) && !empty($_POST['prenom']) && !empty($_POST['email']) && !empty($_POST['adresse']) && !empty($_POST['ville'])) {  
    // Patch XSS  
    $pseudo = htmlspecialchars($_POST['pseudo']);  
    $nom = htmlspecialchars($_POST['nom']);  
    $prenom = htmlspecialchars($_POST['prenom']);  
    $email = htmlspecialchars($_POST['email']);  
    $adresse = htmlspecialchars($_POST['adresse']);  
    $code_postal = htmlspecialchars($_POST['code_postal']);  
    $ville = htmlspecialchars($_POST['ville']);  
    $password = htmlspecialchars($_POST['password']);  
    $password_retype = htmlspecialchars($_POST['password_retype']);  
  
    var_dump($_POST); // Ajout du var_dump pour le débogage
```

Toujours sur la sécurité, j'ai mis en place le hash des mot de passe en utilisant Bcrypt qui est une fonction intégrée à Php; c'est un algorithme de hachage cela garantit un minimum de sécurité par du chiffrement.

```
if ($password === $password_retype) { // Si les deux mots de passe saisis sont bons  
    // On hash le mot de passe avec Bcrypt, via un coût de 12  
    $cost = ['cost' => 12];  
    $password = password_hash($password, PASSWORD_BCRYPT, $cost);
```

J'ai également mis en place une redirection si un utilisateur accède à une page d'administration sans être loggé en administrateur; Cela peut se produire si il connaît ou devine les noms de fichiers:
Je réutilise ma fonction AdminConnected pour rédiriger un éventuel intrus vers la page d'accueil.

```
5  ?>  
6  <?php if (!AdminConnected()) {  
7  |     header('location:../index.php');  
8  }  
q
```

Un peu de Front

J'ai utilisé la bibliothèque Bootstrap pour la mise en forme du site, mettant les éléments à organiser dans des div container (ou *container-fluid* du fait de ma mise en page avec image de fond)
j'ai essentiellement utilisé des *d-flex flex-column*, de **bootstrap** pour ma mise en page.

J'ai réinitialisé les *paramètres CSS* avant toute chose afin d'éviter d'éventuels problèmes de margin liste à points etc...

```
1  /* reset des paramètres CSS*/
2  * {
3      margin: 0;
4      padding: 0;
5      text-decoration: none;
6      list-style: none;
7      justify-content: center;
8 }
```

J'ai également utilisé Bootstrap pour ma Navbar et mes boutons: le fait d'avoir des composants prêts à l'emploi est un aide précieuse pour ne pas avoir à ré inventer la roue pour chaque composant.

```
<header> <!-- Navbar Bootstrap modifiée Transparente - Collapse set sur "expand-sm" -->
<nav class="navbar navbar-expand-sm navbar-dark">
  <a class="navbar-brand" href="../index.php">Junk Food</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav"
  aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNav">
    <ul class="navbar-nav mr-auto">
```

Composante de Web Dynamique

Bien que responsive en grande partie, ce site n'en demeure pas moins un site et non une application dynamique, c'est un dinosaure procédural qui doit charger une page à chaque requête.

Cependant je lui ai intégré une fonctionnalité dynamique grâce à l'utilisation de Javascript:

Cette fonctionnalité permet à l'administrateur d'effectuer une recherche ou de classer des éléments grâce à leurs caractéristiques sans recharger la page et ça change tout...

footer.php

```
1  <!--JavaScript exécuté par navigateur (plugin jQuery
2  améliore la fonctionnalité des tableaux HTML en ajoutant des fonctionnalités de tri,
3  de recherche et de pagination)-->
4  <script type="text/javascript">
5      $(document).ready(function() {
6          $('#data-table').DataTable({
7              paging: true
8          });
9      });

```

Afin d'éviter le blocage du rendu initial de la page Html on place le code Javascript dans le footer.

Notre script nous permet d'avoir une composante dynamique qui réagit sans avoir à recharger notre page, un gain de temps et de ressources certain et un confort d'utilisation au standards du web moderne.

On peut réaliser une recherche dont le résultat s'affiche en temps réel ou classer nos éléments à volonté et ce instantanément.

Gestion des Produits						
Ajouter un plat						
Show	10	entries	Search: <input type="text"/>			
Id ↑↓	Visuel ↑↓	Nom ↑↓	Categorie ↑↓	Prix ↑↓	Actions ↑↓	
31		Poulet Braisé	2	25	Détails	Effacer
33		Curry vert	2	17	Détails	Effacer
36		Pâtes pesto	2	10	Détails	Effacer
38		canapé saumon	1	19	Détails	Effacer
41		Salade	2	50	Détails	Effacer
Showing 1 to 5 of 5 entries				Previous	1	Next

jointure de tables

Une *jointure* est effectuée dans la requête SQL utilisée pour récupérer les informations du panier *cart* et les détails des produits *plats*:

Dans cette requête SQL, la jointure est effectuée avec la clause **JOIN**, reliant la colonne **id_product** de la table *cart* à la colonne *id* de la table *plats*. Cela crée une correspondance entre les produits dans le panier et les détails de ces produits.

Cette requête récupère les colonnes de la table *cart* ainsi que les colonnes de la table *plats* (nom, prix, photo) pour les produits qui correspondent à l'utilisateur actuel. La jointure est réalisée en utilisant les colonnes **id_product (clé étrangère)** et *id*.

```
***** AFFICHAGE DU PANIER *****/
$sid = (int)$_SESSION['id'];
$stmt = $bdd->prepare('SELECT cart.*, plats.nom, plats.prix, plats.photo FROM cart
JOIN `plats` ON `plats`.`id` = `cart`.`id_product`
WHERE user_id=:id;');
|
$stmt->execute(["id" => $sid]);
$query = $stmt->fetchAll();
?
?>
```

On peut donc établir en Bdd des liens entre différentes tables conformément à notre MCD et aux cardinalités.* prévues pour notre projet.

Vulnérabilité de sécurité et veille

II. Veille de sécurité

PHP: La veille technologique pour PHP implique la surveillance continue des mises à jour des nouvelles versions du langage il faut donc mettre à jour sa version de PHP.

phpMyAdmin, rien de bien différent, des sites recensent les vulnérabilités connues dont le site officiel: <https://www.phpmyadmin.net/security/> ils proposent des solutions mais le plus simple reste d'effectuer régulièrement les mises à jour.

Recherche et traduction

Pour ma jointure Sql je recherchais des éléments de compréhension et de contexte, j'ai donc effectué une recherche google:

Environ 20700 000 résultats (0,32 secondes)

Essayez avec cette orthographe : [SqI JOIN tables](#)

 PostgreSQL
<https://www.postgresql.org> › current :

Documentation: 16: 2.6. Joins Between Tables

This query is called a **left outer join** because the **table** mentioned on the left of the **join** operator will have each of its rows in the output at least once ...

 PostgreSQL FR
<https://docs.postgresql.fr> › tutorial-join :

2.6. Jointures entre les tables

Les requêtes peuvent accéder à plusieurs **tables** en même temps ou accéder à la même **table** de façon à ce que plusieurs lignes de la **table** soient traitées en même ...

2.6. Joins Between Tables

Thus far, our queries have only accessed one table at a time. Queries can access multiple tables at once, or access the same table in such a way that multiple rows of the table are being processed at the same time. Queries that access multiple tables (or multiple instances of the same table) at one time are called *join* queries. They combine rows from one table with rows from a second table, with an expression specifying which rows are to be paired. For example, to return all the weather records together with the location of the associated city, the database needs to compare the `city` column of each row of the `weather` table with the `name` column of all rows in the `cities` table, and select the pairs of rows where these values match.^[4] This would be accomplished by the following query:

```
SELECT * FROM weather JOIN cities ON city = name;
```

city	temp_lo	temp_hi	prcp	date	name	location
San Francisco	46	50	0.25	1994-11-27	San Francisco	(-194,53)
San Francisco	43	57	0	1994-11-29	San Francisco	(-194,53)
(2 rows)						

La réponse me paraissait suffisement pertinente pour que je prenne le temps de la lire et de la traduire avec Google Translate

2.6. Jointures entre les tables

Jusqu'à présent, nos requêtes n'ont accédé qu'à une seule table à la fois. Les requêtes peuvent accéder à plusieurs tables à la fois ou accéder à la même table de telle manière que plusieurs lignes de la table soient traitées en même temps. Les requêtes qui accèdent à plusieurs tables (ou à plusieurs instances de la même table) en même temps sont appelées requêtes de jointure. Ils combinent les lignes d'une table avec les lignes d'une deuxième table, avec une expression spécifiant quelles lignes doivent être appariées. Par exemple, pour renvoyer tous les enregistrements météorologiques ainsi que l'emplacement de la ville associée, la base de données doit comparer la colonne ville de chaque ligne de la table météo avec la colonne de nom de toutes les lignes de la table villes, et sélectionner les paires de lignes où ces valeurs correspondent.^[4] Cela serait accompli par la requête suivante :

J'ai donc trouvé une méthode appropriée et adaptable à mon code en allant consulter une documentation simple et claire sur Internet.

Conclusion

Le projet sur lequel j'ai travaillé m'a permis de consolider mes connaissances. J'ai revu une bonne partie des choses apprises lors de ma formation et même plus.

La meilleure manière d'apprendre à coder est de pouvoir réaliser un projet concret. J'ai pu créer une maquette, me questionner sur le modèle conceptuel des données pour élaborer la base de données, ce qui permet de ne pas se tromper pendant le processus de réalisation du projet. J'ai pu utiliser la méthode agile.

J'ai développé mon projet en PHP, le langage que j'ai appris lors de ma formation. J'ai commencé à créer mon arborescence et à mettre en place ma base de données. Pour m'aider dans la partie front-end, j'ai utilisé Bootstrap et sa bibliothèque. J'ai également utilisé un fichier CSS pour personnaliser le code.

Il y a encore beaucoup à améliorer, mon projet a de nombreuses fonctionnalités qui pourraient lui être ajoutées et de nombreux axes d'amélioration.

J'ai pour l'heure repris une formation à Bordeaux où j'habite désormais et où j'apprends Ruby et son framework "Ruby on rails"

L'univers du code et le métier de dév est infini tant par les langages et les framework que par les méthodes et les innovations constantes qu'il propose.

Posez un problème à quatre personnes et chacun aura sa méthode pour le résoudre et même si aucune n'est forcément mauvaise il y en a toujours une plus efficace.