

# Dossier de projet

## Registre des cancers de Limoges

**Sonia Mouhoubi**

**Coding School 1**  
***2021/2022***

# Sommaire

<b>Compétences du référentiel couvertes par le projet</b>	<b>3</b>
<b>Résumé</b>	<b>4</b>
<b>Spécifications fonctionnelles</b>	<b>5</b>
Acteur du projet	5
Présentation générale du projet	5
Domaine d'activité	5
Présentation de l'Entreprise	5
Description de l'existant	5
Aspects graphiques	6
Périmètre du projet	6
Arborescence du site	6
<b>Description des fonctionnalités</b>	<b>7</b>
Fonctionnalités côté utilisateur	7
Fonctionnalités côté administrateur	7
<b>Spécifications techniques</b>	<b>8</b>
Technologies utilisées pour la partie front-end	8
Technologies utilisées pour la partie back-end	8
Environnement de développement	8
Architecture des dossiers	9
Adaptabilité du site aux écrans	10
<b>Réalisation du site</b>	<b>10</b>
Charte graphique	10
Maquette graphique	11
Conception base de données	14
Extraits de code significatifs	15
<b>Sécurisation et veille</b>	<b>25</b>
Faille XSS	27
injection SQL	28
Faille upload	28
Connexion à l'administration	29
Problème rencontré	30
Extrait d'un site anglophone	30
<b>ANNEXES</b>	<b>31</b>
Cahier des charges	31

## Compétences du référentiel couvertes par le projet

Le projet couvre les compétences énoncées ci-dessous.

Pour l'activité type 1, « **Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité** » :

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

Pour l'activité type 2, « **Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité** » :

- Créer une base de données
- Développer les composants d'accès aux données
- Développer la partie back-end d'une application web ou web mobile
- Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce

## Résumé

Le **Registre des Cancers de Limoges** est un site web qui recueille les données liées à la maladie du cancer. Il a été élaboré pour *la responsable scientifique du registre des cancers* de la Haute-Vienne **Tania D'ALMEIDA** et *l'épidémiologiste* **Elfried SALANON**.

Le but de ce projet est d'**informer** et de **sensibiliser** les professionnels de santé, la population de Limoges ainsi que tous les internautes qui se rendront sur le dispositif sur les cancers ainsi que leurs prises en charges et sur les chiffres liés à la maladie. Cela permet aussi de mettre en avant ce qu'est un registre et son utilité.

Sur le site, le visiteur a accès :

- à la présentation du registre
- aux chiffres des cancers de Haute-Vienne
- à de l'actualité diverse
- aux activités de recherche pour le cancer
- à un formulaire de contact

Une **partie administration** a été créée pour l'administrateur afin qu'il puisse avoir la main sur son site. Il a la possibilité de voir, ajouter, modifier et supprimer des articles tels que les chiffres des cancers, les actualités, ou les activités de recherche. Une page unique par article a été créée afin de voir la fiche détaillée. Il a aussi accès à son profil qu'il pourra modifier et supprimer.

# Spécifications fonctionnelles

## Acteur du projet

### Client :

- Tania D'ALMEIDA : Responsable scientifique du registre général des cancers de la Haute-Vienne
- Elfried SALANON : Epidémiologiste

### Développeur :

- Sonia MOUHOUBI : Etudiante en développement web et mobile

## Présentation générale du projet

Création d'un site vitrine pour le registre des cancers de la ville de Limoges. Une partie administration permettra l'ajout d'articles par l'administrateur.

Le projet permettra de mettre en avant ce qu'est un registre des cancers et de sensibiliser la population de Limoges ainsi que tous visiteurs sur le site et les professionnels sur les cancers et les données liées à la maladie.

## Domaine d'activité

Le domaine d'activité est la santé et les recherches liées aux cancers.

## Présentation de l'Entreprise

Centre Hospitalier et Universitaire de Limoges (CHU Limoges)

Cible visé

Le site s'adresse aux utilisateurs (professionnels compris) qui veulent avoir accès aux données des cancers.

## Description de l'existant

Un ancien site avait été créé auparavant mais n'est plus existant. Le site actuel ne s'appuie donc plus sur l'ancien. Un nom de domaine va être acheté ainsi qu'un hébergement web. Un logo a été procuré et du contenu va être fourni afin d'administrer le site.

Plusieurs échanges avec les clients ont été faits afin de répondre aux besoins du projet (aspects graphiques, fonctionnalités à développer pour l'application).

## Aspects graphiques

Les clients désiraient un site au ton clair et épuré. Les couleurs à privilégier étaient le bleu, le vert et le jaune qui rappellent les couleurs du logo et les couleurs liées à la santé.

Le site devait montrer le professionnalisme de leur activité sans mettre trop d'images afin de mener le visiteur directement à l'information.

## Périmètre du projet

Le site sera réalisé en français et ce dernier devra être accessible sur différents supports (mobile, tablette et ordinateur).

## Arborescence du site

Le site comporte :

- Page **d'accueil**
- Page **présentation**
- Page **chiffres des cancers**
- Page **actualité**
  - Page **unique d'actualité**
- Page **activité de recherche : Projet en cours**
- Page **activité de recherche : Projet en passés**
- Page **activité de recherche : Projet en futur**
  - Page **unique par projet**
- Page **contact**
- Page **connexion** à l'administration du site
- Page **admin : profil**
- Page **admin : chiffres des cancers**
  - Page **unique chiffres des cancers**
- Page **admin : actualité**
  - Page **unique : actualité**
- Page **admin : activité de recherche**
  - Page **unique : activité de recherche**
- Page **admin : message contacts**
- Page **admin : formulaire d'ajout d'article**

# Description des fonctionnalités

## Fonctionnalités côté utilisateur

### Affichage des articles postés

Le visiteur a la possibilité de voir les dernières actualités sur la page d'accueil de son site. Il pourra voir afficher les articles des chiffres des cancers, des actualités et des activités des recherches sur les pages dédiées.

### Affichage de la page unique article

Le visiteur peut accéder à la page détaillée de chaque article. La page comportera au minimum un titre est une description et pourra contenir une image ou un fichier et la date du poste.

### Formulaire de contact

L'utilisateur peut demander des informations par le biais d'un formulaire de contact.

## Fonctionnalités côté administrateur

### Formulaire de connexion

L'administrateur du site pourra se connecter à la partie administration du site après authentification grâce à un formulaire de connexion.

### Espace Profil

Après authentification, l'administrateur a accès à son profil. Il aura la possibilité de voir, modifier et supprimer son profil.

### Ajout d'article

Dans la partie administrative du site, l'administrateur pourra ajouter des actualités, des articles liés aux chiffres des cancers et des articles sur les activités de recherche. via un formulaire. Les articles comporteront au minimum une catégorie, un titre, une description et pourront contenir une image ou un fichier et la date du poste. Ils pourront être lus, modifiés et supprimés.

## Ajout de catégorie

L'administrateur aura la capacité de créer de nouvelles catégories ainsi que les consulter. Il pourra aussi les modifier et les supprimer.

# Spécifications techniques

## Technologies utilisées pour la partie front-end

Le site a été réalisé avec les langages **HTML** (HyperText Markup Language), **CSS** (Cascading Style Sheets) et **JavaScript**. Des **media Queries** ont été utilisés pour la responsive du site.

## Technologies utilisées pour la partie back-end

La partie back-end du site a été développée en **PHP** (Hypertext Preprocessor) et **SQL** (Structured Query Language) pour la base de données.

## Environnement de développement

Le site a été développé avec :

**WAMP SERVER** : pour l'environnement en local

- **PhpMyAdmin** : pour l'administration de la base de données

**VISUAL STUDIO CODE** : pour la création du code

**TORTOISE GIT** et **GITHUB** : pour stocker différentes versions de code

**TRELLO** : pour la planification des tâches

**FIGMA** : pour la création de la maquette

**LOOPING** : pour la création du **MCD** et **MLD**

## Recherche

Les recherches pour le développement du site ont été faites avec le navigateur **GOOGLE**.

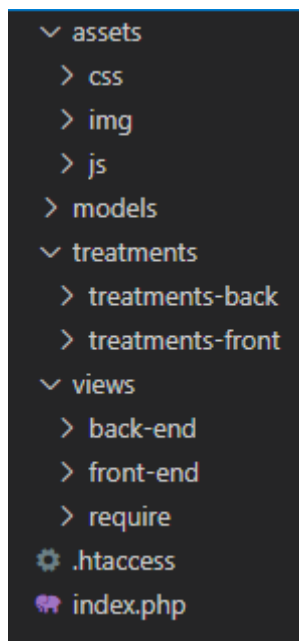


Exemple de site pour la recherche en français : <https://developer.mozilla.org/fr>  
Exemple de site pour la recherche en anglais : <https://stackoverflow.com/>

## Architecture des dossiers

Le dossiers du site sont structurés comme suit :

- Un dossier **assets** qui stocke :
  - un dossier **img** où se trouve les photos et fichiers du site.
  - un dossier **css** qui comporte le css du site
  - un dossier **js** : qui contient le script javascript du site
- Un dossier **models** où se trouve les classes du site qui communiquent avec la base de données.
- Un dossier **treatments** dans lequel se trouvent le traitement, la logique du site qui sont séparés dans un dossier **treatments-front** et **treatments-back**.
- Un dossier **views** qui contient les pages qui vont être affiché sur le site. Dans ce dossier on retrouve les dossiers :
  - **require**
  - **front-end**
  - **back-end**
- Un fichier **.htaccess** pour la réécriture d'URL.
- Un fichier **index.php**.



## Adaptabilité du site aux écrans

Le site a été créé en **Mobile First** afin de se concentrer sur la clarté de l'interface utilisateur. Il utilise des requêtes **media Queries** afin de pouvoir modifier l'apparence du site en fonction de l'écran sur lequel il est projeté. Le site est donc adapté au mobile, tablette et ordinateur. Il peut aussi être projeté sur des écrans de téléviseurs.

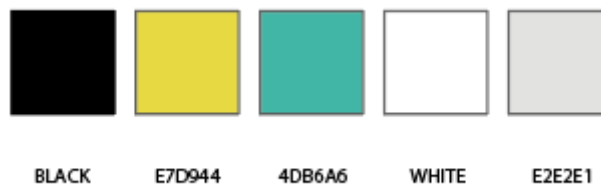
## Réalisation du site

### Charte graphique

La charte graphique a été conçue de la manière suivante :

Police d'écriture général : '**Poppins**', sans-serif

Police d'écriture des titres : **Roboto**, sans-serif



### Maquette graphique

La maquette du site a été réalisée avec le logiciel en ligne **FIGMA**.

Pour la réalisation de la maquette, j'ai effectué une **veille technique** sur les tendances des sites web liés à la santé. Je me suis appuyée sur les screens et les liens que m'avaient envoyés mes clients afin qu'ils puissent avoir un site qui ressemble à leurs attentes.

J'ai tout d'abord élaboré des **wireframes** sans couleurs en **version mobile** puis en **version desktop** pour me faire une idée des pages que comporterait le site. Puis j'ai créé la maquette haute fidélité avec les couleurs et les images fictives.

## Bienvenue sur le registre des cancers de Limoges.

### PRESENTATION DU REGISTRE

#### Définition :

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

#### Objectif :

Est sapiente quis qui doloribus voluptates et sequi libero aut dignissimos veritatis. Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.



Carte France / Limoge

## Actualités



### ACTUALITE 1

Lorem ipsum dolor sit amet  
20/01/2022

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

Toute l'actualité +



### ACTUALITE 2

Lorem ipsum dolor sit amet  
20/01/2022

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

Toute l'actualité +



## Actualités


### ACTUALITE 1

Lorem ipsum dolor sit amet  
20/01/2022



Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

Toute l'actualité +



Présentation
Chiffres des cancers
Activité de recherche
Actualités
Contact

# Bienvenue sur le registre des cancers de Limoges.


## PRESENTATION DU REGISTRE

**Définition :**

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

**Objectif :**

Est sapiente quis qui doloribus voluptates et sequi libero aut dignissimos veritatis. Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.



## Actualités



### ACTUALITE 1

20/01/2022

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

Toute l'actualité +



### ACTUALITE 2

20/01/2022

Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

Toute l'actualité +



## Actualités

### ACTUALITE 1

20/01/2022

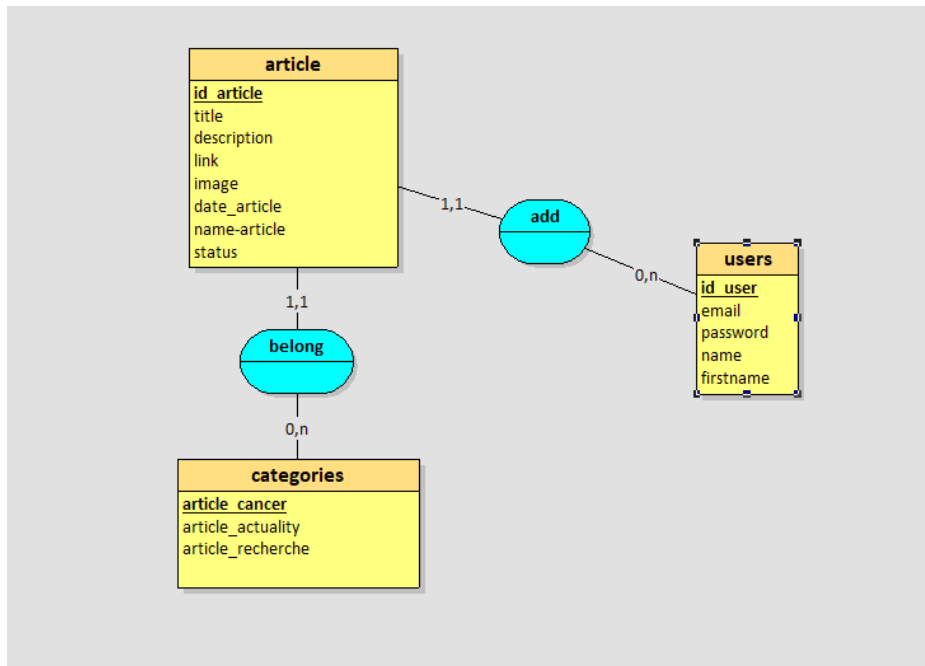


Lorem ipsum dolor sit amet. In Quis nisi ea alias consectetur ut inventore ipsam sit sint vitae eum illum soluta est asperiores optio.

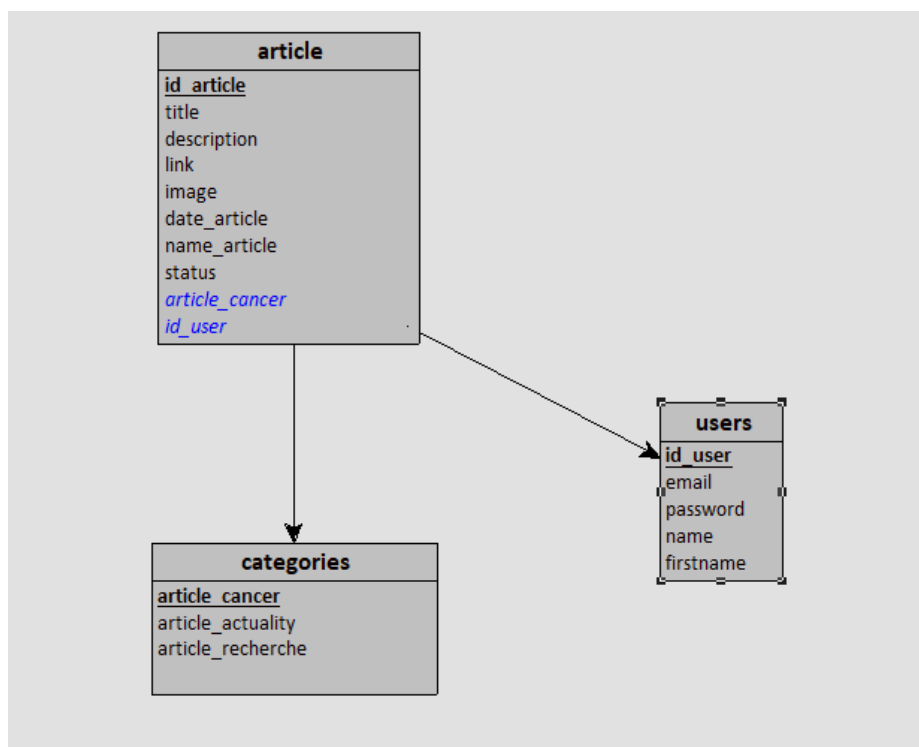
Toute l'actualité +

## Conception base de données

La base de données a été conçue selon la méthode d'analyse et de conception **MERISE**. Afin de pouvoir créer les relations et les dépendances entre les différents acteurs (tables), j'ai mis en place le modèle conceptuel de base de données (**MCD**). Et de ce dernier, j'ai élaboré le modèle logique des données (**MLD**).



MCD



MLD

## Extraits de code significatifs

### Structure du site en HTML

Pour le développement du site, j'ai tout d'abord commencé par créer la structure du site en **HTML**. Dans le **dossier require**, on peut retrouver le **HEADER** et le **FOOTER** qui se retrouvent dans toutes les pages.

```
<body>
  <header>
    <div class="logo">
      <a href="accueil"></a>
    </div>

    <input type="checkbox" id="burger" hidden>
    <label class="menu-burger" for="burger">
      <span id="span1"></span>
      <span id="span2"></span>
      <span id="span3"></span>
    </label>

    <nav class="nav">
      <a href="accueil"></a>

      <ul>
        <li>
          <a href="presentation">Présentation</a>
        </li>
        <li>
          <a href="chiffres-cancers">Chiffres du cancer en Haute-Vienne</a>
        </li>
        <li>
          <a href="activites-recherches">Activité de recherche</a>
        </li>
        <li>
          <a href="actualites">Actualités</a>
        </li>
        <li>
          <a href="contact">Contact</a>
        </li>
      </ul>
    </nav>
  </header>
```

### Aspect visuel en CSS

Puis j'ai reproduit la maquette en utilisant du **CSS**. J'ai commencé par mettre en place un CSS de base qui s'appliquera à toutes les pages puis un CSS par section en utilisant les sélecteurs de classes.

```

/* ..... */
* {
  box-sizing : border-box;
  margin: 0;
  padding: 0;
}
body {
  scroll-behavior: smooth;
  line-height: 1.25;
  margin : 0;
  padding: 0;
  font-family: 'Poppins', sans-serif;
  font-size: 1rem;
}
section {
  padding: 1rem;
}
img {
  max-width: 100%;
  padding: 1rem;
}
a {
  text-decoration : none;
  font-weight : bold;
  font-weight: bolder;
  color: black;
}
ul {
  margin: 0;
  padding: 0;
}
ol {
  list-style: none;
  margin: 0;
  padding: 0;
}

/* Navigation */
.nav ul {
  width: 100%;
  height: 0;
  overflow: hidden;
  transition: ease-in-out 0.2s;
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
}
.nav li {
  width: 100%;
  text-align: center;
}
.nav li a {
  width: 100%;
}
/* LOGO */
.nav img {
  width: 10rem;
  padding: 0;
}
/* Menu BURGER */
.menu-burger {
  z-index: 10;
  position: absolute;
  top : 2em;
  right : 3.5em;
}
.menu-burger span {
  position: absolute;
  display: block;
  width: 40px;
  height: 7px;
  background-color : #4db6a6;
  border-radius: 25px;
}

```

## Responsive avec les Medias Queries et Flex

Afin de rendre le site responsive, j'ai d'abord élaboré le design en version mobile, puis en version tablette et en version grand écran. Créer le site en commençant par la version mobile facilite la conception car généralement ce sont des blocs qui se superposent l'un après l'autre.

Ensuite pour adapter le site en tablette et desktop, j'ai utilisé les médias queries.

```

@media only screen and (min-width: 768px){
/* :::::::::::::::::::::::::::::::::::::::::::::::::: CSS PAR DEFAULT ::

    section {
        /* padding-top: 2rem; */
        padding-bottom: 3rem;
    }
    h2 {
        padding: 2rem 0;
    }

/* :::::::::::::::::::::::::::::::::::::::::::::::::: HEADER ::::::::::

    .nav {
        display: flex;
    }
    .nav ul {
        height: auto;
        flex-wrap: nowrap;
    }
    .nav li {
        width: auto;
        padding: 2rem 1rem 1rem;
    }
    /* LOGO */
    .nav img {
        padding: 0 0.5rem;
    }
    /* Menu BURGER */
    .menu-burger {
        display: none;
    }

/* :::::::::::::::::::::::::::::::::::::::::::::::::: FOOTER ::::::::::

    footer {
        display: flex;
        justify-content: space-around;
    }

```

Pour permettre aux éléments de se mettre côte à côte, j'ai utilisé la propriété **flex** de CSS ce qui permet la flexibilité des éléments.



```

.actualities {
  display: flex;
  flex-wrap: wrap;
}
.actualities h2 {
  width: 100%;
}
.actualities div {
  width: 40%;
  margin: auto;
}
.actualities img {
  width: 50%;
  float: left;
  margin-right: 1rem;
}
.actualities span {
  text-align: left;
}

```

## Interactivité du site en JS

Pour rendre le site interactif (mettre à jour des éléments ou infos sans rechargement de page, avoir une réaction après une action faite par l'utilisateur) j'ai utilisé le langage **JavaScript**.

Dans le formulaire de connexion à l'administration du site, j'ai utilisé la **méthode fetch()**, afin de vérifier les informations entrées par l'utilisateur.

```

fetch("../treatments/treatment-connexion.php", {
  method: "POST",
  body: formData,
})
.then((response) => response.json())
.then((response) => {

```

## Inclure du contenu externe dans un fichier grâce au PHP

Grâce au **langage php** qui permet de produire des pages web dynamiques, j'ai inclus le contenu du HEADER et du FOOTER dans toutes les pages du site. J'ai utilisé la **fonction require** qui permet d'appeler et d'exécuter du code contenu dans un fichier externe.

```
require ('views/require/views-header.php');
```

```
<?php require ('views/require/views-footer.php'); ?>
```

## Réécriture d'url

Afin d'avoir de 'jolies' URL, avec un rendu plus professionnel, j'ai procédé à une réécriture d'URL grâce au fichier **.htaccess**.

```
RewriteEngine On

RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^(.*)$ index.php?url=$1 [NC,L]
```

Dans le fichier **index.php** qui est à la racine du dossier, j'ai créé une fonction pour ne pas faire apparaître l'extension des fichiers. Cela évitera de savoir quel langage serveur est utilisé. Bien que cela ne sécurise pas le site, moins un utilisateur malveillant en sera sur le fonctionnement interne du site, plus il sera difficile de le malmener.

```
<?php
if (isset($_GET['url'])) {

    $name_file = explode("/", $_GET['url']);
    $name_file = end($name_file);

}
else {

    $name_file = "accueil";

}

function render($name_file) {

    $front_pages = ['accueil', 'actualites', 'chiffres-cancers', 'connexion',
    'contact', 'activites-recherches', 'presentation', 'actualite-unique'];

    if(in_array($name_file, $front_pages)) {

        require("views/front-end/views-$name_file.php");

    } else {

        require("views/front-end/views-error404.php");

    }

}

render($name_file);
```

## POO

Afin de pouvoir interagir avec la base de données et donc de pouvoir afficher le contenu du site dynamiquement, j'ai construit et organisé mon code en utilisant la **programmation orienté objet** (POO). Pour ce faire, j'ai utilisé des **classes**.

Une **classe est** un ensemble de code contenant des variables et des fonctions permettant de créer des **objets**. En soi, c'est la somme des propriétés d'un objet et des traitements dont il est capable.

Pour éviter de se répéter on peut faire hériter une classe enfant de la classe mère. La classe enfant hérite donc de la classe mère grâce au mot clé **extends**.

Ici, la classe Category extends de la classe **Db** pour pouvoir se connecter à la base de données. Par convention, on utilise une majuscule au début d'un nom de classe pour les différencier dans le reste du code.

```
<?php

require_once('Db.php');

class Category extends Db {

    private $id_category;
    private $new_category;

    private $idSubCategory;
    private $nameCategory;
    private $nameSubCategory;
```

La classe Article quant à elle, extends de la classe Category ce qui fait qu'elle a hérité de toutes les **propriétés** et **méthodes** de Article ainsi que de Db.

Un **objet est** une représentation d'une chose matérielle ou immatérielle à laquelle on associe des propriétés et des actions.

Exemple :

**Objet** : une voiture.

**Propriété** : roues, moteur, sièges ...

**Action** : rouler, freiner ...

C'est un bloc de code mêlant des variables et des fonctions, appelées respectivement **attributs** et **méthodes**.

Les **attributs** sont les caractéristiques propres à l'objet, ce sont donc ses **propriétés**. Les **méthodes** sont les actions applicables à un objet, ce sont donc les **actions** de l'objet.

## SQL

Pour pouvoir interroger la base de données, j'ai utilisé le **langage SQL**. Ce langage m'a permis d'écrire, de lire, de modifier et de supprimer les données stockées. Avec ce langage, il est possible aussi de modifier la structure de la base de données en ajoutant et modifiant des tables par exemple. Il est aussi possible de gérer les bases de données en créant ou modifiant de nouvelle base. Le système de gestion de base de données (SGBD) qui a été utilisé est **MySQL**. L'interface qui a permis au script PHP d'interroger la base de données via des requêtes SQL est **PDO** (PHP Data Objects).

Si le pilote SGBD MySQL n'est pas déclaré, il faut aller dans le fichier **php.ini** et ajouter la ligne suivante :

```
extension = php_pdo_mysql.dll
```

La ligne suivante doit aussi figurer :

```
extension = php_pdo.dll
```

## PDO

Pour pouvoir se connecter à la base de données, il faut faire une instance de **PDO**. Afin de gérer d'éventuelles erreurs lors de l'instanciation, il faut rattraper l'erreur grâce à **PDOException**.

```

class Dd {
    protected $db;

    public function __construct() {
        try {
            $this->db = new PDO('mysql:host=localhost;dbname=registre-cancers;charset=utf8', 'root', '');
        }
        catch (PDOException $e)
        {
            die('Erreur : ' . $e->getMessage());
        }
    }
}

```

L'un des points forts de PDO est les **requêtes préparées**. Une requête préparée est plus rapide et plus sûre surtout contre les tentatives d'**injections SQL**.

Pour cela, il faut utiliser la méthode **prepare()** de l'objet PDO qui permet donc de préparer une requête. On peut soit mettre dans les **values** d'une requête des points d'interrogation ou le symbole **:** et le nom des champs de la base de données.

Pour la première façon de faire, dans la méthode **execute()**, on place les valeurs dans un tableau qui remplaceront dans l'ordre, les points d'interrogation.

Pour la deuxième façon, on placera dans un tableau associatif les valeurs dont les clés sont les paramètres nommés de la méthode **prepare()** en tant chaîne de caractère.

```

public function getInfosUser($id)
{
    $this->id = $id;

    $req = $this->db->prepare("SELECT * FROM client WHERE id_client = ?");
    $req->execute([$id]);
    $res = $req->fetch(PDO::FETCH_ASSOC);

    return $res;
}

```

Lorsque l'on souhaite afficher des données, on a la possibilité d'utiliser la méthode **fetch()** de la classe **PDOStatement**. Cette méthode permet de récupérer ligne par ligne les résultats. Ou la méthode **fetchAll()** qui retourne un tableau des données. On utilise souvent **PDO::FETCH\_ASSOC** pour dire que l'on veut récupérer les résultats dans un tableau associatif.

```
$res = $req->fetch(PDO::FETCH_ASSOC);
```

## Ajout d'une nouvelle catégorie

Dans le fichier de traitement qui gère le formulaire des catégories, j'ai créé une fonction qui me permet d'ajouter une nouvelle catégorie. Dans une variable, je récupère les données grâce à la superglobale POST. Pour plus de sécurité, j'intègre le POST dans la fonction **valid\_data()** que j'ai créé pour contrer les failles web.

```
function register_new_category() {  
    if(isset($_POST['registerCat']))  
    {  
        $new_category = valid_data($_POST['newCategory']);  
  
        if(isset($new_category) && !empty($new_category)) {  
            if(!empty($category)) {  
                $category = new Category;  
                $category->registerNewCategory($new_category);  
            }  
            else {  
                $_SESSION['message'] = "Cette catégorie existe déjà !";  
            }  
        }  
    }  
}
```

## Modification d'un catégorie

Pour la modification de catégorie, j'ai récupéré l'ID de la catégorie dans l'URL afin de pouvoir modifier la bonne catégorie.

```

function update_category() {

    $category = new Category;
    $id_url = explode("/", $_GET['url']);
    $id_url = end($id_url);

    $res = $category->get_category_by_ID($id_url);

    if(isset($_POST['update_cat']))
    {
        $cat = valid_data($_POST['category']);

        if(isset($cat) && !empty($cat)) {
            $category->update_category($_POST['category'], $id_url);

            $_SESSION['msg'] = "<p>La catégorie a été modifiée avec succès.</p>";
        }
    }
}

```

## Suppression d'un article

Pour la suppression d'un article, j'ai récupéré l'ID de l'article dans l'URL afin de pouvoir supprimer le bon article.

```

function delete_article() {

    $article = new Article;
    $id_url = explode("/", $_GET['url']);
    $id_url = end($id_url);

    if(!empty($_POST['delete'])) {

        $article->delete_article($id_url);
    }

    $_SESSION['msg'] = "<p>L'article a bien été supprimé.</p>";
}

```

## Affichage des articles d'actualités

Pour afficher les actualités, j'ai utilisé la boucle **foreach()** afin de récupérer les infos de la base de données. Puis, j'ai affiché les informations qui m'intéressaient en faisant un **echo** des valeurs du tableau récupéré grâce au foreach.

```

<?php foreach ($articles as $value) { ?>

    <div>
        <h2><?= valid_data($value['name_article'])?></h2>

        <span><?= valid_data($value['date'])?></span>

        <p><?= valid_data($value['description'])?></p>

        <a href="">En savoir +</a>
    </div>

<?php } ?>

```

## Sécurisation et veille

Durant le développement du projet, des recherches ont été faites afin de sécuriser le site des cyberattaques. Des sites comme StackOverflow, OWASP, php.net, developer.mozilla, OpenClassRoom ont répondu à mes problématiques.

Exemple de recherche faite en anglais et en français :

- <https://owasp.org/www-project-top-ten/>
  - **A01:2021-Broken Access Control** moves up from the fifth position; 94% of applications were tested for some form of broken access control. The 34 Common Weakness Enumerations (CWEs) mapped to Broken Access Control had more occurrences in applications than any other category.
  - **A02:2021-Cryptographic Failures** shifts up one position to #2, previously known as Sensitive Data Exposure, which was broad symptom rather than a root cause. The renewed focus here is on failures related to cryptography which often leads to sensitive data exposure or system compromise.
  - **A03:2021-Injection** slides down to the third position. 94% of the applications were tested for some form of injection, and the 33 CWEs mapped into this category have the second most occurrences in applications. Cross-site Scripting is now part of this category in this edition.
  - **A04:2021-Insecure Design** is a new category for 2021, with a focus on risks related to design flaws. If we genuinely want to “move left” as an industry, it calls for more use of threat modeling, secure design patterns and principles, and reference architectures.
  - **A05:2021-Security Misconfiguration** moves up from #6 in the previous edition; 90% of applications were tested for some form of misconfiguration. With more shifts into highly configurable software, it's not surprising to see this category move up. The former category for XML External Entities (XXE) is now part of this category.
  - **A06:2021-Vulnerable and Outdated Components** was previously titled Using Components with Known Vulnerabilities and is #2 in the Top 10 community survey, but also had enough data to make the Top 10 via data analysis. This category moves up from #9 in 2017 and is a known issue that we struggle to test and
- <https://openclassrooms.com/fr/courses/6179306-securisez-vos-applications-web-avec-lowasp/6520368-stoppez-le-cross-site-scripting-xss>



## Définissez le cross-site scripting ?



Les **attaques cross-site scripting** ou **XSS** sont faites pour prendre le contrôle de votre **navigateur**. Un attaquant qui y parvient aura potentiellement accès à vos **cookies** et à vos **sessions** qui peuvent contenir des **données sensibles** ! Ces attaques peuvent également permettre d'apporter des modifications non autorisées à une application web et créer des liens qui vous mèneront vers des **sites malveillants** !

### Découvrez les attaques utilisant XSS

Avec une attaque XSS, un attaquant va essayer de prendre le contrôle de votre navigateur en injectant un **script JavaScript** dans l'application web. Il pourra l'injecter directement dans un formulaire, mais il peut également l'injecter dans l'URL, l'en-tête HTTP ou d'autres parties du framework utilisé.

Contrairement aux injections SQL, il ne s'agit pas de requêtes et de commandes SQL sur une base de données. **Une faille XSS s'exécute dans le code de l'application web.**

Revenons à la page de connexion avec le nom d'utilisateur et le mot de passe. Au lieu du nom d'utilisateur, le pirate va entrer :

```
<script> I haxx000red you </script>
```

- <https://stackoverflow.com/questions/601300/what-is-sql-injection#:~:text=SQL%20injection%20happens%20when%20you.error%20than%20in%20a%20vulnerability>.

### How does it cause vulnerabilities?

It can lead to vulnerabilities because attackers can send values to an application that they know will be interpolated into a SQL string. By being very clever, they can manipulate the result of queries, reading data or even changing data that they shouldn't be allowed to do.

Example in PHP:

```
$password = $_POST['password'];  
$id = $_POST['id'];  
$sql = "UPDATE Accounts SET PASSWORD = '$password' WHERE account_id = $id";
```

Now suppose the attacker sets the POST request parameters to "password=xyzy" and "id=account\_id" resulting in the following SQL:

```
UPDATE Accounts SET PASSWORD = 'xyzy' WHERE account_id = account_id
```

Although I expected `$id` to be an integer, the attacker chose a string that is the name of the column. Of course now the condition is true on *every* row, so the attacker has just set the password for *every* account. Now the attacker can log in to anyone's account -- including privileged users.

Pour la sécurité, des validations frontend avec JS et validation HTML avec l'attribut type de l'input et l'attribut required ont été faites. Côté backend des validations en PHP ont aussi été vérifiées.

Pour se prémunir des failles de sécurité telle que :

- **Faillle XSS** : Le **cross-site scripting** consiste à injecter du contenu malveillant en JavaScript dans une page provoquant différentes actions comme la redirection vers un autre site pour de l'hameçonnage, bloquer l'accès au site, prendre le contrôle d'un site web pour ajouter ou supprimer du contenu, ou encore pour récupérer des données transmises par les utilisateurs.
- **Injection SQL** : L'injection SQL est une technique qui permet d'injecter des éléments notamment du code de type SQL dans les champs de formulaires web ou dans les liens de pages afin de les envoyer au serveur. De cette façon, les attaquants outrepassent les contrôles de sécurité et parviennent à afficher ou à modifier des éléments présents dans une base de données, par exemple des mots de passe ou des coordonnées bancaires.
- **Faillle UPLOAD** : La **faille upload** est une **faille** permettant d'**uploader** des fichiers avec une extension non autorisée. Elle est due à la mauvaise configuration du script d'upload ou à l'absence complète de sécurité.

J'ai créé pour les entrées et les sorties de données, une fonction appelée **valid-data()** qui permet de se prémunir de ces failles.

```
function valid_data($data) {  
    $data = trim($data);  
    $data = stripslashes($data);  
    $data = htmlspecialchars($data);  
    $data = strip_tags($data);  
    return $data;  
}
```

## Faillle XSS

Pour la **faille XSS**, j'ai utilisé les fonctions suivante :

- **trim()** : supprime les espaces et autres caractères prédéfinis des deux côtés d'une chaîne.

- **stripslashes()** : Supprime les antislashes d'une chaîne.
- **htmlspecialchars()** : Convertit les caractères spéciaux en entités HTML
- **strip\_tags()** : Supprime les balises HTML et PHP d'une chaîne

## injection SQL

Afin de protéger nos requêtes contre les **injections SQL**, j'ai utilisé la **méthode quote()** de la classe PDO qui se charge d'ajouter les guillemets autour des chaînes, et d'éviter les injections en doublant les guillemets à l'intérieur de la chaîne.

J'ai aussi utilisé les **déclarations préparées**. Une instruction déclarée est un modèle de requête SQL dont on spécifie des paramètres à un stade ultérieur pour l'exécuter.

On prépare donc les requêtes avant de les exécuter comme ci-dessous :

```
public function getCategory() {
    $req = $this->db->prepare("SELECT * FROM `category` ORDER BY `id_category` DESC");
    $req->execute();
    $res = $req->fetchAll(PDO::FETCH_ASSOC);

    return $res;
}
```

## Faible upload

Pour les fichiers du site (image ou document), j'ai vérifié la taille de fichier et le type d'extension. Des restrictions ont été faites pour la taille ainsi que le type de fichier. De plus, j'ai généré un nom aléatoire pour chaque fichier uploadé grâce aux fonctions **md5()**, **uniqid()** et **rand()** et enregistré le nom dans la base de données.

```

if(isset($_POST['register']))
{
    // Avant d'upload l'image on initialise une taille max à 5mo et les formats autorisés
    $maxSize = 50000;
    $validExt = array('.jpg', '.jpeg', '.png', '.svg');

    if($_FILES['image']['error'] > 0)
    {
        $_SESSION['msg'] = "Une erreur est survenue lors du transfert.";
    }

    $fileSize = $_FILES['image']['size'];

    if($fileSize > $maxSize)
    {
        $_SESSION['msg'] = "<p>Le fichier ne doit pas dépassé 5 Mo.</p>";
    }

    $fileName = $_FILES['image']['name'];
    $fileExt = '.' . strtolower(substr(strrchr($fileName, '.'), 1));

    if(!in_array($fileExt, $validExt))
    {
        $_SESSION['msg'] = "<p>Le format du fichier n'est pas accepté.</p>";
    }

    $tmpName = $_FILES['image']['tmp_name'];
    $uniqueName = md5(uniqid(rand(), true)); // Générer des noms d'images aleatoires

    $fileName = 'public/img/'.$uniqueName.$fileExt;

```

## Connexion à l'administration

Pour la connexion à l'administration du site, une authentification avec une adresse mail et un mot de passe long est demandé à l'administrateur avec une majuscule, une minuscule, un chiffre et un caractère spécial minimum.

Le mot de passe est vérifié grâce à la fonction **\$password\_verify** : fonction qui permet de vérifier qu'un mot de passe correspond à un **hachage**.

Dans l'espace profil de l'admin, l'administrateur à la possibilité de modifier son email ainsi que son mot de passe. Le nouveau mot de passe sera haché grâce à la fonction **hashed\_password()**.

```

// Hachage du mot de passe
$hashed_password = password_hash($password, PASSWORD_DEFAULT);

```

## Problème rencontré

Durant la conception du site, j'ai rencontré des situations problématiques auxquels j'ai pu trouver des solutions grâce aux sites web suivants :

- Warning concernant PDOStatement::execute() :

<https://stackoverflow.com/questions/2713566/warning-pdostatementexecute-sqlstatehy093-invalid-parameter-number-num>

- Gestion d'erreurs et des exceptions

<https://fr.smartworldclub.net/11697000-handling-errors-and-exceptions-with-asp-net-mvc>

- Message du **var\_dump()** concernant un array

<https://stackoverflow.com/questions/53663705/php-notice-array-to-string-conversion-in>

## Extrait d'un site anglophone

Le site suivant parle des failles upload et comment s'en prévenir :

<https://www.funinformatique.com/en/upload-flaw-how-to-use-and-protect-it-part-1/>

### Extrait du site en anglais

How does the Upload flaw work?

The goal of this flaw is to upload a file with an unauthorized extension. (For example a php code) so as to have access to the target server. If the upload form of your site is not secure, then a hacker could easily have fun uploading a malicious PHP file (web shell for example) which would allow him to take total control of your web application, and your server.

## 1) Bypass MIME verification

The MIME type of a file is the type of content that a file is made up of. For example, the MIME type of gif images is "image / gif" and the MIME type of jpg images is "image / jpeg"

Some Upload scripts only check if the MIME corresponds to the authorized file types, on the other hand a mime check is not sufficient because a hacker can bypass this check

A malicious user can bypass this check. It can upload a PHP file by tricking the server into believing the file is a JPEG image.

Traduction en français :

Comment fonctionne la faille Upload ?

Le but de cette faille est de télécharger un fichier avec une extension non autorisée. (Par exemple un code php) afin d'avoir accès au serveur cible. Si le formulaire d'upload de votre site n'est pas sécurisé, alors un pirate pourrait facilement s'amuser à uploader un fichier PHP malveillant (web shell par exemple) lequel lui permettrait de prendre le contrôle total de votre application web, et de votre serveur.

## 1) Contourner la vérification MIME

Le type MIME d'un fichier est le type de contenu dont un fichier est fait. Par exemple, le type MIME des images gif est "image/gif" et le type MIME des images jpg est "image/jpeg"

Certains scripts de téléchargement ne vérifient que si le MIME correspond aux types de fichiers autorisés, par contre une vérification mime n'est pas suffisant car un pirate peut contourner cette vérification

. Un utilisateur malintentionné peut contourner cette vérification. Il peut télécharger un fichier PHP en incitant le serveur à croire que le fichier est une image JPEG.

# ANNEXES

## Cahier des charges

Cahier des charges pour son le registre des cancers de Limoges

ACTEURS DU PROJET

Clients :

- Tania D'ALMEIDA Responsable scientifique du registre général des cancers de la Haute-Vienne
- Elfried SALANON

Développeur :

- Sonia MOUHOUBI

1 - Présentation générale du projet

Création d'un site vitrine pour le registre des cancers de la ville de Limoges.

Une partie administration permettra l'ajout d'articles par l'administrateur.

Le projet permettra de mettre en avant ce qu'est un registre des cancers et de sensibiliser la

population de Limoges ainsi que tous visiteurs sur le site et les professionnels sur les

cancers et les données liées à la maladie.

2 - Présentation de l'Entreprise

Centre Hospitalier et Universitaire de Limoges (CHU Limoges)

3 - Dans quel domaine êtes-vous ?

Dans le domaine de la santé.

4 - Expliquer quel va être le rôle du site internet dans la stratégie de l'Entreprise ?

Informar la population de Limoges et les spécialistes sur les cancers.

5 - Indiquer une date butoir pour la fin du projet :

Le projet se déroule du 30 mai au 31 juin 2022.

6 - Avez-vous un site web ?

Il y a un site au nom de : [www.registre-cancer-limousin.fr](http://www.registre-cancer-limousin.fr) qui nous redirige vers [https://www.contacter-medecin-de-garde.org/ville/limoges\\_87000/](https://www.contacter-medecin-de-garde.org/ville/limoges_87000/)

La cliente n'a plus accès à son nom de domaine. Elle devra se procurer un nouveau nom de

domaine ainsi qu'un l'hébergement.

Du contenu ainsi que des fichiers à télécharger seront fournis par la cliente.

En bonus, un système de newsletter pourra être mis en place.

7 - Qui s'occupe de l'hébergement de ce site ? de sa maintenance ? Êtes-vous actuellement suivi par un prestataire ?

La cliente ou Elfried Salanon.

8 - Concernant les aspects ergonomie et graphisme du site, quelles seraient les couleurs principales du site ? Fournir deux ou trois couleurs à l'image de

votre entreprise.

Couleur au ton clair. Privilégier le bleu, vert et jaune qui rappellent le logo et les couleurs liés à la santé.

Le site doit être épuré, professionnel, pas trop d'images, qui mène directement à l'information.

9 - Avez-vous un logo ?

Oui, un logo a été fourni.

10 - Arborescence du site

- Navigation :

- Logo
- Barre de navigation
- Barre de recherche
- Page d'accueil
- Page à propos
- Page données des cancers
- Page actualité
- Page activités de recherche
- Page administration
- Page contact

Footer :

- Coordonnées du CHU
- Mentions légales
- RGPD

11 - Contenus présents sur le site. Que devez-vous présenter sur votre site en termes

de contenus: documents téléchargeables

(plaquettes, pdf..), vidéos, images, diaporama, .. ?

Contenus sur le registre des cancers et les cancers.

Logos des organismes partenaires. Une demande auprès de ces organismes doit être faite

afin de voir les procédures légales à suivre.

Document au format pdf ou publisher.

Ajout de liens internes et externes pour le référencement (SEO).