

Dossier projet

TP - Concepteur développeur d'applications

Walid BOUGUERRA
04/06/2024

Ce dossier est réalisé dans le cadre de l'obtention du titre professionnel : Concepteur Développeur d'Applications, il présente un projet effectué durant ma formation au Greta Seine-et-Marne de Lognes.

Pour réaliser ce projet, j'ai mis en œuvre différentes compétences professionnelles :

- Développer une application sécurisée
- Concevoir et développer une application sécurisée organisée en couches
- Préparer le déploiement d'une application sécurisée

Sommaire

Sommaire.....	1
I. Introduction.....	2
Présentation du projet.....	2
Gestion et conduite du projet.....	3
II. Analyse des besoins.....	4
Diagramme de cas d'utilisation.....	4
Description de cas d'utilisation.....	5
Maquettes.....	7
Diagramme de séquence système.....	8
Diagramme de classe analyse.....	8
III. Conception.....	9
Diagramme de séquence conception.....	9
Diagramme de classe conception.....	10
MCD.....	11
MLD.....	12
Plan de test.....	13
Architecture logicielle.....	14
Choix technologiques.....	14
IV. Développement.....	15
Mise en place du MVC.....	15
Structure.....	15
Routeur et autoloader.....	15
Contrôleurs.....	17
Modèles.....	17
Vues.....	18
Client.....	19
Admin.....	20
Livreur.....	21
Code page d'accueil.....	21
Contrôleur.....	22
Modèle.....	22
Vue.....	22
Sécurisation de l'application.....	23
Hachage.....	23
Faille XSS.....	24
Injection SQL.....	25
Envoi de fichiers malicieux.....	25

Tests.....	26
Exécution du plan de test.....	26
Tests unitaires.....	28
Fonction à tester :.....	28
Classe de test :.....	28
Résultats du test :.....	29
Déploiement.....	29
Dockerfile.....	30
V. Conclusion.....	30
Difficultés.....	30
Améliorations.....	31
Bilan.....	31

I. Introduction

Présentation du projet

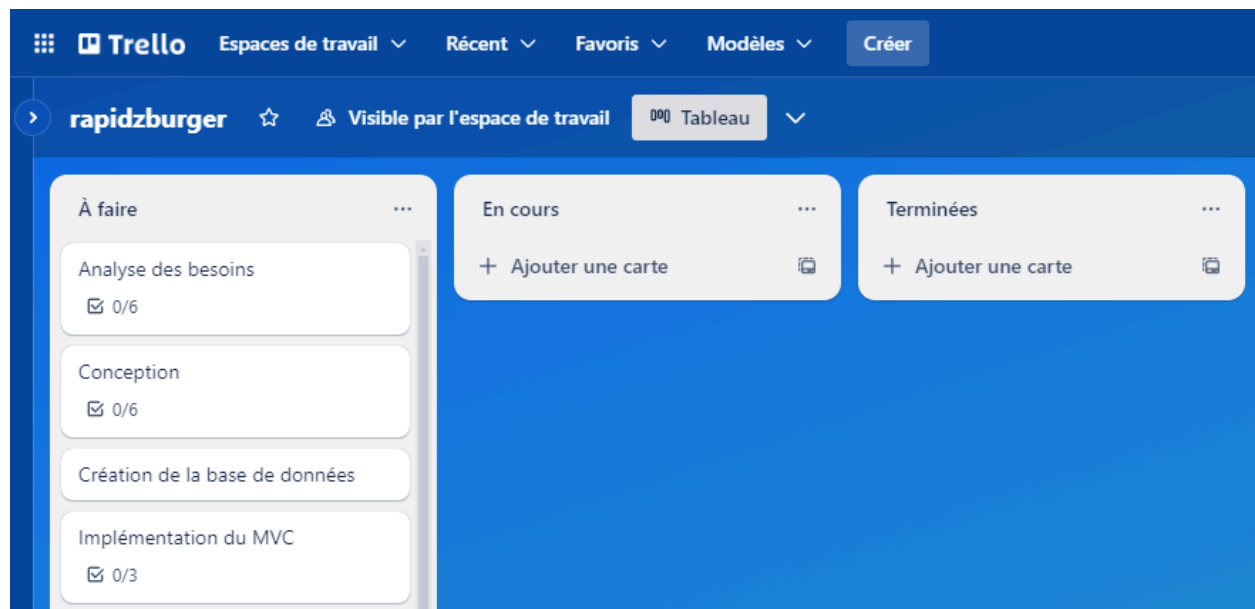
Un fast-food nommé RapidzBurger souhaite accroître son activité, pour atteindre son objectif elle souhaite mettre en place un site web permettant au clients de commander des burgers directement en ligne, cela permettrait d'augmenter la visibilité du restaurant en ayant une présence en ligne, mais aussi de faciliter la gestion des commandes via une interface web.

En tant que développeur, ma mission est d'apporter une solution en développant un site web de A à Z qui répond aux exigences de l'entreprise.

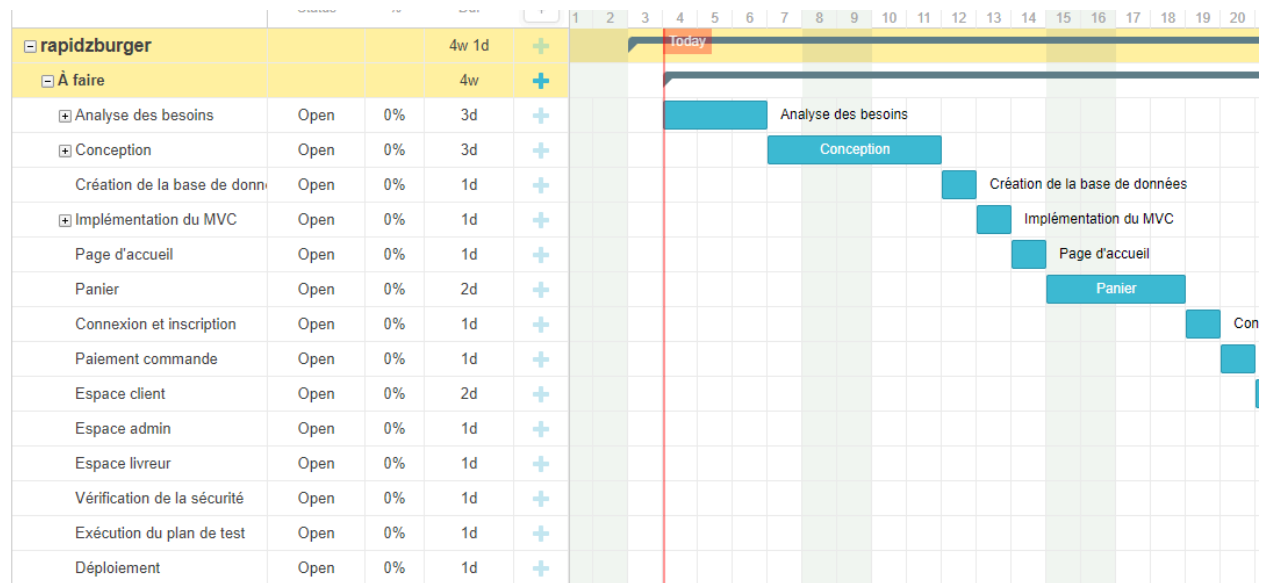
Pour réaliser un tel projet, je dois m'organiser, pour ce faire, j'ai mis en place une gestion et une conduite de projet.

Gestion et conduite du projet

J'ai utilisé Trello, c'est un outil de gestion de projet en ligne, il m'a permis de créer un tableau Kanban qui m'aide à visualiser le travail, limiter le travail en cours et maximiser l'efficacité.



J'ai ordonné les tâches à effectuer en fonction de leur priorité dans la réalisation du projet. À partir de ce tableau, j'ai mis en place un diagramme de Gantt qui constitue mon planning avec la prévision du temps de réalisation de chaque tâche du tableau Kanban.



Après m'être organisé pour gérer le projet, je peux commencer le projet par la première phase qui est l'analyse des besoins.

II. Analyse des besoins

Avant de développer l'application, une modélisation s'impose que ce soit dans la phase d'analyse ou de conception, pour ce faire, je vais utiliser UML qui est un langage de modélisation.

Au travers des différentes phases du projet, je vais réaliser des diagrammes uml pour représenter l'architecture et le fonctionnement de l'application graphiquement.

Le point de départ de mon processus de développement est la réflexion autour des cas d'utilisation, l'objectif est de comprendre les besoins du client pour rédiger un cahier des charges fonctionnel.

Pour analyser les besoins, nous devons répondre à deux questions :

- à quoi sert le système ? Pour définir les cas d'utilisations.
- qui utilise le système ? Pour définir les acteurs.

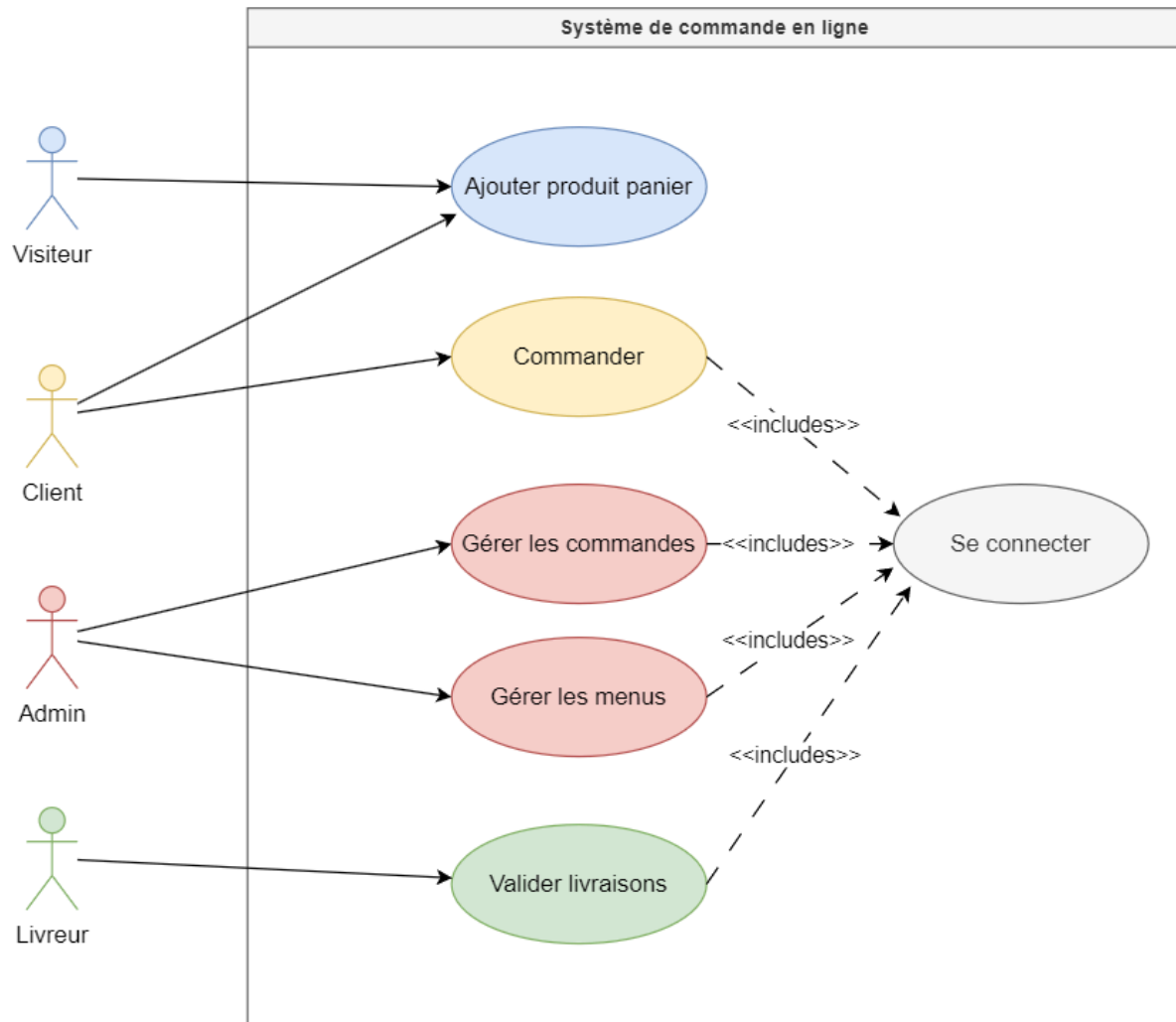
Les cas d'utilisations présentent les fonctionnalités principales du système, les acteurs sont les entités qui interagissent avec le système.

Pour répondre à ces questions, nous allons construire trois éléments :

- un diagramme de cas d'utilisation
- la description textuelle d'un cas d'utilisation
- un diagramme de séquence des scénarios d'utilisations

Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation permet de représenter graphiquement les cas d'utilisation, les acteurs du système ainsi que leurs associations.



Nous avons une première vision de l'application, mais elle reste très sommaire et manque d'informations, on va donc détailler les cas d'utilisations avec une description plus approfondie.

Description de cas d'utilisation

Nous allons faire une description textuelle des scénarios des cas d'utilisations pour représenter leurs déroulements.

Voici les scénarios pour le cas d'utilisation "commander" :

Nom	Commander
Acteur	Client
Pré-condition	Panier contenant au moins un produit
Données en entrée	Produits dans le panier
Scénario principal	<p>1. L'utilisateur clique sur le bouton "Procéder au paiement" dans son panier.</p> <p>Scénario alternatif : Client déjà connecté → Reprend à l'étape 4</p> <p>2. Le système redirige l'utilisateur vers la page de connexion.</p> <p>3. L'utilisateur saisit son email et son mot de passe puis il clique sur le bouton "Connexion".</p> <p>Scénario d'erreur : email ou mot de passe vide 3a. Le système affiche un message d'erreur : "Veuillez remplir tous les champs de connexion". → Retour à l'étape 2</p> <p>Scénario d'erreur : email ou mot de passe incorrect(s) 3b. Le système affiche un message d'erreur : "Email ou mot de passe incorrect(s) !". → Retour à l'étape 2</p> <p>4. Le système redirige le client vers la page de validation de commande.</p> <p>5. Le client saisit son adresse, numéro de téléphone et numéro de carte bancaire puis il clique sur le bouton "Payer".</p> <p>Scénario d'erreur : Les informations ne sont pas corrects 5a. Le système affiche un message d'erreur : "Les informations ne sont pas correctes !". → Retour à l'étape 4</p> <p>6. Le système redirige le client vers son espace avec son historique de commandes.</p>

En plus de mieux comprendre les cas d'utilisations, des éléments d'interface utilisateur interviennent dans les scénarios, ce qui m'a aidé pour la réalisation de maquettes graphiques.

Maquettes

The image displays two wireframe screenshots of a web application for 'RapidzBurger'.

Top Screenshot: Panier (Cart)

- Browser title: RapidzBurger
- URL: <https://rapidzburger/panier>
- Navigation links: [Accueil](#) | [Panier](#) | [Connexion](#) | [Inscription](#)
- Section: **Panier**
- Items list:
 - Item One
 - Item Two
 - Item Three
- Total: **Total : 50 €**
- Action button: **Procéder au paiement**

Bottom Screenshot: Livraison (Delivery) and Paiement (Payment)

- Browser title: RapidzBurger
- URL: <https://rapidzburger/checkout>
- Navigation links: [Accueil](#) | [Panier](#) | [Commandes](#)
- Section: **Livraison**
- Form fields:
 - Adresse:
 - Téléphone:
- Section: **Paiement**
- Form fields:
 - Numéro carte bancaire:
- Total: **Total commande : 50 €**
- Action button: **Payer**

Diagramme de séquence système

Le dernier outil pour réfléchir aux cas d'utilisation est le diagramme de séquence, il permet de représenter graphiquement les messages entre les acteurs et le système de manière chronologique.

On peut voir ici le déroulement du scénario du cas d'utilisation "commander" avec les données nécessaires et les erreurs qui peuvent survenir comme par exemple lorsque le client saisit des informations incorrectes.

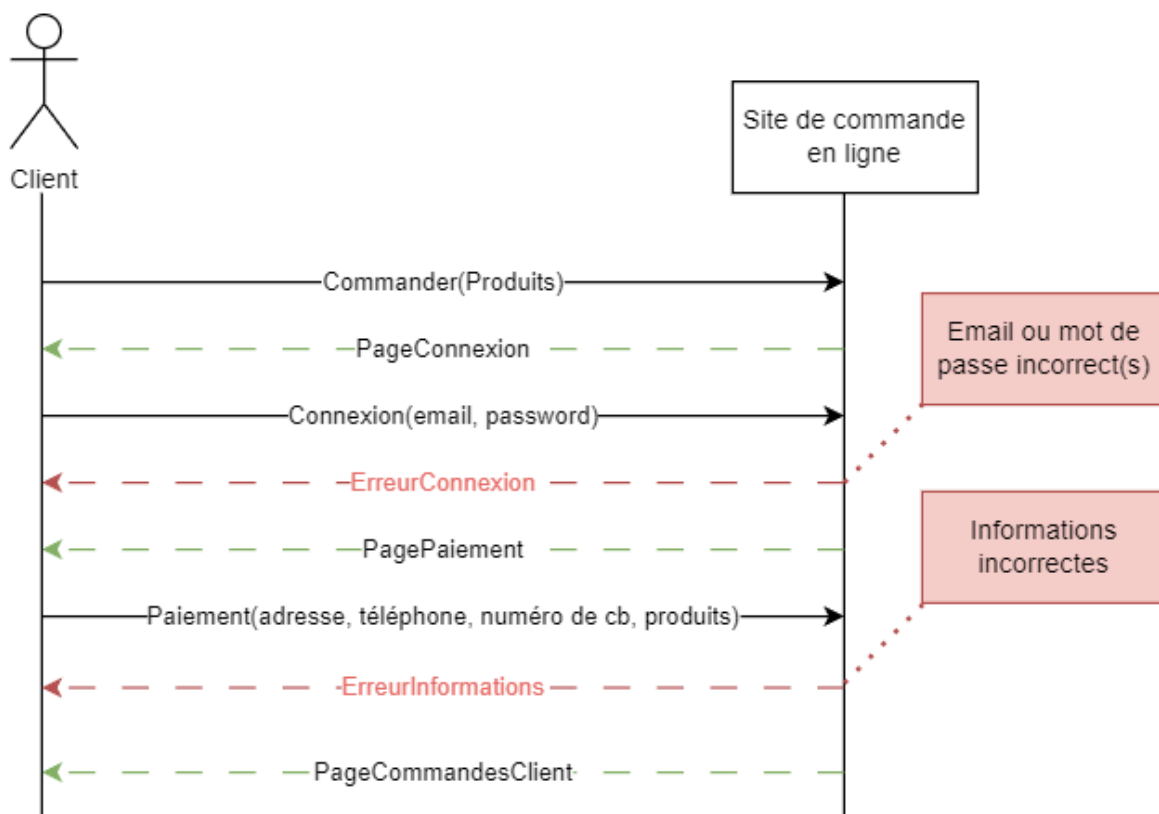
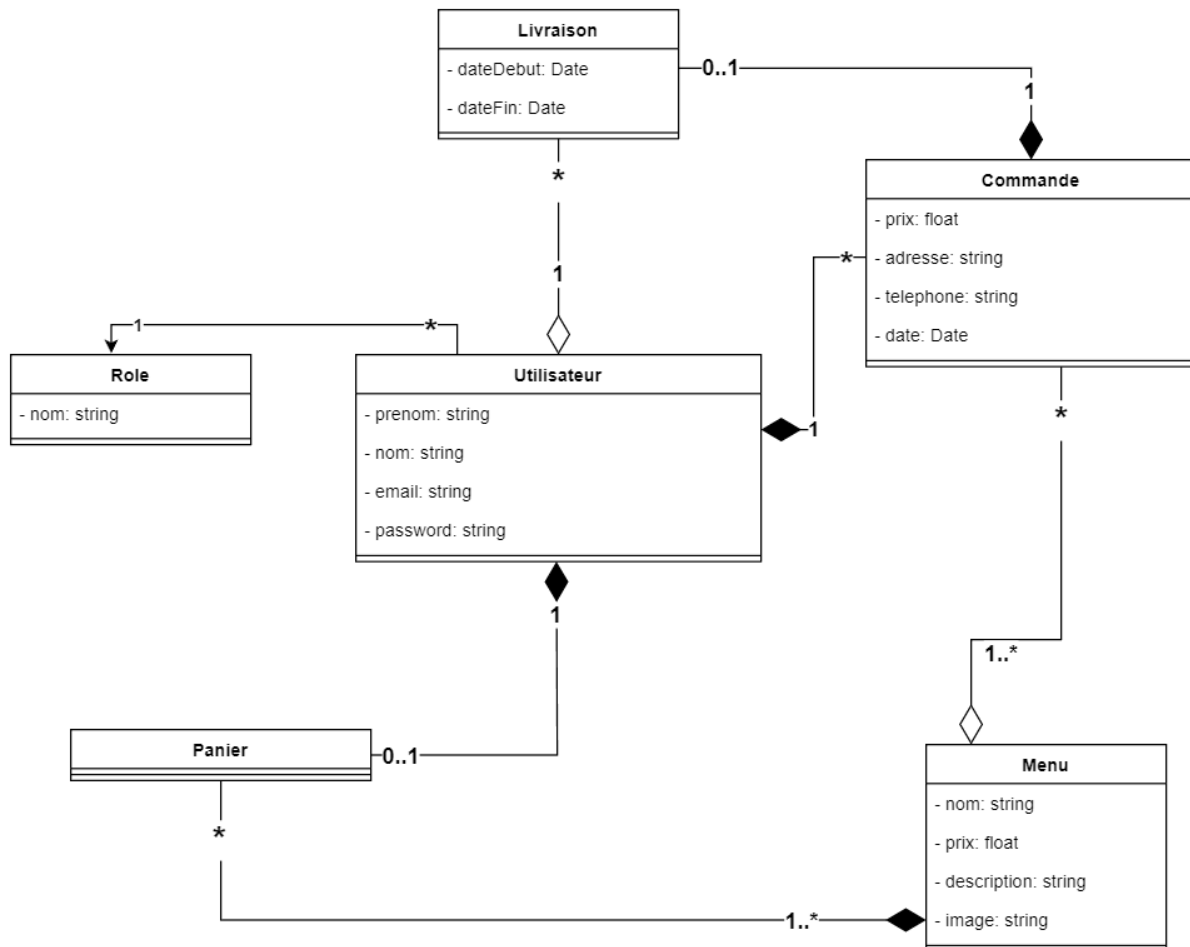


Diagramme de classe analyse

Après cette phase d'analyse, nous avons une meilleure vision de notre application ce qui nous permet de représenter le système comme un ensemble d'objets interagissant, pour ce faire, j'ai mis en place un diagramme de classe du domaine qui représente la structure interne du logiciel.

On a défini 6 classes avec leurs associations pour constituer notre système, cela va nous aider pour la phase de conception et de développement.



III. Conception

Diagramme de séquence conception

Voici le diagramme de séquence en phase de conception, il permet de décrire la réalisation des cas d'utilisation représentés par le diagramme de classes, on a une représentation temporelle des échanges de messages entre les objets du système de façon chronologique.

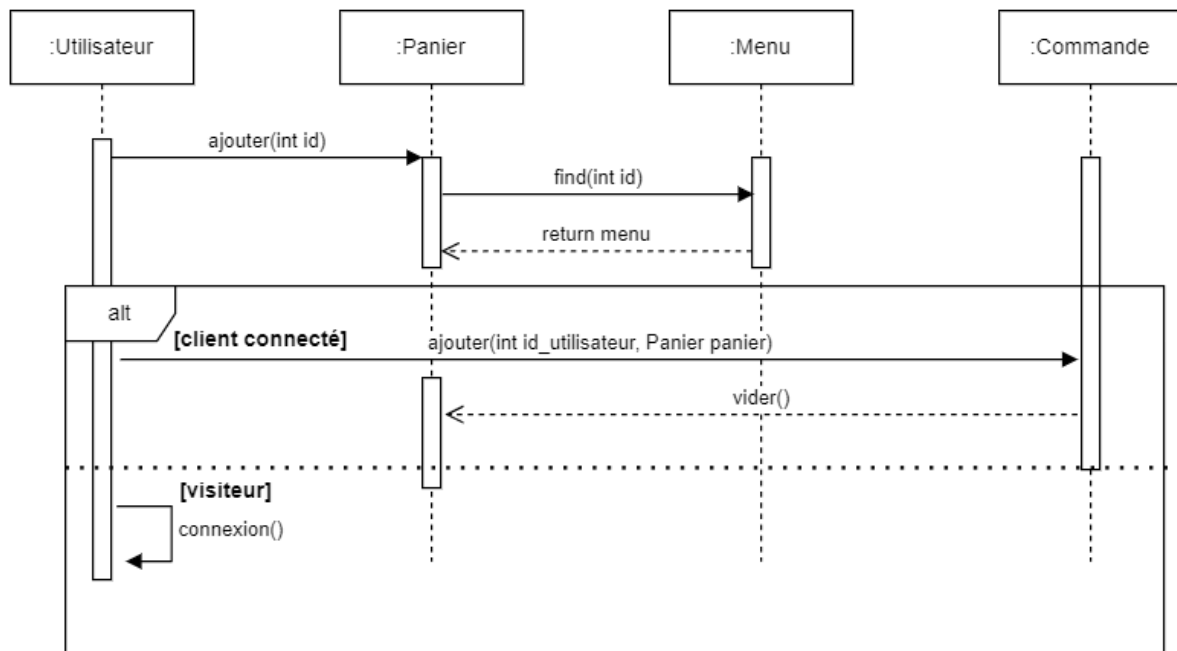
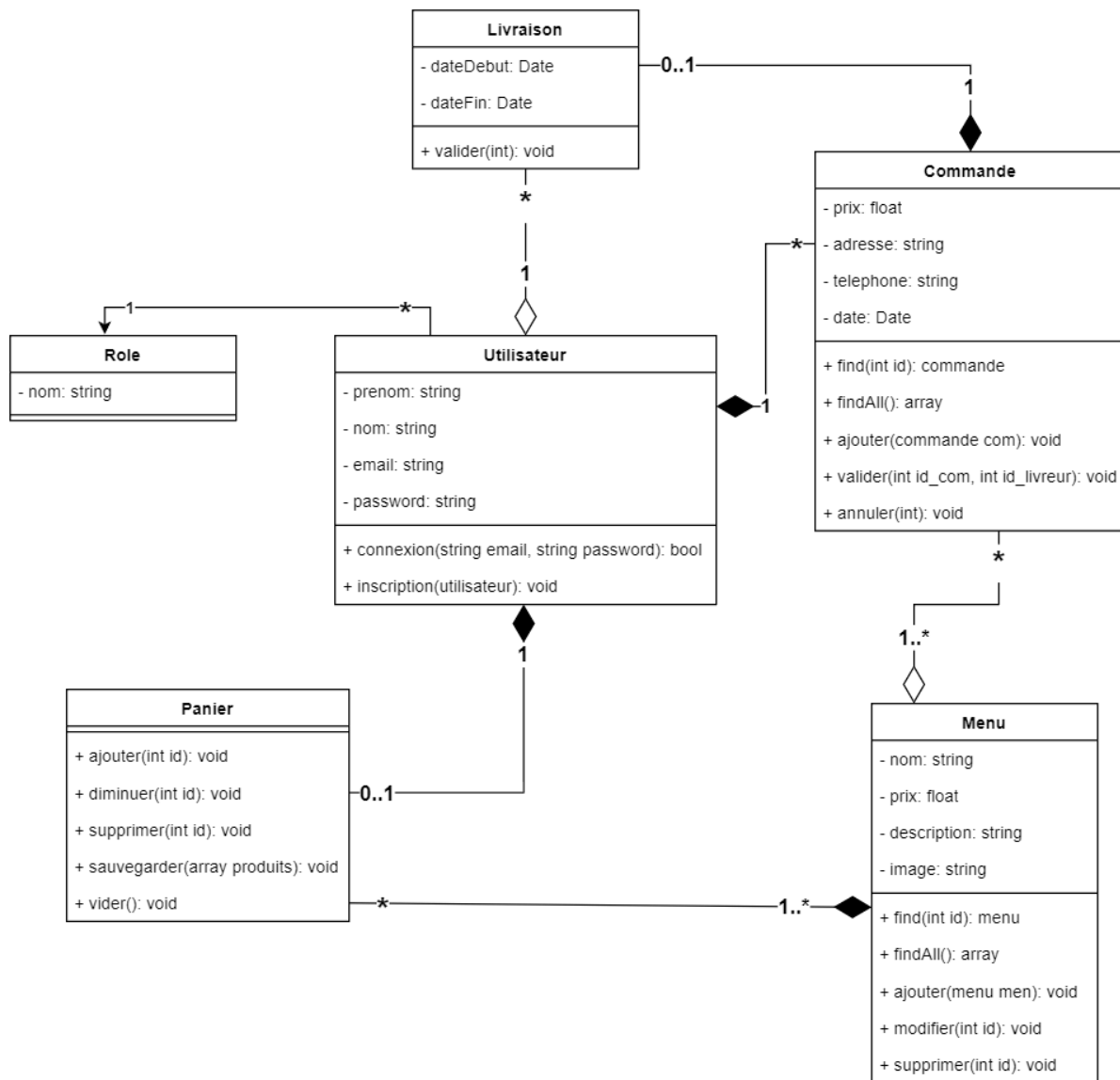


Diagramme de classe conception

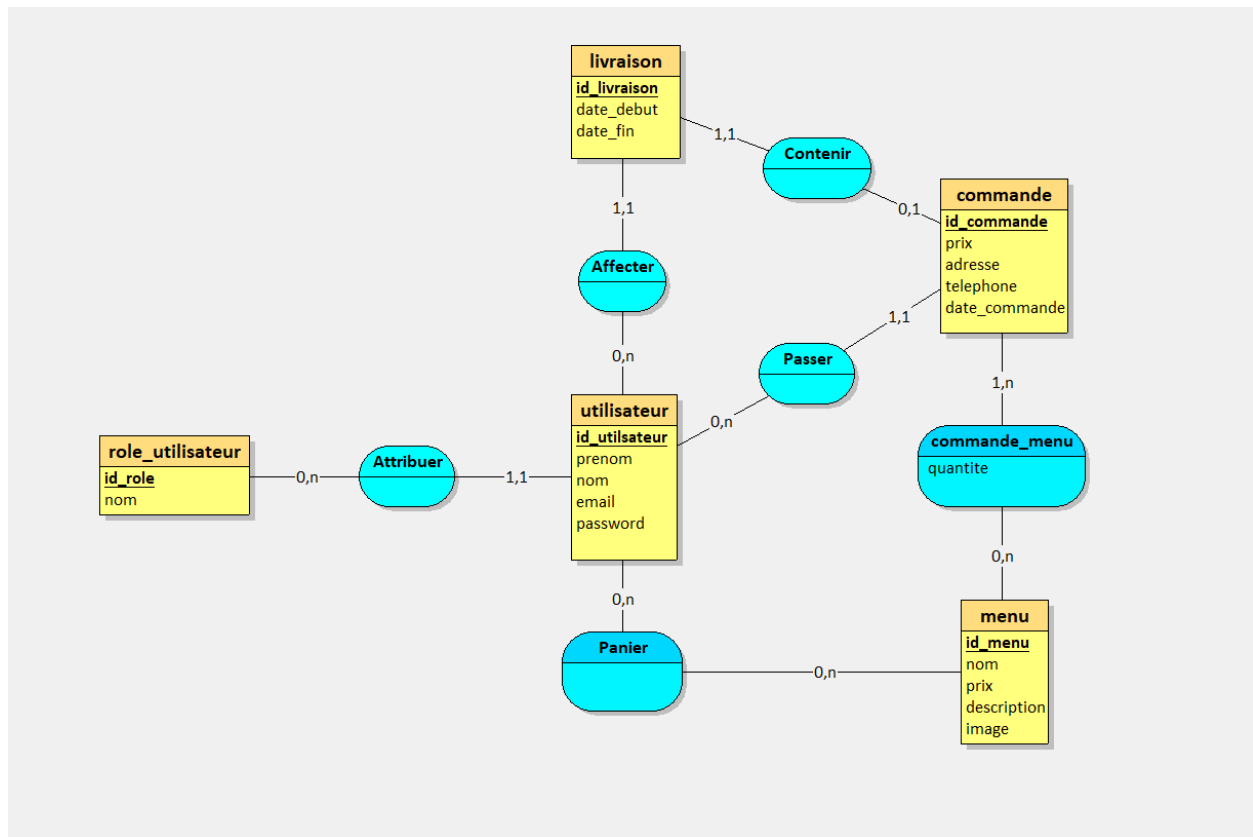
Suite au diagramme de séquence conception où nous avons défini les messages entre objets qui se traduisent en méthodes, nous pouvons désormais les ajouter à notre diagramme de classe qui est maintenant complet.



MCD

Nous allons utiliser une base de données pour stocker les données de notre application, avant de la mettre en place, il faut la modéliser, pour cela, nous allons créer un MCD , c'est une des premières étapes lors de la conception de base de données avec Merise.

On peut retrouver nos 6 entités qui correspondent aux 6 classes du diagramme de classes, on a aussi défini leurs relations avec les cardinalités associées.



MLD

Pour que nos entités deviennent des tables de notre base de données, nous allons les traduire textuellement, c'est le MLD.

MLD

```

menu = (id_menu INT, nom VARCHAR(250), prix DECIMAL(4,2), description TEXT, image VARCHAR(250));
role_utilisateur = (id_role BYTE, nom VARCHAR(250));
utilisateur = (id_utilisateur INT, prenom VARCHAR(250), nom VARCHAR(250), email VARCHAR(250), password VARCHAR(250), #id_role);
commande = (id_commande INT, prix DECIMAL(4,2), adresse VARCHAR(250), telephone VARCHAR(10), date_commande DATETIME, #id_utilisateur);
livraison = (id_livraison INT, date_debut DATETIME, date_fin DATETIME, #id_commande, #id_utilisateur);
commande_menu = (#id_menu, #id_commande, quantite BYTE);
Panier = (#id_utilisateur, #id_menu);
  
```

Plan de test

Suite aux phases précédentes, on a réussi à définir les fonctionnalités de l'application, lors de développement de celles-ci, il faudra les tester avant de les mettre en production pour vérifier qu'elles remplissent correctement leur fonction.

Pour ce faire j'ai mis en place un plan de test qu'il faudra exécuter lors de la phase de développement.

Fonctionnalité	Étapes	Données en entrée	Résultat attendues	Résultat obtenu
Connexion (email et mot de passe corrects)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe : test	L'utilisateur devrait être redirigé sur son espace client avec ses commandes.	
Connexion (email ou mot de passe incorrects)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe : incorrect	L'utilisateur devrait être redirigé sur la page de connexion avec un message d'erreur "Email ou mot de passe incorrect(s) !"	
Connexion (email ou mot de passe vide)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe :	L'utilisateur devrait être redirigé sur la page de connexion avec un message d'erreur "Veuillez remplir tous les champs !"	
Commander (utilisateur non-connecté)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande.	- Panier avec des produits	L'utilisateur est redirigé vers la page de connexion pour se connecter	
Commander (panier vide)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande.	- Panier vide	L'utilisateur reste sur la page panier	

Commander (utilisateur connecté avec un panier)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande. 3. Il saisit son adresse et son numéro de téléphone pour la livraison 4. Il saisit son numéro de carte bancaire pour le paiement 5. Il clique sur le bouton "Payer"	Adresse : 2 rue du test Numéro : 0612345678 Numéro de cb : 1111-1111-1111-1111	L'utilisateur est redirigé vers son espace client avec sa nouvelle commande inscrite	
---	--	--	--	--

Architecture logicielle

Pour l'architecture de l'application, j'ai fait le choix du MVC, c'est un modèle de conception logiciel qui repose sur la séparation des différentes couches d'une application : la logique métier et l'affichage, ce qui permet une meilleure répartition du travail et une maintenance améliorée.

Choix technologiques

OS : Windows 10 car c'est mon OS principal.

Éditeur de code : VSCode, je suis habitué à cet éditeur avec mes raccourcis et extensions pour être plus productif.

Environnement de développement : Laragon, c'est un environnement portable, isolé, rapide et puissant pour Windows qui me permet de bénéficier d'un serveur web Apache, PHP et MySQL.

Outils de versionning : Git & GitHub pour sauvegarder les versions du projet.

Back-end :

- PHP 8 car c'est le principal langage de script côté serveur pour créer des sites web, il est simple à utiliser, stable et bien maintenu.
- MySQL 8 pour ma base de données, car c'est un SGBD polyvalent, sécurisé et bien documenté.

Front-end : Bootstrap 5.3 est un framework css qui me permet de créer des interfaces responsives rapidement et facilement grâce à des classes, un système de grille et de composants.

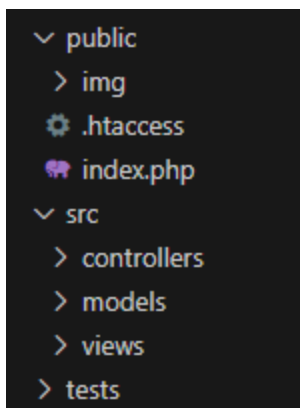
IV. Développement

Mise en place du MVC

J'ai fait le choix de ne pas utiliser de framework PHP pour rester simple et léger avec la structure de mon projet sans être limité par un framework.

Structure

La première étape de mon développement est la mise en place de la structure des dossiers.



Le dossier "public" contient les images, le fichier index.php qui est le point d'entrée de l'application ainsi qu'un fichier .htaccess qui permet de récupérer les paramètres de l'URL pour mon routeur.

Le dossier "src" contient la structure MVC, il est séparé du fichier index.php pour pas que les visiteurs ne puissent accéder à ce dossier.

Routeur et autoloader

La deuxième étape est la construction de mon routeur dans le fichier index.php qui me permet d'appeler un contrôleur, une méthode associée avec un potentiel id.

Je récupère ces informations avec les paramètres de l'URL, le premier paramètre est le contrôleur, le second, l'action et le troisième un id.


```
// Router
$url = $_GET['url'] ?? null;
$url = explode("/", filter_var($url, FILTER_SANITIZE_URL));

$controller = $url[0] ?? null;
$action = $url[1] ?? null;
$id = $url[2] ?? null;

if (file_exists('../src/controllers/' . $controller . 'Controller.php') && !empty($controller)) {
    $controller = 'Controllers\\' . $controller . 'Controller';
    $controller = new $controller();
    if (isset($action)) {
        if (method_exists($controller, $action)) {
            if (isset($id)) {
                $controller->$action($id);
            } else {
                $controller->$action();
            }
        } else {
            $controller->index();
        }
    } else {
        $controller->index();
    }
} else {
    $controller = new Controllers\\MenuController();
    $controller->index();
}
```

Pour appeler ces classes, je dois les instancier dans le fichier index.php, pour cela, j'utilise l'autoloader de composer qui est un gestionnaire de dépendances pour PHP.

```
require '../vendor/autoload.php';
```

```
"autoload": {
    "psr-4": {
        "App\\": "src/",
        "Controllers\\": "src/controllers/",
        "Models\\": "src/models/"
    }
},
```

Contrôleurs

La troisième étape est la mise en place des contrôleurs qui sont les intermédiaires entre les modèles et les vues.

La classe Controller contient une méthode render() qui permet de rendre une vue avec des données, cette classe est héritée par tous les autres contrôleurs pour bénéficier de cette méthode.

```
<?php
namespace Controllers;
class Controller
{
    public function render($view, $data = [])
    {
        $nav = $this->navbarRender();

        ob_start();

        extract($data);

        include '../src/views/' . $view . '.php';

        $contenu = ob_get_clean();

        include '../src/views/layouts/default.php';
    }
}
```

Modèles

Les données viennent des modèles qui se chargent de faire des requêtes à la base de données pour lire, ajouter, modifier ou supprimer des données.

```
namespace Models;
use PDO;

abstract class Model {
    private $dsn="mysql:dbname=rapidzburger;host=localhost;charset=utf8";
    private $login="root";
    private $password="root";
    protected $pdo;
    protected $table;

    public function __construct(){
        if($this->pdo == null){
            $this->pdo = new PDO($this->dsn, $this->login, $this->password);
            $this->pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
            $this->pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,PDO::FETCH_OBJ);
        }
    }

    public function findById(int $id)
    {
        $query = $this->pdo->prepare("SELECT * FROM $this->table WHERE id = :id");
        $query->execute(['id' => $id]);
        return $query->fetch();
    }
}
```

Vues

La méthode render affiche une vue qui est un fichier php qui contient principalement du HTML avec du code PHP pour afficher dynamiquement les données.


Les vues représentent aussi l'implémentation des maquettes.

Découvrez nos burgers

Faites-vous livrer nos burgers frais préparés à partir de recettes élaborées par des chefs reconnus.



Menus



Steakhouse
12.00 €


Le plaisir d'une viande de bœuf grillée à la flamme, d'une sauce barbecue, de bacon, des tranches de cheddar fondu et d'oignons croustillants.

1x Steakhouse

1x Moyennes frites

1x Coca-Cola sans sucres (40cl)

Ajouter au panier



Double Cheese Bacon
13.00 €


Ce sandwich va à l'essentiel. Deux viandes de bœuf grillées à la flamme, des tranches de cheddar fondu et du bacon.

1x Steakhouse

1x Moyennes frites

1x Coca-Cola sans sucres (40cl)

Ajouter au panier



Chicken Spicy
12.00 €


Une sauce Spicy Andalouse (relevée), un pain avec éclats de maïs, des tranches de cheddar fondu et un poulet 100% issu de filet.

1x Steakhouse

1x Moyennes frites

1x Coca-Cola sans sucres (40cl)

Ajouter au panier



Indian Chicken Steakhouse
13.00 €

Un pain moelleux, une sauce curry, des oignons croustillants, des tranches de bacon, des tranches de cheddar fondu et un poulet 100% issu de filet !

1x Steakhouse

1x Moyennes frites

1x Coca-Cola sans sucres (40cl)

Ajouter au panier

© 2024 RapidzBurger

Panier

Produit	Quantité	Prix
Menu : Double Cheese Bacon	<div>- 1 +</div>	13 € <div>X</div>
Menu : Steakhouse	<div>- 1 +</div>	12 € <div>X</div>

Total : 25 €

Procéder au paiement



Panier

Mes commandes

Déconnexion

Livraison

Adresse

Téléphone

Paie ment

Numéro carte bancaire

Total commande : 25 €

Payer



Panier

Mes commandes

Déconnexion

Mes commandes

En attentes

Menus	Prix	Date
<ul style="list-style-type: none">1x Menu Steakhouse1x Menu Double Cheese Bacon	25 €	2024-06-06 14:16:54


En cours

Menus	Prix	Date
-------	------	------

Livrées

Menus	Prix	Date
-------	------	------

Admin



Rapidz
Burger

Commandes

Menus

Déconnexion

Commandes

En attentes

Numéro	Menus	Prix	Date	Client	Livraison
8	<ul style="list-style-type: none">1x Menu Steakhouse1x Menu Double Cheese Bacon	25 €	2024-06-06 14:16:54	Walid Bouguerra 0600000000 2 rue du test	<div>Jean Pierre</div> <div>ValiderAnnuler</div>

En cours

Numéro	Menus	Prix	Date	Client	Livreur
--------	-------	------	------	--------	---------

Livrées

Numéro	Menus	Prix	Client	Livreur	Date livraison
--------	-------	------	--------	---------	----------------

Livreur



Rapidz
Burger

Déconnexion

Livraisons

Date	Menus	Client
2024-06-06 14:18:15	<ul style="list-style-type: none">1x Menu Steakhouse1x Menu Double Cheese Bacon	Walid Bouguerra 2 rue du test 0600000000

Valider

Code page d'accueil

Voici le code pour l'affichage de la page d'accueil :

Contrôleur

```
class MenuController extends Controller {

    private $menuModel;

    public function __construct()
    {
        $this->menuModel = new MenuModel();
    }

    // Page d'accueil
    public function index()
    {
        $menus = $this->menuModel->findAll();
        $this->render('home', compact('menus'));
    }
}
```

Modèle

```
class MenuModel extends Model {
    protected $table = "menu";
}
```

Vue

```
<section class="mt-4">
    <h2 class="mb-3">🍔 Menus</h2>
    <div class="row">
        <?php foreach($menus as $menu): ?>
            <div class="col-lg-3 col-md-6 col-sm-12 mb-2">
                <div class="card h-100" style="max-width: 18rem;">
                    nom ?>">
                    <div class="card-body d-flex flex-column">
                        <h5 class="card-title"><?=$menu->nom ?></h5>
                        <h6><?=$menu->prix ?> €</h6>
                        <p class="card-text"><?=$menu->description ?></p>
                        <a href="/panier/add/<?=$menu->id ?>" class="btn btn-danger mt-auto">🛒 Ajouter au panier</a>
                    </div>
                </div>
            </div>
        <?php endforeach; ?>
    </div>
</section>
```

Sécurisation de l'application

Pour sécuriser l'application, j'ai suivi les recommandations de l'OWASP, une organisation qui est consacrée à la sécurité des applications web.

Hachage

Les mots de passe ne doivent pas être stockés en clair dans la base de données, car s'il y a une faille, un attaquant pourrait récupérer le couple identifiant/mot de passe en clair et accéder à l'application, mais aussi l'essayer sur d'autres sites.

Le hachage est un algorithme permettant de chiffrer de manière irréversible via une fonction de hachage cryptographique ce qui rend la tâche très difficile pour l'attaquant de connaître le mot de passe d'origine.

Lors de l'inscription d'un utilisateur, j'utilise la fonction PHP `password_hash()` pour hacher les mots de passe.

password_hash

(PHP 5 >= 5.5.0, PHP 7, PHP 8)

`password_hash` — Crée une clé de hachage pour un mot de passe

Description

```
password_hash(#\[\SensitiveParameter\] string $password, string|int|null $algo, array $options = []): string
```

```
public function add(string $prenom, string $nom, string $email, string $password)
{
    $query = $this->pdo->prepare("INSERT INTO $this->table VALUES(null, 1, :prenom, :nom, :email, :password)");
    $query->execute([
        'prenom' => $prenom,
        'nom' => $nom,
        'email' => $email,
        'password' => password_hash($password, PASSWORD_DEFAULT)
    ]);
}
```

Ensuite, lors de la connexion pour vérifier si le mot de passe entré correspond à celui haché dans la base de données; j'utilise la fonction `password_verify()`.

password_verify

(PHP 5 >= 5.5.0, PHP 7, PHP 8)

password_verify — Vérifie qu'un mot de passe correspond à un hachage

Description

```
password_verify(#[\SensitiveParameter] string $password, string $hash): bool
```

```
$query = $this->pdo->prepare("SELECT * FROM $this->table WHERE email = :email");
$query->execute(['email' => $email]);
$user = $query->fetch();
if (!empty($user)) {
    if (password_verify($password, $user->password)) {
        return $user;
    } else {
        return null;
    }
}
```

Faible XSS

Une faille XSS consiste à injecter du code malveillant HTML ou JavaScript via les paramètres d'entrée côté client. Pour m'en prémunir, j'ai utilisé la fonction htmlspecialchars() de PHP.

htmlspecialchars

(PHP 4, PHP 5, PHP 7, PHP 8)

htmlspecialchars — Convertit les caractères spéciaux en entités HTML

```
$nom = htmlspecialchars($_POST['nom']);
$prix = htmlspecialchars($_POST['prix']);
$description = htmlspecialchars($_POST['description']);
```

Injection SQL

L'une des injections les plus connues est l'injection SQL qui consiste à injecter du code SQL qui permet à l'attaquant de modifier des requêtes afin d'accéder à la base de données.

Pour se protéger de cette attaque, j'ai utilisé des requêtes préparées qui permettent d'isoler les données par rapport aux requêtes.

```
$query = $this->pdo->prepare("INSERT INTO menu VALUES(null, :nom, :prix, :description, :image)");  
$query->execute([  
    'nom' => $nom,  
    'prix' => $prix,  
    'description' => $description,  
    'image' => $image  
]);
```

Envoi de fichiers malicieux

L'administrateur du site peut ajouter des menus via un formulaire d'ajout, dans celui-ci, il doit ajouter une image par le biais d'un input de type file, cela peut être une faille, car un utilisateur par erreur ou par malveillance pourrait envoyer un fichier malicieux sur le serveur.

Pour m'en prémunir, j'effectue une vérification de l'extension du fichier ainsi que de la taille.

```
$fileName = $_FILES['image']['name'];
$fileTempName = $_FILES['image']['tmp_name'];
$fileSize = $_FILES['image']['size'];
$fileError = $_FILES['image']['error'];
$fileType = $_FILES['image']['type'];

$newFileName = str_replace(" ", "-", strtolower($_POST['nom']));

$fileExt = explode('.', $fileName);
$fileActualExt = strtolower(end($fileExt));

$allowedExt = ['jpg', 'jpeg', 'png', 'webp'];

if (in_array($fileActualExt, $allowedExt)) {
    if ($fileError === 0) {
        if ($fileSize < 50000) {
            $newFileName.= '.' . $fileActualExt;
            $fileDestination = 'img/' . $newFileName;
            move_uploaded_file($fileTempName, $fileDestination);

            $nom = htmlspecialchars($_POST['nom']);
            $prix = htmlspecialchars($_POST['prix']);
            $description = htmlspecialchars($_POST['description']);
            $this->menuModel->add($nom, $prix, $description, $newFileName);
        }
    }
}
```

Tests

Exécution du plan de test

J'ai exécuté le plan de test pour vérifier que le résultat obtenu correspond bien au résultat attendu si c'est le cas, je mets la fonctionnalité en production sinon je fais une correction.

Fonctionnalité	Étapes	Données en entrée	Résultat attendues	Résultat obtenu
Connexion (email et mot de passe corrects)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe : test	L'utilisateur devrait être redirigé sur son espace client avec ses commandes.	L'utilisateur est redirigé sur son espace client avec ses commandes.
Connexion (email ou mot de passe incorrects)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe : incorrect	L'utilisateur devrait être redirigé sur la page de connexion avec un message d'erreur "Email ou mot de passe incorrect(s) !!"	L'utilisateur est redirigé sur la page de connexion avec un message d'erreur "Email ou mot de passe incorrect(s) !!"
Connexion (email ou mot de passe vide)	1. L'utilisateur se rend la page de connexion en cliquant sur le bouton "Se connecter" de la barre de navigation. 2. Il saisit son email et son mot de passe 3. Il clique sur le bouton "Connexion"	email : test@gmail.com mot de passe :	L'utilisateur devrait être redirigé sur la page de connexion avec un message d'erreur "Veuillez remplir tous les champs !"	L'utilisateur est redirigé sur la page de connexion avec un message d'erreur "Veuillez remplir tous les champs !"
Commander (utilisateur non-connecté)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande.	- Panier avec des produits	L'utilisateur est redirigé vers la page de connexion pour se connecter	L'utilisateur est redirigé vers la page de connexion pour se connecter
Commander (panier vide)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande.	- Panier vide	L'utilisateur reste sur la page panier	L'utilisateur est redirigé vers la page d'accueil
Commander (utilisateur connecté avec un panier)	1. L'utilisateur se rend dans son panier 2. Il clique sur le bouton "Procéder au paiement" pour valider sa commande. 3. Il saisit son adresse et son numéro de téléphone pour la livraison 4. Il saisit son numéro de carte bancaire pour le paiement 5. Il clique sur le bouton "Payer"	Adresse : 2 rue du test Numéro : 0612345678 Numéro de cb : 1111-1111-1111-11 11	L'utilisateur est redirigé vers son espace client avec sa nouvelle commande inscrite	L'utilisateur est redirigé vers son espace client avec sa nouvelle commande inscrite

Tests unitaires

J'ai aussi mis en place des tests unitaires avec PHPUnit qui est un framework de tests unitaires dédié à PHP pour vérifier que les exécutions de bouts de codes correspondent aux assertions prédéfinies.

Fonction à tester :

```
// Vérification de la validité du mot de passe
public function verifPassword(string $password)
{
    $pattern = "/^(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9])(?=.*?[#?!@$%^&*~]).{8,}$/";

    if (preg_match($pattern, $password)) {
        return true;
    } else {
        throw new \Exception('Le mot de passe n\'est pas valide !');
        return false;
    }
}
```

Classe de test :

```
<?php
use Controllers\ClientController;
use PHPUnit\Framework\TestCase;

class ClientControllerTest extends TestCase {

    public function testVerifPassword()
    {
        $clientController = new ClientController();
        $this->expectException(\Exception::class);
        $clientController->verifPassword("test");
    }
}
```

Résultats du test :

```
Runtime:      PHP 8.1.10

.                                                    1 / 1 (100%)

Time: 00:00.507, Memory: 6.00 MB

OK (1 test, 1 assertion)
PS C:\Users\Admin\Desktop\rapidzburger> 
```

Le mot de passe "test" n'est pas valide, la fonction lance bien une exception.

```
$clientController->verifPassword("-Secr3t.");
```

```
There was 1 failure:

1) ClientControllerTest::testVerifPassword
Failed asserting that exception of type "Exception" is thrown.

FAILURES!
Tests: 1, Assertions: 1, Failures: 1.
PS C:\Users\Admin\Desktop\rapidzburger> 
```

Ici le mot de passe "-Secre3t." est valide et aucune exception n'est lancée.

Déploiement

Docker est un outil qui peut emballer une application et ses dépendances dans un conteneur isolé qui pourra être exécuté sur n'importe quel environnement, on a l'avantage que notre code marchera partout sans problème de compatibilité.

J'ai utilisé Docker pour déployer l'application, pour ce faire, j'ai créé une Image docker qui contient tout ce qui est nécessaire au fonctionnement de l'application : le code, les dépendances et les configurations. Pour créer cette image, il faut créer un Dockerfile, un fichier qui est la recette avec toutes les instructions pour créer notre image.

Dockerfile

```

1 FROM debian:stable-slim
2 LABEL version="1.0"
3 ARG APT_FLAGS="-q -y"
4 ARG DOCUMENTROOT="/var/www/html"
5 RUN apt-get update -y && apt-get install ${APT_FLAGS} apache2
6 RUN apt-get install ${APT_FLAGS} mariadb-server
7 COPY db/database.sql /
8 RUN apt-get install ${APT_FLAGS} \
9     php-mysql \
10    php && \
11    rm -f ${DOCUMENTROOT}/index.html && \
12    apt-get autoclean -y
13 COPY public ${DOCUMENTROOT}/public
14 COPY src ${DOCUMENTROOT}/src
15 EXPOSE 80
16 WORKDIR /var/www/html
17 ENTRYPOINT service mariadb start && mariadb < /database.sql && apache2ctl -D FOREGROUND

```

Après avoir construit l'image avec la commande "docker build -t rapidzburger .", docker crée un conteneur qu'on peut démarrer avec la commande "docker run -d --name rapidzburger_c -p 8080:80 rapidzburger", on peut ensuite ouvrir l'application sur le port définit : localhost:8080.

Cette image pourra être partagée avec d'autres développeurs pour travailler sur l'application avec le même environnement.

V. Conclusion

Difficultés

J'ai eu des difficultés dans les premières modélisations de mes diagrammes, ils étaient trop compliqués et peu lisibles avec l'abondance d'informations pas toujours nécessaires, on perdait donc leurs utilités.

J'ai repris ces diagrammes en les rendant faciles à comprendre en les simplifiant, pour ce faire, j'ai retiré les informations inutiles ou évidentes.

Améliorations

Le développement d'une application mobile fait partie d'une future amélioration du système, elle permettrait de faciliter le travail des livreurs avec leurs livraisons détaillées directement sur leurs téléphones.

Améliorer le processus de déploiement en mettant en place un pipeline CI/CD, c'est une pratique Devops qui consiste en l'intégration continue et le déploiement continu pour automatiser et accélérer le développement de logiciels.

Bilan

Ce que je retiens de ce projet est l'importance de la modélisation dans les phases d'analyse et de conception dans le développement d'application, tous les documents qui en découle permettent de collaborer plus facilement avec les autres, ça facilite aussi le développement, car on a une idée précise et graphique de ce qu'on doit développer.