# DOSSIER PROJET

# Développeur Web et Web Mobile

Lanza Eric

Session Septembre – Mai 2020

|  |
| --- |
| **SOMMAIRE** |

1. Compétences du référentiel couvertes par le projet
2. Cahier des charges
   1. Présentation du projet
   2. Spécifications fonctionnelles
      * Fonctionnement de l’application
      * Users stories
      * Arborescence du site web
      * Charte graphique
      * Maquettage
   3. Spécifications techniques

* Editeur de code
* Versionning et stockage
* Choix de l’architecture logicielle
* Langages web

1. Réalisation du projet
   1. Mise en œuvre du contrôleur frontal
      * Routage
      * Analyse des paramètres transmis
      * Gestion des erreurs
   2. Développement front end
      * Réalisation et développement d’une interface web
        + Structure type d’une page html
        + Référencement naturel ou SEO
        + Personnalisation du thème de Bootstrap avec SASS
        + Affichage d’image miniature avec JavaScript
   3. Développement back end
      * Création de la base de données
      * Développer les composants d’accès aux données

|  |
| --- |
| 1. **COMPETENCES DU RERERENTIEL COUVERTES PAR LE PROJET** |

**Activités types 1**

***Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité***

Maquetter une application

* La maquette prend en compte les spécificités fonctionnelles décrites dans les cas d'utilisation ou les scénarios utilisateur
* La maquette est conforme à l'expérience utilisateur et à l'équipement ciblé
* La maquette respecte les principes de sécurisation d’une interface utilisateur

Réaliser une interface utilisateur web statique et adaptable

* L'interface est conforme à la maquette de l'application
* Les bonnes pratiques de structuration sont respectées y compris pour le web mobile
* Les pages web s’adaptent à la taille de l’écran
* Le site respecte les règles de référencement naturel
* La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité

Développer une interface utilisateur web dynamique

* Les pages web sont conformes à l’expérience utilisateur y compris pour l’expérience mobile
* L'architecture de l'application répond aux bonnes pratiques de développement et de sécurisation d'application web
* L’application web est optimisée pour les équipements mobiles
* La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité
* La veille sur les vulnérabilités connues permet d’identifier des failles potentielles

**Activités types 2**

***Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité***

Créer une base de données

* La base de données est conforme au schéma physique
* Les scripts de création de bases de données et d’insertion des données de test s’exécutent sans erreurs
* Les besoins de sécurité du SGBD sont exprimés selon l’état de l’art et les exigences de sécurité identifiées
* La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité

Développer les composants d’accès aux données

* Les traitements relatifs aux manipulations des données répondent aux fonctionnalités décrites dans le dossier de conception technique
* Les composants d’accès à la base de données suivent les règles de sécurisation reconnues
* La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité
* La veille sur les vulnérabilités connues permet d’identifier des failles potentielles

Développer la partie back-end d’une application web ou web mobile

* Les bonnes pratiques de développement sont respectées
* Les composants serveur contribuent à la sécurité de l’application
* Le code source des composants est documenté ou auto-documenté
* La démarche de recherche permet de résoudre un problème technique ou de mettre en œuvre une nouvelle fonctionnalité
* La veille sur les vulnérabilités connues permet d’identifier des failles potentielles

|  |
| --- |
| CAHIER DES CHARGES |

* 1. **Présentation du projet**

J’ai choisi de vous présenter un projet personnel, ayant l’avantage d’aborder toutes les étapes de développement d’un site web et de remplir les critères d’évaluation de mes compétences professionnelles.

Mon projet est un guide sur la reconnaissance des champignons.

A travers ce site je partage une de mes passions et j’offre à ceux qui le désirent de participer à l’élaboration d’une base de données collaboratives, regroupant photos et description d’espèces.

* 1. **Spécifications fonctionnelles**

**Fonctionnement de l’application**

Comme le site propose une documentation illustrée, et collaborative je dois pouvoir stocker ces informations dans une base de données.

Pour commencer j’ai définit trois types d’utilisateurs :

* Visiteur
* Membre
* Administrateur

**Fonctionnalités en fonction des permissions**

Les 3 types d’utilisateurs ont accès à l’espace public

* Une page d’accueil
* A la liste des espèces donnant accès au détail de chacune des
  + Fiches descriptives

**Les membres** visualisent les informations de profil, la liste des fiches qu’ils ont créées, accèdent à travers des liens aux pages :

* + Ajouter une fiche descriptive
  + Modifier une fiche descriptive
  + Modifier ou supprimer son profil

*Restrictions*

Lors de la création d’une fiche descriptive par un membre, celui-ci peut la consulter mais ne sera accessible dans l’espace public qu’après validation par l’administrateur.

**L’administrateur** peut visualiser l’ensemble des comptes utilisateurs, la liste des fiches présentes sur le site. En outre il a la possibilité de masquer une fiche dans l’espace public, supprimer un compte ou une fiche descriptive.

**Users stories**

Cette étape me permet de décrit les fonctionnalités à développer.

|  |  |  |
| --- | --- | --- |
| **En tant que…** | **Je souhaite…** | **Afin de…** |
| Visiteur | Consulter la liste des espèces | Accéder à leurs caractéristiques. |
| Visiteur | M’inscrire | Créer mon profil et accéder à d’autres fonctionnalités. |
| Membre | Me connecter | Gérer mon espace |
| Membre | Me déconnecter | Quitter mon espace |
| Membre | Voir mon profil | Voir mes informations  personnelles |
| Membre | Modifier mon profil | Changer/corriger mes  Informations personnelles |
| Membre | Créer une fiche descriptive | Collaborer au développement du site en créant une documentation illustrée. |
| Membre | Modifier une fiche descriptive | Mettre à jour le contenu d’une fiche détaillée |
| Membre | Supprimer une fiche |  |
| Membre | Supprimer mon profil | Quitter la communauté |
| Administrateur | Consulter la listes des utilisateurs | Voir les inscriptions |
| Administrateur | Supprimer un utilisateur | Modérer les utilisateurs |
| Administrateur | Consulter la liste des fiches | M’assurer du contenu |
| Administrateur | Supprimer une fiche | Modérer le contenu |
| Administrateur | Autoriser la publication d’une fiche descriptive | De la rendre accessible au visiteur |

**Arborescence du site web**

**Charte graphique**

C’est l’identité visuel du site, j’ai choisi des couleurs en accord avec la nature, une courte vidéo en page d’accueil permet de retenir l’attention du visiteur.

Pour personnaliser mon site, j’ai choisi d’intégrer des images d’arrière-plan légèrement floutées pour agrémenter les pages.

**Code couleurs**

**Liens de navigation**

rgb(255, 255, 255): liens

 rgb(224, 139, 94) : Liens actif

 rgb(214, 206, 195) : Liens au survol

**Autres**

 rgb(85, 84, 84) : couleur de texte

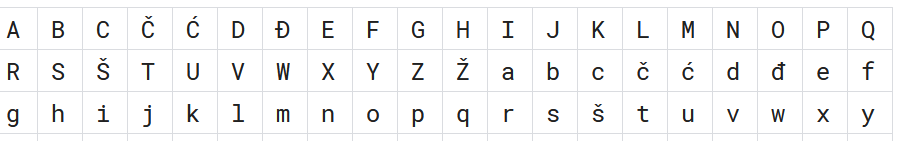
 rgb(63, 80, 62) : bouton de pagination, titre de chaque vignettes.

 rgba(39, 54, 26, 0.8) : formulaires

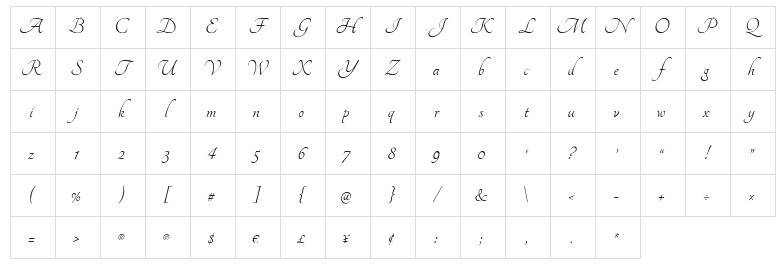
**Police de caractères**

J’ai choisi une combinaison de 3 polices de caractères.

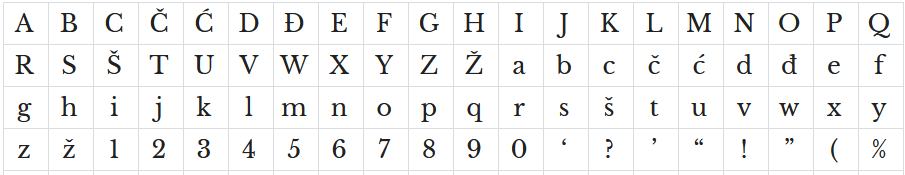
**Roboto** que j’utilise pour la barre de navigation et le pied de page.



**Tangerine** employé pour le titre principal et secondaire de la page d’accueil.



**Libre Baskerville** appliqué sur le contenu de chaque page.

****

**Maquettage**

|  |  |
| --- | --- |
|  | J’ai utilisé **Adobe XD** pour réaliser les maquettes dédiées aux support mobile et Desktop.  J’ai tout d’abord découpé visuellement chaque page en plusieurs zones, cette organisation appelée le **zoning** me simplifiera la tache lors du développement.  En effet cela me permet d’identifier les différents blocs qui constituent la page.  J’ai ensuite réalisé le **Mockup**, rendu visuel étant le plus proche possible du produit final, après m’être inspiré en parcourant quelques sites et divers ouvrages. |

**Aperçu du Zoning de la page d’accueil**

**Mockup de la page d’accueil**

****

* 1. **Spécifications techniques**

**Editeur de code**

|  |  |
| --- | --- |
|  | Pour développer mon projet j’utilise Vscode un éditeur open source, il prend en charge presque tous les principaux langages de programmation et fourni des extensions téléchargeables gratuitement. |

**Gestion de version**

|  |  |
| --- | --- |
|  | Je me sers de **Git**, un logiciel gestion de versions.  Il me permet de réaliser des sauvegardes régulières de mon projet en local et sur un serveur d’hébergement comme **GitHub.**  C’est un outil incontournable pour tout développeur, il offre en outre, la possibilité de partager son code avec d'autres développeurs. |
|  |

**Choix de l’architecture logicielle**

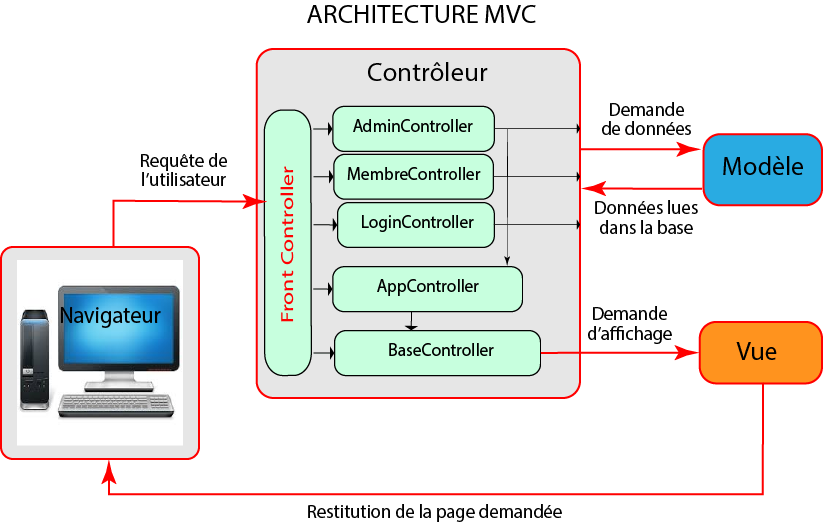
Pour la réalisation de mon projet j’ai choisi une architecture MVC se décomposant en trois parties : Modèle**,** Vue, Contrôleur.

Ainsi je sépare les responsabilités, le code est mieux organisé, plus lisible et donc plus facile à maintenir.

Le **Modèle** encapsule la logique métier ainsi que l’accès aux données.

La **Vue** permet l'interaction avec l’utilisateur, présentation, saisie et validation des données.

Le **Contrôleur** fait le lien entre l’utilisateur et le reste de l’application.



**Langages web**

Langage pour le développement front-end

|  |  |
| --- | --- |
|  | Dans la première étape de développement, je structure mes documents, mets en forme le contenu de mes pages tel que je l’ai défini dans la phase de maquettage.  Les zonings élabore l’architecture de mes pages, pour cela j’utilise **HTML** qui est un langage de balise. |

|  |  |
| --- | --- |
| **CSS – SASS - BOOTSTRAP** | |
|  | Le langage **CSS** change l’apparence des éléments HTLM, donne du style à mes pages, c’est grâce à ce langage que je vais pouvoir appliquer l’apparence esthétique réalisé avec mes mockups. |
|  |  |
|  | Ce préprocesseur CSS, je l’emploi conjointement avec la méthodologie BEM (Bloc Elément Modifier), il rend mon code plus lisible mieux **organisé et surtout structuré.** |
|  |  |
|  | Avec ce Framework CSS je développe et je conçois des pages rapidement, et j’écris moins de code.  L’utilisation des grilles m’a permis de découper et d’organiser mes pages web en fonction du support numérique.  Je me suis servi de plusieurs composants pour illustrer mes pages, comme Bootstrap exploite jQuery, j’obtiens un rendu dynamique pour certains composants. |
|  |  |
|  | JavaScript a été utile pour afficher un rendu des images lors de la sélection à partir d’un formulaire. |

|  |  |
| --- | --- |
|  | J’emploi Twig, moteur de Template pour personnaliser, assembler, à l’aide de blocs, mes pages web. Le résultat offre un code plus clair et lisible, en évitant les répétitions, par ailleurs, le traitement des données est séparé. |

Langage pour le développement back end

|  |  |
| --- | --- |
|  | PHP est un langage coté serveur que je peux utiliser conjointement avec ma base de donnée relationnelle de type SQL. |

|  |
| --- |
| 1. **REALISATION DU PROJET** |

* 1. **Mise en œuvre du contrôleur frontal**

J’ai mis en œuvre un Contrôleur Frontal à partir du fichier « index.php » qui constitue le point d'entrée unique du site.

**Routage**

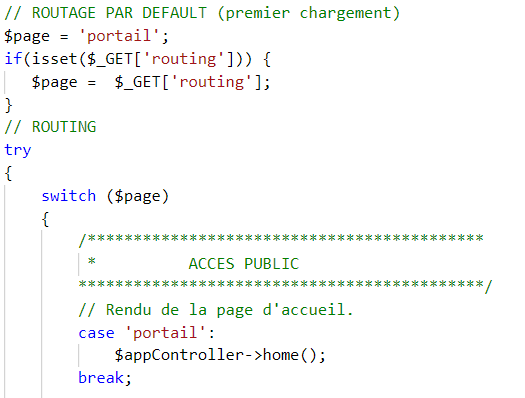
Son rôle permet de centraliser la gestion des requêtes entrantes.

Les paramètres de la route sont transmis dans L’URL sous forme de clé valeur.

Voici un exemple d’URL

https://.../index.php?routing=portail

Un bloc « try » entoure la commande « switch » dans lequel je parcours et vérifie l’existence de la route passer dans l’URL.

****

**Analyse des paramètres transmis**

Dans cet exemple suivant, je souhaite afficher la page listant toute les espèces de champignons, sous forme de vignettes en apportant quelques informations (comestibilité, nom commun).

Je vérifie les paramètres fournis, sélectionne et appelle **l’action** à réaliser.

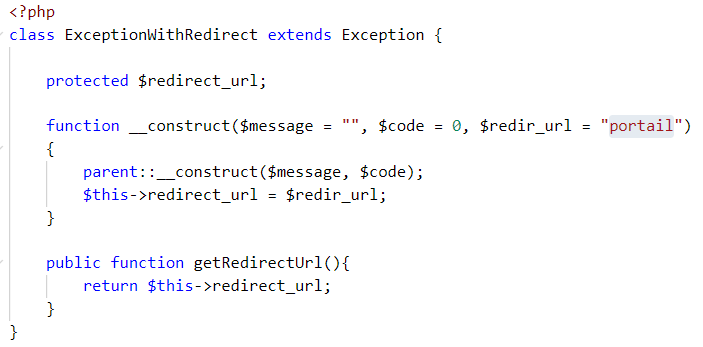


**Gestion des erreurs**

Dans l’exemple vu précédemment, dans le cas où la requête est incohérente je signale l’erreur à l’utilisateur en **lançant une exception avec** ("throw").

J’ai édité une classe nommée « ExceptWhithRedirect » qui hérite de la classe « Exception » fourni par PHP, ceci me permet d’ajouter un lien de redirection dans la page d’erreur.

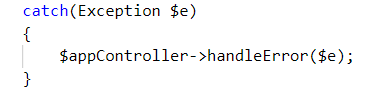
J’instancie la classe avec « new ».



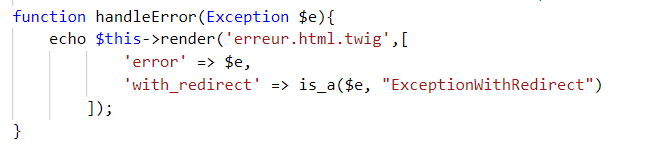
La méthode constructeur est automatiquement appelée au moment de l’instanciation de la classe « ExceptWhithRedirect ».

Par ailleurs j’ai besoin du constructeur parent (Exception) pour attraper l’erreur il est donc nécessaire de faire appel à « parent::\_\_construct() ».

**J’attrape l’erreur avec** ("catch").

****

Je fais le rendu de la page d’erreur grâce à l’objet Twig.



L’utilisateur consulte la page d’erreur et active la redirection vers une autre page du site.



* 1. **Développement front-end**

Mon projet est bâti sur une architecture MCV, pour constituer mes pages j’utilise twig qui sépare le code PHP du code HTML.

**Réalisation et développement d’une interface web**

**Structure type d’une page html**

Un document HTML se compose de plusieurs balises :

* **<!DOCTYPE html>** : définit le type de document
* **<html ></html>** : délimite le début et la fin de la page
* **<head></head>** : définit le codage des caractères, contrôle la mise en page sur les navigateurs mobiles, fourni des informations sur l’auteur du site, indique les liens vers les feuilles de style, informe sur le contenu de la page.
* **<body></body>** : contient la partie visible de la page, aussi appelée « corps de page ». C'est ce qui sera affiché à l’écran.

J’édite dans un fichier Twig la structure commune a toutes les pages du site.

*default.html.twig*



**Référencement naturel ou SEO**

Je cherche à améliorer la visibilité de mon site sur les pages de résultats des moteurs de recherche et plate-forme de réseaux sociaux**.**

**J’ai** ajouté dans la balise **< head>** de mon layout parent, 3 blocs (title, meta et opengraph), qui seront renseigné dynamique lors de la génération du rendu par twig.

*Default.html.twig*

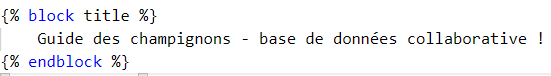


Je dois à présent définir le titre et les métas donnés pour chaque template enfant du site.

Voici un exemple pour la réalisation de la page regroupant la liste des espèces.

**Balise <title>**

La balise <title> est importante. Elle est utilisée par les moteurs de recherche pour indexer évaluer et positionner la page web et les navigateurs pour donner un titre a la page au niveau de l’onglet, comme titre des marque-pages (favoris), etc. Elle doit refléter le contenu de la page et contenir des mots clés pertinents.



**Balise meta description**

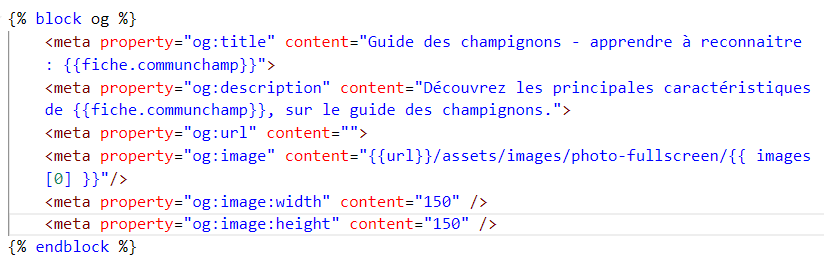
Elle renseigne les moteurs de recherche sur le sujet traité et sera ajouté par Google dans la page de résultat pour détailler le contenu.



**Balise meta robots**

Elle donne les directives aux moteurs de recherche sur la manière d'explorer, indexer et parcourir les liens sur une page.

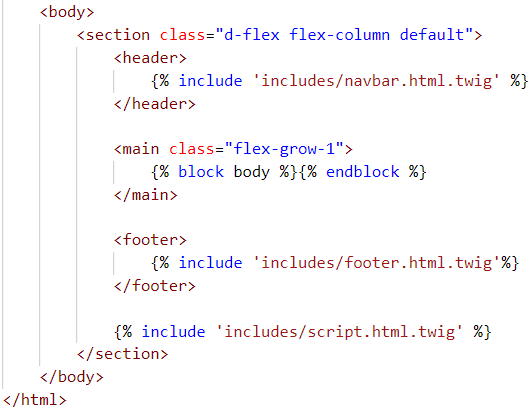
**Balise Open Graph**

Les balises Open Graph permettent d’améliorer la présentation de mes pages web lorsque des liens sont partagées sur les réseaux sociaux. 

**Le corps de page**

L'élément **<body>** représente le contenu principal du document HTML, visible par l’utilisateur.

*default.html.twig*

****

Dans cet exemple j’ai placé dans un conteneur flexible la balise **<header>, <main> et <footer**>, ces balises dites sémantiques définissent le rôle du contenu qu’ils englobent.

Le block body affiche le contenu principal de la page, j’ai ajouté dans la balise <main> une classe « flex-grow-1 » ceci permet de remplir l’espace restant, ainsi le footer se positionnera toujours en bas de page.

**Personnalisation du thème de Bootstrap avec SASS**

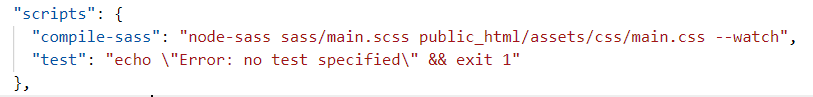
Je souhaite ajouter les couleurs définies dans la charte graphique, modifier les couleurs du composant de pagination ainsi que de la barre de navigation.

Pour réaliser cette opération proprement, je dois intégrer les **fichiers sources** de Bootstrap dans mon projet.

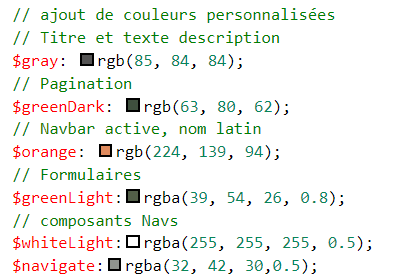
Travaillant sous Windows, j’installe au préalable **Node JS** pour bénéficier du gestionnaire de package **NPM**.

J’installe Bootstrap avec la commande : $ npm install bootstrap

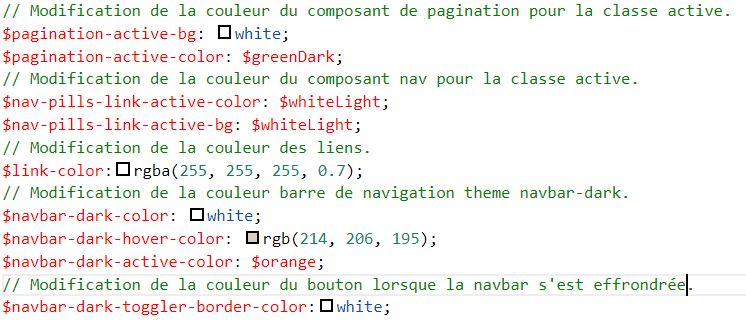
Puis **SASS**, le préprocesseur CSS : npm install -g sass pour compiler mon code CSS et celui de bootstrap et enfin je définir dans le fichier « package.json » un script pour automatiser la compilation.



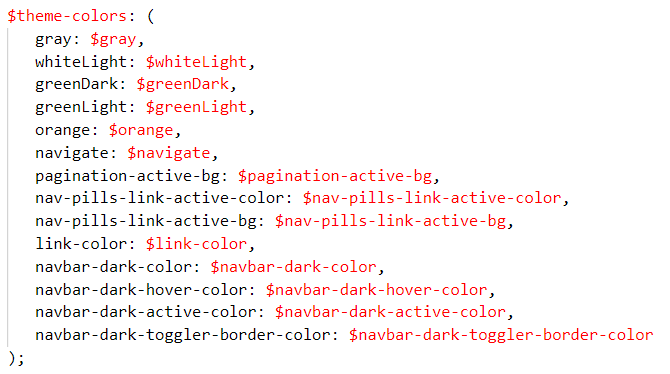
Je crée un fichier que je nomme « change-colors-bootstrap.scss » et j’ajoute les couleurs personnalisées.



J’attribut ces couleurs aux composants cités en amont.



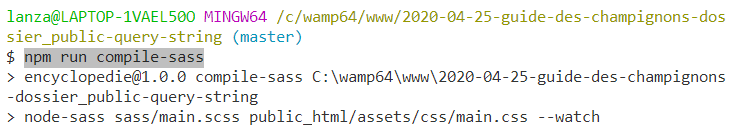
Je dois maintenant l’ajouter au thème par défaut de Bootstrap



Et pour terminer j’importe Bootstrap :



Je lance dans le terminal le compilateur SASS, pour obtenir un fichier CSS combinant mon code CSS et celui de Bootstrap.



**Affichage d’image miniature avec javascript**

* 1. **Développement back end**

**Création de la base de données**

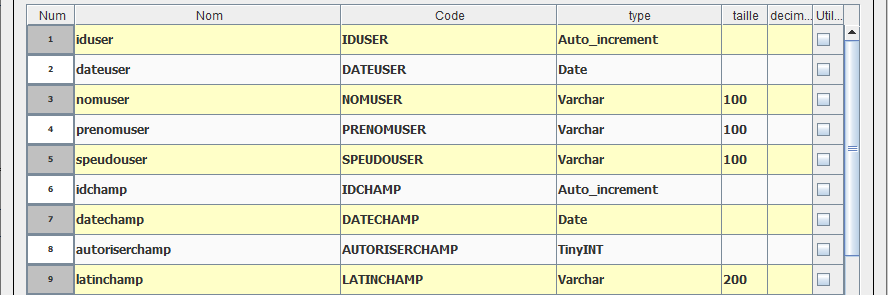
J’ai choisi d’utiliser une base de données de type relationnel pour conserver les informations de profils utilisateurs, la documentation relative aux différentes fiches descriptives.

J’utilise l’utilitaire JMerise pour modéliser ma base de données.

Je recueille et regroupe les informations à traiter en un ensemble cohérent qui devront figurées sur le site.

Je récence trois groupes de données liés les uns aux autres.

Celle des utilisateurs, fiches descriptives et images puis je rédige le dictionnaire de données**.**

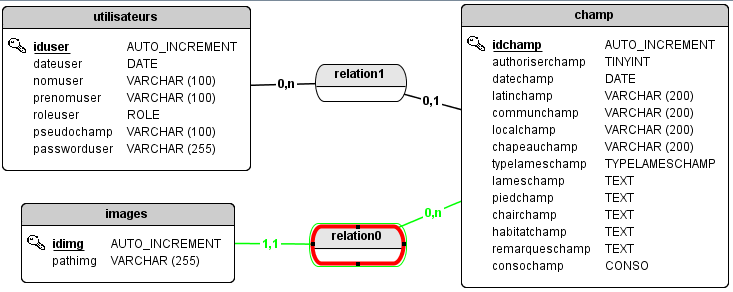
****

**Le modèle conceptuel de données**

J’élabore le Modèle Conceptuel de Donnée (MCD) basé sur deux notions principales : les **entités** et les **associations**.

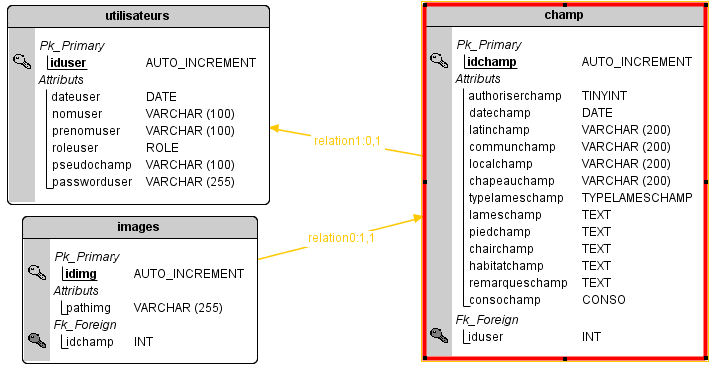
Je définis les relations :

* 1 utilisateur possède de 0 à n fiche mais une fiche appartient à 0 ou 1 utilisateur.
* 1 fiche possède 0 à n images mais une image appartient à 1 et 1 seule fiche.

****

**Le modèle physique des données**

A cette étape, les **entités** deviennent des **tables**, les identifiants deviennent des **clés primaires**.

L’utilitaire génère les tables en fonction des cardinalités décrites ci-dessus. ****

**Le script**

Je récupère le script généré par JMerise et j’ajoute deux contraintes de table pour la suppression d’enregistrement.

* « ON DELETE SET NULL », Lorsque l’utilisateur supprime son profil, la clé étrangère des fiches qu’il aura créée sera définit a « NULL ».
* « On DELETE CASCADE », si l’utilisateur supprime une fiche, les données images liées, seront automatiquement détruites.



**Développement des composants d’accès aux données**

Les requêtes préparées

Stockage mot de passe

Deploiment Serveur php mysql 000webhost.com

**ANNEXES**

**Maquettage**