

# **Sommaire**

1. Tableau de Synthèse
2. Attestation de Travail
3. Description succincte des projets
  - Projet 1 : 1ClickAllEat
  - Projet 2 : LDAP – Telora
  - Projet 3 : Portfolio
4. Veille technologique
5. Annexe
  - Projet 1 :
    -
  - Projet 2 :
    -
  - Projet 3 :
    -

1. Tableau de Synthèse  
(a insérer une fois imprimer )

	Gérer le patrimoine informatique	Répondre aux incidents et aux demandes d'assistance et d'évolution	Développer la présence en ligne de l'organisation	Travailler en mode projet	Mettre à disposition des utilisateurs un service informatique	Organiser son développement professionnel
<b>Portfolio</b>			X	X		X
<b>1ClickAllEat</b>			X	X	X	X
<b>Logiciel LDAP</b>	X	X		X	X	X

## 2. Attestation de Travail

A insérer : attestation fournie par l'entreprise ou le tuteur de stage/alternance si applicable.

### 3. Descriptions Succinctes des Projets

#### Projet 1 : 1ClickAllEat

Contexte : Projet réalisé en autonomie dans le cadre du BTS SIO SLAM, Pour répondre à un besoin concret : fluidifier la gestion des réservations et des commandes dans la restauration, tout en offrant une expérience utilisateur moderne, rapide et sécurisée.

Partenaire : Aucun (projet individuel)

Environnement technique :

- Front-end : Blade (Laravel), Bootstrap 5, HTML5, CSS3, responsive design
- Back-end : Laravel 10 (PHP 8), architecture MVC
- Base de données : SQLite (développement), MySQL/PhpMyAdmin (production)
- Outils : Breeze (authentification), Faker (données de test), GitHub, VS Code, hébergement HTTPS

Liste des tâches :

- Analyse du besoin, rédaction du cahier des charges et du diagramme de gantt
- Modélisation de la base de données et création des migration Laravel
- Développement des fonctionnalités principal : réservation, commande, gestion des utilisateurs, gestion des restaurants/tables/menus

- Implémentation de l'authentification sécurisée multi-profils (clients, restaurateurs)
- Développement d'un tableau de bord pour restaurateur (statistique, gestion des réservations et commandes)
- Tests unitaires, validation des formulaires et gestion des erreurs
- Déploiement sur serveur distant, configuration du HTTPS et optimisation de la sécurité
- Rédaction de la documentation utilisateur et technique

Difficultés rencontrées / solutions :

- Découverte et prise en main de Laravel (documentation officielle, tutoriels, forums)
- Gestion des droits et de la sécurité (middlewares, tests, validation)
- Déploiement et configuration serveur (documentation OVH, Let's Encrypt, adaptation .env)
- Pas de maquette initiale, adaptation de templates Bootstrap (recherche d'exemples, tests responsive)

Compétences couvertes :

- Développement de composants d'interface
- Création d'interfaces utilisateurs/administrateur
- Optimisation de l'expérience utilisateur
- Intégration de services externes
- Architecture de base de données relationnelle
- Optimisation des requêtes
- Administration du patrimoine informatique
- Protection des informations sensibles
- Gestion des autorisations
- Documentation des processus (UML, MERISE)

### Compétences développées :

- Maîtrise du framework Laravel
- Gestion autonome d'un projet complexe
- Mise en place d'interfaces par rôle
- Sécurisation des données utilisateurs
- Capacité de mise en ligne et de sécurisation une fois le projet en production

### Bilan personnel/Perspective d'amélioration :

- Fier d'avoir mené ce projet seul, en développant des fonctionnalités avancées et en progressant énormément sur Laravel et l'écosystème PHP.
- Gain d'autonomie, de rigueur, et de capacité à résoudre des problèmes complexes.
- Pour la suite : intégrer un paiement en ligne, notifications avancées, API mobile, personnalisation des pages par restaurant.

## Projet 2 : Portfolio Web

Contexte : Dans le cadre de mon BTS SIO option SLAM, j'ai réalisé un site portfolio afin de valoriser mes compétences, mes projets et ma démarche de veille technologique. Ce projet est une initiative personnelle, sans partenaire, qui répond à la fois à un besoin professionnel (présentation lors de recherches de stage ou d'emploi) et pédagogique (synthèse de mes acquis techniques et méthodologiques).

Environnement technique :

- Langages : PHP, HTML5, CSS3
- Frameworks & outils : Bootstrap 5 (pour le responsive et les composants UI), XAMPP (serveur local), JavaScript (pour les interactions dynamiques)
- Gestion de versions : Github, Gitdesktop
- Organisation : Arborescence MVC simplifiée, séparation du code, utilisation de includes pour le header/footer

Objectifs et fonctionnalités principales :

- Présenter de façon claire et attractive mes projets, compétences et expériences
- Mettre en avant la veille technologique et mes outils de suivi
- Proposer une navigation fluide, moderne et responsive, compatible tous supports
- Illustrer l'évolution de mes compétences à travers des modals détaillés pour chaque projet

Démarche et tâches réalisées :

- Analyse des portfolios concurrents pour définir les attentes en matière de design et d'ergonomie
- Création de la structure HTML et mise en place du routage via PHP

- Intégration de Bootstrap pour accélérer le développement et garantir la responsivité
- Développement de modals personnalisés pour la section veille et les projets
- Mise en place d'un formulaire de contact sécurisé (validation des champs, protection contre les injections)
- Ajout d'effets visuels et d'animations CSS pour améliorer l'expérience utilisateur
- Tests sur différents navigateurs et appareils pour assurer la compatibilité
- Rédaction de la documentation utilisateur et technique

Difficultés rencontrées / solutions :

- Conflits entre Bootstrap et styles personnalisés : Résolus par l'utilisation de classes spécifiques et de la propriété

### **!important en CSS.**

- Gestion des modals et de l'accessibilité : Nécessité d'adapter le code pour garantir un affichage correct et une navigation clavier.

Bilan personnel :

- J'ai considérablement renforcé ma rigueur en développement front-end, en apprenant à structurer mon code et à anticiper les problèmes d'ergonomie.
- J'ai découvert l'importance du responsive design et de l'accessibilité, en testant mon site sur mobile et en optimisant la navigation pour tous les utilisateurs.
- Ce projet m'a permis de prendre conscience de l'importance de la veille technologique, que j'ai intégrée directement dans le site via une section dédiée et des modals interactifs.



- Enfin, j'ai appris à documenter mon travail, à présenter mes réalisations de façon professionnelle, et à utiliser mon portfolio comme outil de communication lors de mes démarches de stage ou d'emploi.

## 4. Veille Technologique

Outils de veille utilisés :

- Feedly (agrégateur de flux RSS)
- LinkedIn, Twitter (veille réseaux sociaux professionnels)
- Google Alerts
- Forums : StackOverflow, OpenClassrooms

Sujets de veille suivis :

- PHP : nouveautés (PHP 8.3), sécurité (CVE-2024-4577), bonnes pratiques, frameworks (Laravel)
  - Sécurité web : failles récentes, bonnes pratiques, outils d'audit
- JavaScript : nouveautés ES2024, pipeline operator, Temporal API, sécurité XSS
  - Sécurité web : failles récentes, bonnes pratiques, outils d'audit
  - Tendances développement web : frameworks, accessibilité, performance

## 6. Annexes

### Annexes Projets

- 1ClickAllEat
  - Cahier des charges 1ClickAllEat (p.12 à 15)
  - Diagrammes UML (p.16 à 23)
  - Extraits de code commentés (p.24 à 31)
  - Diagramme de gantt (p.32)
  - Méthode merise (p.33 à 36)
  - Documentation utilisateur (p.37 à 46)
- Ldap - Telora
- Portfolio

### Annexes Veille

- Copies d'articles lus (PDF, captures)
- Tableaux comparatifs de frameworks
- Notes personnelles de synthèse

## **Annexe 2 : 1ClickAllEat**

### **Cahier des Charges - Projet 1ClickAllEat**

#### **1. Contexte**

La startup 1ClickAllEat nous mandate pour développer une application de réservation de restaurant au préalable. Cette solution vise à simplifier et rendre plus accessibles les restaurants traditionnels qui sont concurrencés par les fast-foods.

#### **2. Objectif Principal**

Proposer une solution optimisée pour plusieurs restaurants qui proposent des menus différents. L'application permettra d'optimiser les échanges entre les différents acteurs du restaurant (administration, service, cuisine, clients).

#### **3. Public Cible**

Restaurateurs et leur personnel

#### **4. Expression des Besoins**

L'application 1ClickAllEat doit offrir une gestion complète et optimisée des restaurants, permettant de suivre en temps réel les menus, les commandes et les réservations. L'ensemble des opérations sera parfaitement synchronisé avec les horaires d'ouverture, assurant une expérience fluide et sur-mesure pour chaque client.

La plateforme fournira aux restaurateurs les outils nécessaires à l'ajustement des offres, à la gestion des réservations et au suivi des commandes, accessibles en ligne pour un pilotage efficace des établissements à tout moment.

#### **5. Fonctionnalités**

## 5.1. Fonctionnalités MVP (Minimum Viable Product)

- **Gestion des restaurants**
  - Création et paramétrage des profils de restaurants
  - Configuration des horaires d'ouverture
  - Gestion des tables et capacités
- **Authentification des utilisateurs**
  - Système d'inscription/connexion sécurisé
  - Récupération de mot de passe
  - Différents niveaux d'accès selon les rôles
- **Accès spécifique restaurant**
  - Interface dédiée aux restaurateurs
  - **Tableau de bord de gestion**
  - Statistiques de base sur l'activité
- **Accès spécifique client**
  - Interface utilisateur intuitive
  - Recherche et filtrage des restaurants
  - Historique des commandes/réservations
- **Gestion de la carte/des menus**
  - Classification des articles au menu par catégorie
  - Ajout/modification/suppression d'articles
  - Gestion des prix et disponibilités
- **Gestion des commandes/réservations**

- Création et suivi des réservations
- Gestion des statuts des commandes
- Notifications aux clients et restaurateurs

## **5.2. Fonctionnalités MLP (Minimum Lovable Product)**

- **Génération de QR code**
  - Lien d'accès à la commande pour chaque carte de restaurant
  - QR code dynamique et personnalisable
- **Responsive**
  - Adaptation à tous les formats d'écran
  - Expérience utilisateur optimisée sur mobile, tablette et desktop
- **Paramétrage de la charte graphique**
  - Personnalisation des couleurs et logo du restaurant
  - Adaptation visuelle de la carte selon l'identité du restaurant
- **Paiement en ligne via API Stripe**
  - Intégration complète avec le package Laravel Cashier
  - Gestion sécurisée des transactions
  - Suivi des paiements et remboursements

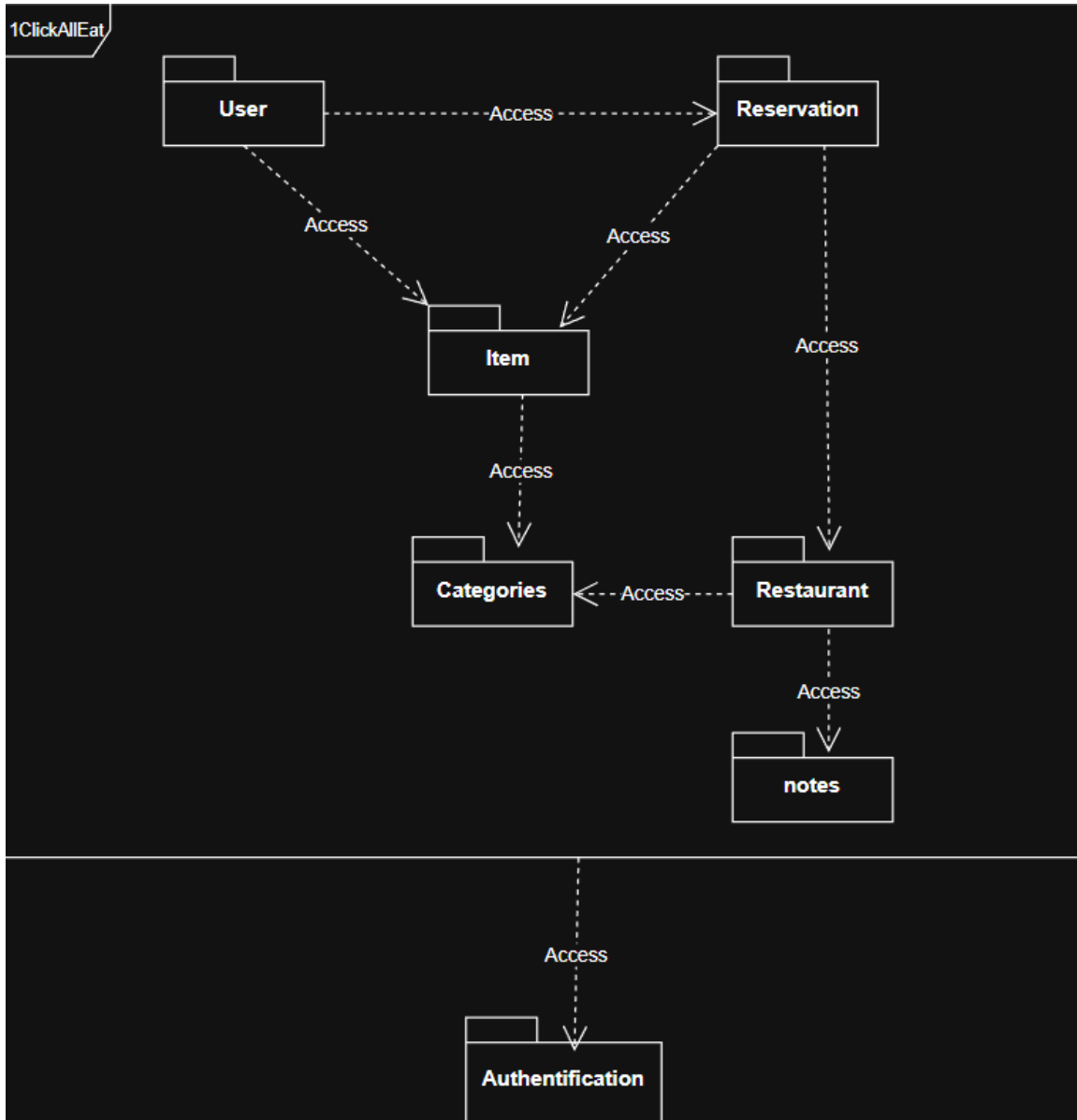
## **6. Contraintes Techniques**

- **Framework de développement**
  - Laravel (Back-end)
  - Interface utilisateur moderne et intuitive

- **Déploiement**
  - Sur un nom de domaine dédié
  - Hébergement sécurisé et performant
- **Intégration continue**
  - Automatisation des migrations de base de données
  - Gestion du vidage des caches de Laravel
  - Optimisation des fichiers pour l'environnement de production
- **Template**
  - Design responsive et moderne
  - Adaptable à l'identité visuelle des différents restaurants
- **Sécurité**
  - Certificat SSL (HTTPS) pour sécuriser les communications
  - Protection des données personnelles
- **Tests unitaires**
  - Mise en place de tests automatisés
  - Code qui teste le code

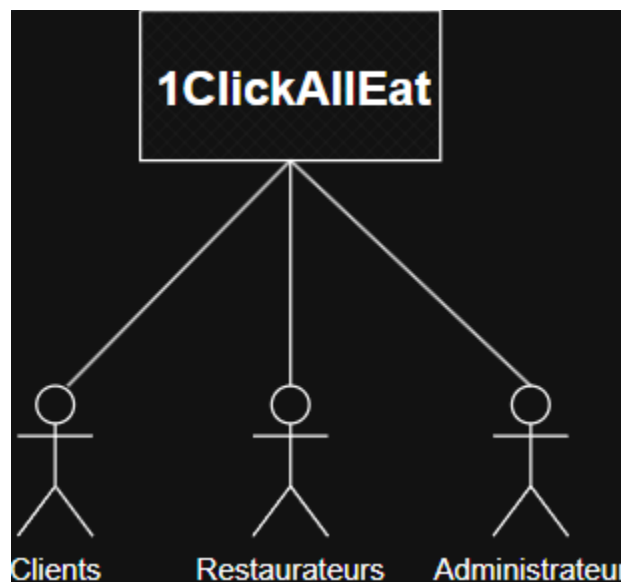
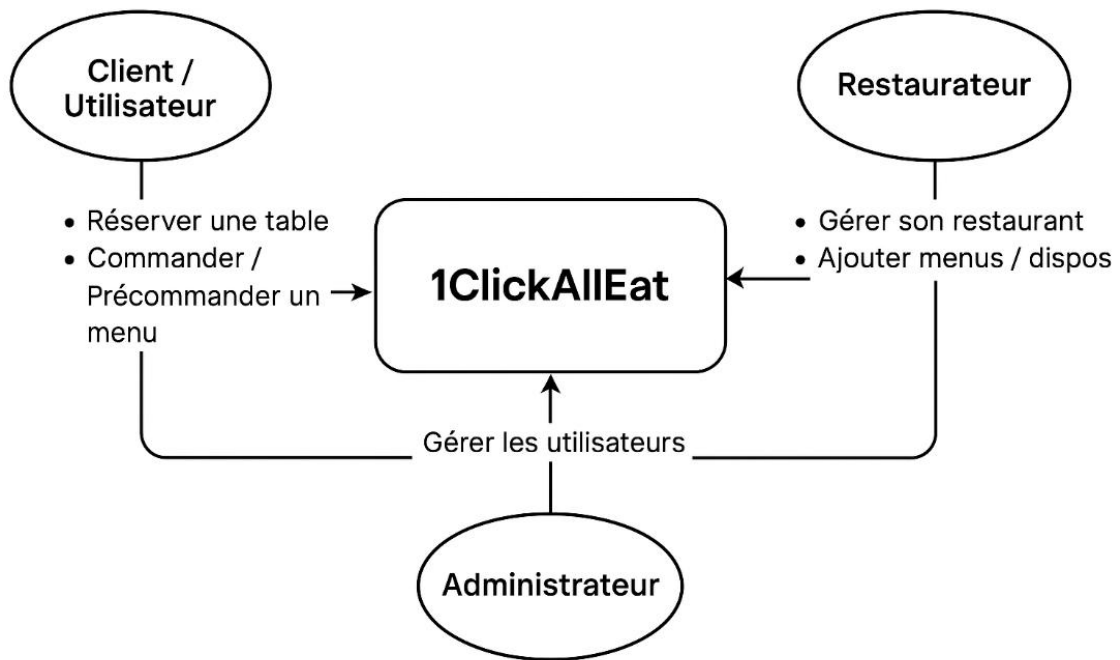
## Diagramme UML :

- Diagramme de package

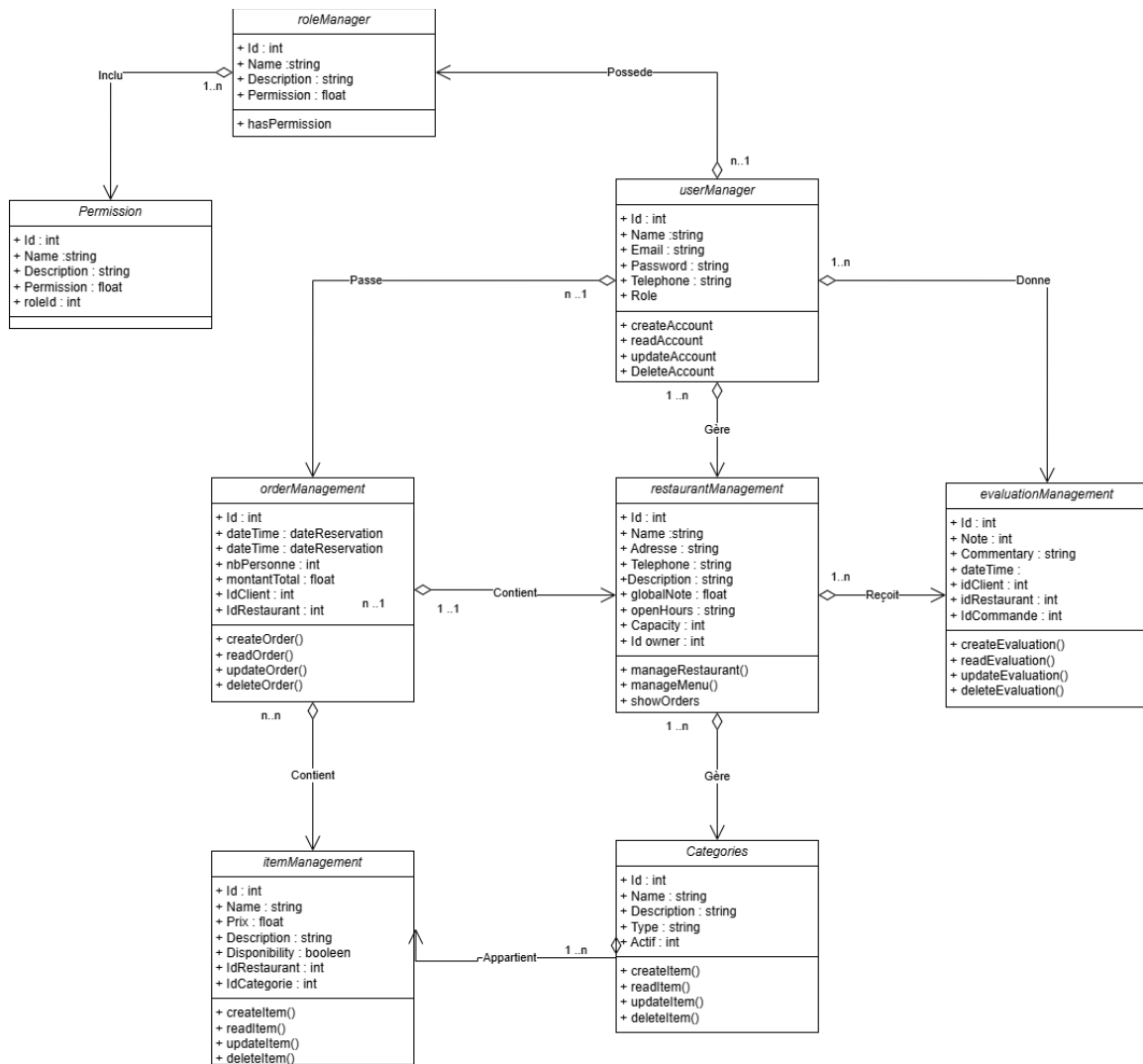




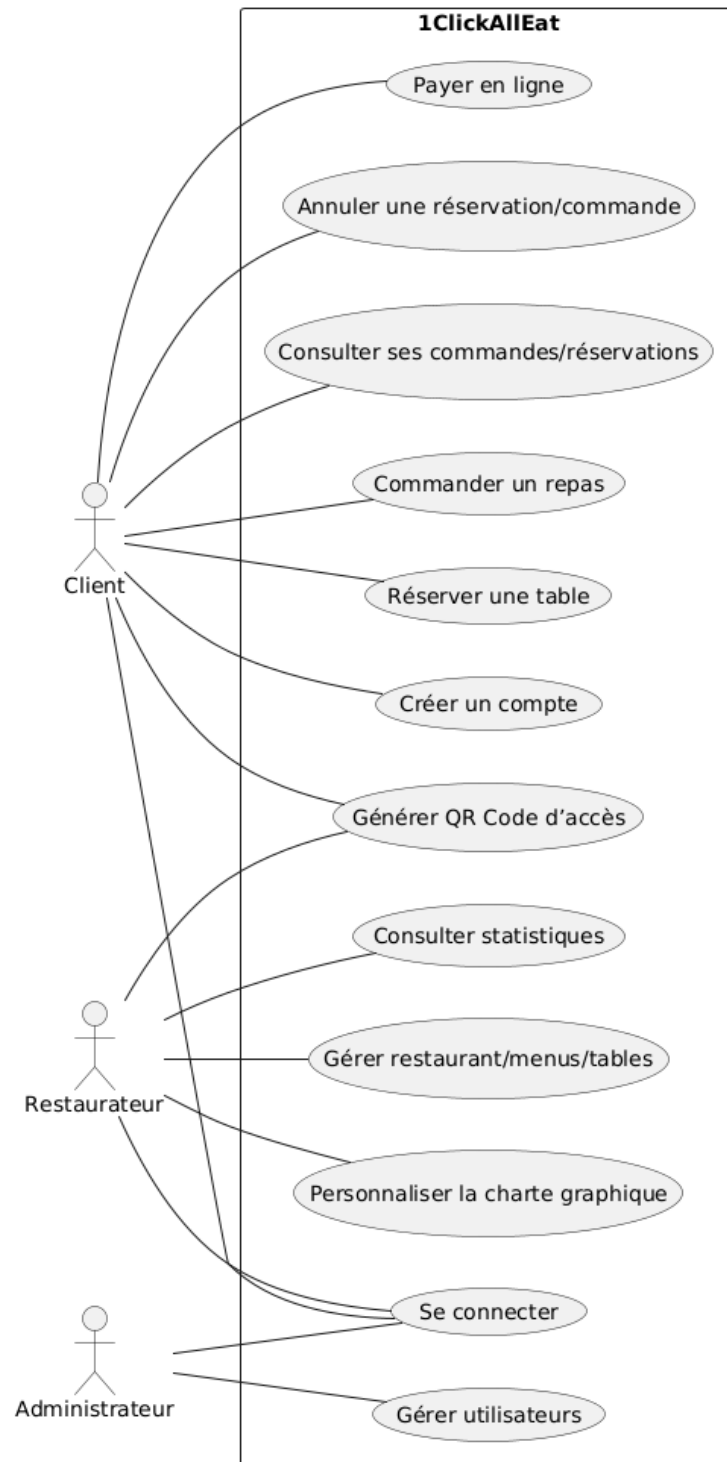
- Diagramme de contexte



○ Diagramme de classe



- Use\_case



## Plan de Test Qualité – 1ClickAllEat

### Objectif

Fournir un cadre structuré pour valider que l'application 1ClickAllEat répond à toutes les exigences fonctionnelles (MVP, MLP), techniques et de qualité : gestion des restaurants, authentification, accès différencié, gestion des menus/commandes, responsive, personnalisation graphique, sécurité, CI/CD, etc.

### Périmètre du test

#### Fonctionnalités MVP

- Gestion des restaurants (création, modification, suppression)
- Authentification et gestion des utilisateurs (clients, restaurateurs)
- Accès différencié (dashboard restaurateur, espace client)
- Gestion des menus/cartes (catégorisation, affichage)
- Gestion des commandes (création, suivi, annulation)

#### Fonctionnalités MLP

- Génération et affichage du QR Code pour accès rapide
- Design responsive (mobile, tablette, desktop)
- Personnalisation graphique (charte, couleurs, logos)
- Paiement en ligne (API Stripe)

#### Contraintes techniques

- Laravel (PHP 8+)
- Déploiement HTTPS (certificat SSL)
- Intégration continue (migrations, cache, optimisation)

- Tests unitaires/fonctionnels automatisés

## Méthodologie

- Tests manuels et automatisés : PHPUnit,
- Tests fonctionnels : Vérification de chaque exigence du cahier des charges
- Tests d'intégration : Vérification de la communication entre modules
- Tests de performance : Simulation de charge, temps de réponse
- Tests de sécurité : SSL, authentification, gestion des rôles
- Tests de compatibilité/responsive : Desktop, tablette, mobile, principaux navigateurs
- Environnement de préproduction : Identique à la production

## Tableau du plan de test

Périmètre testé	Date	Description du test	Résultat attendu	Résultat obtenu	Réussite/Échec	Testé par
Authentification utilisateur	2025-04-10	Connexion/déconnexion (client, restaurateur, admin)	Accès sécurisé, redirection correcte, session isolée	À renseigner	À renseigner	QA
Gestion des restaurants	2025-04-11	CRUD restaurant via dashboard restaurateur	Opérations sans erreur, affichage immédiat des changements	À renseigner	À renseigner	QA
Accès restaurateur	2025-04-12	Dashboard, gestion menus, commandes, réservations	Accès réservé, outils visibles uniquement pour restaurateur	À renseigner	À renseigner	QA
Accès client	2025-04-12	Consultation menus, réservation, commande	Accès restreint, affichage personnalisé selon le restaurant	À renseigner	À renseigner	QA

Périmètre testé	Date	Description du test	Résultat attendu	Résultat obtenu	Réussite/Échec	Testé par
Gestion menus/cartes	2025-04-13	CRUD catégories/articles, affichage dynamique	Modifications visibles, catégorisation correcte	À renseigner	À renseigner	QA
Gestion des commandes	2025-04-14	Passer/annuler commande, suivi en temps réel	Commande enregistrée, notification cuisine, historique mis à jour	À renseigner	À renseigner	QA
Génération QR Code	2025-04-15	Génération, scan, redirection vers menu/commande	QR code lisible, redirection correcte	À renseigner	À renseigner	QA
Responsive design	2025-04-16	Tests sur desktop, tablette, mobile, Chrome, Firefox, Safari	Interface cohérente, aucune perte de fonctionnalité	À renseigner	À renseigner	QA
Personnalisation graphique	2025-04-17	Modification couleurs, logos, polices via interface	Rendu conforme à la charte définie	À renseigner	À renseigner	QA
Paiement en ligne (Stripe)	2025-04-18	Transaction test, validation, génération reçu	Paiement validé, reçu généré, sécurité respectée	À renseigner	À renseigner	QA
CI/CD et déploiement	2025-04-19	Migrations, cache, optimisation lors d'un déploiement	Scripts CI déclenchés, déploiement sans interruption, rollback possible	À renseigner	À renseigner	DevOps
Sécurité SSL/API	2025-04-20	Vérification HTTPS, sécurité Stripe, rôles	Aucune faille détectée, accès sécurisé, logs d'accès complets	À renseigner	À renseigner	Sécurité

### Exécution, reporting et validation

- Les anomalies sont tracées dans l'outil de suivi (GitHub Issues...).
- Un rapport de test est généré à chaque fin de cycle.
- La mise en production n'est validée que si tous les tests critiques sont passés et les anomalies bloquantes corrigées.

## Conclusion

Ce plan de test garantit la conformité de l'application 1ClickAllEat avec les attentes fonctionnelles et techniques du projet. Il sera mis à jour à chaque évolution majeure du produit ou retour utilisateur.

- Extrait de code

## 1. Réservation d'une table

php

CopyInsert

// Extrait de ReservationController.php

```
public function store(Request $request, Restaurant $restaurant)
{
    $validated = $request->validate([
        'table_id' => ['required', 'exists:restaurant_tables,id'],
        'date_reservation' => ['required', 'date', 'after:now'],
        'heure_reservation' => ['required', 'date_format:H:i'],
    ]);

    $table = RestaurantTable::findOrFail($validated['table_id']);

    if (!Auth::check()) {
        return redirect()->back()->with('error', 'Vous devez être connecté pour réserver une table.');
```

```
    }

    $dateTime = $validated['date_reservation'] . ' ' . $validated['heure_reservation'];
```

```
    $exists = $table->reservations()
```

```
        ->where('date_reservation', $dateTime)
```

```
        ->whereIn('status', ['pending', 'confirmed'])
```

```
        ->exists();
```



```

if ($exists) {

    return redirect()->back()->with('error', 'Cette table est déjà réservée pour ce créneau.');
```

```

}
```

```

$reservation = Reservation::create([

    'user_id' => Auth::id(),

    'restaurant_id' => $restaurant->id,

    'table_id' => $table->id,

    'date_reservation' => $dateTime,

    'status' => 'pending',

]);
```

```

// Création automatique d'une commande liée à la réservation

$order = new \App\Models\FoodOrder([

    'client_id' => Auth::id(),

    'restaurant_id' => $restaurant->id,

    'status' => 'pending',

    'total_price' => 0

]);

$order->save();
}
```

### Explication détaillée

Cette méthode permet à un utilisateur connecté de réserver une table :

- Elle valide que la table existe et que la date/heure sont correctes.
- Elle vérifie que l'utilisateur est bien connecté.

- Elle contrôle que la table n'est pas déjà réservée à ce créneau.
- Si tout est OK, elle crée la réservation et génère automatiquement une commande vide associée, facilitant le workflow côté client et restaurateur.

## 2. Affichage différencié des commandes et réservations (client vs restaurateur)

php

CopyInsert

// Extrait de FoodOrderController.php

```
public function index()
{
    if (Auth::user()->role->name === 'client') {
        $orders = Auth::user()->clientFoodOrders()->with(['restaurant', 'items'])->latest()->get();
        $reservations = \App\Models\Reservation::with(['restaurant', 'table'])
            ->where('user_id', Auth::id())
            ->orderBy('date_reservation', 'asc')
            ->get();
        return view('orders.index', compact('orders', 'reservations'));
    } else {
        $orders = Auth::user()->restaurantFoodOrders()->with(['client', 'items'])->latest()->get();
        return view('orders.index', compact('orders'));
    }
}
```

## Explication détaillée

Cette méthode affiche les commandes et réservations dans l'espace utilisateur :

- Si l'utilisateur est un **client** : il voit ses propres commandes et réservations.
- Si c'est un **restaurateur** : il voit toutes les commandes passées dans son restaurant.
- Cela permet une expérience personnalisée et sécurisée selon le rôle, en limitant l'accès aux bonnes informations.

### 3. Passage d'une commande (création d'une commande à partir du panier)

php

CopyInsert

// Extrait de FoodOrderController.php

```
public function store(Request $request)
{
    $validated = $request->validate([
        'restaurant_id' => ['required', 'exists:restaurants,id'],
        'items' => ['required', 'array'],
    ]);

    $itemIds = array_keys($validated['items']);
    $items = Item::findMany($itemIds);
    $selectedItems = [];
    foreach ($items as $item) {
        $qty = intval($validated['items'][$item->id]['quantity'] ?? 0);
```

```

if ($qty > 0) {
    $selectedItems[] = [
        'id' => $item->id,
        'quantity' => $qty,
        'price' => $item->effective_price,
    ];
}
}

if (count($selectedItems) === 0) {
    return back()->with('error', 'Veuillez sélectionner au moins un plat.');
```

}

```

$restaurant = Restaurant::findOrFail($validated['restaurant_id']);

// Création de la commande

$order = FoodOrder::create([
    'client_id' => Auth::id(),
    'restaurant_id' => $restaurant->id,
    'status' => 'pending',
    'total_price' => array_sum(array_map(fn($i) => $i['quantity'] * $i['price'], $selectedItems)),
]);

// Ajout des plats à la commande (OrderItem)

foreach ($selectedItems as $item) {
    $order->items()->create([
        'item_id' => $item['id'],
```

```

        'quantity' => $item['quantity'],

        'price' => $item['price'],

    ));

}

return redirect()->route('orders.index')->with('success', 'Commande enregistrée !');
}

```

### Explication détaillée

- Cette méthode gère la validation du panier et la création d'une commande.
- Elle vérifie que le panier contient bien des plats, puis crée la commande avec le total calculé.
- Chaque plat sélectionné est ajouté à la commande via la relation `items()`.
- Cela garantit la cohérence des données et permet un suivi précis des commandes côté cuisine.

## 4. Annulation d'une réservation

php

CopyInsert

```

// Extrait de ReservationController.php

public function cancel(Reservation $reservation)
{
    // Vérifie que seul le propriétaire de la réservation peut l'annuler
    if ($reservation->user_id !== Auth::id()) {

```

```

        abort(403);
    }

    $reservation->status = 'cancelled';

    $reservation->save();

    return redirect()->back()->with('success', 'Réservation annulée.');
```

### Explication détaillée

- Cette méthode permet à un utilisateur d’annuler sa propre réservation.
- Elle vérifie l’autorisation (propriétaire uniquement).
- Elle change le statut en “cancelled” et sauvegarde la modification.
- Cela protège la logique métier et évite les suppressions accidentelles ou malveillantes.

## 5. Ajout d’un plat au panier (extrait simplifié)

php

CopyInsert

// Extrait de CartController.php

```

public function add(Request $request, $itemId)
{
    $item = Item::findOrFail($itemId);

    $cart = session()->get('cart', []);

    $cart[$itemId] = [
        'name' => $item->name,
```

```
'quantity' => ($cart[$itemId]['quantity'] ?? 0) + 1,  
'price' => $item->effective_price,  
];  
session(['cart' => $cart]);  
return back()->with('success', 'Plat ajouté au panier.');
```

### Explication détaillée

- Cette méthode ajoute un plat au panier stocké en session.
- Si le plat existe déjà, on incrémente la quantité.
- Le panier est persistant tant que la session de l'utilisateur reste active.
- Cela permet une expérience utilisateur fluide et rapide, sans base de données pour le panier temporaire.

○ Diagramme de gantt

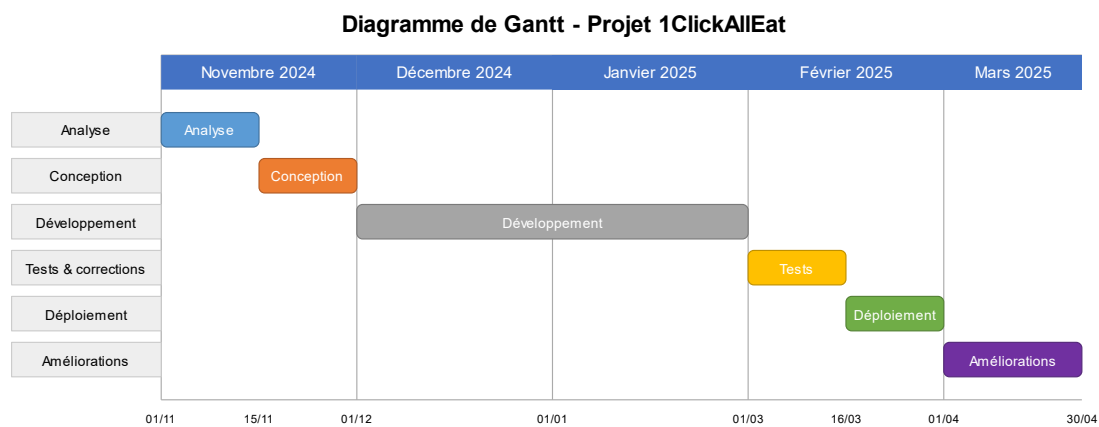
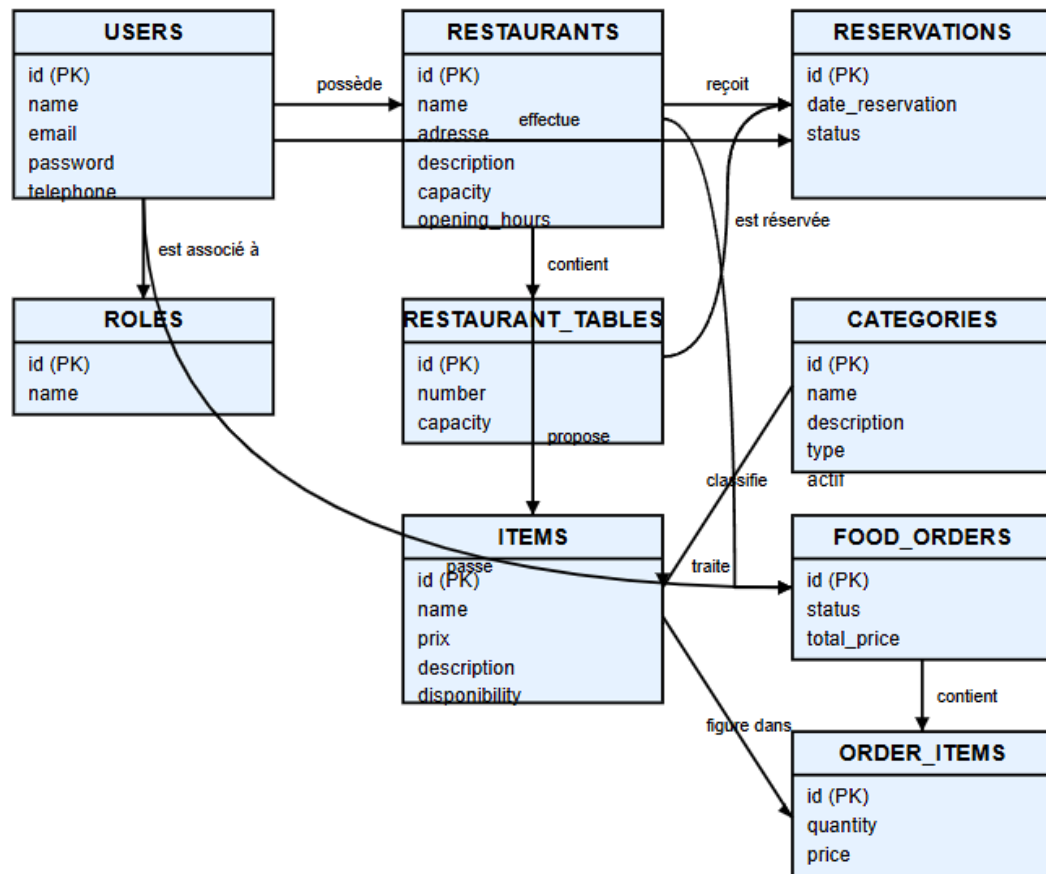




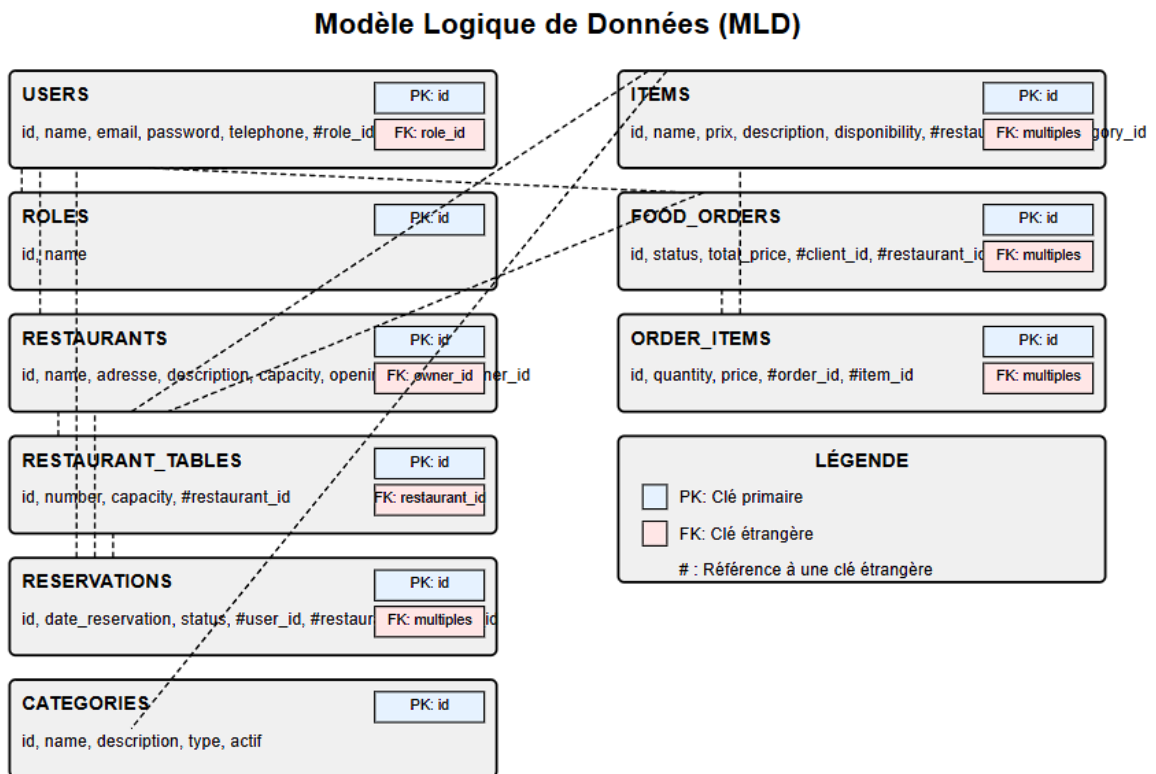
Diagramme Merise :

Dictionnaire de données : (a imprimer sur 3 page)

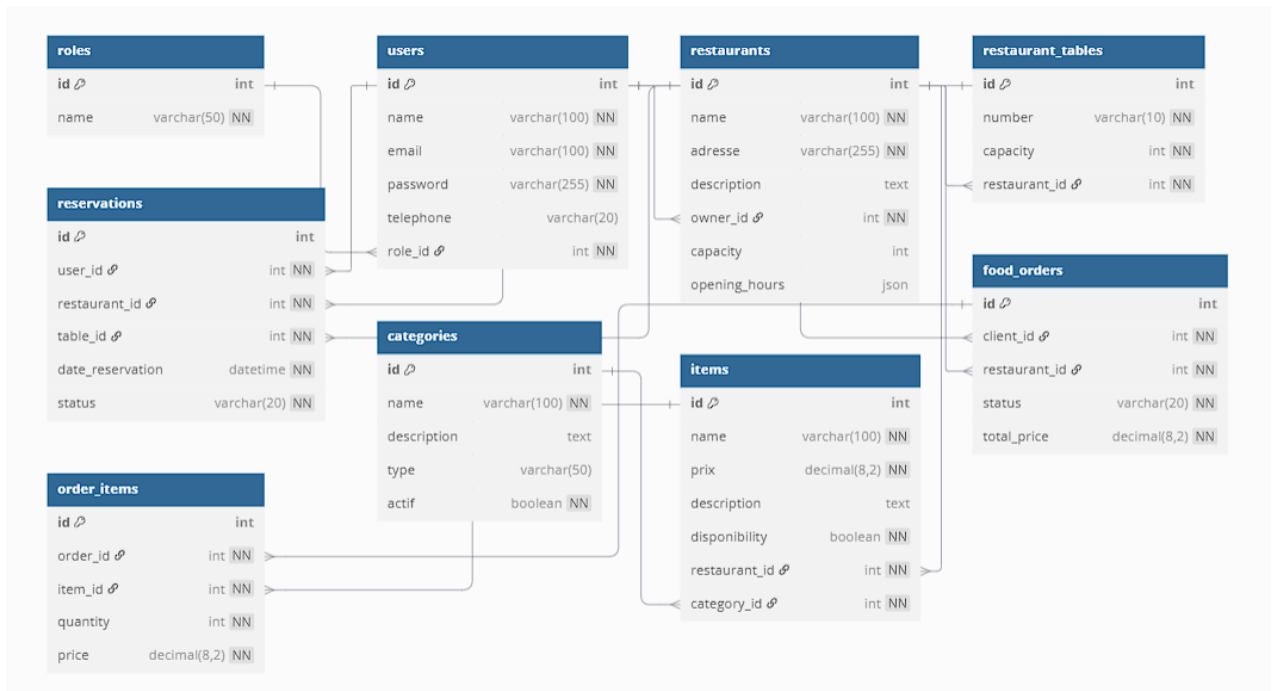
- MCD (Modèle conceptuel de données) :



- MLD (Modèle logique de données) :



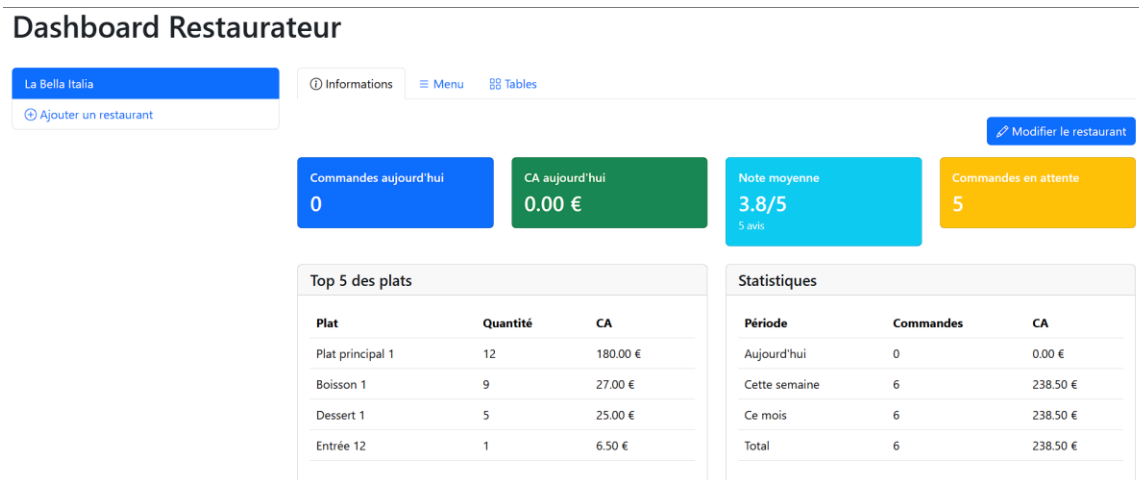
- MPD (Modele physique de données) :



# Documentation utilisateur complète de 1ClickAllEat

## Section 1: Interface Restaurateur

### 1. Dashboard principal



### 2. Gestion des tables

**Dashboard Restaurateur**

La Bella Italia

Ajouter un restaurant

Informations Menu Tables

Ajouter une table

Tables du restaurant				
Numéro	Capacité	Statut	Réservation en cours	Actions
10	5 personnes	Occupée	27/04/2025 14:25 (? pers.)	<a href="#">✎</a> <a href="#">✖</a>
12	12 personnes	Disponible	-	<a href="#">✎</a> <a href="#">✖</a>
6	10 personnes	Occupée	26/04/2025 12:33 (? pers.)	<a href="#">✎</a> <a href="#">✖</a>

### 3. Gestion du menu

**Dashboard Restaurateur**

La Bella Italia

Ajouter un restaurant

Informations Menu Tables

Ajouter un plat

Entrées			
Nom	Description	Prix	Actions
Entrée 12	Délicieuse entrée du restaurant La Bella Italia	6.50 €	<a href="#">✎</a> <a href="#">✖</a>

Plats principaux			
Nom	Description	Prix	Actions
Plat principal 1	Spécialité du chef du restaurant La Bella Italia	15.00 €	<a href="#">✎</a> <a href="#">✖</a>

Desserts			
----------	--	--	--

## 4. Gestion des commandes

Commandes en cours				
N°	Client	Total	Statut	Actions
#1	Jean Dupont	21.50 €	En préparation	<a href="#">Détails</a>
#2	Thomas Bernard	74.00 €	En attente	<a href="#">Détails</a>
#3	Thomas Bernard	74.00 €	En attente	<a href="#">Détails</a>
#4	Thomas Bernard	28.00 €	En attente	<a href="#">Détails</a>
#5	Thomas Bernard	41.00 €	En attente	<a href="#">Détails</a>
#7	Thomas Bernard	0.00 €	En attente	<a href="#">Détails</a>

## 5. Paramètres du restaurant

Modifier le restaurant

Nom du restaurant

La Bella Italia

Adresse

23 rue du test

Description

Authentique cuisine italienne avec des pâtes fraîches et des pizzas cuites au feu de bois.

Horaires d'ouverture

Lundi-Vendredi 9h-20h

Exemple : Lun-Ven 9h-22h, Sam-Dim 10h-23h

Capacité (nombre de places)

80

Mettre à jour le restaurant

Annuler

## Section 2: Interface Client

### 1. Parcourir les restaurants

1ClickAllEat Restaurants

Thomas Bernard

Commandez en ligne facilement


Découvrez les meilleurs restaurants près de chez vous et commandez en quelques clics.

Voir tous les restaurants

Restaurants populaires

## 2. Commander en ligne

### Restaurants populaires




**Sushi Master** ★ 4.2 (5)  
📍 8 Avenue de Tokyo, 75008 Paris

★★★★☆ par Client Test le 20/02/2025  
Service un peu lent mais la nourriture est correcte.

★★★★★ par Thomas Bernard le 28/03/2025  
Excellent service et nourriture délicieuse ! Je recommande vivement.

Voir le menu




**Le Bistrot Français** ★ 4.2 (5)  
📍 22 Rue Saint-Michel, 75005 Paris

★★★★★ par Client Test le 16/04/2025  
Ambiance chaleureuse et cuisine de qualité. Parfait pour un dîner en famille ou entre amis.

★★★★☆ par Thomas Bernard le 14/04/2025  
Restaurant correct mais sans plus. Les plats manquent un peu de saveur.

Voir le menu



**Spice of India** ★ 4.2 (5)  
📍 5 Rue des Épices, 75010 Paris

★★★★☆ par Client Test le 05/03/2025  
Service efficace et nourriture de qualité. L'ambiance pourrait être améliorée.

★★★★☆ par Thomas Bernard le 05/03/2025  
Qualité moyenne. Certains plats étaient bons, d'autres décevants.


Voir le menu

## 3. Suivi des commandes et réservations

### Mes réservations de table

Restaurant	Table	Date	Statut	Action
La Bella Italia	6	26/04/2025 12:33	En attente	Annuler
La Bella Italia	10	27/04/2025 14:25	En attente	Annuler
La Bella Italia	12	30/04/2025 10:10	Annulée	-

### Mes commandes

 Voir les restaurants

Numéro	Restaurant	Table	Personnes	Date et heure	Statut	Total	Actions
7	La Bella Italia	Non assignée	- pers.	-	En attente	0.00 €	Détails
5	La Bella Italia	Non assignée	- pers.	-	En attente	41.00 €	Détails
4	La Bella Italia	Non assignée	- pers.	-	En attente	28.00 €	Détails
3	La Bella Italia	Non assignée	- pers.	-	En attente	74.00 €	Détails
2	La Bella Italia	Non assignée	- pers.	-	En attente	74.00 €	Détails

## 4. Profil utilisateur

### Mon Profil

Informations du compte

#### Profile Information

Update your account's profile information and email address.

Name

Email

Sauvegarder

### Modifier le mot de passe

#### Update Password

Ensure your account is using a long, random password to stay secure.

Current Password

New Password

Confirm Password

Save

### Supprimer le compte

#### Delete Account

Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain.

Delete Account



## Section 1: Interface Restaurateur

### 1. Dashboard principal

Le Dashboard Restaurateur est votre centre de contrôle principal pour gérer votre établissement sur la plateforme 1ClickAllEat. Il affiche:

- Le nom de votre restaurant (ex: La Bella Italia)
- Navigation principale: Informations, Menu, Tables
- Statistiques essentielles:
  - Commandes aujourd'hui (nombre)
  - Chiffre d'affaires (CA) du jour
  - Note moyenne et nombre d'avis (ex: 3.8/5 basé sur 5 avis)
  - Commandes en attente (nombre)
- Top 5 des plats les plus vendus avec quantités et CA générés
- Statistiques de performance par période (aujourd'hui, semaine, mois, total)

Le bouton "Modifier le restaurant" en haut à droite vous permet d'accéder rapidement aux paramètres de votre établissement.

### 2. Gestion des tables

L'onglet "Tables" vous permet de visualiser et gérer l'ensemble des tables de votre restaurant:

- Liste complète avec: Numéro, Capacité, Statut (Disponible/Occupée)
- Affichage des réservations en cours pour les tables occupées
- Pour chaque table, vous pouvez:
  - Modifier les informations (icône crayon)

- Supprimer la table (icône poubelle)
- Bouton "Ajouter une table" pour créer une nouvelle table avec:
  - Numéro de table
  - Capacité (nombre de personnes)

Cette vue vous aide à optimiser l'occupation de votre restaurant et à suivre les réservations en temps réel.

### **3. Gestion du menu**

L'onglet "Menu" vous permet de créer et gérer l'ensemble de votre carte:

- Organisation par catégories: Entrées, Plats principaux, Desserts, etc.
- Pour chaque plat, vous visualisez:
  - Nom du plat
  - Description
  - Prix
  - Actions (modifier/supprimer)
- Bouton "Ajouter un plat" pour enrichir votre menu avec:
  - Nom du plat
  - Description détaillée
  - Prix en euros
  - Catégorie (à sélectionner dans le menu déroulant)

Gardez votre menu à jour pour offrir la meilleure expérience à vos clients.

#### 4. Gestion des commandes

La section "Commandes en cours" vous permet de suivre toutes les commandes reçues:

- Numéro de commande
- Nom du client
- Montant total
- Statut (En préparation, En attente)
- Action "Détails" pour consulter le détail complet

Cette vue vous aide à traiter efficacement les commandes et à suivre leur progression.

#### 5. Statistiques et performances

Le dashboard principal affiche des indicateurs clés de performance:

- Top 5 des plats les plus populaires avec:
  - Nom du plat
  - Quantité vendue
  - Chiffre d'affaires généré
- Récapitulatif financier par période:
  - Aujourd'hui (0 commandes, 0.00 €)
  - Cette semaine (6 commandes, 238.50 €)
  - Ce mois (6 commandes, 238.50 €)
  - Total (6 commandes, 238.50 €)

Ces données vous permettent d'analyser les tendances et d'optimiser votre offre.

## 6. Paramètres du restaurant

Le formulaire "Modifier le restaurant" vous permet de mettre à jour les informations essentielles:

- Nom du restaurant
- Adresse complète
- Description détaillée
- Horaires d'ouverture (format suggéré: Lun-Ven 9h-22h, Sam-Dim 10h-23h)
- Capacité totale (nombre de places)

Gardez ces informations à jour pour que les clients puissent trouver facilement votre établissement.

## Section 2: Interface Client

### 1. Parcourir les restaurants

La page d'accueil client présente:

- Bannière principale "Commandez en ligne facilement"
- Courte description: "Découvrez les meilleurs restaurants près de chez vous et commandez en quelques clics"
- Bouton "Voir tous les restaurants" pour accéder au catalogue complet
- Section "Restaurants populaires" présentant les établissements les plus appréciés
- Filtres de recherche avec:
  - Nom du restaurant
  - Note minimale

- Adresse/Ville (Ex: Paris)
- Tri (A-Z)

Cette interface permet aux clients de découvrir rapidement des restaurants qui correspondent à leurs préférences.

## **2. Réserver une table**

La section "Mes réservations de table" permet aux clients de:

- Visualiser leurs réservations existantes avec:
  - Nom du restaurant
  - Numéro de table
  - Date et heure
  - Statut (En attente, Annulée)
  - Action possible (Annuler)
- Réserver de nouvelles tables dans les restaurants participants

Les réservations sont synchronisées avec le dashboard restaurateur, permettant une gestion efficace des places.

## **3. Commander en ligne**

Les clients peuvent passer commande directement depuis la plateforme:

- Parcourir le menu du restaurant organisé par catégories
- Ajouter des plats à leur panier
- <sup>2</sup>Définir la quantité pour chaque article
- Voir le total de leur commande en temps réel
- Valider et payer en ligne

#### **4. Suivi des commandes et réservations**

La section "Mes commandes" permet aux clients de:

- Voir l'historique complet de leurs commandes
- Consulter le statut de chaque commande (En attente, En préparation)
- Accéder aux détails complets (articles, prix, restaurant)
- Voir le montant total de chaque commande
- Les commandes sont numérotées pour faciliter le suivi

Cette transparence améliore l'expérience client et réduit les demandes d'information.

#### **5. Profil utilisateur**

Le menu utilisateur (accessible depuis l'avatar en haut à droite) permet d'accéder à:

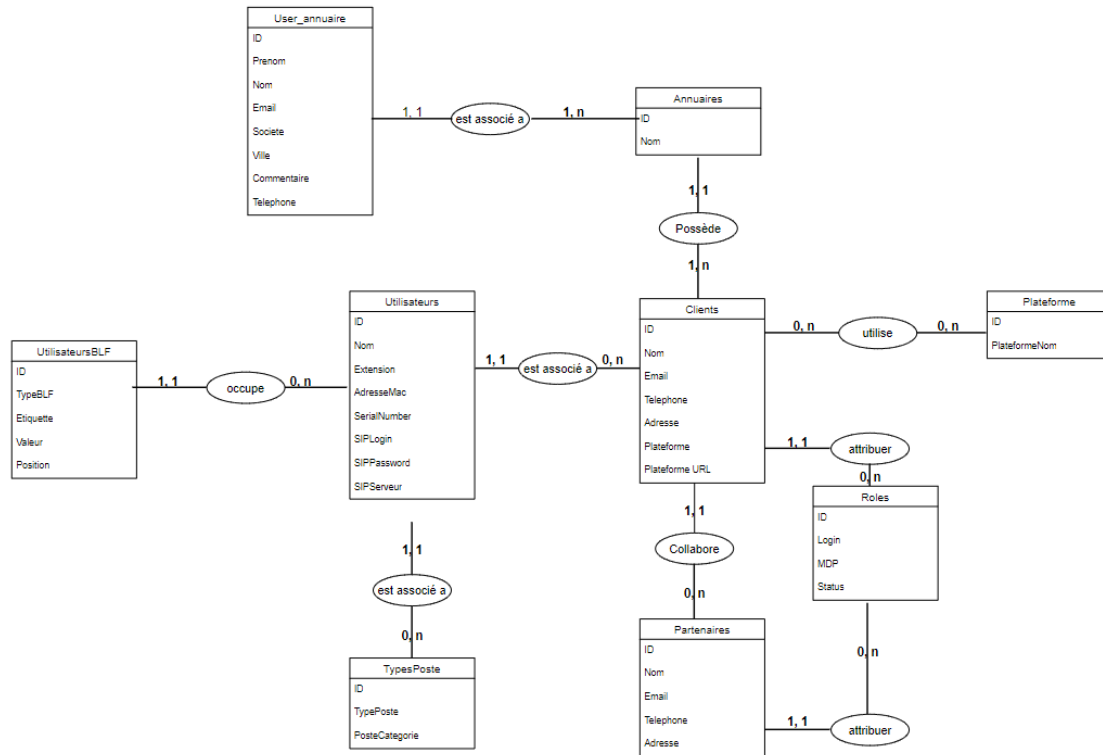
- "Mon profil": informations personnelles
- "Mes commandes": historique des commandes
- "Déconnexion": quitter la session en cours

Les clients peuvent ainsi gérer facilement leur compte et accéder à leur historique.

## **Annexe \* : Telora**

**Cahier des charges :**

## Diagramme UML :





**Merise**

## Annexe Veille :

### PHP : Nouveautés, Sécurité, Bonnes Pratiques

#### PHP 8.3 : Nouveautés clés (<https://www.php.net/releases/8.3/en.php>)

- Constantes typées : PHP 8.3 permet désormais de typer explicitement les constantes de classe, renforçant ainsi la robustesse du typage.
- Clonage profond des propriétés readonly : Amélioration de la gestion des propriétés en lecture seule, facilitant la duplication d'objets complexes.
- Fonction `json_validate()` : Nouvelle fonction pour valider la syntaxe JSON sans le décoder, optimisant ainsi les performances.
- Améliorations générales : Optimisations de performance, corrections de bugs et nettoyage du code.

### New `json_validate()` function [RFC](#) [Doc](#)

PHP < 8.3

```
function json_validate(string $string): bool {
    json_decode($string);

    return json_last_error() === JSON_ERROR_NONE;
}

var_dump(json_validate('{ "test": { "foo": "bar" } }')); // true
```

PHP 8.3

```
var_dump(json_validate('{ "test": { "foo": "bar" } }')); // true
```

`json_validate()` allows to check if a string is syntactically valid JSON, while being more efficient than `json_decode()`.

### Deep-cloning of readonly properties [RFC](#)

PHP < 8.3

```
class PHP {
    public string $version = '8.2';
}

readonly class Foo {
    public function __construct(
        public PHP $php
    ) {}

    public function __clone(): void {
        $this->php = clone $this->php;
    }
}

$instance = new Foo(new PHP());
$cloned = clone $instance;

// Fatal error: Cannot modify readonly property Foo::$php
```

PHP 8.3

```
class PHP {
    public string $version = '8.2';
}

readonly class Foo {
    public function __construct(
        public PHP $php
    ) {}

    public function __clone(): void {
        $this->php = clone $this->php;
    }
}

$instance = new Foo(new PHP());
$cloned = clone $instance;

$cloned->php->version = '8.3';
```

`readonly` properties may now be modified once within the magic `__clone` method to enable deep-cloning of readonly properties.

Vulnérabilité CVE-2024-4577 (<https://nvd.nist.gov/vuln/detail/cve-2024-4577>)

- Description : Faille critique affectant PHP-CGI sur Windows, permettant à un attaquant de passer des options malveillantes à l'exécutable PHP, pouvant conduire à l'exécution de code arbitraire.
- Versions concernées : PHP 8.1 < 8.1.29, PHP 8.2 < 8.2.20, PHP 8.3 < 8.3.8.
- Recommandation : Mettre à jour vers les versions corrigées dès que possible.

## CVE-2024-4577 Détail

### Description

Dans les versions PHP 8.1.\* avant 8.1.29, 8.2.\* avant 8.2.20 et 8.3.\* avant 8.3.8, lors de l'utilisation d'Apache et de PHP-CGI sous Windows, si le système est configuré pour utiliser certaines pages de code, Windows peut utiliser le comportement « Best-Fit » pour remplacer des caractères dans la ligne de commande fournie aux fonctions de l'API Win32. Le module PHP CGI peut interpréter ces caractères comme des options PHP, ce qui peut permettre à un utilisateur malveillant de transmettre des options au binaire PHP en cours d'exécution, et ainsi révéler le code source des scripts, exécuter du code PHP arbitraire sur le serveur, etc.

CVE-2024-52301 – Injection d'arguments via register\_argc\_argv  
(<https://news.icodia.com/actualites-icodia/faille-de-securite-sur-le-framework-laravel>)

- Date de publication : 21 novembre 2024
- Versions affectées :
  - Laravel 6 < 6.20.45
  - Laravel 7 < 7.30.7
  - Laravel 8 < 8.83.28
  - Laravel 9 < 9.52.17
  - Laravel 10 < 10.48.23
  - Laravel 11 < 11.31.0
- Description : Une vulnérabilité critique permet à un attaquant d'injecter des arguments arbitraires en exploitant la directive PHP register\_argc\_argv. Cela peut conduire à l'exécution de commandes non autorisées ou à l'accès à des informations sensibles.
- Solution : Mettre à jour Laravel vers une version corrigée. Si la mise à jour immédiate n'est pas possible, désactivez register\_argc\_argv dans la configuration PHP, en étant conscient des effets secondaires potentiels.



Le framework Laravel fait l'objet d'une faille de sécurité de criticité haute, il est important de le mettre à jour.

Laravel est un framework PHP open source, conçu pour faciliter le développement d'applications web. Il offre une structure organisée et des outils pour des tâches courantes telles que la gestion de bases de données, les routages, l'authentification, etc.

La faille CVE-2024-52301 est une vulnérabilité critique récemment identifiée dans Laravel. Elle concerne une faiblesse liée à l'injection arbitraire d'arguments dans certaines versions du framework, en exploitant des données d'entrée mal validées. Le problème réside dans l'exploitation du comportement de PHP lorsque la directive register\_argc\_argv est activée. Cela permet aux attaquants de manipuler des variables de requête pour injecter des arguments malveillants dans l'environnement de l'application. Cette faille peut permettre à un attaquant d'exécuter des commandes avec des permissions élevées, de modifier des informations sensibles, ou encore de se connecter sans autorisation.

JavaScript : ES2024, Pipeline Operator, Temporal API  
([https://www.w3schools.com/js/js\\_2024.asp](https://www.w3schools.com/js/js_2024.asp))

## ES2024 : Nouveautés majeures

- `Object.groupBy()` et `Map.groupBy()` : Nouvelles méthodes pour regrouper les éléments d'un objet ou d'une map selon une fonction de regroupement.
- Temporal API : Introduction de nouvelles classes pour une gestion précise des dates et heures, remplaçant l'objet `Date`.

### JavaScript `Object.groupBy()`

#### Example

```
// Create an Array
const fruits = [
  {name:"apples", quantity:300},
  {name:"bananas", quantity:500},
  {name:"oranges", quantity:200},
  {name:"kiwi", quantity:150}
];

// Callback function to Group Elements
function myCallback({ quantity }) {
  return quantity > 200 ? "ok" : "low";
}

// Group by Quantity
const result = Object.groupBy(fruits, myCallback);
```

Try it Yourself »

### JavaScript `Map.groupBy()`

#### Example

```
// Create an Array
const fruits = [
  {name:"apples", quantity:300},
  {name:"bananas", quantity:500},
  {name:"oranges", quantity:200},
  {name:"kiwi", quantity:150}
];

// Callback function to Group Elements
function myCallback({ quantity }) {
  return quantity > 200 ? "ok" : "low";
}

// Group by Quantity
const result = Map.groupBy(fruits, myCallback);
```

Try it Yourself »

## Pipeline Operator (|>) (<https://github.com/tc39/proposal-pipeline-operator>)

- **Fonctionnalité** : Permet de chaîner les appels de fonctions de manière plus lisible, en passant la sortie d'une fonction comme entrée de la suivante.
- **Avantage** : Améliore la clarté du code en évitant les imbrications complexes.

### Pourquoi un opérateur de canalisation

Dans l'enquête « State of JavaScript 2020 », la quatrième réponse la plus fréquente à la question « [Que manque-t-il actuellement à JavaScript ?](#) » était un opérateur de pipe. Pourquoi ?

Lorsque nous effectuons des opérations consécutives (par exemple, des appels de fonction) sur une valeur en JavaScript, il existe actuellement deux styles fondamentaux :

- passer la valeur comme argument à l'opération (imbriquer les opérations s'il y a plusieurs opérations),
- ou appeler la fonction comme une méthode sur la valeur (en enchaînant plusieurs appels de méthode s'il existe plusieurs méthodes).

Autrement dit, `three(two(one(value)))` par rapport à `value.one().two().three()`. Cependant, ces styles diffèrent considérablement en termes de lisibilité, de fluidité et d'applicabilité.

### L'imbrication profonde est difficile à lire

Le premier style, l'imbrication, est généralement applicable : il fonctionne pour n'importe quelle séquence d'opérations : appels de fonction, arithmétique, littéraux de tableau/objet, `await` etc. `yield`

Cependant, l'imbrication est difficile à lire lorsqu'elle devient profonde : le flux d'exécution se déplace de droite à gauche, contrairement à la lecture de gauche à droite d'un code normal. S'il y a plusieurs arguments à certains niveaux, la lecture oscille même entre les deux sens : notre regard doit sauter à gauche pour trouver le nom d'une fonction, puis à droite pour trouver des arguments supplémentaires. De plus, l'édition ultérieure du code peut s'avérer complexe : il faut trouver le bon emplacement pour insérer de nouveaux arguments parmi les nombreuses parenthèses imbriquées.

### Le chaînage des méthodes est limité

Le deuxième style, le chaînage de méthodes, n'est utilisable que si la valeur possède les fonctions désignées comme méthodes de sa classe. Cela limite son applicabilité. Mais lorsqu'il s'applique, grâce à sa structure postfixée, il est généralement plus utilisable et plus facile à lire et à écrire. L'exécution du code s'effectue de gauche à droite. Les expressions profondément imbriquées sont démantelées. Tous les arguments d'un appel de fonction sont regroupés avec le nom de la fonction. Modifier ultérieurement le code pour insérer ou supprimer d'autres appels de méthode est simple : il suffit de placer le curseur à un endroit précis, puis de saisir ou de supprimer une suite de caractères contiguës.

En effet, les avantages du chaînage de méthodes sont si attractifs que certaines bibliothèques populaires déforment leur structure de code spécifiquement pour permettre davantage de chaînages de méthodes. L'exemple le plus frappant est [jQuery](#), qui reste la bibliothèque JS la plus populaire au monde. Le cœur de jQuery est un unique objet superflu contenant des dizaines de méthodes, qui renvoient toutes le même type d'objet afin de permettre le chaînage continu. Il existe même un nom pour ce style de programmation : [les interfaces fluides](#).

Malheureusement, malgré toute sa fluidité, le chaînage de méthodes ne peut pas à lui seul prendre en charge les autres syntaxes de JavaScript : appels de fonctions, arithmétique, littéraux de tableaux/objets, `await` etc. `yield`. De cette façon, le chaînage de méthodes reste limité dans son applicabilité.

### Les opérateurs de canalisations combinent les deux mondes

L'opérateur de pipe tente de marier la commodité et la facilité du chaînage de méthodes avec la large applicabilité de l'imbrication d'expressions.

La structure générale de tous les opérateurs pipe est `value |> e1 |> e2 |> e3`, où `e1`, `e2` et `e3` sont des expressions qui prennent des valeurs consécutives comme paramètres. L'opérateur effectue ensuite un certain degré de magie pour « piper » `value` du côté gauche vers le côté droit.

Temporal API ([http://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects/Temporal](http://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Temporal))

- Objectif : Fournir une API moderne pour la gestion des dates et heures, avec prise en charge des fuseaux horaires et des calendriers.
- Avantages : Précision accrue, évite les bugs liés à l'objet Date, et offre plus de flexibilité pour les calculs temporels.