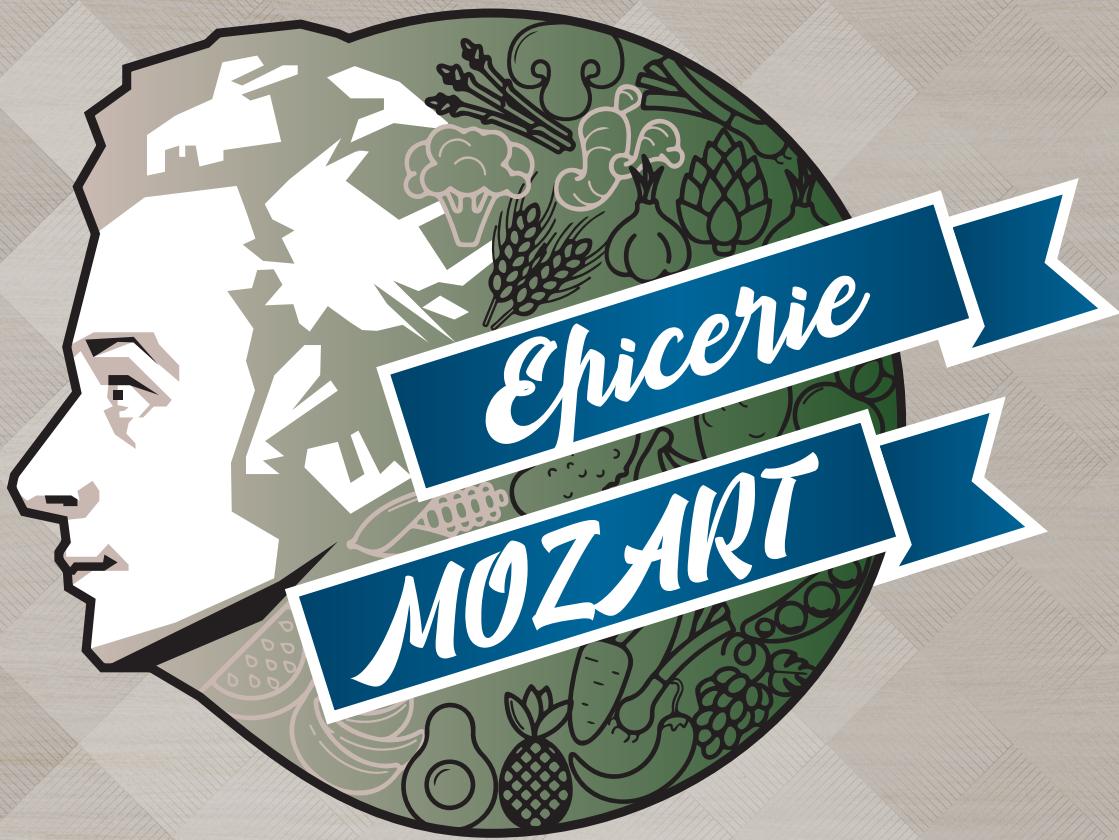


Titre professionnel Développeur Web / Web mobile



DOSSIER DE PROJET

DIDIER
RENY

EESC CCI Longwy

SOMMAIRE

COMPÉTENCES DU RÉFÉRENTIEL	4
Liste des compétences du référentiel couvertes par le projet.....	5
INTRODUCTION	6
Bienvenue à l'Épicerie Mozart.....	7
CAHIER DES CHARGES DESIGN	8
Contexte de l'entreprise et objectifs.....	9
Logiciels utilisés	10
Charte graphique	10
Arborescence du site	13
Maquettage.....	14
BASE DE DONNÉES	16
Dictionnaire des données.....	17
Le MCD (Modèle Conceptuel des Données).....	18
Le MLD (Modèle Logique des Données).....	19
Le MPD (Modèle Physique des Données).....	20
Environnement du projet	21
RÉALISATION DU PROJET	22
Le modèle MVC (Modèle-Vue-Contrôleur).....	23
Création des entités et de la base de données.....	24
Les événements côté Back-End	28
Les événements	32
Sécurité	39
Jeu d'essai - Modification d'un événement par l'administrateur.....	40
VEILLE ET RECHERCHES	42
Les outils de veille	43
Recherche sur site anglophone	44
Conclusion et Remerciements	45

COMPÉTENCES DU RÉFÉRENTIEL

Liste des compétences du référentiel couvertes par le projet

	Activités types	Compétences professionnelles												
1	Développer la partie Front-End d'une application web ou web mobile en intégrant les recommandations de sécurité	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%; text-align: center;">1</td><td>Maquetter une application</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">2</td><td>Réaliser une interface utilisateur web statique et adaptable</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">3</td><td>Développer une interface utilisateur web dynamique</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">4</td><td>Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> </table>	1	Maquetter une application	<input checked="" type="checkbox"/>	2	Réaliser une interface utilisateur web statique et adaptable	<input checked="" type="checkbox"/>	3	Développer une interface utilisateur web dynamique	<input checked="" type="checkbox"/>	4	Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce	<input checked="" type="checkbox"/>
1	Maquetter une application	<input checked="" type="checkbox"/>												
2	Réaliser une interface utilisateur web statique et adaptable	<input checked="" type="checkbox"/>												
3	Développer une interface utilisateur web dynamique	<input checked="" type="checkbox"/>												
4	Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce	<input checked="" type="checkbox"/>												
2	Développer la partie Back-End d'une application web ou web mobile en intégrant les recommandations de sécurité	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%; text-align: center;">5</td><td>Créer une base de données</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">6</td><td>Développer les composants d'accès aux données</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">7</td><td>Développer la partie Back-End d'une application web ou web mobile</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> <tr> <td style="text-align: center;">8</td><td>Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce</td><td style="background-color: #557744; color: white; text-align: center;"><input checked="" type="checkbox"/></td></tr> </table>	5	Créer une base de données	<input checked="" type="checkbox"/>	6	Développer les composants d'accès aux données	<input checked="" type="checkbox"/>	7	Développer la partie Back-End d'une application web ou web mobile	<input checked="" type="checkbox"/>	8	Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce	<input checked="" type="checkbox"/>
5	Créer une base de données	<input checked="" type="checkbox"/>												
6	Développer les composants d'accès aux données	<input checked="" type="checkbox"/>												
7	Développer la partie Back-End d'une application web ou web mobile	<input checked="" type="checkbox"/>												
8	Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce	<input checked="" type="checkbox"/>												

INTRODUCTION

Bienvenue à l'Épicerie Mozart

L'épicerie solidaire MOZART est une initiative sociale qui permet à des personnes en situation de précarité et des familles monoparentales de faire leurs courses à prix réduit dans un espace convivial. Plus qu'une simple épicerie, c'est un lieu de rencontre et d'échange pour les membres de la communauté locale. Cependant, pour toucher plus de personnes et faire connaître ses activités, l'épicerie solidaire doit être présente sur le web.



C'est pourquoi l'épicerie a décidé de mettre en place un site internet afin d'améliorer sa visibilité et de présenter ses motivations et ses valeurs au travers de leur action concrète.

Le site web sera conçu pour refléter les valeurs de convivialité, de partage et de solidarité de l'épicerie. Nous voulons créer un environnement accueillant qui invite les visiteurs à se joindre à notre communauté.

En plus de présenter l'épicerie solidaire, le site comprendra une section consacrée aux événements. Les visiteurs pourront les consulter et s'inscrire sur le site, renforçant leur sentiment d'appartenance à la communauté, puis réserver leur place en ligne. Nous espérons que cette fonctionnalité rendra plus facile la participation à nos activités et leur organisation.

Enfin, l'épicerie souhaite que la réalisation d'un site internet permette de toucher plus de personnes et d'élargir la portée de ses actions, convaincus qu'il aidera à partager son message et ses valeurs afin de récolter des dons.

CAHIER DES CHARGES

DESIGN

Contexte de l'entreprise et objectifs

L'ASSOCIATION

L'épicerie solidaire Mozart se présente comme un commerce de **proximité classique**, elle permet à des publics en difficulté économique et des familles monoparentales de réaliser leurs courses en choisissant les produits qu'ils souhaitent consommer et en les payant **10% à 30% de leur valeur marchande habituelle.**

Dans l'épicerie solidaire, des produits variés et de qualité sont en rayon :

- **Alimentaire**

- Fruits et légumes
- Produits frais
- Produits secs
- Viandes, Poissons

- **Hygiène**

- **Vêtements**

Tous les produits sont issus de la solidarité grâce à des dons de magasins, d'industriels, de grossistes, **de particuliers**, etc.
Une enveloppe financière attribuée par l'état est utilisée pour compléter les approvisionnements.

L'épicerie offre également un espace de rencontre favorisant l'insertion sociale et professionnelle ainsi que l'**organisation d'événements** tels que des ateliers cuisine, un salon de coiffure éphémère ou des soins médicaux.

L'OBJECTIF

L'objectif du projet est de permettre à l'épicerie Mozart d'accroître sa visibilité sur internet à travers un environnement en ligne qui reflète ses valeurs d'inclusion et de communauté, accueillant et chaleureux.

- Informer sur les événements et ateliers
- Permettre la réservation de place aux événements et soins médicaux
- Encourager les dons via une plateforme dédiée
- Inciter les personnes à devenir bénévoles
- Développer de nouveaux partenariats

La section consacrée aux événements sera un élément clé du site.
L'épicerie souhaite créer un espace qui permettra aux utilisateurs de découvrir les événements organisés et de réserver leur place en ligne.
Elle souhaite que cette fonctionnalité soit facile d'utilisation et accessible à tous.

Logiciels utilisés



Adobe Photoshop
Logiciel de traitement d'images



Adobe Illustrator
Logiciel de création graphique vectorielle.



Adobe XD
Outil de conception vectorielle pour les applications web et mobiles



Affinity Publisher
Logiciel de publication assistée par ordinateur

Charte graphique

LOGOTYPE

L'épicerie a souhaité actualiser son logo tout en conservant l'image du célèbre compositeur qui donne son nom à l'enseigne.

J'ai donc travaillé dans cette optique, avec également pour objectif d'y associer une vision de la terre nourricière et des éléments visuels liés à l'activité de l'entreprise. J'ai opté pour des formes et des silhouettes stylisées, représentant des aliments comme des fruits, des légumes, des épices ou d'autres produits de base. Ces éléments s'intègrent harmonieusement autour de la représentation de Mozart, créant ainsi une fusion visuelle.

La typographie s'inspire là encore du logo précédent mais avec plus d'homogénéité.

Les couleurs naturelles et harmonieuses parachèvent la cohésion de l'ensemble



La Police de caractère du Logo

Camping Regular

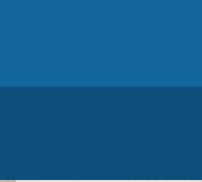
n'est utilisée que pour lui, Elle n'apparaît nulle part ailleurs sur le site



LES COULEURS

Le choix de ces couleurs a été soigneusement étudié pour créer une identité visuelle cohérente, engageante et en phase avec les valeurs de l'épicerie.

En associant ces trois couleurs, nous avons créé une harmonie visuelle qui renforce l'identité du commerce. Elles se complètent et s'équilibrivent mutuellement, créant une atmosphère chaleureuse et accueillante pour nos clients. De plus, cette combinaison de couleurs est polyvalente et s'adapte facilement à différents supports


#13679c
R 19 V 103 B 156


#0e4e7a
R 14 V 78 B 122

Le **bleu lapis-lazuli** est une teinte profonde et riche, évoquant la confiance et la stabilité. Elle représente l'engagement envers la qualité des produits et services de l'établissement. Elle évoque également la sérénité, ce qui est important dans son approche axée sur le bien-être et l'équilibre.


#7d876c
R 125 V 135 B 108


#506a45
R 80 V 106 B 69


#225529
R 34 V 85 B 41

Le **vert olive**, quant à lui, incarne la vitalité, la croissance et la durabilité. Cette couleur naturelle relie l'épicerie à son engagement envers l'environnement et une alimentation responsable. Elle rappelle la fraîcheur des produits qu'elle propose, ainsi que son attachement à la préservation des ressources précieuses. Le vert olive symbolise également la santé et le bien-être, soulignant l'implication envers une alimentation équilibrée et nourrissante.


#ccc7c2
R 204 V 199 B 194

Enfin, la couleur **bois naturel** complète la palette, apportant chaleur et authenticité à l'identité visuelle. Elle rappelle le lien profond avec la nature, l'attachement à la simplicité et à l'entraide. Cette couleur évoque également la durabilité et l'intemporalité, soulignant la volonté de promouvoir des valeurs durables et une consommation responsable.

LE GRAPHISME

Les étiquettes et les labels sont des éléments couramment utilisés pour fournir des informations, des certifications et des garanties sur les produits. Ils symbolisent la transparence, la confiance et l'attention portée aux détails. En intégrant ces éléments dans la charte graphique, l'épicerie souhaite symboliser son engagement envers ses bénéficiaires, en leur offrant une expérience de consommation claire et rassurante.

Ce choix graphique permet également de créer une esthétique vintage et rassurante. Cela reflète l'attachement à des pratiques durables, à des produits de qualité et à une relation de confiance avec les fournisseurs, les bénévoles et les bénéficiaires de l'Épicerie Mozart.



LA TYPOGRAPHIE

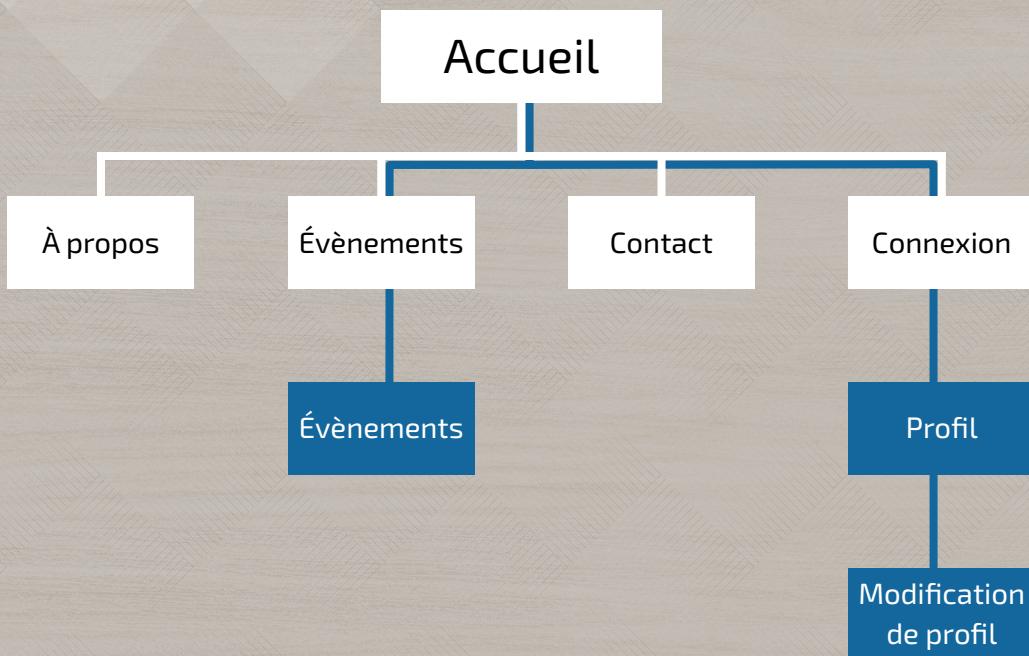
Les polices ont été choisies sur Google Fonts pour plusieurs raisons:

- un système d'intégration simple qui permet de facilement ajouter la police de caractère choisie au site web en utilisant le code fourni, sans avoir à télécharger ou à héberger les fichiers de police.
- Elles sont spécialement optimisées pour une utilisation en ligne. Elles sont légères, ce qui réduit le temps de chargement de votre site web. De plus, elles sont compatibles avec la plupart des navigateurs.

Exo 2 Regular, LIGHT, Bold, Black

Mochiy Pop One Regular

Arborescence du site



Lorsqu'un visiteur se rend sur le site il a accès à l'arborescence principale du site (en blanc) avec des pages d'informations, de contact et de connexion.

Sur la page de connexion il a la possibilité de s'inscrire puis de se connecter pour accéder à une arborescence spécifique (en bleu) comportant une page profil, modification de profil, et une page 'Évènements' qui comportera des informations supplémentaires et des possibilités de réservations.

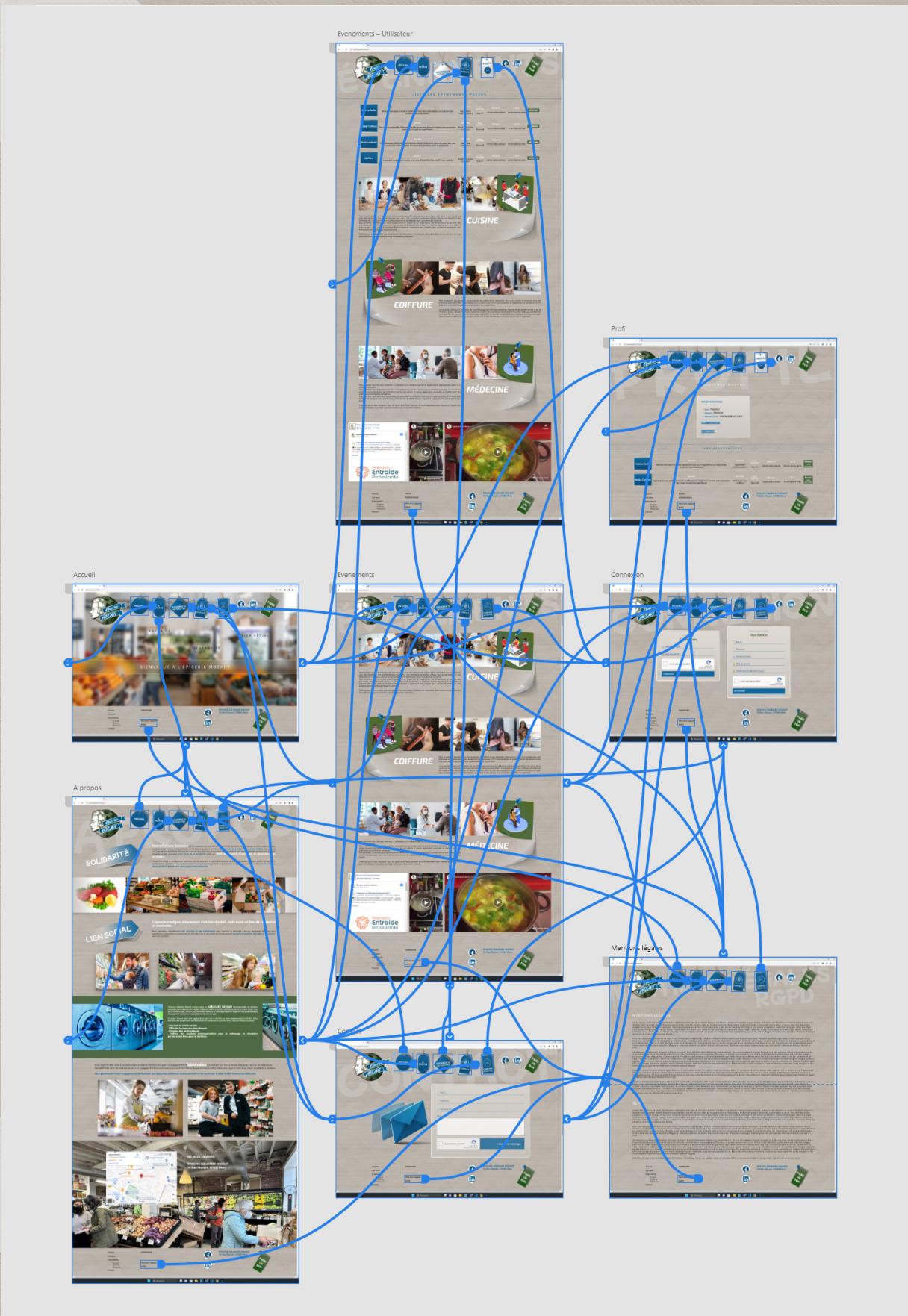
L'interface d'administration ne sera accessible que via une adresse URL particulière qui n'apparaîtra pas sur le site. En s'y connectant, l'administrateur accédera à une arborescence qui lui est propre dans sa totalité.

Maquettage

Après une étape de prototypage et la réalisation d'un wireframe, visant à définir un zoning des différents composants de l'interface du site, j'ai réalisé une maquette qui permet d'avoir une vision complète de l'interface utilisateur prévue en regroupant le wireframe et tous les éléments de la charte graphique.

Ce maquettage est réalisé de plusieurs façon afin de définir le '*Responsive Design*' qui permet d'adapter une page web aux différents supports sur lesquels elle peut être visualisée et garantir ainsi son ergonomie.





BASE DE DONNÉES

Le processus de conception de base de données commence par la création d'un **dictionnaire des données**, suivi de la création d'un **MCD (Modèle Conceptuel des Données)**, qui représente les concepts et les relations du système d'informations. Ensuite, le MCD est converti en un **MLD (Modèle Logique des Données)**, qui spécifie les détails de mise en œuvre dans un modèle relationnel. Enfin, le MLD est traduit en un **MPD (Modèle Physique des Données)**, qui représente la structure physique de la base de données et prépare son implémentation dans un **SGBD (Système de Gestion de Base de Données)**. Chaque étape ajoute des détails et des spécifications supplémentaires pour passer d'une vision conceptuelle à une mise en œuvre concrète de la base de données.

Dictionnaire des données

Un dictionnaire des données est un document qui recense et définit les termes utilisés dans un contexte spécifique, faisant écho aux besoins recueillis auprès du client. Il fournit des descriptions détaillées des différents éléments de données dont leur nom, leur signification, leur format, leur type et parfois des commentaires particuliers.

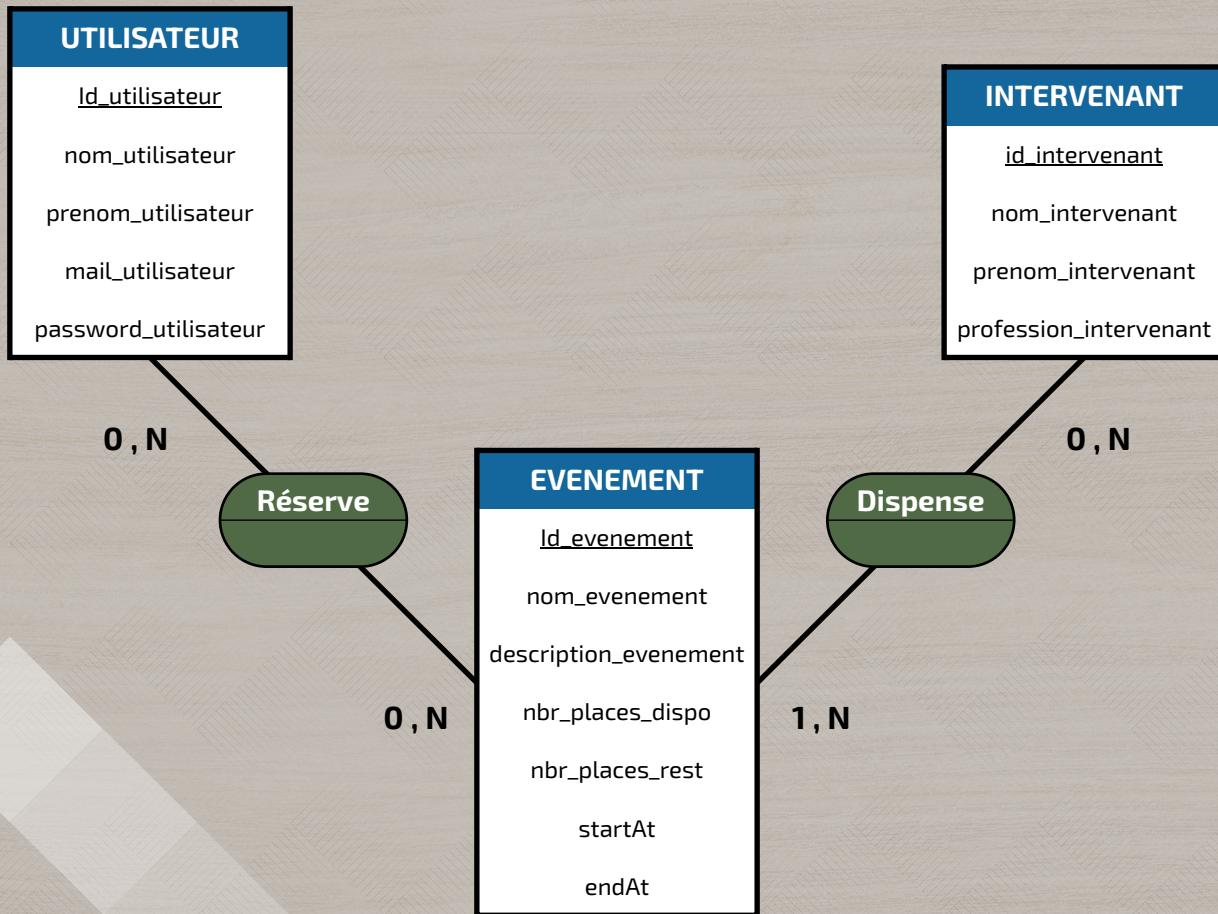
Le tableau qui suit représente un dictionnaire de données épuré, exempt de doublon ou de donnée superflue.

Nom	Désignation	Type	Taille
UTILISATEUR			
nom_utilisateur	Nom de famille de l'utilisateur	Texte	55
prenom_utilisateur	Prénom de l'utilisateur	Texte	55
mail	Adresse email de l'utilisateur	Texte	150
password	Mot de passe de l'utilisateur	Texte	255
EVENEMENT			
nom_evenement	Désignation de l'évènement	Texte	55
description	Description de l'évènement	Texte	750
nbr_places_dispo	Nombre de places disponibles	Entier	
nbr_places_rest	Nombre de place restantes	Entier	
startAt	Date et heure de début	Date et heure	
endAt	Date et heure de fin	Date et heure	
INTERVENANT			
nom_intervenant	Nom de famille de l'intervenant	Texte	55
prenom_intervenant	Prénom de l'intervenant	Texte	55
profession_intervenant	Profession de l'intervenant	Texte	70

Le MCD (Modèle Conceptuel des Données)

Le Modèle Conceptuel des Données est une représentation graphique visant à montrer comment les différents éléments sont liés entre eux... Il se concentre sur les concepts et les relations entre les différentes entités (ou objets) indépendamment de la technologie utilisée pour stocker les données. Le MCD utilise un ensemble de règles et de diagrammes codifiés pour sa représentation:

- **les entités**
- **les propriétés (ou champs)** qui sont la liste des données de l'entité dont la première est toujours l'**id** (l'identifiant)
- **les relations** qui relient les entités entre elles et spécifient la nature de ce lien (le plus souvent par un mot)
- **les cardinalités** qui indiquent combien d'occurrences d'une entité peuvent être associées à une occurrence de l'entité liée dans une relation.



Ici nous pouvons définir les cardinalités entre EVENEMENT et UTILISATEUR comme suit:

- **Un utilisateur peut réserver au minimum 0 et au maximum une infinité (N) d'évènements.**
- **Un évènement peut avoir minimum 0 et maximum une infinité (N) de réservations d'utilisateur.**

Et les cardinalités entre EVENEMENT et INTERVENANT comme suit:

- **Un intervenant peut dispenser au minimum 0 et au maximum une infinité (N) d'évènements.**
- **Un évènement peut être dispensé au minimum par 1 intervenant et au maximum par une infinité (N).**

Le MLD (Modèle Logique des Données)

Le Modèle Logique des Données est une représentation textuelle de la base de données à informatiser qui suit le travail d'analyse du MCD.

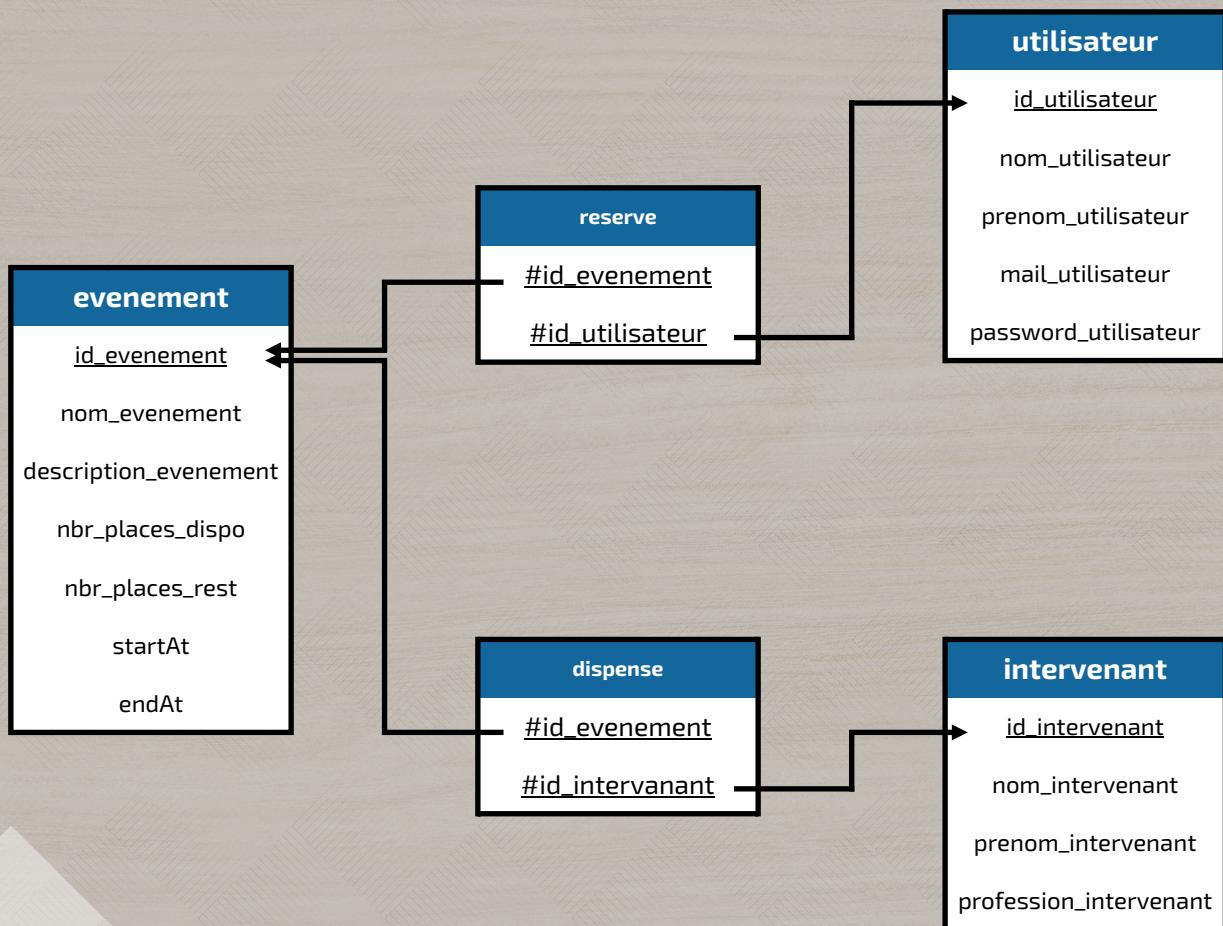
Il se présente sur plusieurs lignes où chacune représente une table et commence toujours par le nom de celle-ci. Les champs sont ensuite listés entre parenthèses et séparés par des virgules. Les clés primaires sont toujours placées en tête de liste et soulignées, tandis que les clés étrangères sont préfixées d'un dièse (#).

```
utilisateur = ( id_utilisateur, nom_utilisateur, prenom_utilisateur, mail_utilisateur,  
password_utilisateur )  
reserve = ( #id_evenement, #id_utilisateur )  
evenement = ( id_evenement, nom_evenement, description_evenement,  
nbr_place_dispo, nbr_place_rest, startAt, endAt )  
dispense = ( #id_evenement, #id_intervenant )  
intervenant = ( id_intervenant, nom_intervenant, prenom_intervenant,  
profession_intervenant )
```

Le MPD (Modèle Physique des Données)

Le Modèle Physique des Données est un modèle qui représente la structure physique des données dans un Système de Gestion de Base de Données (SGBD). Contrairement au MCD qui est indépendant de la technologie de stockage, le MPD tient compte des contraintes spécifiques du SGBD. De fait, des changements s'opèrent dans la terminologie et les entités se transforment en tables.

A l'instar du MCD, le MPD est une représentation graphique qui cette fois fait la synthèse avec le MLD.



Environnement du projet



Visual Studio Code

Éditeur de code extensible développé par Microsoft.



HTML5 (HyperText Markup Language 5)

Langage de balisage servant à composer et structurer une page web.



CSS3 (Cascading Style Sheets)

Langage servant à présenter, mettre en page, styliser ou animer une page web.



PHP (Hypertext Preprocessor)

Langage de script côté serveur utilisé pour développer des contenus web dynamiques ainsi que pour interagir avec les bases de données.



JavaScript

Langage de script principalement côté client utilisé pour ajouter de l'interactivité et de la dynamique aux pages web.



Doctrine

ORM (object-relational mapping) pour PHP qui facilite la gestion des opérations de base de données.



Twig

Moteur de templates pour PHP qui permet de séparer la logique de la présentation.



Composer

Logiciel gestionnaire de dépendances en PHP qui permet d'installer les bibliothèques dont le projet a besoin.



WAMP (Windows Apache MySQL PHP)

Ensemble de logiciels qui permet de configurer un environnement de développement web sous Windows (serveur web Apache, base de données MySQL, langage PHP).



phpMyAdmin

Interfaces pour gérer les Systèmes de Gestion de Base de Données MySQL.



git

Logiciel de gestion de versions décentralisé permettant de conserver l'historique de son projet et le travail collaboratif.

RÉALISATION DU PROJET

Le modèle MVC (Modèle-Vue-Contrôleur)

Le modèle MVC est un modèle d'architecture très largement utilisé dans le développement pour organiser et structurer ses applications. Il divise l'application en trois composants principaux : le modèle, la vue et le contrôleur.

• Modèle :

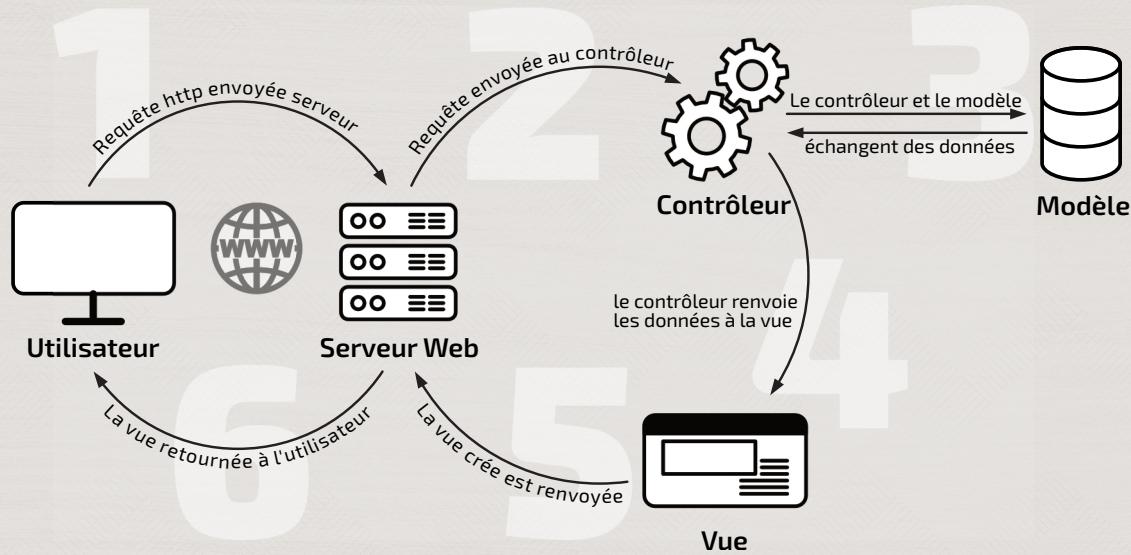
Le *modèle* représente la partie de l'application qui gère les données et la logique. Il est responsable de la manipulation, de la gestion et de la persistance des données. Cela peut inclure l'accès à la base de données, le traitement des données, les calculs et les règles. Le modèle ne doit pas dépendre de la vue ou du contrôleur, ce qui le rend indépendant de l'interface utilisateur.

• Vue :

La *vue* est responsable de l'interface utilisateur de l'application. Elle est chargée de présenter les données au format approprié pour les utilisateurs. La vue récupère les données pertinentes du modèle et les formate pour les afficher de manière compréhensible et visible. La vue ne doit pas contenir de logique complexe. Elle se contente d'afficher les données fournies par le contrôleur.

• Contrôleur :

Le *contrôleur* agit comme un intermédiaire entre le modèle et la vue. Il reçoit les entrées de l'utilisateur et déclenche les actions appropriées dans le modèle. Il récupère les données nécessaires du modèle et les transmet à la vue pour affichage. Le contrôleur gère également les interactions utilisateur, tels que les formulaires, les requêtes HTTP, etc. Il prend les décisions en fonction des actions de l'utilisateur et met à jour le modèle si besoin.



Création des entités et de la base de données

Pour construire la base de données, il est d'abord nécessaire de configurer le fichier **config.php** du dossier **config** de mon projet. Il contient les clés telles que "doctrine.user", "doctrine.dbname", "doctrine.mdp", correspondant à des configurations spécifiques pour la bibliothèque Doctrine ORM qui définissent les informations de connexion à la base de données.

```
"doctrine.user" => "root",
"doctrine.dbname" => "epicerieV2",
"doctrine.mdp" => "",
"doctrine.driver" => "pdo_mysql",
```

Il faut ensuite créer via phpMyAdmin une **Nouvelle base de données**. Dans mon cas elle se nomme **epicerieV2**.

Je peux alors créer mes différentes entités dans le dossier **model>entity** correspondant au MPD précédemment développé.

Vous pouvez voir ci-contre l'exemple de la **class Evenement**.

Ici j'écris les commentaires qui sont des annotations utilisées par Doctrine pour définir la **class Evenement** comme une entité persistante. Je spécifie le nom et les propriétés de l'entité en tant que colonnes correspondantes dans la table, avec le typage de données appropriées. Cela permet à Doctrine de créer et de gérer automatiquement la correspondance entre les objets Evenement et les enregistrements de la table "evenement" dans la base de données.

```

/**
 * @ORM\Entity
 * @ORM\Table(name="evenement")
 */
class Evenement
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="IDENTITY")
     * @ORM\Column(type="integer")
     * @var integer
     */
    private int $id;

    /**
     * @ORM\Column(type="string", length="55")
     * @var string
     */
    private string $nom;

    /**
     * @ORM\Column(type="string", length="750")
     * @var string
     */
    private string $description;

    /**
     * @ORM\Column(type="integer")
     * @var integer
     */
    private int $nbr_places_dispo;

    /**
     * @ORM\Column(type="integer")
     * @var integer
     */
    private int $nbr_places_rest;

    /**
     * @ORM\Column(type="datetime")
     * @var
     */
    private $startAt;

    /**
     * @ORM\Column(type="datetime")
     * @var
     */
    private $endAt;
}

```

La class **Evenement** se poursuit ainsi:

```
/*
 * Many to Many Evenements-Users.
 * @ORM\ManyToMany(targetEntity="User", inversedBy="evenements",
cascade={"merge"})
 * @ORM\JoinTable(name="evenements_users")
 * @var Collection<int, User>
 */
private Collection $users;

/**
 * Many to Many Evenements-Intervenants.
 * @ORM\ManyToMany(targetEntity="Intervenant", inversedBy="evenements")
 * @ORM\JoinTable(name="evenements_intervenants")
 * @var Collection<int, Intervenant>
 */
private Collection $intervenants;

public function __construct()
{
    $this->users = new ArrayCollection();
    $this->intervenants = new ArrayCollection();
}
```

Ici est définie une relation Many-to-Many entre l'entité **Evenements** et la class cible (**target entity**) **User** dans Doctrine ORM. De même qu'entre **Evenements** et **Intervenant**.

Dans les deux cas il est spécifié la table de jointure (**JoinTable(name="")**) et les propriétés nécessaires pour représenter cette relation (**private Collection**) qui représente la collection d'utilisateurs ou d'intervenants associés à un événement.

Cela permet d'associer un événement à plusieurs utilisateurs ou intervenants et vice versa, en utilisant une table de jointure pour stocker les relations entre les deux entités.

Enfin sont créées et initialisées les collections **\$users** et **\$intervenants** avec de nouvelles instances d'**ArrayCollection** de Doctrine. Cela garantit que chaque fois qu'un objet de cette class **Evenement** est créé, les collections associées sont également initialisées et prêtes à être utilisées pour stocker des objets relatifs aux utilisateurs et aux intervenants liés à un événement.

Après avoir créé toutes les entités nécessaires, je termine par 2 commandes dans le terminal:

```
php vendor/bin/doctrine orm:generate-entities ./ --regenerate-entities
```

Cela exécute une commande spécifique pour générer (ou régénérer) des entités de modèle à l'aide de Doctrine ORM. Cela permet de créer automatiquement des classes PHP qui représentent les tables de la base de données, en utilisant les annotations et les configurations définies dans le modèle de données. Doctrine va parcourir les mappings de mes entités et générer automatiquement le code des classes correspondantes, y compris les méthodes getter et setter pour les propriétés définies dans les annotations. Cela signifie qu'il est inutile d'écrire manuellement les méthodes getter et setter pour chaque propriété, Doctrine se chargeant de les générer.

La régénération des entités est également utile lorsque l'on apporte des modifications à nos mappings ou que l'on ajoute de nouvelles propriétés à nos entités. En exécutant cette commande, on s'assure que les class entités reflètent correctement les modifications apportées aux mappings, et que les méthodes getter et setter sont générées ou mises à jour en conséquence.

--regenerate-entities indique à Doctrine de régénérer les entités existantes. Si cette option est spécifiée, les entités déjà générées seront écrasées et remplacées par de nouvelles entités générées (un fichier reste toutefois disponible). Dans le cas contraire seules les entités manquantes seront générées.

```
php vendor/bin/doctrine orm:schema-tool:create
```

Cette commande spécifique de Doctrine traite le schéma et génère la sortie SQL. Elle ne sera utilisée qu'une seule fois.

```
php vendor/bin/doctrine orm:schema-tool:update --force
```

Cette commande spécifique de Doctrine permet de mettre à jour le schéma de base de données en fonction des entités définies dans le code. Cela signifie que les tables de la base de données seront créées, modifiées ou supprimées en fonction des modifications apportées aux entités.

--force indique à Doctrine d'appliquer les modifications directement sans demander de confirmation supplémentaire.

Les évènements côté Back-End

Dans un espace dédié à lui seul, l'administrateur du site à la possibilité de visualiser plusieurs informations et de créer différentes choses:

- Voir la liste des utilisateurs et les supprimer si besoin.
- Enregistrer des intervenants, les afficher, les modifier ou les supprimer.
- Créer des événements, les présenter, voir les réservations de chacun, les modifier, les supprimer.
- ...

LE CRUD

Les intervenants et les évènements obéissent à une structure communément utilisée pour gérer les instances d'entités en base de données:

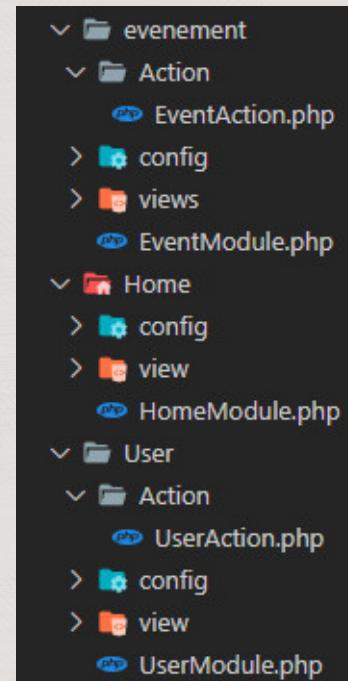
le **CRUD** pour **Create, Read, Update, Delete**.

Cela régit l'action de Crédation de nouvelles données dans le système, de Lecture, de Mise à jour et de Suppression des données existantes.

Dans mon projet, les classes qui centralisent les méthodes des contrôleurs sont situés dans des dossiers **Action**, eux-mêmes dans des dossiers répondant au nom de chaque entité.

Dans le cas des évènements, c'est dans le fichier **EventAction.php** que les méthodes sont écrites.

Les 3 pages suivantes présentent les méthodes **addEvent** (create), **EventUpdate** (update), **listEvent** (read) et **eventDelete** (delete) commentées.



```

public function addEvent(ServerRequestInterface $request) {
    //Récupère la méthode utilisée pour la requête (POST ou GET)
    $method = $request->getMethod();

    //Si le formulaire a été soumis
    if ($method === 'POST') {
        //On récupère le contenu de $_POST (les valeurs saisies dans le formulaire)
        $data = $request->getParsedBody();

        //On instancie le Validator en lui passant le tableau de données à valider
        $validator = new Validator($data);

        //On fixe les règles à respecter sur chaque input du formulaire, si il y en a, et
        //on récupère les erreurs ou null
        $errors = $validator
            ->required('nom', 'description', 'nbr_places_dispo', 'startAt', 'endAt')
            ->getErrors();
        //Si il y a des erreurs, on crée un Toast par erreur et on redirige l'utilisateur
        //afin d'afficher les messages
        if($errors) {
            //Boucle sur le tableau d'erreurs
            foreach($errors as $error) {
                //Création du Toast
                $this->toaster->makeToast($error->toString(), Toaster::ERROR);
            }
            //Redirection
            return $this->redirect('evenement.addEvent');
        }
        //On créer un nouveau évènement
        $new = new Evenement();
        //On récupère l'objet qui représente l'intervenant choisi
        $intervenant = $this->intervenantRepository->find($data['intervenant']);
        //infos de l'évènement
        $new->setNom($data['nom'])
            ->setDescription($data['description'])
            ->addIntervenant($intervenant)
            ->setNbrPlacesDispo($data['nbr_places_dispo'])
            ->setNbrPlacesRest($data['nbr_places_dispo'])
            ->setStartAt(new DateTime($data['startAt'])))
            ->setEndAt(new DateTime($data['endAt'])));
        //Préparation à l'enregistrement en base de données
        $this->manager->persist($new);
        //Enregistrement en base de données
        $this->manager->flush();

        //Création d'un toast de succès
        $this->toaster->makeToast('Évènement ajouté avec succès', Toaster::SUCCESS);

        //on finit par rediriger
        return $this->redirect('evenement.addEvent');
    }

    //On récupère les intervenants
    $intervenants = $this->intervenantRepository->findAll();
    $evenements = $this->listEvent();

    //On rend la vue en passant le tableau en paramètre
    return $this->renderer->render('@evenement/eventAdmin', [
        'intervenants' => $intervenants,
        'evenements' => $evenements
    ]);
}

```

```

public function EventUpdate(ServerRequestInterface $request) {
    //On récupère l'id passé en paramètre de requête
    $id = $request->getAttribute('id');
    //On récupère l'évènement qui correspond à l'id
    $evenement= $this->evenementRepository->find($id);
    //On récupère la method utilisée par la requête
    $method = $request->getMethod();

    //Si le formulaire à été soumis
    if ($method === 'POST') {
        //On récupère les données saisies dans le formulaire ($_POST)
        $data = $request->getParsedBody();
        //On récupère l'intervenant choisi
        $intervenant = $this->intervenantRepository->find($data['intervenant']);
        // On calcule le nombre de places restantes en fonction du nombre de places disponibles
        // nombre de places restantes avant la modif + (nombre de places disponibles après la modif - le nombre de places disponibles avant la modif)
        $nbr_places_restantes = $evenement->getNbrPlacesRest() +
        ($data['nbr_places_dispo'] - $evenement->getNbrPlacesDispo());

        if ($data['nbr_places_dispo'] < 1) {
            // On vérifie que le nombre de places disponibles n'est pas inférieur à 1
            // Si c'est le cas, on affiche un message d'erreur
            $this->toaster->makeToast("Le nombre de places disponibles<br>ne peut pas être inférieur à 1", Toaster::ERROR);
        } elseif ($nbr_places_restantes < 0) {
            // On vérifie que le nombre de places restantes ne devient pas négatif après la modification
            // Si c'est le cas, on affiche un message d'erreur
            $this->toaster->makeToast("Le nombre de places disponibles<br>ne peut pas être modifié<br>sous le nombre de places réservées", Toaster::ERROR);
        } else {
            //Sinon on modifie les données de l'évènement et son intervenant
            $evenement->setNom($data['nom'])
                ->setDescription($data['description'])
                ->setIntervenants($intervenant)
                ->setNbrPlacesDispo($data['nbr_places_dispo'])
                ->setNbrPlacesRest($nbr_places_restantes)
                ->setStartAt(new DateTime($data['startAt'])))
                ->setEndAt(new DateTime($data['endAt']));

            //On enregistre les modifications
            $this->manager->flush();

            //On crée un Toast de success pour l'utilisateur
            $this->toaster->makeToast('Évènement modifié avec succès', Toaster::SUCCESS);

            //On redirige sur la liste des évènements
            return $this->redirect('evenement.addEvent');
        }
    }
    //On récupère les intervenants en base de données pour les utiliser dans le menu select de la vue
    $intervenants = $this->intervenantRepository->findAll();

    //On retourne la vue avec l'évènement que l'on souhaite modifier
    return $this->renderer->render('@evenement/eventUpdate', [
        'evenement' => $evenement,
        'intervenants' => $intervenants
    ]);
}

```

```

public function listEvent(): array {
    //On récupère les évènements
    $evenements = $this->evenementRepository->findAll();

    //On rend la vue en passant le tableau d'évènements en paramètre
    return $evenements;
}

```

```

public function eventDelete(ServerRequestInterface $request) {

    //On récupère l'id passez en paramètre de requête
    $id = $request->getAttribute('id');
    //On récupère l'évènement qui correspond à l'id
    $evenement = $this->evenementRepository->find($id);

    //On prépare l'objet à être supprimé de la base de données
    $this->manager->remove($evenement);
    //On execute la suppression
    $this->manager->flush();

    //On créer un Toast success pour l'administrateur
    $this->toaster->makeToast('Évenement supprimé avec succès', Toaster::SUCCESS);

    //liste des évènements mise à jour
    return $this->redirect('evenement.addEvent');
}

```

LES ROUTES

La class **EventModule** écrite dans le fichier **EventModule.php** présent dans le dossier **evenement**, est responsable de la configuration et de la gestion des routes liées aux évènements.

Les routes sont libellées comme l'exemple ci-dessous:

```

$EventAction = $container->get(EventAction::class);

$this->router->get('/admin/eventAdmin', [$EventAction, 'addEvent'], 'evenement.addEvent');
$this->router->post('/admin/eventAdmin', [$EventAction, 'addEvent']);

```

Une instance de la class **EventAction** est récupérée à partir du conteneur de dépendances.

Les deux routes permettent d'accéder à la même méthode **addEvent** du contrôleur **\$EventAction**, mais en utilisant des méthodes HTTP différentes (GET et POST). Cela permet de traiter différemment les requêtes GET (affichage du formulaire) et POST (traitement des données soumises) pour la même URL **/admin/eventAdmin**.

En GET, **evenement.addEvent** est le nom de la route. Cela permet d'identifier de manière unique cette route dans l'application.

COTÉ ADMINISTRATEUR

L'administrateur peut donc créer un événement grâce à un formulaire lui permettant d'entrer les informations requises.

Ce formulaire est présent dans une page `html.twig` dont vous pouvez voir le code sur la page ci-contre.

Cette page hérite d'un autre modèle appelé `admin.layout.html.twig` qui comporte la barre de navigation. Cela signifie donc que le contenu de cette page sera inséré à l'endroit où se trouve le bloc `{% block body %}` dans le modèle parent `admin.layout.html.twig`.

Le block `{% block stylesheet %}` comporte les feuilles de style CSS qui s'appliquent à la page.

Chaque `input` et `textarea` permet de taper les informations nécessaires sauf l'`input` final de type `submit` qui permet lui de valider. L'élément `select` permet lui d'accéder à la liste déroulante des intervenants enregistrés au préalable grâce à un autre formulaire.

`{{ csrf() }}` est une fonction implémentée en PHP qui génère un jeton CSRF (Cross-Site Request Forgery - Falsification de requêtes intersites) qui est utilisé pour protéger le formulaire contre les attaques CSRF.

The screenshot shows a light gray modal window centered on a dark background. The window contains fields for event creation:

- Nom de l'évènement:** A text input field.
- Descriptif (succinct, 750 caractères maximum):** A large text area.
- Intervenant:** A dropdown menu with the placeholder "Choisir un intervenant".
- Nombre de places:** A text input field.
- Date de début:** A date input field with placeholder "jj/mm/aaaa --:--" and a calendar icon.
- Date de fin:** A date input field with placeholder "jj/mm/aaaa --:--" and a calendar icon.
- CRÉER**: A blue rectangular button at the bottom.

```

{% extends 'admin.layout.html.twig' %}

{% block stylesheet %}
<link rel="stylesheet" href="{{ assets('/assets/css/eventAdmin.css') }}" />
<link rel="stylesheet" href="{{ assets('/assets/css/eventAdmin_responsive.css') }}" />
{% endblock %}

{% block body %}
<h1>Création d'Évènements</h1>

{# Formulaire création d'évènements #-}



<form action="" method="post">
    <div class="champ">
        <label>Nom de l'évènement:</label>
        <input type="text" name="nom" id="nom" required="">
    </div>
    <div class="champ2">
        <label>Descriptif (succinct, 750 caractères maximum):</label><br>
        <textarea name="description" id="description" cols="67" rows="5" required=""></textarea>
    </div>
    <div class="champ2">
        <label>Intervenant:</label><br>
        <select name="intervenant" required="">
            <option value="" selected>Choisir un intervenant</option>
            {% for intervenant in intervenants %}
                <option value="{{ intervenant.id }}">{{ intervenant.nom }}  

                    {{ intervenant.prenom }} - {{ intervenant.profession }}</option>
            {% endfor %}
        </select>
    </div>
    <div class="champ">
        <label>Nombre de places:</label>
        <input type="number" name="nbr_places_dispo" id="nbr_places_dispo" required="">
    </div>
    <div class="champ">
        <label>Date de début:</label>
        <input type="datetime-local" name="startAt" id="startAt" required="">
    </div>
    <div class="champ">
        <label>Date de fin:</label>
        <input type="datetime-local" name="endAt" id="endAt" required="">
    </div>
    {{ csrf() }}
    <input type="submit" value="CRÉER" class="inputSub">


```

COTÉ UTILISATEURS

L'utilisateur connecté peut voir les évènements disponibles qu'il peut réserver sur la page EVENEMENTS du site. Celle-ci n'affiche pas ces informations si le visiteur n'est pas connecté.

Sur l'extrait de code de cette page que vous pouvez voir ci-dessous, le block `{% if user %}` est une instruction conditionnelle qui vérifie si la variable user existe et a une valeur non null. Si c'est le cas, le contenu à l'intérieur du bloc sera affiché. Sinon, il sera ignoré.

La boucle `{% for evenement in evenements %}` itère sur chaque élément de la variable evenements. À chaque itération, le contenu à l'intérieur du bloc sera exécuté. Cela affichera tous les évènements.

Les expressions Twig comme `{{ evenement.nom }}` affichent les valeurs des propriétés de l'objet evenement, telles que le nom, prenom, etc.

Le reste de la page qui n'est pas compris dans le block `{% if user %}` est le contenu lisible de tous, connecté ou non.

Les pages suivantes 36-37 montrent l'affichage obtenu et le code CSS correspondant qui permet de styliser la page.

```
{% extends 'layout.html.twig' %}

{% block meta %}
<meta name="description" content="Les évènements et actions de l'épicerie solidaire Mozart de Metz en faveur d'un public en difficulté"/>
{% endblock %}

{% block stylesheet %}
<link rel="stylesheet" href="{{ assets('/assets/css/evenement.css') }}"/>
<link rel="stylesheet" href="{{ assets('/assets/css/evenement_responsive.css') }}"/>
{% endblock %}

{% block title %}
<title>Évènements et ateliers organisés régulièrement et bénévolement</title>
{% endblock %}

{% block body %}

<div id="h1titre">
    <h1>ÉVÈNEMENTS</h1>
</div>

{% if user %}
<!-- LISTE DES ÉVÈNEMENTS -->
<h2 id="eventitre">LISTE DES ÉVÈNEMENTS PRÉVUS</h2>
<table class="dataTable">
    <tbody>
        {% for evenement in evenements %}
            <tr class="tableColor">
                <td class="etiquette">{{ evenement.nom }}</td>
                <td class="descrip"><h6>Descriptif: </h6><br>{{ evenement.description }}</td>
                <td><h6>Intervenant: </h6><br>
```

```

{%
    for intervenant in evenement.intervenants %
        {{ intervenant.nom }}
        {{ intervenant.prenom }}
        <br>
        ( {{ intervenant.profession }} )
    {% endfor %}
</td>
<td class="dispo"><h6>Places disponibles: </h6><br>{{ evenement.nbrPlacesRest }} sur
{{ evenement.nbrPlacesDispo }}</td>
<td><h6>Débutera le: </h6><br>{{ evenement.startAt }}</td>
<td class="fin"><h6>Finira le: </h6><br>{{ evenement.endAt }}</td>
<td>
<a class="boutModif" href="{{ path('user.reserver', {id: evenement.id}) }}"
    &nbsPRÉSERVER&nbsP</a>
</td>
</tr>
{% endfor %}
</tbody>
</table>
{% endif %}

<!-- CUISINE -->
<div id="cuisine">
    <div class="photo_cuis1"></div>
    <div class="photo_cuis2"></div>
    <div class="photo_cuis3"></div>
    <div class="postit_cuis1"></div>
    <div class="article1">
        <article>
            Notre atelier cuisine est heureux de vous accueillir pour des séances de cuisine à prix
            abordable. Nous travaillons avec des bénévoles non professionnels pour offrir une prestation
            accessible à tous tout en permettant à ces bénévoles de s'impliquer dans un projet solidaire et de
            développer leurs compétences culinaires.
            <br>
            Nous sommes conscients que cuisiner peut être un moyen de se réapproprier son
            alimentation et de faire des économies. Nos chefs animateurs et animatrices sont passionnés et
            mettront tout en œuvre pour vous aider à préparer des repas sains et équilibré. Nous proposons
            également des conseils pour acheter et entreposer des aliments de manière économique et durable.
            <br><br>
            N'hésitez pas à nous rendre visite et à profiter de notre atelier cuisine à prix
            abordable. Nous serons heureux de vous accueillir et de vous transmettre nos connaissances
            culinaires !
        </article>
    </div>
    <div class="titre1">
        <h2>
            CUISINE
        </h2>
    </div>
    <div class="flip1">
        
    </div>
</div>

```

```

body{
    width: 100vw;
    background-image: url('/assets/img/Background.jpg');
}

/*|||||||||MAIN|||||||||*/
main{
    position: relative;
    padding-top: 14%;
}

}h1{
    position: absolute;
    font-family: 'Mochiy Pop P One', sans-serif;
    font-size: 11.3vw;
    z-index: -1;
    color: rgb(204, 199, 194, 0.7);
    rotate: -5deg;
    margin-top: -14%;
}

/*CUISINE*/
#cuisine {
    width: 80vw;
    min-width: 800px;
    height: 600px;
    margin-left: auto;
    margin-right: auto;
    margin-top: 2%;
    margin-bottom: 6%;
    display: grid;
    grid-template-columns: 14% 14% 14% 14% 14% 14% 1fr ;
    grid-template-rows: 60% 1fr 1fr;
    gap: 0px 0px;
    font-family: 'Exo 2', sans-serif;
}

}.photo_cuis1 {
    grid-area: 1 / 1 / 2 / 3;
    background-image: url(/assets/img/événements/cuisine1.jpg);
}

}.photo_cuis2 {
    grid-area: 1 / 3 / 2 / 5;
    background-image: url(/assets/img/événements/cuisine2.jpg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: contain;
}

}.photo_cuis3 {
    grid-area: 1 / 5 / 2 / 7;
    background-image: url(/assets/img/événements/cuisine3.jpg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: contain;
}

}.postit_cuis1 {
    grid-area: 1 / 6 / 2 / 8;
    background-image: url(/assets/SVG/postit_cuisine.svg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: contain;
    z-index: 2;
}

}.article1 {
    grid-area: 2 / 1 / 4 / 5;
    font-weight: 500;
    text-align: justify;
    font-size: 1em;
}

}.titre1 {
    grid-area: 2 / 5 / 3 / 7;
    font-size: 3em;
    font-style: italic;
    color: white;
    padding-left: 12%;
    z-index: 2;
}

}.flip1 {
    grid-area: 1 / 5 / 4 / 8;
    justify-self: end;
    align-self: end;
}

```

LISTE DES ÉVÉNEMENTS PRÉVUS

Event Type	Description	Intervenant	Places disponibles	Débutera le	Finira le	Action
Cuisine facile	Préparation de plats simples pour menus complets et rapides à réaliser	Cuisinier Marie (Chef cuisinière)	4 sur 7	09/07/2023 à 09:30	09/07/2023 à 17:30	RÉSERVER
Coiffure Homme	Prestation coiffure pour homme uniquement	Maître coiffeur Edouard (Coiffeur)	8 sur 9	10/07/2023 à 10:30	10/07/2023 à 17:00	RÉSERVER
Consultations médicales	Visites Médicales pour enfants et parents. RÉSERVÉES AUX FAMILLES MONOPARENTALES UNIQUEMENT. Merci de vous munir des cartes de santé et de tous les documents médicaux dont vous disposez.	Médecin Grégory (Docteur généraliste)	12 sur 12	29/06/2023 à 09:00	29/06/2023 à 12:00	RÉSERVER

CUISINE

Notre atelier cuisine est heureux de vous accueillir pour des séances de cuisine à prix abordable. Nous travaillons avec des ingrédients de saison et des méthodes simples qui sont accessibles et permettent à ces bénévoles de s'impliquer dans un projet solidaire et de développer leurs compétences culinaires. Nous sommes conscients que cuisiner peut être un moyen de se réapproprier son alimentation et de faire des économies. Nos chefs animateurs et animatrices sont passionnés et mettront tout en œuvre pour vous aider à préparer des repas sains et équilibrés. Nous proposons également des conseils pour acheter et entreposer des aliments de manière économique et durable.

N'hésitez pas à nous rendre visite et à profiter de notre atelier cuisine à prix abordable. Nous serons heureux de vous accueillir et de vous transmettre nos connaissances culinaires !

```

/*Liste des évènements*/
.eventitre{
    text-align: center;
    color:#13679c;
    text-transform: uppercase;
    font-family: 'Exo 2', sans-serif;
    font-weight: 600;
    position: relative;
    width: 100%;
    white-space: nowrap;
    display: block;
    animation: elastic 1s ease-in-out forwards;
    border-top: solid 1px;
    border-bottom: solid 1px;
    padding: 1%;
    margin-top: 3%;
}

@keyframes elastic {
    0% {
        letter-spacing: 80px;
        color: rgba(255, 255, 255, 0);
    }
    70% {
        letter-spacing: 0px;
    }
    100% {
        letter-spacing: 10px;
        color:#13679c;
        border-top: solid 1px;
        border-bottom: solid 1px;
    }
}

.dataTables {
    table-layout:auto;
    border-spacing: 10px 30px;
    margin-left: auto;
    margin-right: auto;
    margin-top: 1%;
    width: 80%;
    text-align: center;
    font-family: 'Exo 2', sans-serif;
}

td {
    padding: 1% 0%;
    border-bottom: 1px solid white;
}

.etiquette{
    border-radius: 6px;
    background: radial-gradient(circle at 100% 100%, #13679c 0, #13679c 4px, transparent 4px) 0% / 8px 8px no-repeat,
    radial-gradient(circle at 0 100%, #13679c 0, #13679c 4px, transparent 4px) 100% 0% / 8px 8px no-repeat,
    radial-gradient(circle at 100% 0, #0e4e7a 0, #0e4e7a 4px, transparent 4px) 0% 100% / 8px 8px no-repeat,
    radial-gradient(circle at 0 0, #0e4e7a 0, #0e4e7a 4px, transparent 4px) 100% 100% / 8px 8px no-repeat,
    linear-gradient(#13679c, #0e4e7a) 50% 50% / calc(100% - 8px) calc(100% - 13px) no-repeat,
    linear-gradient(#13679c, #0e4e7a) 50% 50% / calc(100% - 13px) calc(100% - 8px) no-repeat,
}

```

linear-gradient(45deg, rgb(236, 235, 234) 0%, #13679c 100%);
color: white;
font-size: 1.2em;

}

.descrip{
width: 45%;

}

.dispo{
width: 6%;

}

.tableColor {
background-filter: blur(1px);
font-weight: 600;

}

td:last-child{
text-align : center;
margin: 2%;

}

.bout{
color: red;
font-size: 1.2em;
font-weight: 700;
text-decoration: none;

}

.bout:hover{
color: yellow;
font-weight: 900;

}

.boutModif{
background-color: #506a45;
color: white;
font-size: 1em;
font-weight: 700;
text-decoration: none;
text-align : center;
padding: 5% 5%;

}

.boutModif:hover{
background-color: #13679c;
padding: 12% 5%;

}

h6{
color: white;
font-size: 0.8em;
font-weight: 500;

}

La barre de navigation du site reste visible même si on scroll dans la page.

Pour conserver une bonne visibilité des menus, l'arrière-plan se floute lorsque l'on scrolle dans la page grâce à un script JavaScript et au CSS.

JAVASCRIPT

```
const nav = document.getElementById('flou')

function auScroll(event) {
    if (window.pageYOffset < 60) {
        nav.classList.remove('scroller')
    } else {
        nav.classList.add('scroller')
    }
}
window.addEventListener('scroll', auScroll)
```

HTML

```
<div id="flou"></div>
```

CSS

```
#flou{
    position: absolute;
    position: fixed;
    width: 100vw;
    height: 230px;
    z-index: 4;
}

#flou.scroller{
    backdrop-filter: blur(8px);
}
```

Ce script JavaScript :

- Sélectionne un élément HTML ayant l'ID "flou" à l'aide de `document.getElementById('flou')` et le stocke dans la constante nav.
- Définit une fonction `auScroll` qui sera exécutée lorsqu'un événement de scroll se produira sur la fenêtre.
- Vérifie dans la fonction `auScroll` si la valeur de défilement vertical de la fenêtre (`window.pageYOffset`) est inférieure à 60 pixels.
 - Si c'est le cas, il supprime la classe CSS "scroller" de l'élément HTML sélectionné par nav.
 - Sinon, il ajoute la classe CSS "scroller" à l'élément HTML sélectionné par nav.
- Ajoute un écouteur d'événement (addEventlistener) sur la fenêtre pour écouter les événements de défilement (scroll). Lorsque l'événement se produit, la fonction `auScroll` est appelée.

En CSS, le sélecteur `#flou.scroller` sélectionne l'élément qui a à la fois l'ID "flou" et la classe "scroller". La propriété `backdrop-filter` permet d'appliquer l'effet de flou à l'arrière-plan de l'élément.



PROTECTION CONTRE LES ATTAQUES PAR INJECTION SQL

Pour effectuer des opérations de base de données, Doctrine échappe automatiquement les valeurs des paramètres en veillant à ce qu'elles soient traitées de manière sécurisée.

CAPTCHA

Un CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) empêche les attaques de type brute-force, où un attaquant tente de soumettre une grande quantité de données en essayant différentes combinaisons pour trouver une vulnérabilité. Il rend aussi plus difficile une attaque par déni de service (DoS) visant à rendre un service ou une ressource indisponible en submergeant le système ciblé de demandes ou de trafic excessif.

JETON CSRF

La faille CSRF est une attaque qui exploite le fait qu'un site web fait confiance à une demande provenant d'un utilisateur authentifié sans vérifier l'origine de cette demande.

Par exemple si je suis connecté à un site web A et que tout en étant connecté, je visite un autre site B malveillant, ce site B envoie une requête HTTP au site A pour effectuer une action indésirable, telle que modifier mon mot de passe, ou autres actions malveillantes sous mon nom. La raison est que lorsque je suis connecté à certains sites, mon navigateur envoie automatiquement les cookies d'authentification à ce site avec chaque requête HTTP. Ces cookies permettent au site de m'identifier et de savoir que je suis authentifié. L'attaque CSRF exploite cela en utilisant mon navigateur pour effectuer des actions sans mon consentement.

Pour se protéger contre les attaques CSRF, j'utilise un mécanisme appelé jeton CSRF. Ce jeton est un identifiant unique généré par le serveur et inclus dans chaque formulaire ou requête qui modifie l'état du site (comme une demande POST). Lorsque je soumets un formulaire, le jeton CSRF est également envoyé avec la requête.

Lorsque le serveur reçoit la requête, il vérifie que le jeton CSRF fourni correspond à celui attendu. Si les jetons ne correspondent pas ou si aucun jeton n'est fourni, le serveur peut considérer la requête comme suspecte et la rejeter.

Jeu d'essai - Modification d'un événement par l'administrateur

L'administrateur peut voir la liste des événements et leur nombre de places restantes suite aux réservations des utilisateurs. Il a la possibilité de modifier ces informations.

Il souhaite changer le nom et le nombre de places disponibles pour l'évènement **Cuisine Facile** qui a 3 réservations. En cliquant sur modifier il obtient une page comportant le même formulaire que pour la création mais cette fois pré-remplie avec les informations de l'évènement en question.

	Nom:	Description:	Intervenant:	Places disponibles:	Début/Fin:		
X	Cuisine facile	Préparation de plats simples pour menus complets et rapides à réaliser	Cristine Marie (Chef cuisinier)	17 sur 20	09/07/2023 à 09:30	09/07/2023 à 17:30	Modifier
X	Coiffure Homme	Prestation coiffure pour homme uniquement	Manon Agathe (Coiffeur)	8 sur 9	10/07/2023 à 10:30	10/07/2023 à 17:00	Modifier
X	Consultations médicales	Visites Médicales pour enfant et parents. RÉSERVÉES AUX FAMILLES MONOPARENTALES UNIQUEMENT. Merci de vous munir des carnet de santé et de tous les documents médicaux dont vous disposez	Manon Greg (Docteur généraliste)	12 sur 12	29/06/2023 à 09:00	29/06/2023 à 12:00	Modifier

Données entrées	Données attendus	Données reçues	Comparaison attendus / reçus
Tous les champs ne sont pas remplis	Message indiquant de remplir les champs vides	Message indiquant de remplir les champs vides	identique
Le nombre de places est inférieur au nombres de réservations	Message d'erreur indiquant l'impossibilité d'effectuer cette action	Message d'erreur indiquant l'impossibilité d'effectuer cette action	identique
Le nombre de places est supérieur ou égal au nombre de réservations et tous les champs sont remplis	Message de confirmation de modification et affichage de la liste d'événements mises à jour	Message de confirmation de modification et affichage de la liste d'événements mises à jour	identique

La tentative de modifier le nombre de places disponible sous le nombre de réservations déjà effectuées se solde par un message d'erreur l'avertissant qu'il ne peut pas effectuer cette action et le formulaire reste vierge de toute modification.

Après avoir indiqué une valeur correct et changé également le nom, la modification est validée. Un message de succès apparaît. Le nombre de places disponibles à bien diminué de 20 à 14 et les 3 réservations ont été conservées.

Intervenant:	Places disponibles:	Débutera le:	Finira le:	
Cuisto Marie (Chef cuisinière)	11 sur 14	09/07/2023 à 09:30	09/07/2023 à 17:30	Modifier
Maindargent Edouard (Coiffeur)	8 sur 9	10/07/2023 à 10:30	10/07/2023 à 17:00	Modifier
Maison Greg (Docteur généraliste)	12 sur 12	29/06/2023 à 09:00	29/06/2023 à 12:00	Modifier

VEILLE ET RECHERCHES

Les outils de veille

La veille permet de rester informé des diverses nouveautés dans les domaines qui nous intéressent, d'anticiper leur évolution ou encore d'être informé de découvertes importantes voir indispensables.

A cette fin j'utilise plusieurs outils:



Il me permet d'obtenir des informations de sources fiables directement par mail sur mon téléphone portable, facile et efficace à consulter.



Grâce à l'application disponible sur smartphone, cet agrégateur de flux RSS me permet comme Google Alerts d'obtenir des informations fiables par domaine.



Le Codeur Blog regroupe diverses informations sur les milieux du développement, graphisme, design ...



Quora permet créer, d'éditer et d'organiser des questions-réponses. Le site organise les questions-réponses par sujets et permet aux utilisateurs de collaborer et de les visualiser.

Recherche sur site anglophone

TWIG

There are two kinds of delimiters: { % ... % } and {{ ... }}. The first one is used to execute statements such as for-loops, the latter outputs the result of an expression.

Control Structure

A control structure refers to all those things that control the flow of a program - conditionals (i.e. if / elseif / else), for-loops, as well as things like blocks. Control structures appear inside { % ... % } blocks.

Template Inheritance

The most powerful part of Twig is template inheritance. Template inheritance allows you to build a base "skeleton" template that contains all the common elements of your site and defines blocks that child templates can override. The extends tag is the key here. It tells the template engine that this template "extends" another template. When the template system evaluates this template, first it locates the parent. The extends tag should be the first tag in the template.

<https://twig.symfony.com/doc/3.x/templates.html>

Il existe deux types de délimiteurs : { % ... % } et {{ ... }}. Le premier est utilisé pour exécuter des instructions telles que des boucles for, le second affiche le résultat d'une expression.

Structure de contrôle

Une structure de contrôle fait référence à tous les éléments qui contrôlent le flux d'un programme - les conditions (c'est-à-dire if / elseif / else), les boucles for, ainsi que les éléments comme les blocs. Les structures de contrôle apparaissent à l'intérieur des blocs { % ... % }.

Héritage de modèle

La partie la plus puissante de Twig est l'héritage de modèle. L'héritage de modèle vous permet de créer un modèle "squelette" de base qui contient tous les éléments communs de votre site et définit les blocs que les modèles enfants peuvent remplacer.

La balise extends est ici la clé. Elle indique au moteur de modèle que ce modèle "étend" (extends) un autre modèle. Lorsque le système de modèle évalue ce modèle, il localise d'abord le parent. La balise extends doit être la première balise du modèle.

Conclusion et Remerciements

je souhaite exprimer ma profonde gratitude pour l'opportunité qui m'a été offerte d'acquérir des compétences essentielles dans le domaine du développement web.

Ce projet que j'ai réalisé dans le cadre de cette formation, en concevant et en développant un site internet pour une épicerie solidaire, a été un véritable test de mes compétences nouvellement acquises. Grâce à celles-ci, j'ai pu créer un site fonctionnel, sécurisé et esthétiquement plaisant, répondant aux besoins spécifiques de l'épicerie solidaire. Avec l'usage de différents langages, frameworks et logiciels, cette expérience pratique a consolidé mes acquis en graphisme et design, et développé de nouvelles capacités en développement web.

Les formateurs, experts dans leur domaine, ont su transmettre leur savoir-faire de manière claire et concrète, en me guidant tout au long du processus d'apprentissage. J'ai grandement apprécié la disponibilité et le soutien apportés par l'équipe pédagogique.