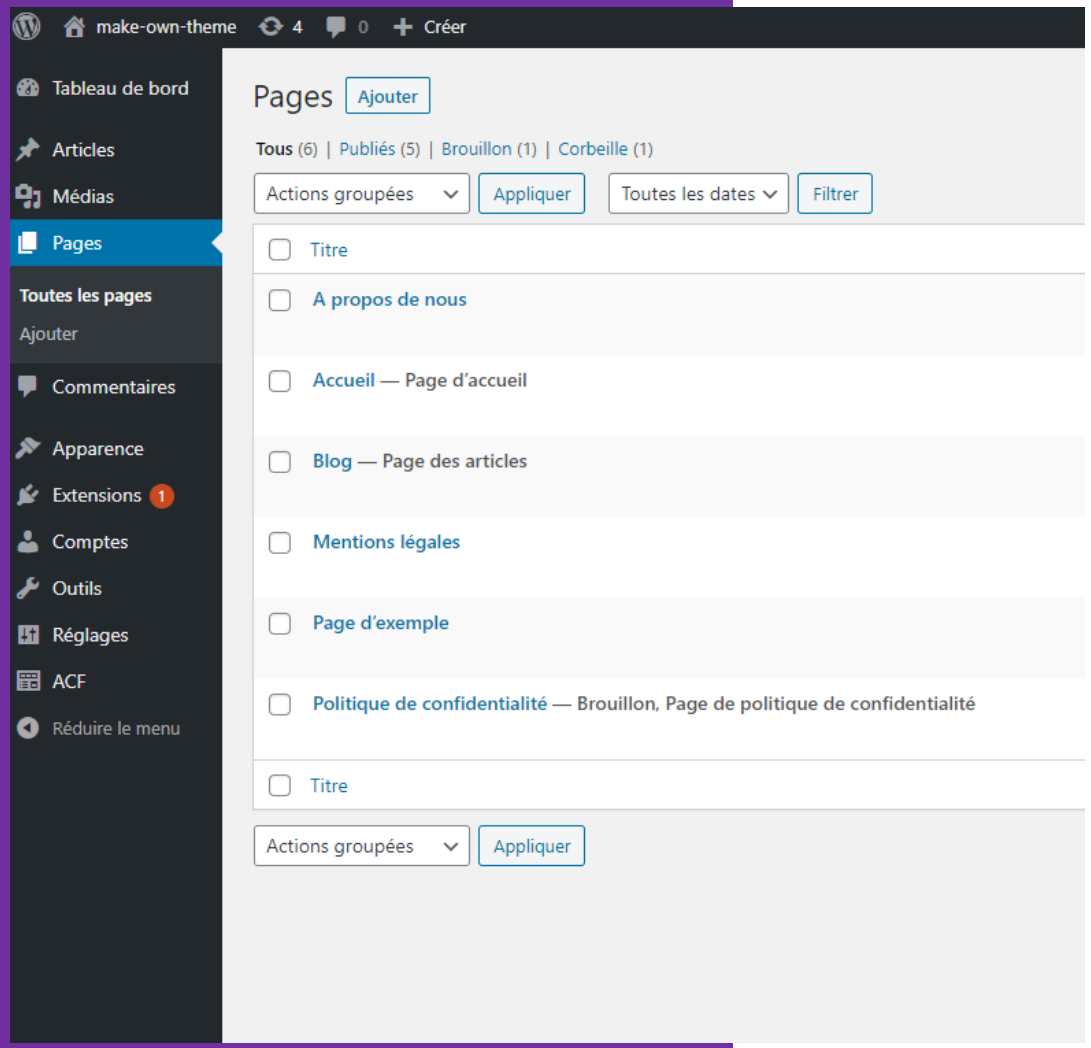
Wordpress

Création de la page des articles (single.php)

Paramétrages avant conception de la page des articles



Avant de prévoir la mise en place de la page qui servira de « template » pour les articles, il faut créer une page qui regroupera tous les articles du blog.

Sur le backoffice, il faut aller sur « Pages » puis « ajouter » pour créer une nouvelle page.

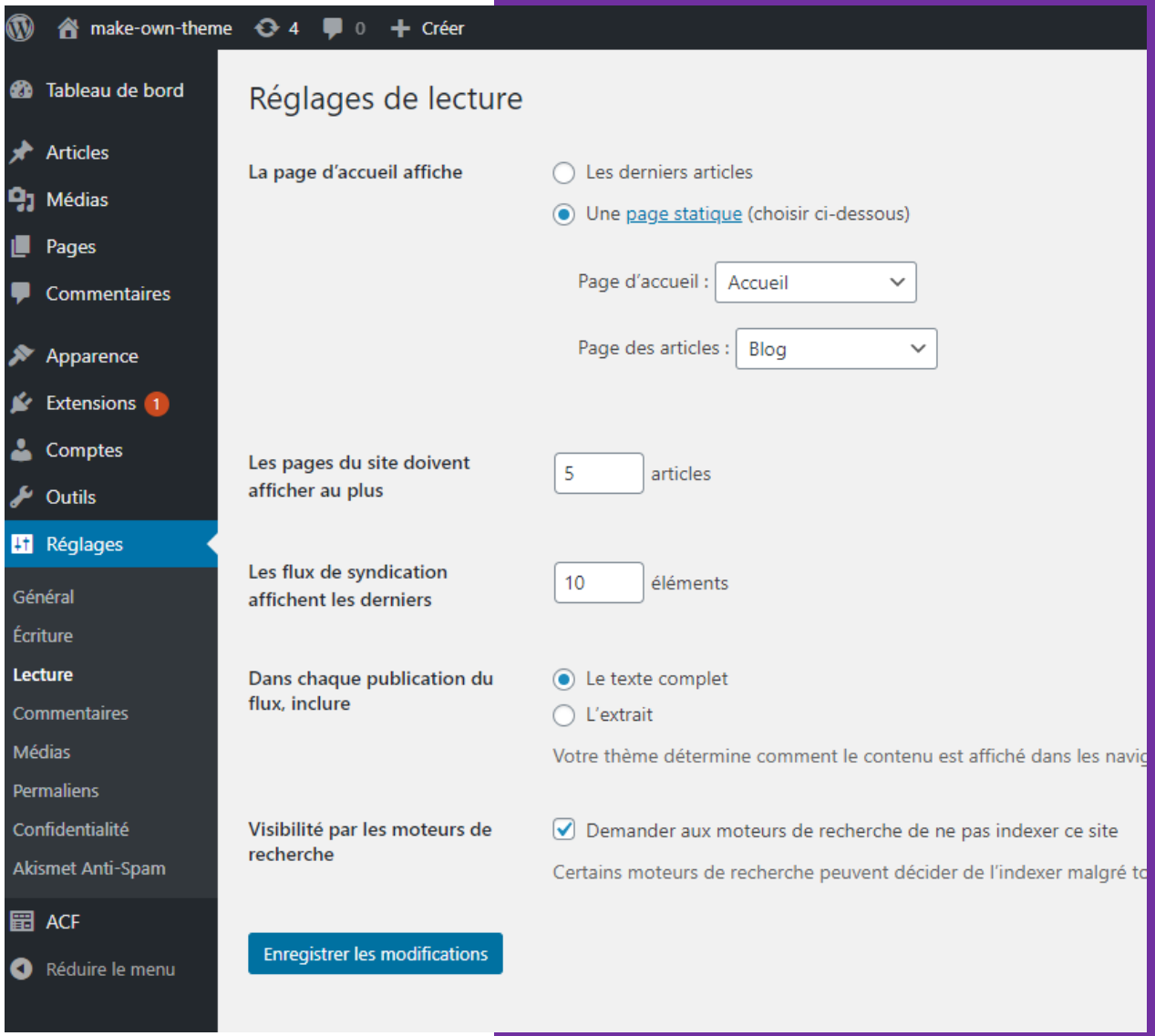
Cette nouvelle page sera nommée « blog ».

Paramétrages avant conception de la page des articles

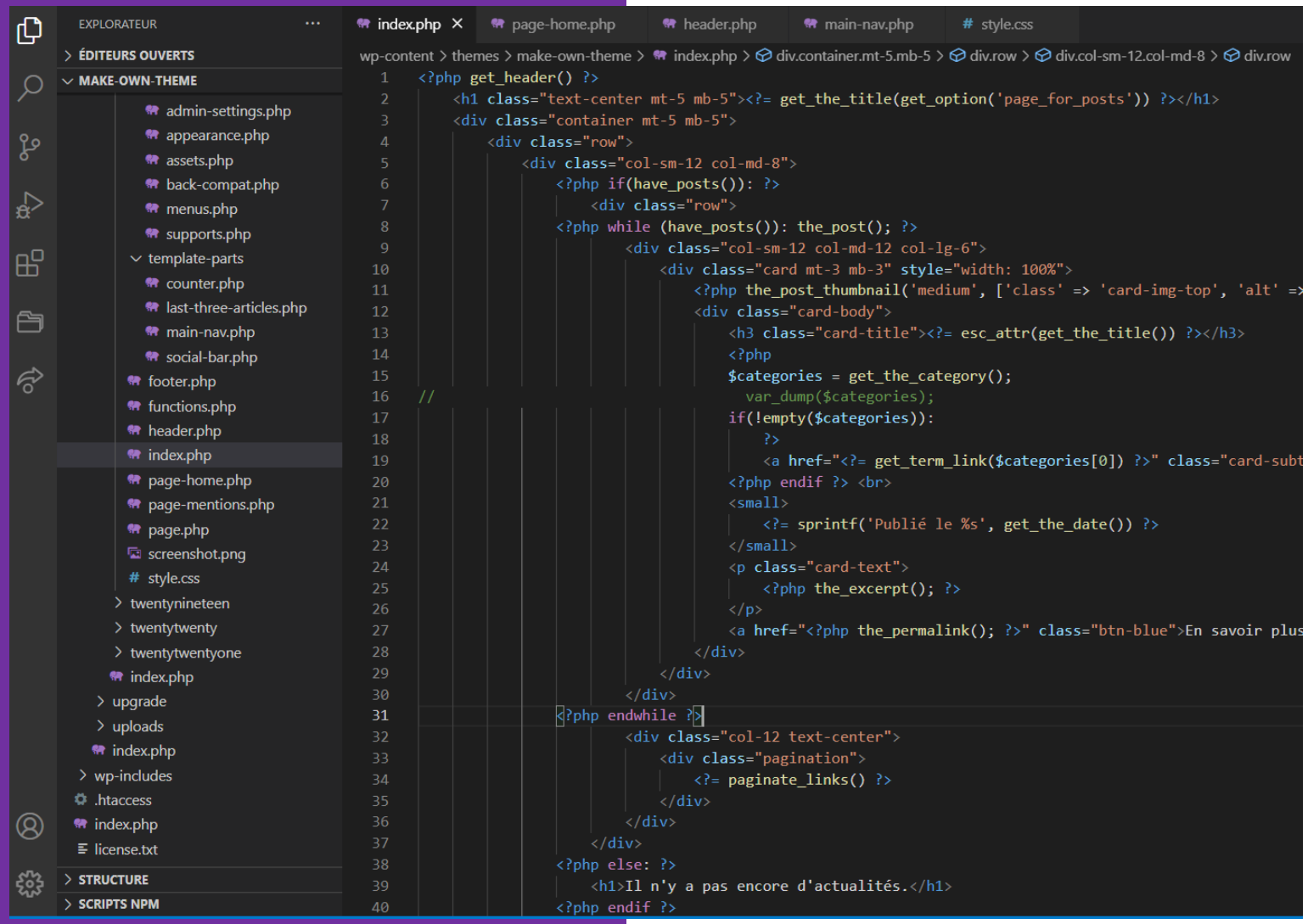
La page « blog » étant créée, il faut se rendre dans « réglages » puis « lecture » pour paramétrer les pages qui seront utilisées pour le site.

En choisissant « page statique », il est possible de choisir la page qui servira d'accueil et la page qui servira pour les articles.

Normalement, « blog » doit apparaître dans les choix possible pour la page des articles. Il suffit de laisser sur « blog » comme choix d'affichage pour la page des articles et d'enregistrer les modifications pour permettre à Wordpress d'aller chercher la bonne page.



Page pour le blog avec “index.php”



Désormais c’est sur la page « index.php » que Wordpress ira chercher la page pour afficher les articles du blog.

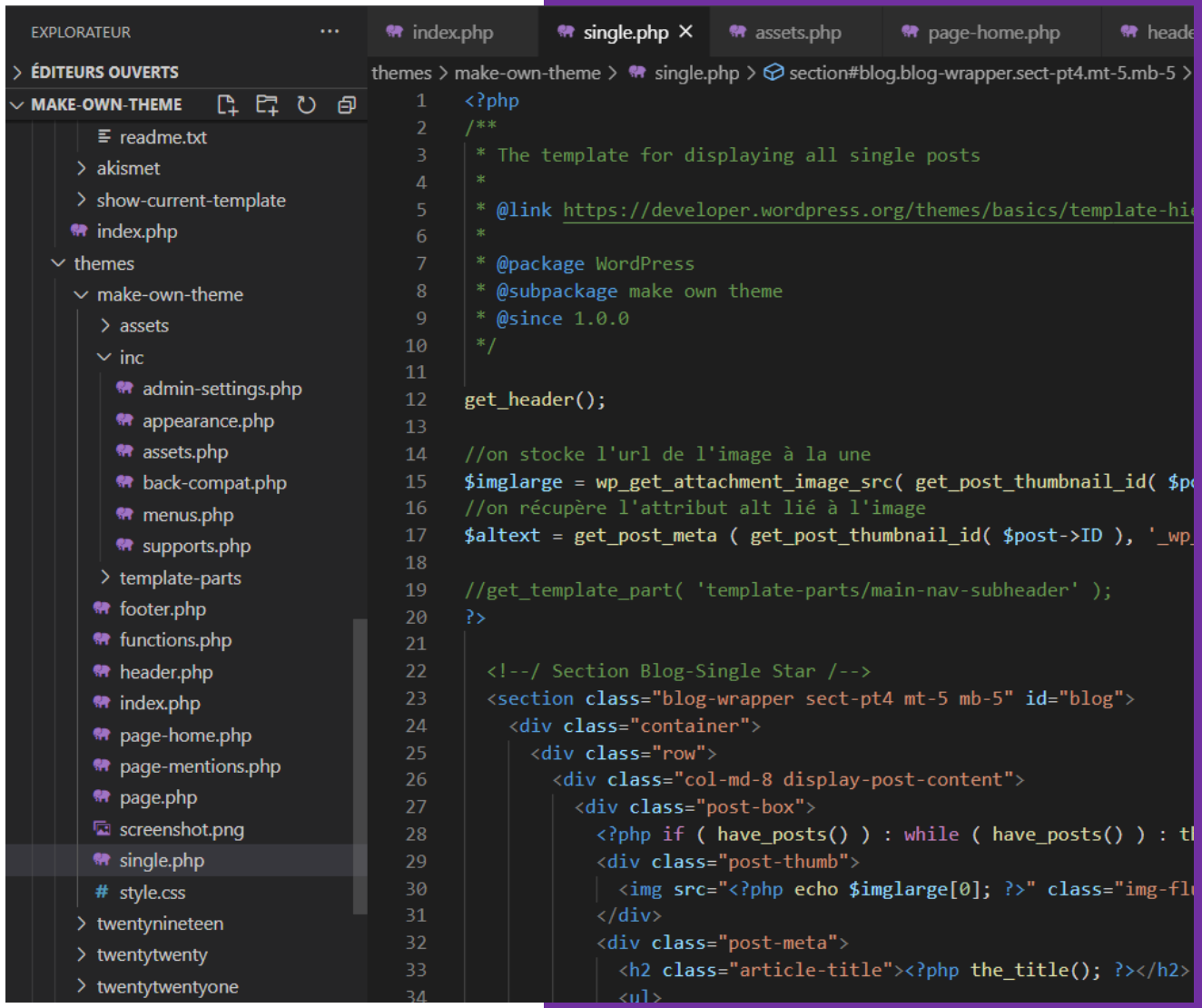
C’est donc dans ce fichier qu’il faudra gérer l’affichage des articles grâce à un système de boucle.

Pour afficher un article au complet, il faut mettre en place le fichier « single.php ».

Page d’un article complet avec “single.php”

Le fichier « single.php » est à mettre en place au même niveau qu’ « index.php ».

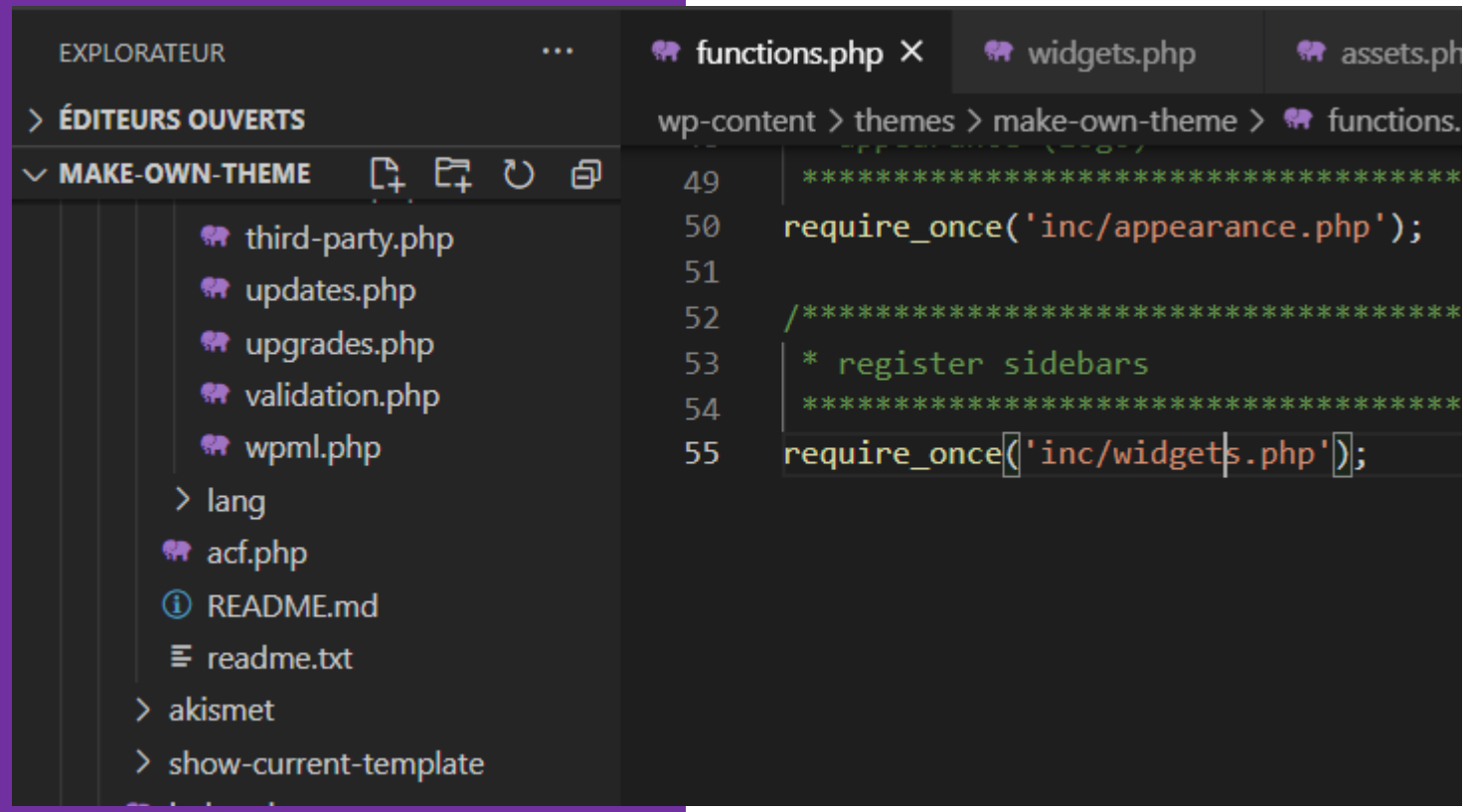
C’est ce fichier qui affichera l’article au complet, un formulaire pour poster un commentaire mais aussi les commentaires laissés par les utilisateurs.



Wordpress

Déclaration et enregistrement d'une zone de widget pour la Sidebar

Mise en place de la sidebar dans le theme avec “functions.php”



Pour tout les ajouts de support au sein de Wordpress, il est essentiel de passer par le fichier « functions.php » pour les déclarer.

Le fichier « widgets.php » qui servira pour déclarer la ou les sidebars du site.

La fonction « require_once » permets d’aller chercher le fichier nécessaire pour faire fonctionner le code.

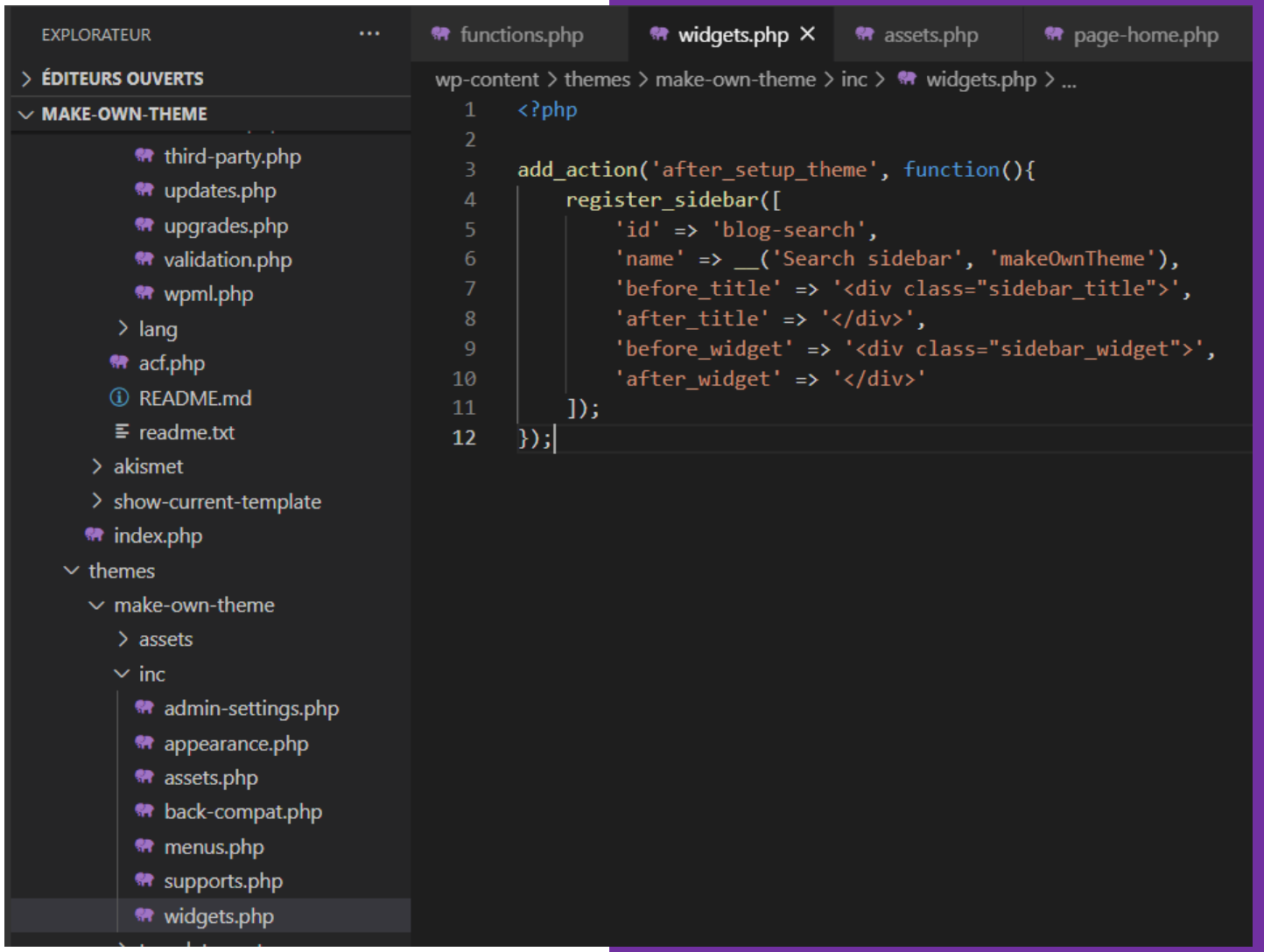
Mise en place de la sidebar dans le theme avec “widgets.php”

C’est dans le fichier « widgets.php » que va être ajouté l’appel au « hook » de Wordpress pour y greffer l’enregistrement de la sidebar.

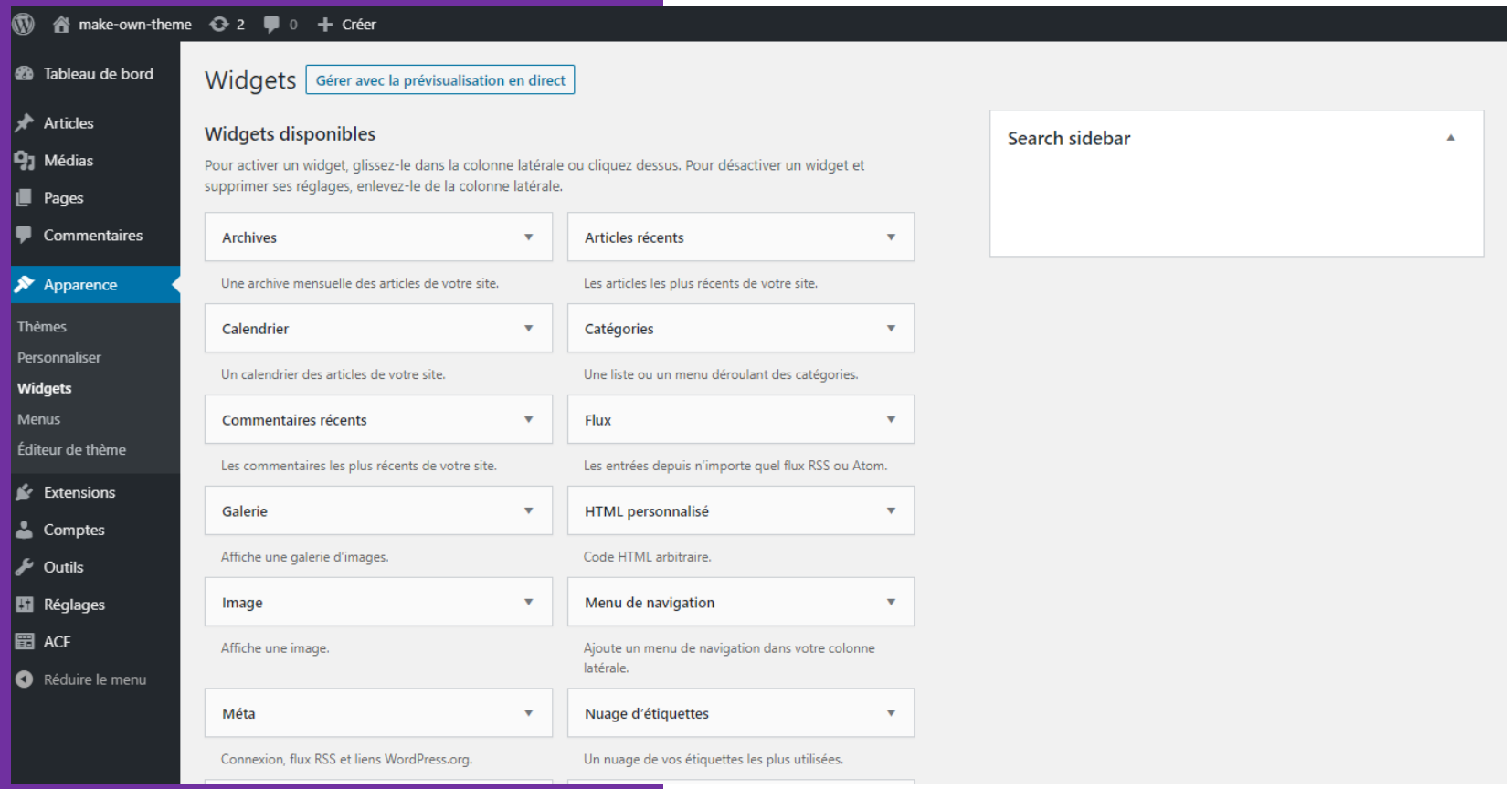
L’ « id » et le « name » s’est pour identifier la nouvelle sidebar.

Les paramètres « before_title » et « after_title » permettent de définir l’élément HTML qui englobera la sidebar.

Les deux derniers paramètres ont la même utilité que les deux précédents mais pour les widgets qui se retrouveront dans la sidebar.



Zone de widget dans le backoffice



Une fois que la sidebar est déclarée, l'onglet « widgets » doit apparaître dans le menu d'apparence.

En sélectionnant celui-ci, vous arriverez sur le menu de la capture.

A droite se trouvent les différents « widgets » disponibles.

A gauche, la sidebar qui vient juste d'être déclarée.

Il ne reste plus qu'a glisser/déposer les widgets nécessaires sur l'onglet de la sidebar.

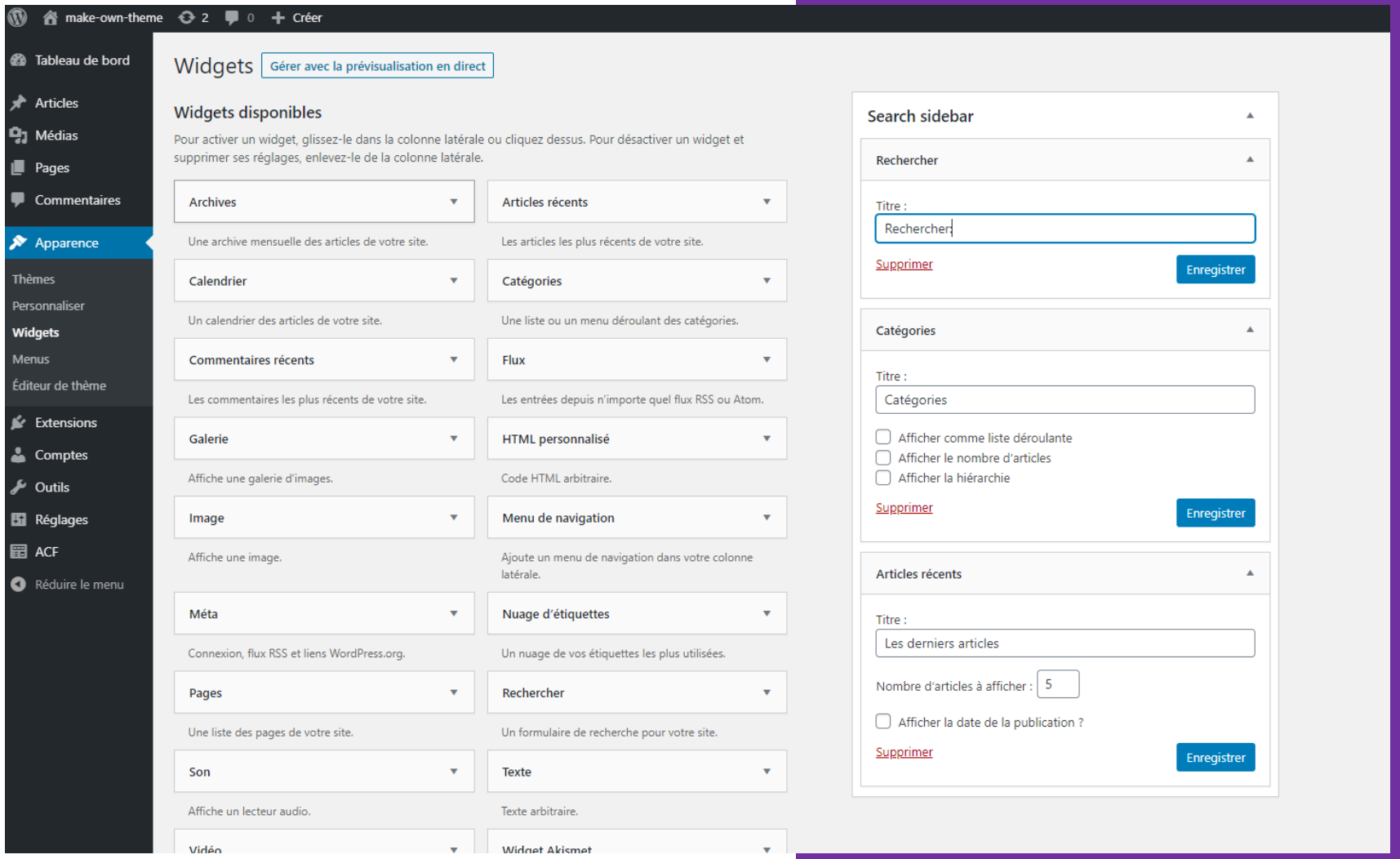
Zone de widget, sidebar

A partir du moment ou les « widgets » sont déposés dans la sidebar, un paramétrage peut être mis en place.

Comme par exemple écrire un titre ou le laisser vide.

Pour « les catégories », sélectionner plusieurs types d'affichages pour l'utilisateur.

Et pour « les articles récents » décider du nombre d'articles à afficher.



Wordpress

Création et inclusion du fichier sidebar.php

Sans “sidebar.php”

```
<div class="col-sm-12 col-md-4 widgets">
    <?php dynamic_sidebar('blog-search') ?>
    <!-- <?php //get_sidebar('blog-search') ?> -->
</div>
```

Sur le code qui n’est pas commenté, la fonction « dynamic_sidebar » ira cherché la sidebar qui est nommée « blog-search ».

Cette fonction a l’avantage d’aller chercher une sidebar de façon dynamique en récupérant celle par défaut qui se trouve dans le thème ou en allant récupérer une en particulier si elle est spécifiée dans ses paramètres.

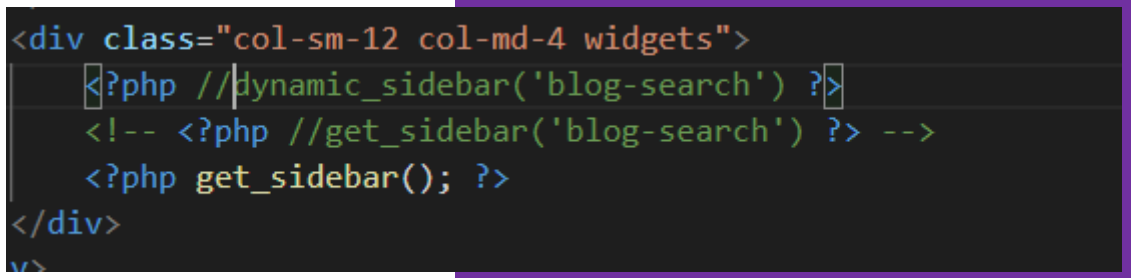
Elle peut être très utile si il faut afficher plusieurs sidebar à différents endroits d’un site.

Avec “sidebar.php”

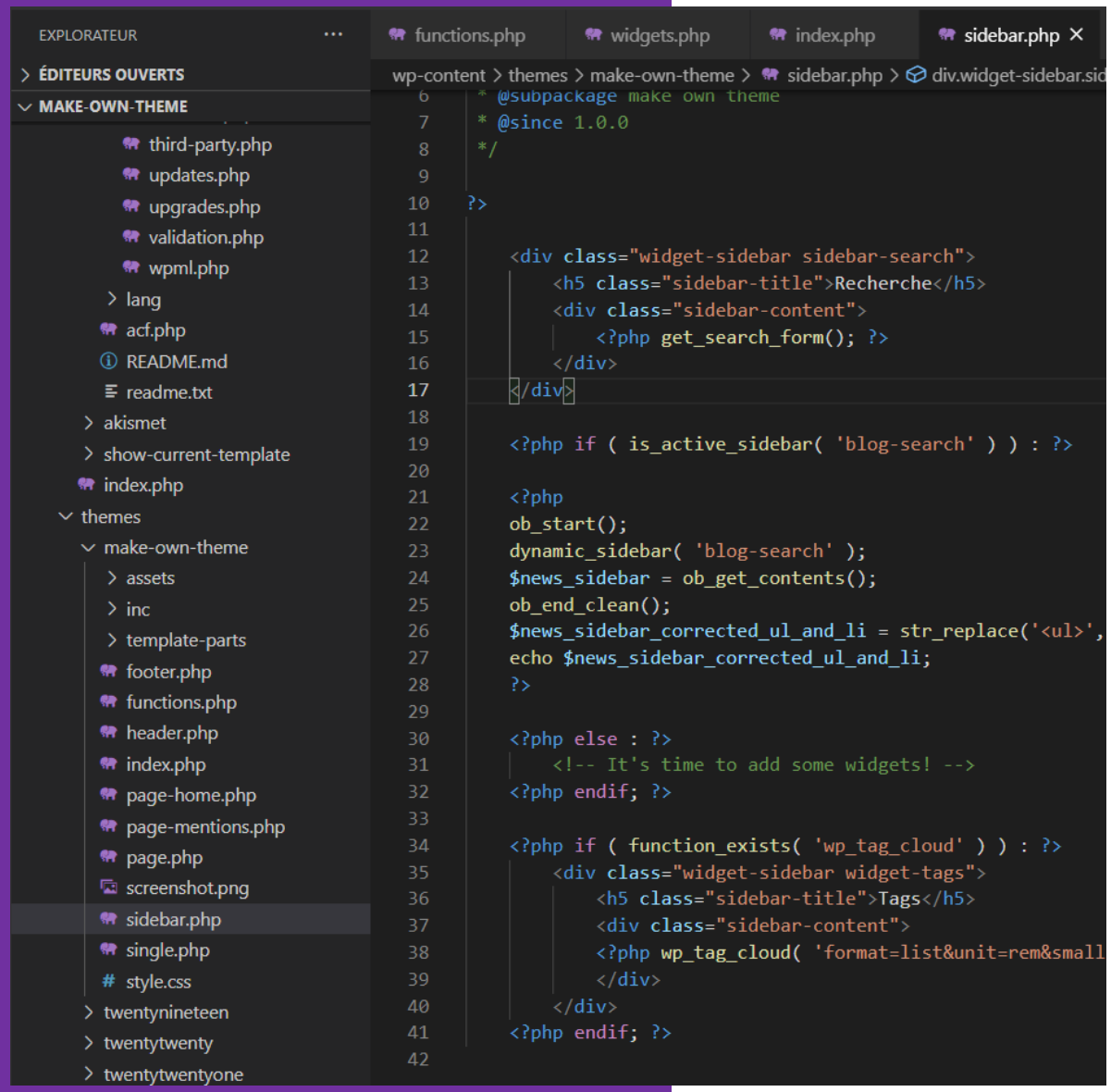
Tout comme la fonction « dynamic_sidebar », la fonction « get_sidebar » permets de récupérer une sidebar lorsqu’elle est spécifiée dans les paramètres.

Si il n’y est mis aucuns paramètres, cette fonction ira chercher le template du thème pour la sidebar par défaut.

Ce template par défaut, c’est le fichier « sidebar.php » qui est utilisé pour gérer l’affichage de la sidebar par défaut.



Avec “sidebar.php”



Sur cette capture, voici un exemple de code qui permets de gérer l’affichage de la sidebar.

Une vérification est faite en amont pour être sur que la sidebar est bien activée dans le back office.

Ensuite l’affichage est géré à la volée pour les « ul » et « li » pour être remplacé par une structure HTML plus claire.

Wordpress

Création et inclusion du fichier searchform.php

searchForm.php

```php
<div class="widget-sidebar sidebar-search">
    <h5 class="sidebar-title">Recherche</h5>
    <div class="sidebar-content">
    <!-- display search form -->
        <?php get_search_form(); ?>
    </div>
</div>
```

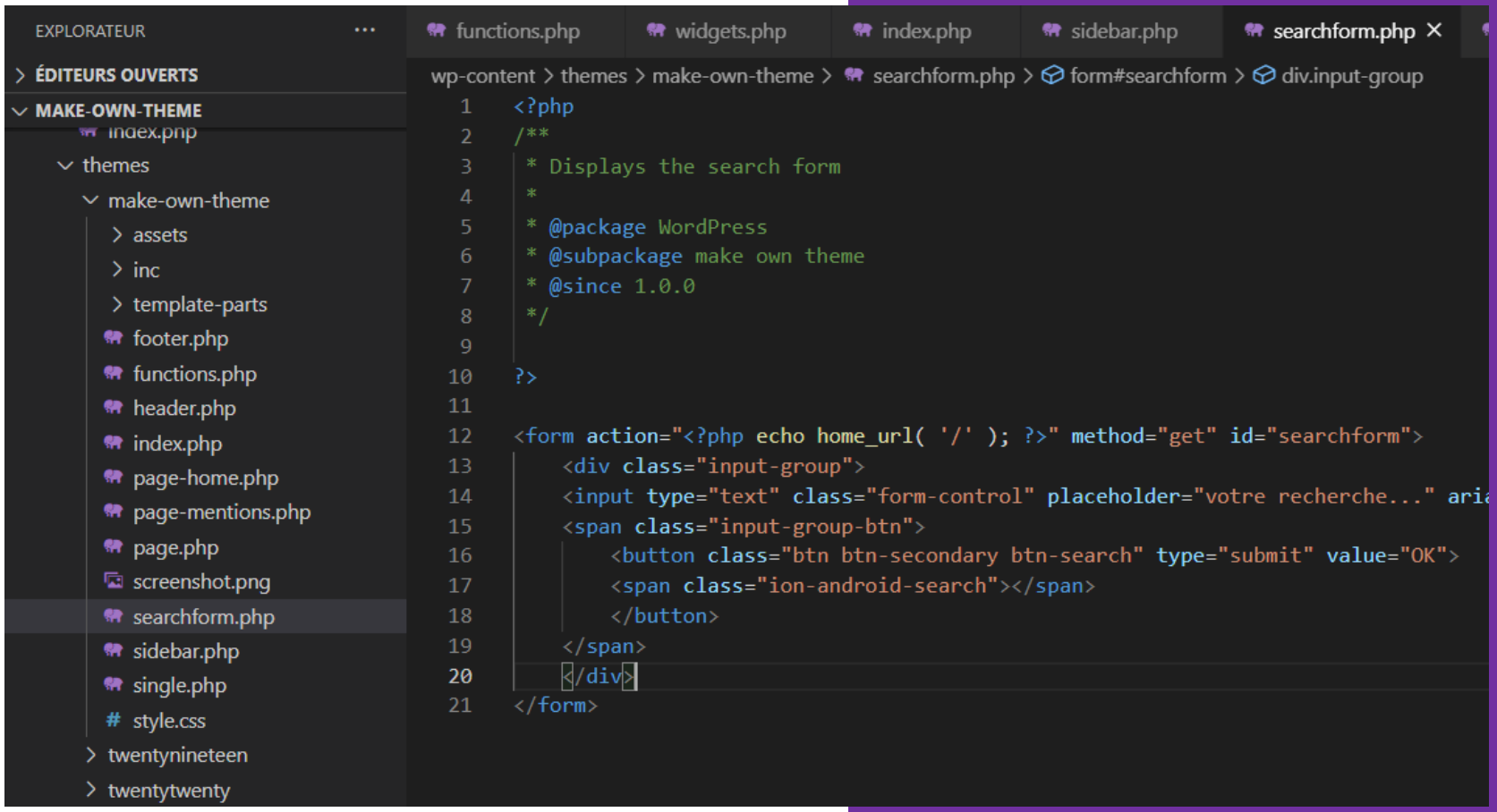
Dans la sidebar ou ailleurs sur le site, il est possible de mettre en place un formulaire de recherche.

C'est la fonction « get_search_form » qui se chargera de récupérer le formulaire à afficher.

Elle ira tout simplement chercher le fichier « searchform.php » pour gérer l'affichage.

searchForm.php

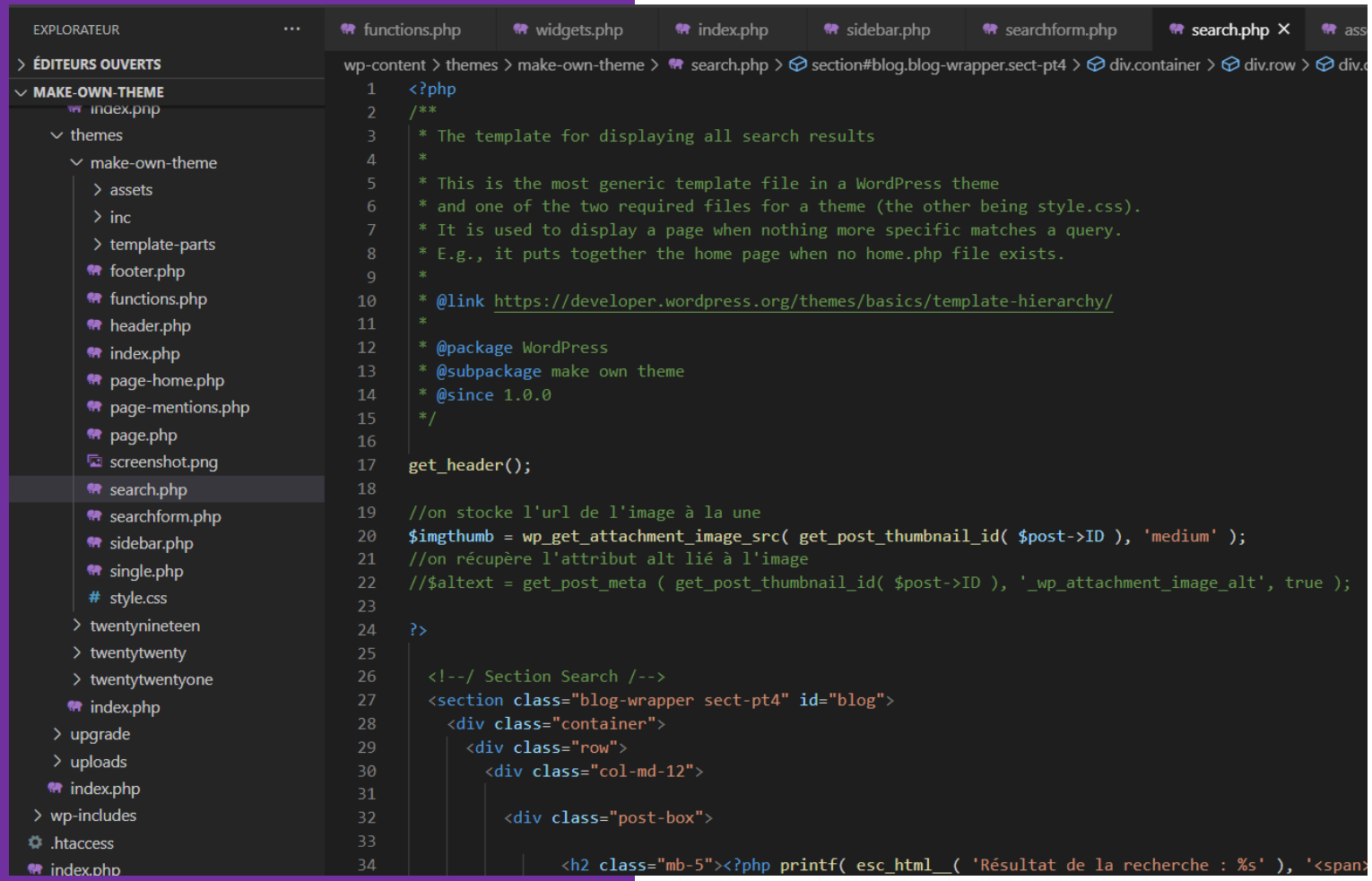
C’est dans ce fichier « searchform.php » que le formulaire est mis en place pour permettre à l’utilisateur de faire se recherche.



Wordpress

Création d’un fichier search.php pour l’affichage des résultats de recherche

Search.php



Pour afficher les résultats de la recherche qui ont été passé via le formulaire de recherche, il faut créer le fichier « search.php ».

Wordpress

Utilisation de la boucle avec des arguments pour affichage dans index.php

Boucle pour affichages sur index.php

```php
<?php if(have_posts()): ?>
    <div class="row">
<?php while (have_posts()): the_post(); ?>
        <div class="col-sm-12 col-md-12 col-lg-6">
            <div class="card mt-3 mb-3" style="width: 100%">
                <?php the_post_thumbnail('medium', ['class' => 'card-img-top', 'alt' => '']); ?>
                <div class="card-body">
                    <h3 class="card-title"><?= esc_attr(get_the_title()) ?></h3>
                    <?php
                    $categories = get_the_category();
                      var_dump($categories);
                    if(!empty($categories)):
                        ?>
                        <a href="<?= get_term_link($categories[0]) ?>" class="card-subtitles mb-2 text-muted"><?= $categories[0]-
                    <?php endif ?> <br>
                    <small>
                        <?= sprintf('Publié le %s', get_the_date()) ?>
                    </small>
                    <p class="card-text">
                        <?php the_excerpt(); ?>
                    </p>
                    <a href="<?php the_permalink(); ?>" class="btn-blue">En savoir plus</a>
                </div>
            </div>
        </div>
<?php endwhile ?>
        <div class="col-12 text-center">
            <div class="pagination">
                <?= paginate_links() ?>
            </div>
        </div>
    </div>
<?php else: ?>
    <h1>Il n'y a pas encore d'actualités.</h1>
<?php endif ?>
```

Pour afficher les résultats d'une requête sur les articles, il faut tout d'abord utiliser la fonction « have_posts » pour vérifier si il y a des articles.

Ensuite pour afficher article par article, il faut utiliser la boucle « while » (ou « tant que » en français).

C'est donc a chaque itération de cette boucle qu'un article sera affiché.

Boucle pour affichages sur index.php

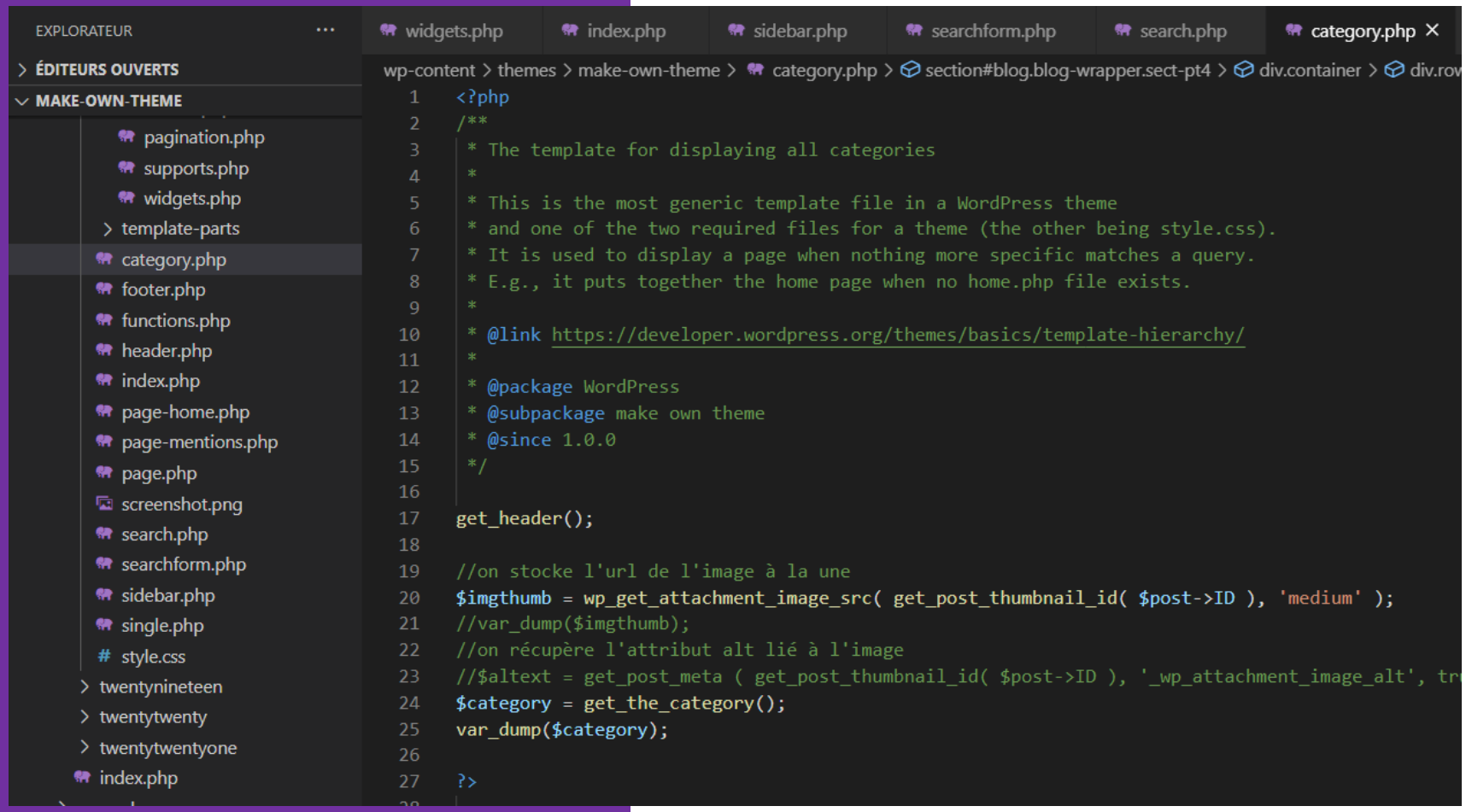
Pour chaque itération, il faut afficher l'image à la une avec « the_post_thumbnail », la catégorie avec « get_the_category », la date de publication avec « the_date », l'extrait de l'article avec « the_excerpt » et un lien vers l'article complet avec « the_permalink ».

```php
<?php while (have_posts()): the_post(); ?>
        <div class="col-sm-12 col-md-12 col-lg-6">
            <div class="card mt-3 mb-3" style="width: 100%">
                <?php the_post_thumbnail('medium', ['class' => 'card-img-top', 'alt' => '']); ?>
                <div class="card-body">
                    <h3 class="card-title"><?= esc_attr(get_the_title()) ?></h3>
                    <?php
                    $categories = get_the_category();
                      var_dump($categories);
                    if(!empty($categories)):
                        ?>
                        <a href="<?= get_term_link($categories[0]) ?>" class="card-subtitles mb-2 text-muted"><?= $categories[0]-
                    <?php endif ?> <br>
                    <small>
                        <?= sprintf('Publié le %s', get_the_date()) ?>
                    </small>
                    <p class="card-text">
                        <?php the_excerpt(); ?>
                    </p>
                    <a href="<?php the_permalink(); ?>" class="btn-blue">En savoir plus</a>
                </div>
            </div>
        </div>
<?php endwhile ?>
```

Wordpress

Création de la page category.php

Category.php



Pour gérer l'affichage de tous les articles d'une catégorie donnée, il faut mettre en place le fichier « category.php ».

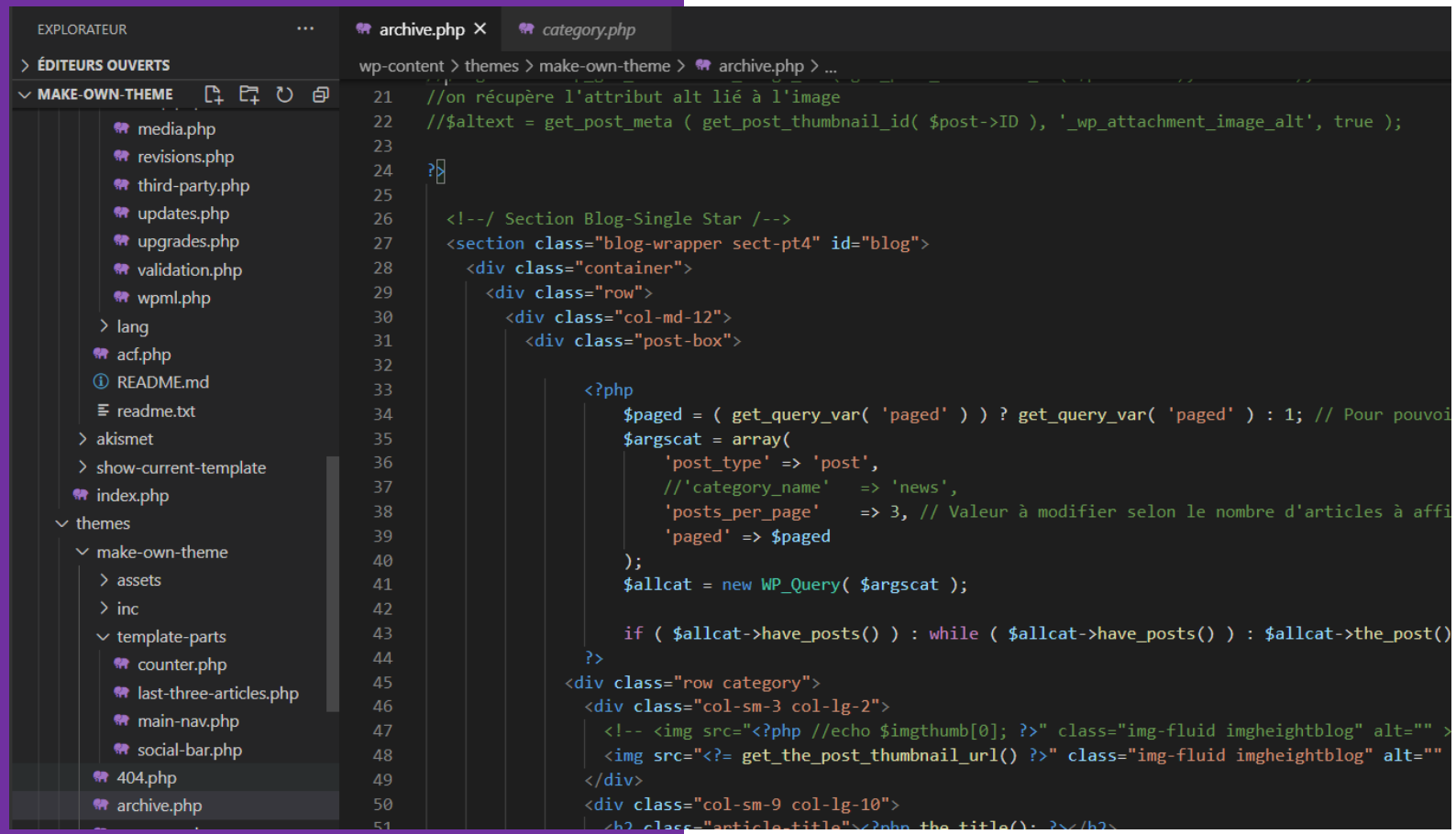
C'est dans ce fichier qu'il faut faire la requête qui va allé chercher les articles de la catégorie concernée.

Ensuite, il ne reste plus qu'a mettre en place l'affichage si des articles existent bien dans la catégorie.

Wordpress

Création de la page archive.php

Archive.php



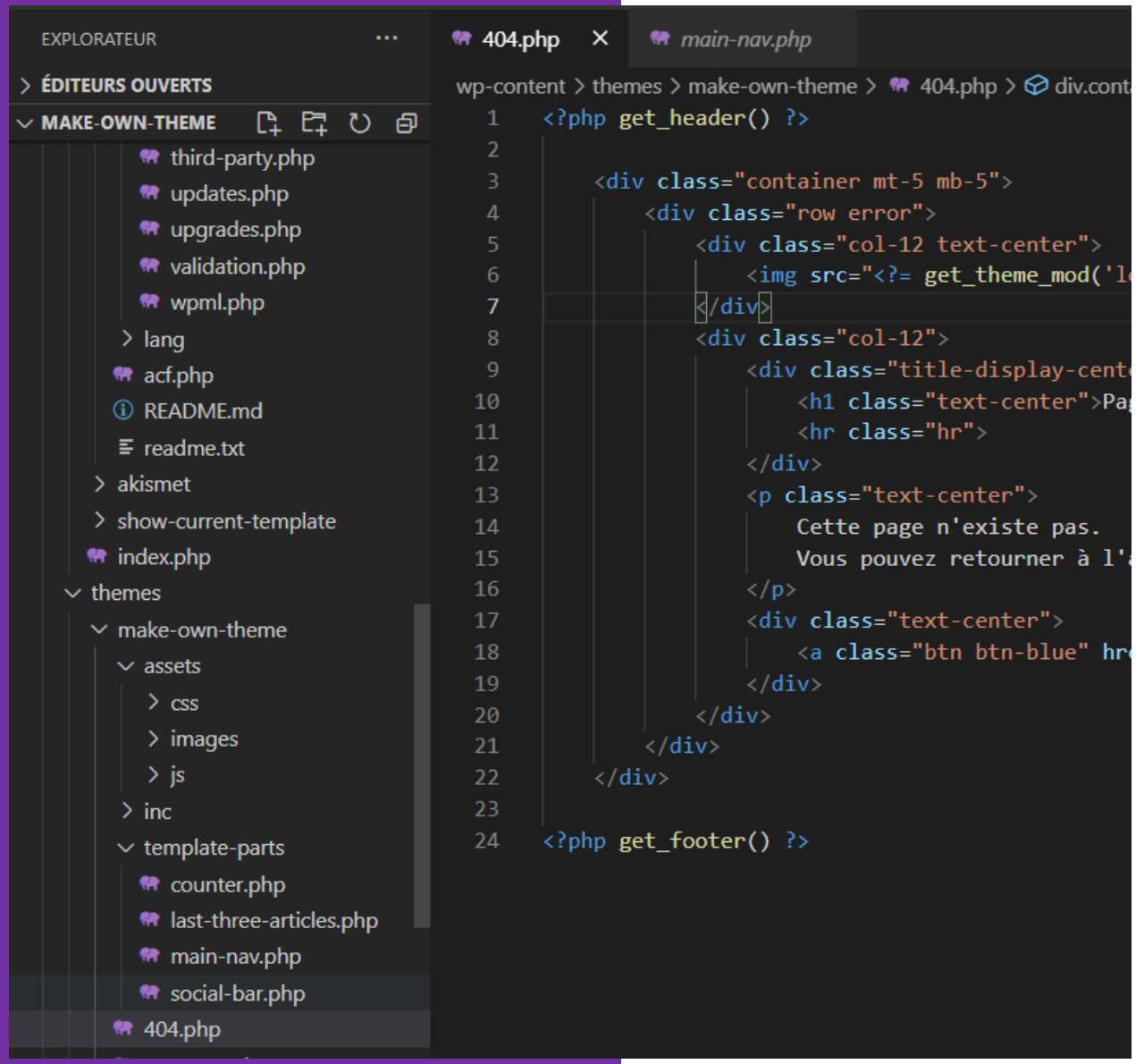
La page « archive.php » comme son nom l'indique, permets d'afficher tous les articles en les archivant soit par dates et/ou catégories.

C'est dans ce fichier qu'il faut faire une requête qui va allé chercher tous les articles.

Ensuite, il ne reste plus qu'a mettre en place l'affichage si des articles existent.

Wordpress

Création de la page 404.php



404.php

Quand un utilisateur tape une mauvaise url ou tout simplement quand il est redirigé vers une mauvaise adresse, il est essentiel de lui indiqué qu'une erreur est survenue.

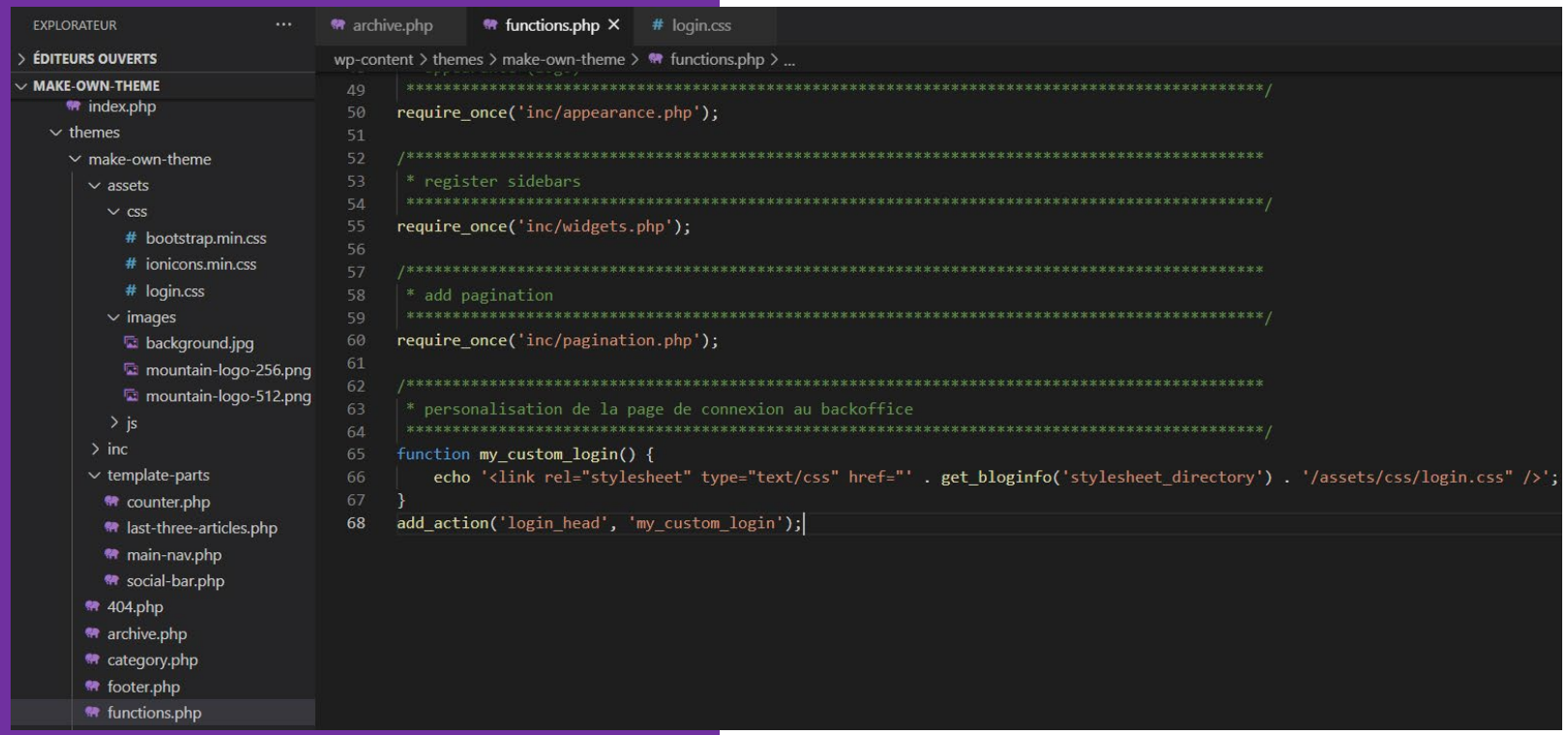
En créant le fichier « 404.php », Wordpress sera quel affichage utiliser lorsqu'une url n'est pas bonne.

Dans ce fichier, il faut en général mettre un texte explicatif pour prévenir l'utilisateur et un bouton pour qu'il puisse revenir à la page d'accueil.

Wordpress

Personnalisation de la page login.php

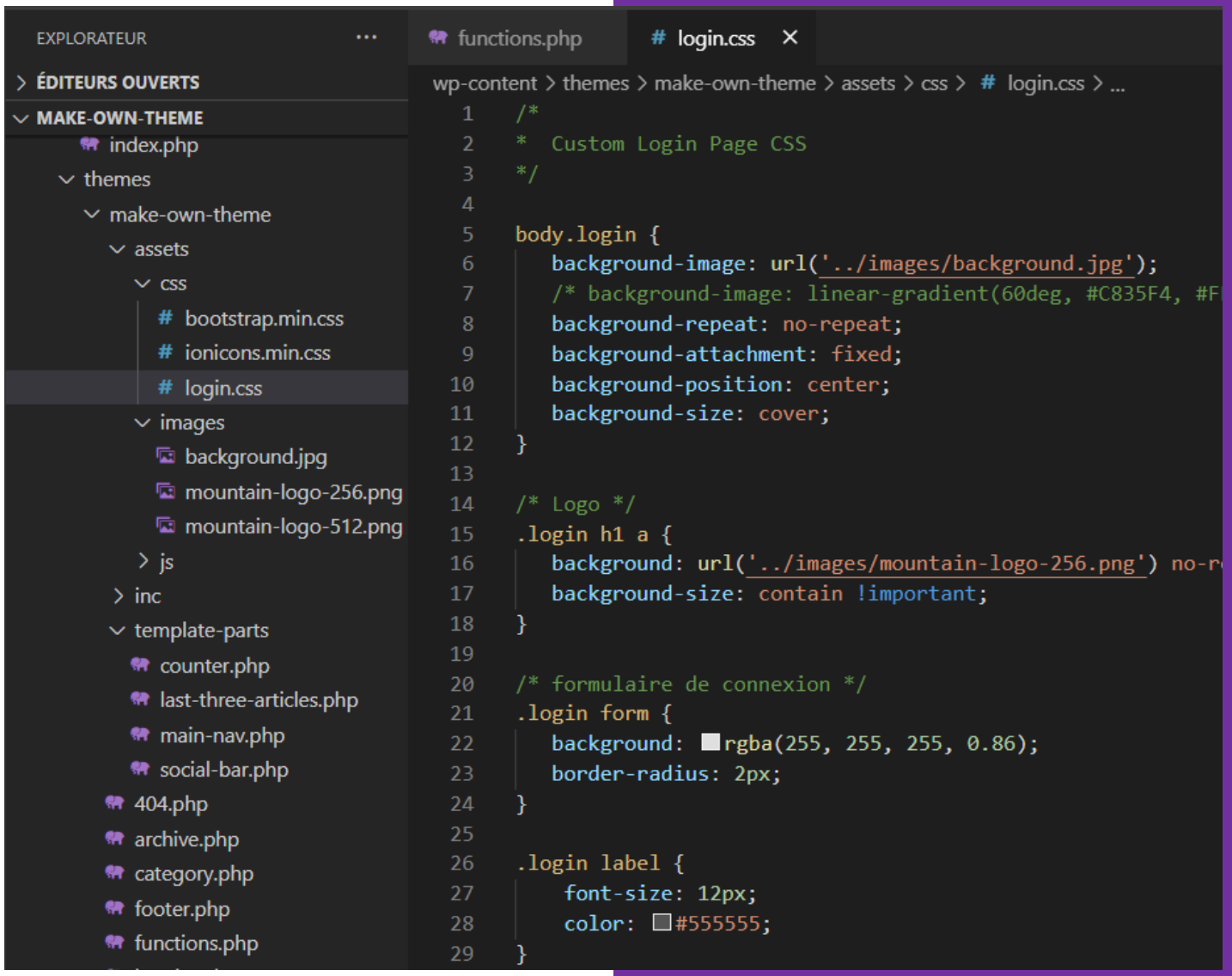
Login.php



Pour modifier la mise en forme de la page de login du backoffice propre à Wordpress. Il faut se brancher sur le « hook » de la page de connexion : « login_head » pour y ajouter le style custom que l'on veut appliquer.

Login.php

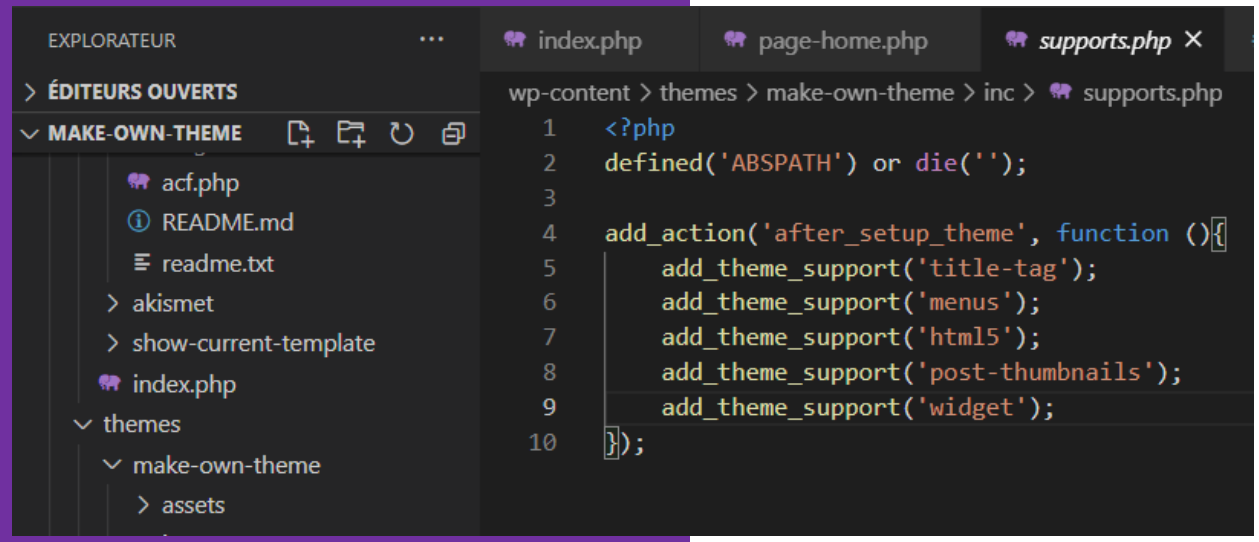
Une fois que la feuille de style est crée et liée au CMS, il ne reste plus qu'a scruter les classes utilisées sur la page de « login » du backoffice pour modifier le style de celle-ci.



Wordpress

Déclaration de la zone du menu de navigation principal

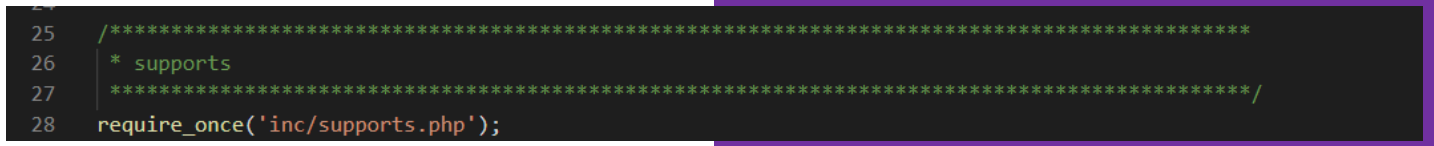
Ajout du support pour le menu (supports.php)



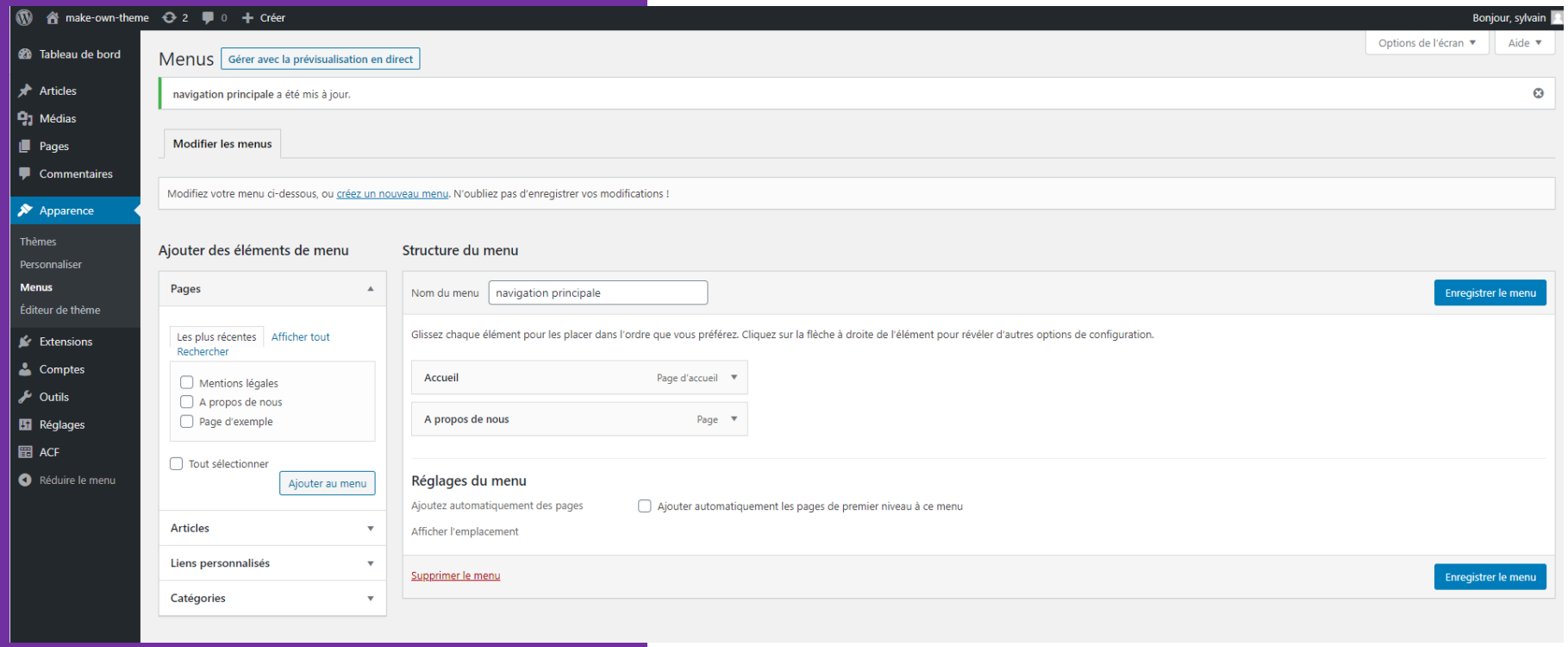
Pour pouvoir ajouter ou modifier les menus sur le backoffice, il faut ajouter le support dans le thème grâce à la fonction « add_theme_support » et en lui passant « menus » comme argument.

Ajout du support du menu (functions.php)

Une fois le support des menus ajouté dans « supports.php », il ne faut pas oublier le « require » dans le fichier « functions.php » pour être sur que les ajouts seront faits.



Création du menu dans le backoffice



Une fois que le support pour le menu est déclaré dans le thème, la zone de menu est a mettre en place en se rendant dans Apparence > Menus.

Il faut déterminer un nom pour le menu.

Puis ensuite, lui attribuer les pages correspondantes et enregistrer le menu.

Wordpress

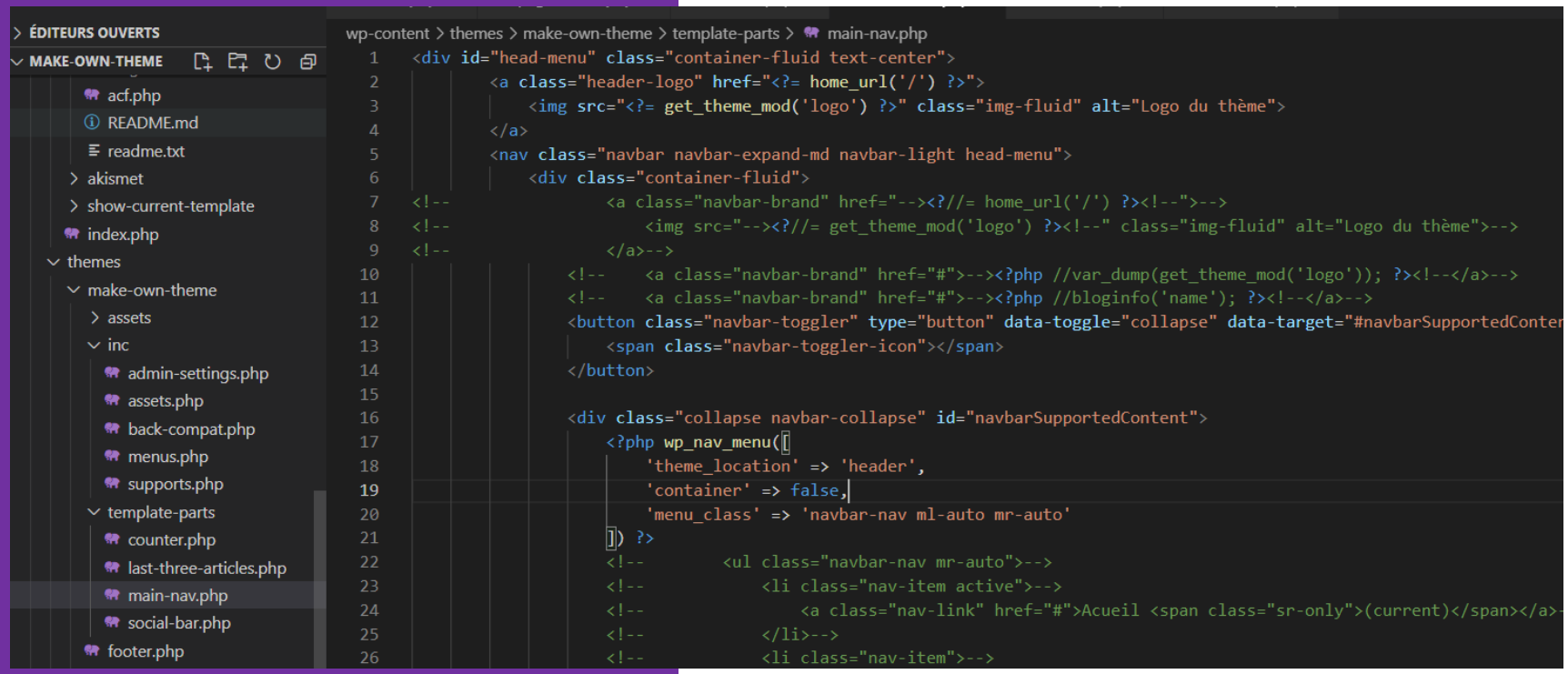
Intégration d’un menu simple

Mise en place du menu dans header.php

Pour le menu de navigation,
c'est au niveau du fichier
« header.php » qu'il faudra
ajouter avec la fonction
« get_template_part » l'appel au
fichier du template du menu.

```
26
27  <body id="page-top">
28  <header>
29          <?php
30          // Output the main navigation menu
31          get_template_part( 'template-parts/main-nav' );
32          ?>
33  </header>
```

Création du template pour le menu



Le fichier « main-nav.php » est à créer dans le dossier « template-parts ».

Sur cette capture, c'est une barre de navigation de Bootstrap qui a été utilisé pour la structure.

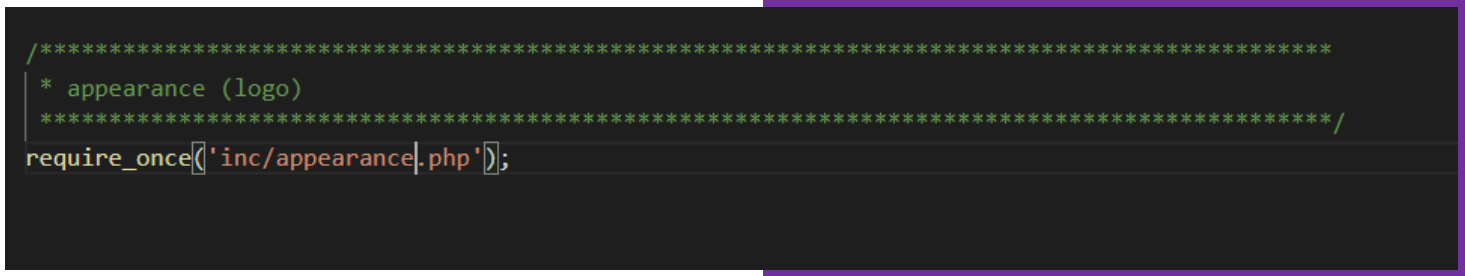
La fonction « wp_nav_menu » permet d'appeler un menu en précisant la localisation du menu dans le thème et de rajouter des classes propres à Bootstrap pour styliser le menu.

Ajout d’un logo dynamique (functions.php)

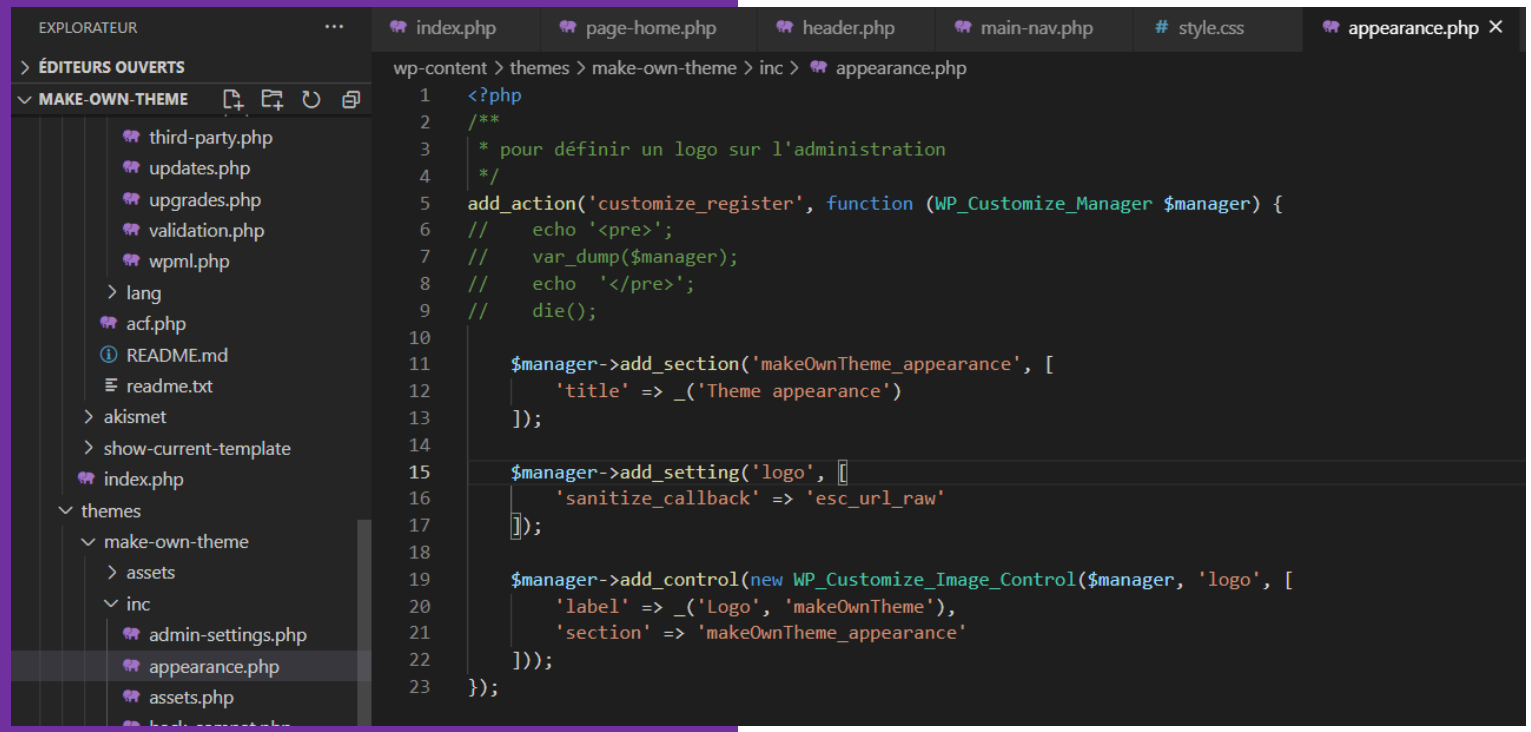
Un petit plus qui peut être appréciable pour l’utilisateur, en lui permettant d’ajouter le logo qu’il souhaite dans le backoffice.

Pour ce faire, il faut commencer par faire un « require_once » du fichier « appearance.php » dans le fichier « functions.php ».

Ensuite, la création du fichier « appearance.php » ce fait dans le dossier « inc » qui est à la racine du dossier du thème.



Ajout d’un logo dynamique (appearance.php)



Le fichier «appearance.php » vas permettre d’ajouter une action qui donnera la possibilité d’ajouter un logo via le menu apparence dans le backoffice.

La première fonction « add_section » qui est branchée sur le manager vas ajouter la section.

Ensuite, un paramètre va se greffer pour nettoyer l’url et s’assurer que c’est un slug.

Et pour finir, l’ajout de contrôles pour l’utilisateur du backoffice.

Ajout d’un logo dynamique (functions.php)

Pour ajouter un logo personalisé, il faut aller dans « apparence » puis « personnaliser » dans le backoffice.

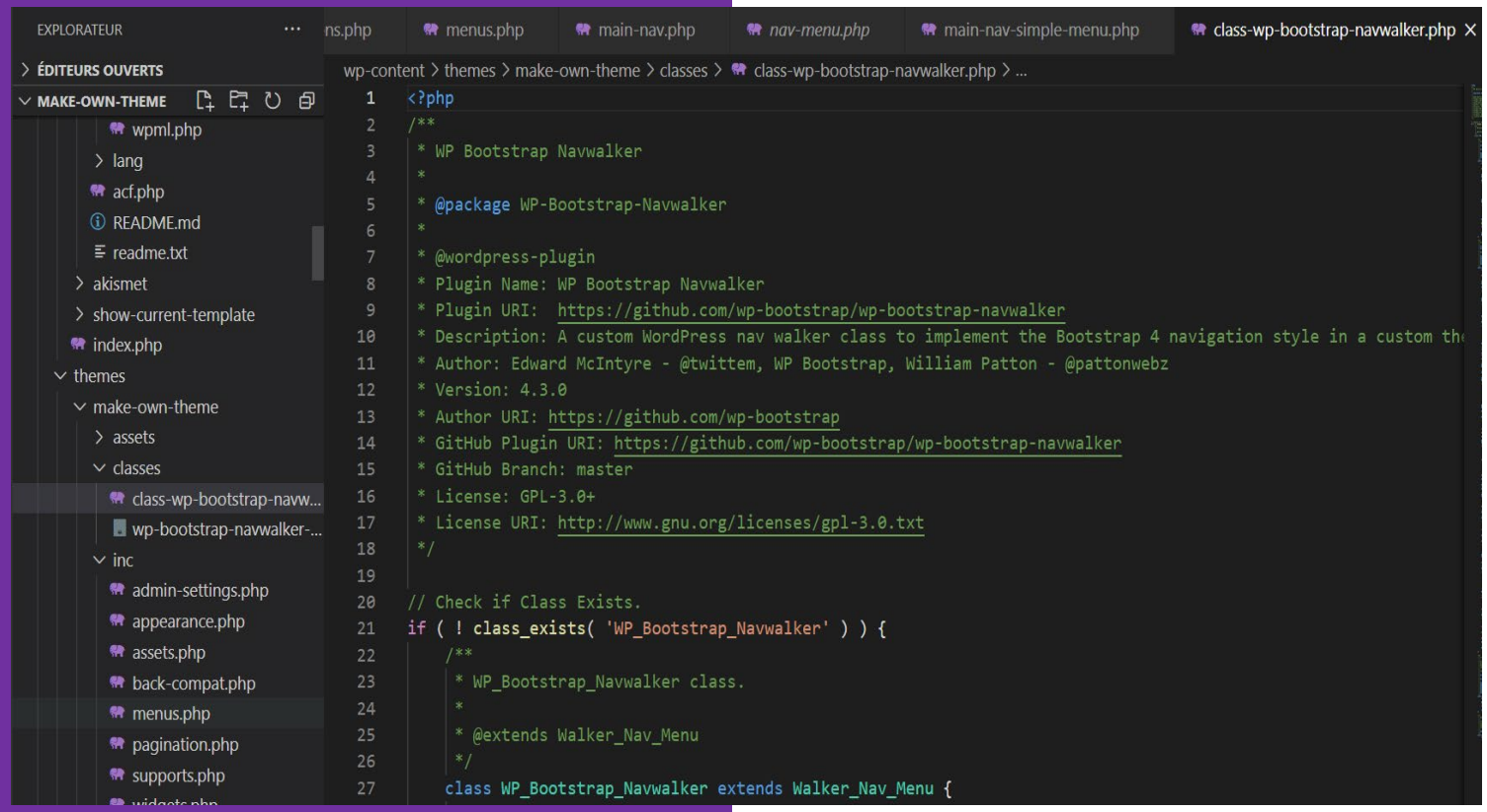
L’onglet « theme appearance » est désormais disponible. Il faut cliquer dessus pour pouvoir changer le logo.

Pour finir, un simple glisser/déposer de l’image qui servira de logo, un ajout du texte alternatif et un enregistrement des modifications permettront d’ajouter le logo sur les pages du site.

Wordpress

Intégration d’un menu complexe avec un walker pour Bootstrap 4

Télécharger le walker



Pour télécharger le « walker » via ce lien https://github.com/wp-bootstrap/wp-bootstrap-navwalker pour avoir la possibilité de customisé encore plus un menu dans Wordpress par exemple en rajoutant de la profondeur dans la hiérarchie des liens du menu.

Une fois le walker téléchargé, il faut dézipper le dossier pour récupérer le fichier qui est necessaire.

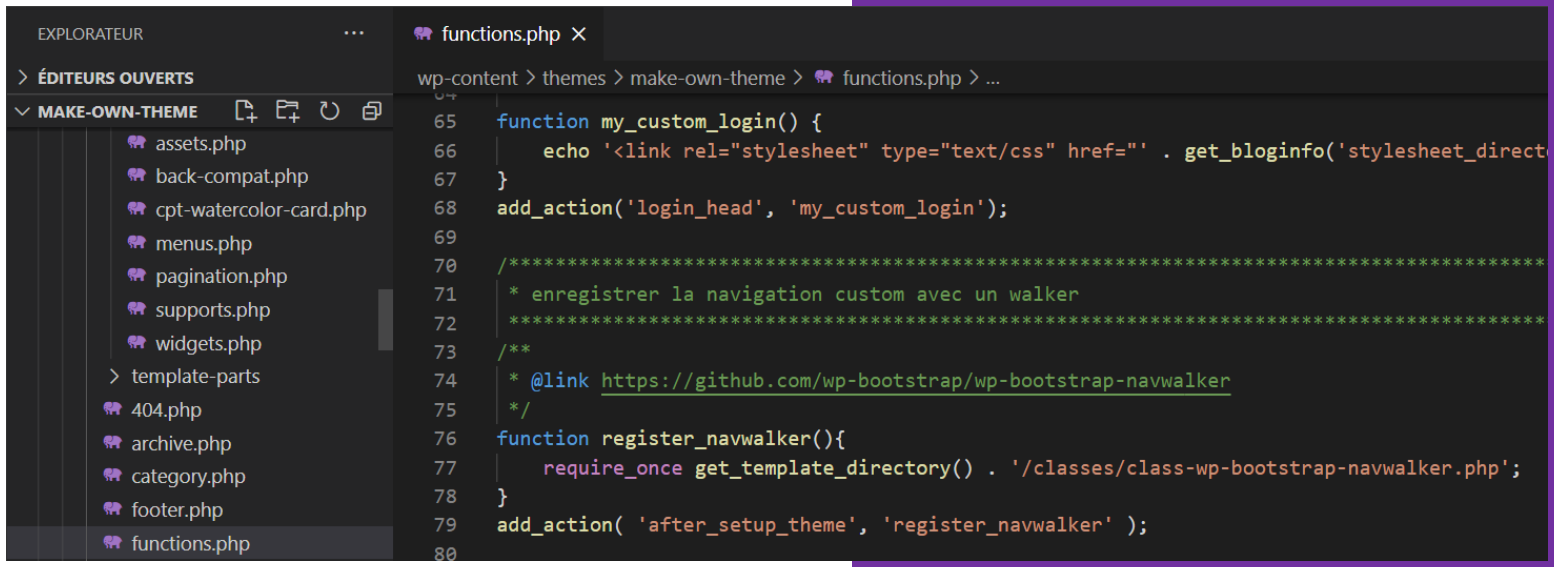
Ici, c'est le fichier: « class-wp-bootstrap-navwalker.php ».

Insertion dans “functions.php”

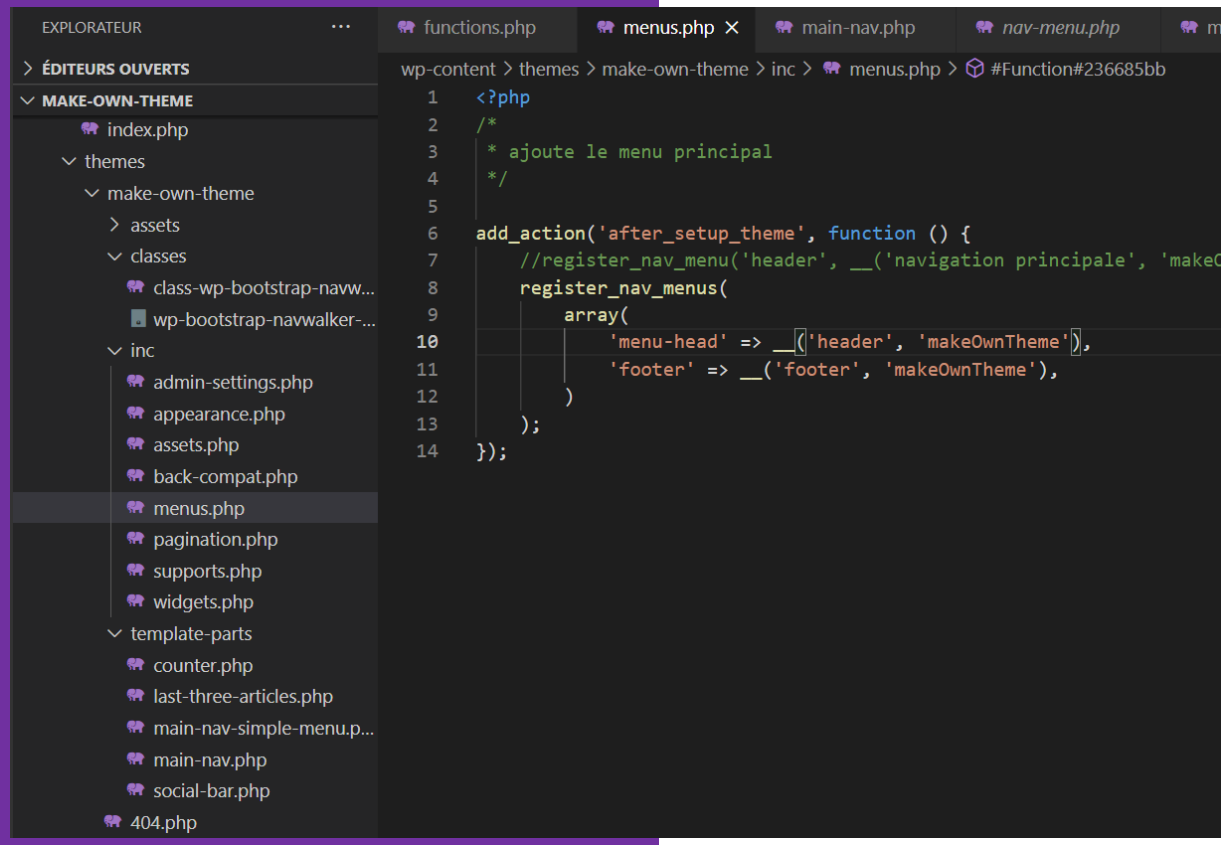
Il faut désormais mettre le fichier téléchargé dans le dossier « classes ».

Il faut enregistrer ce « walker » dans le « hook » de Wordpress pour y avoir accès.

Penser à bien récupérer le bon fichier grâce à la fonction « get_template_directory » puis l’url qui corresponds à la structure ou se trouve le fichier.



Enregistrement des menus



Comme pour le menu simple, il faut veiller à bien enregistrer le menu pour Wordpress.

Avec la fonction « register_nav_menus » qui permets d'enregistrer plusieurs menus.

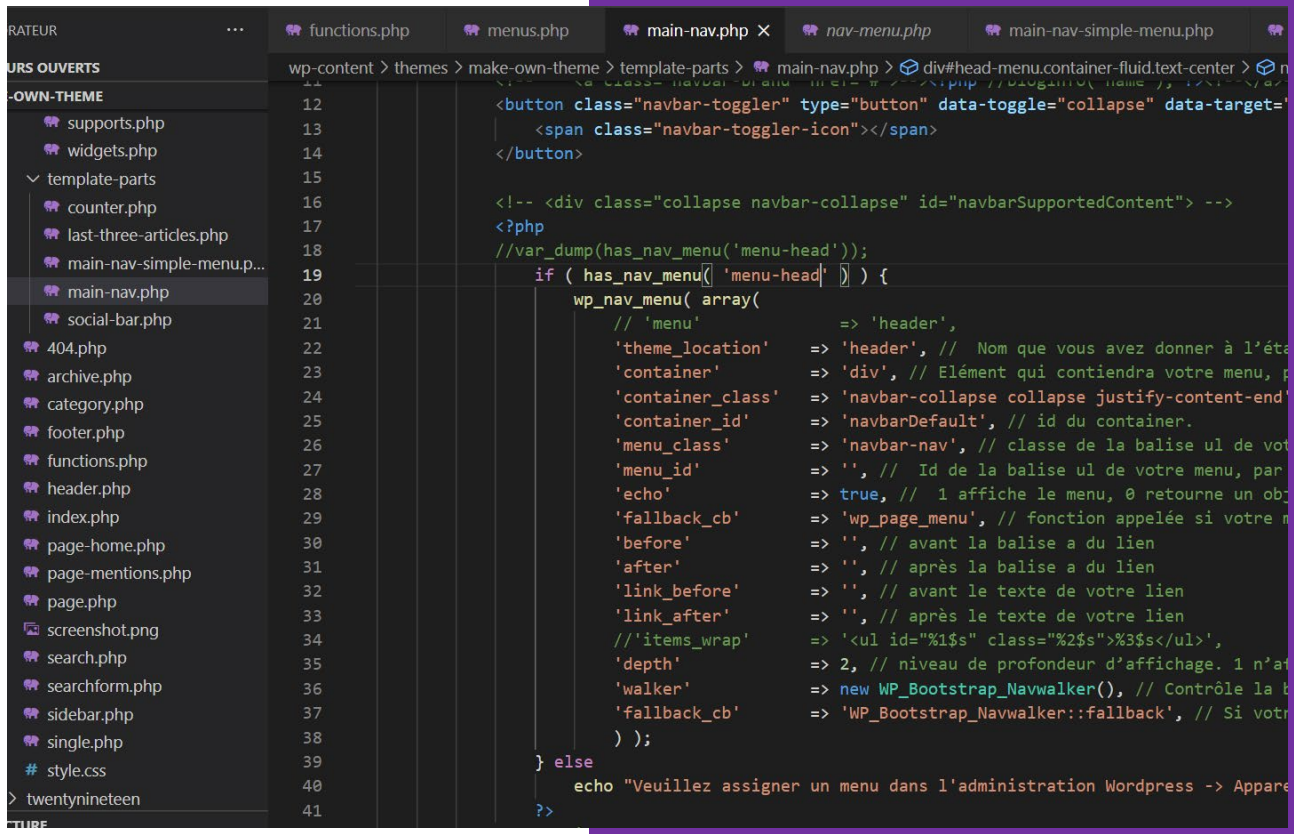
Cette fonction prends en paramètres un tableau PHP qui comprend a chaque ligne le nom du menu et sa localisation dans le thème.

Mise en place du menu dans le header

Tous d'abord, la vérification de l'existance du menu avec la fonction « has_nav_menu ».

Ensuite avec la fonction « wp_nav_menu », un tableau PHP permets de configurer les différentes options du menu.

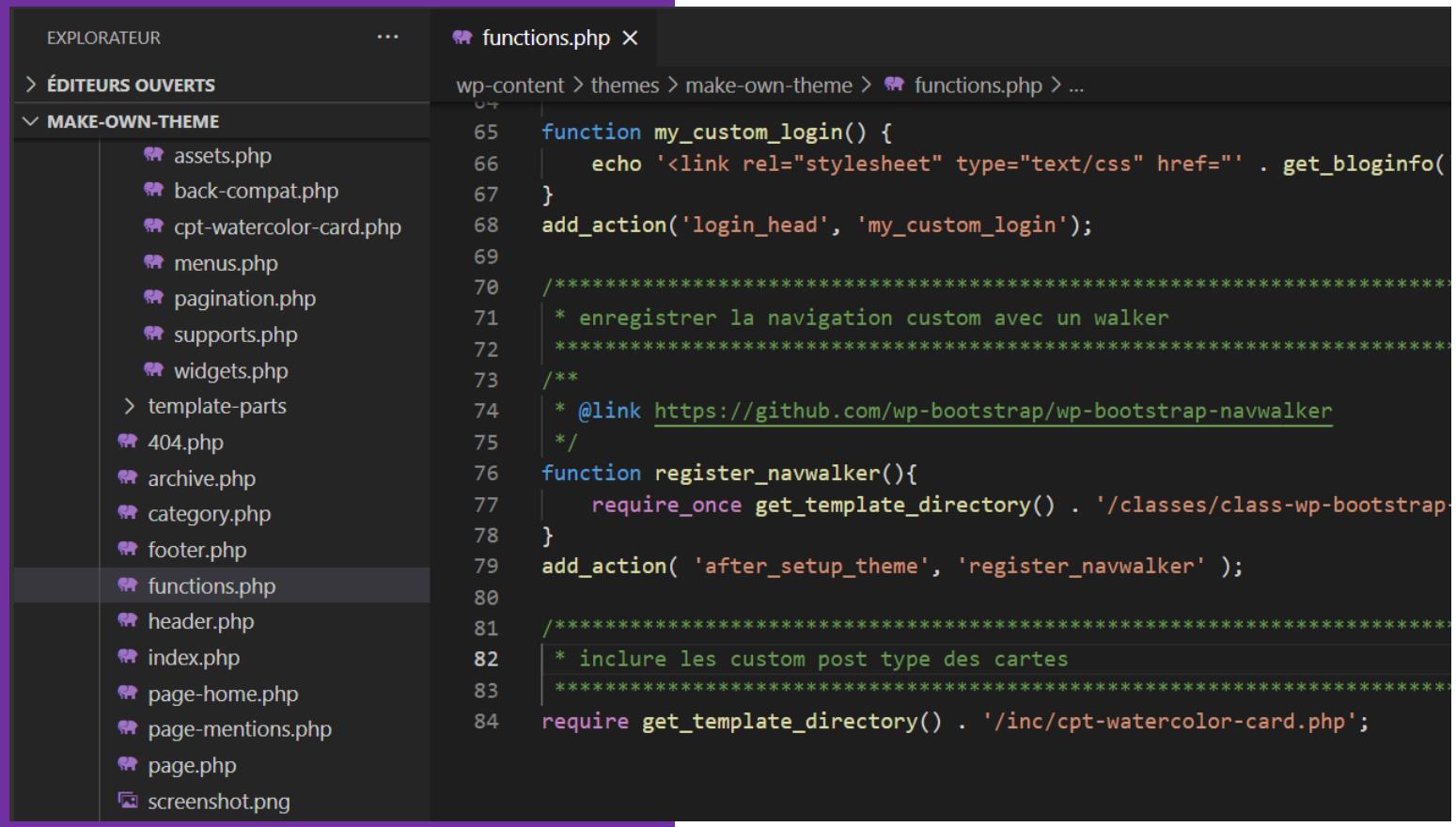
Comme sa localisation, les classes utilisées dans le menu et le « walker » utilisé pour contrôler la barre de navigation.



Wordpress

Création from Scratch d'un Custom Post Type (functions.php) avec ses taxonomies

Inclure les “custom post type”



Qu’est-ce qu’un « custom post type » ?

C’est un type de contenu customisé qui adopte le comportement des articles de Wordpress.

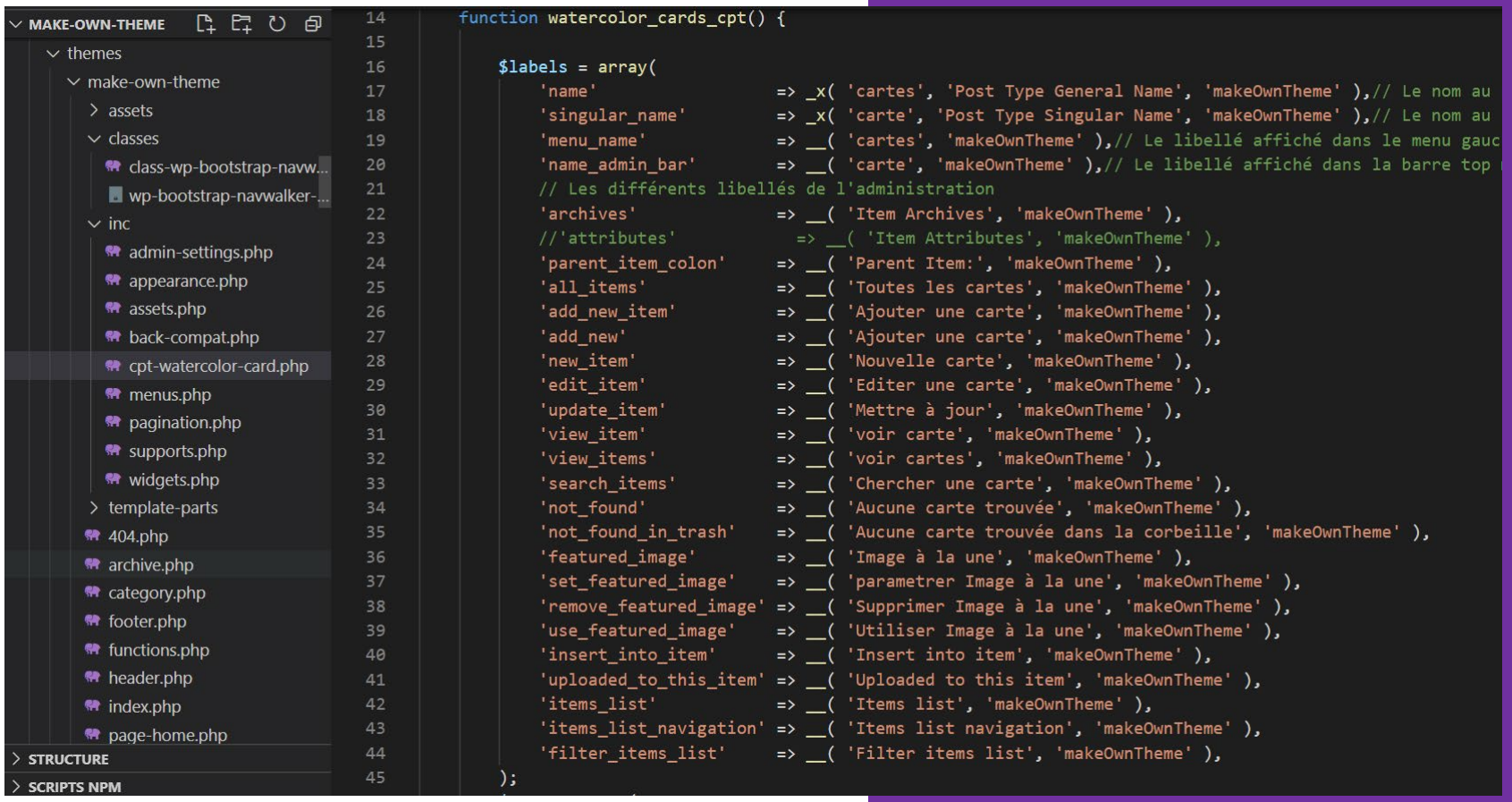
Pour commencer, il faut créer le fichier « cpt-watercolor-card » dans le dossier « inc » puis ne pas oublier de l’inclure dans le fichier « functions.php ».

Le fichier “cpt-watercolor-card”

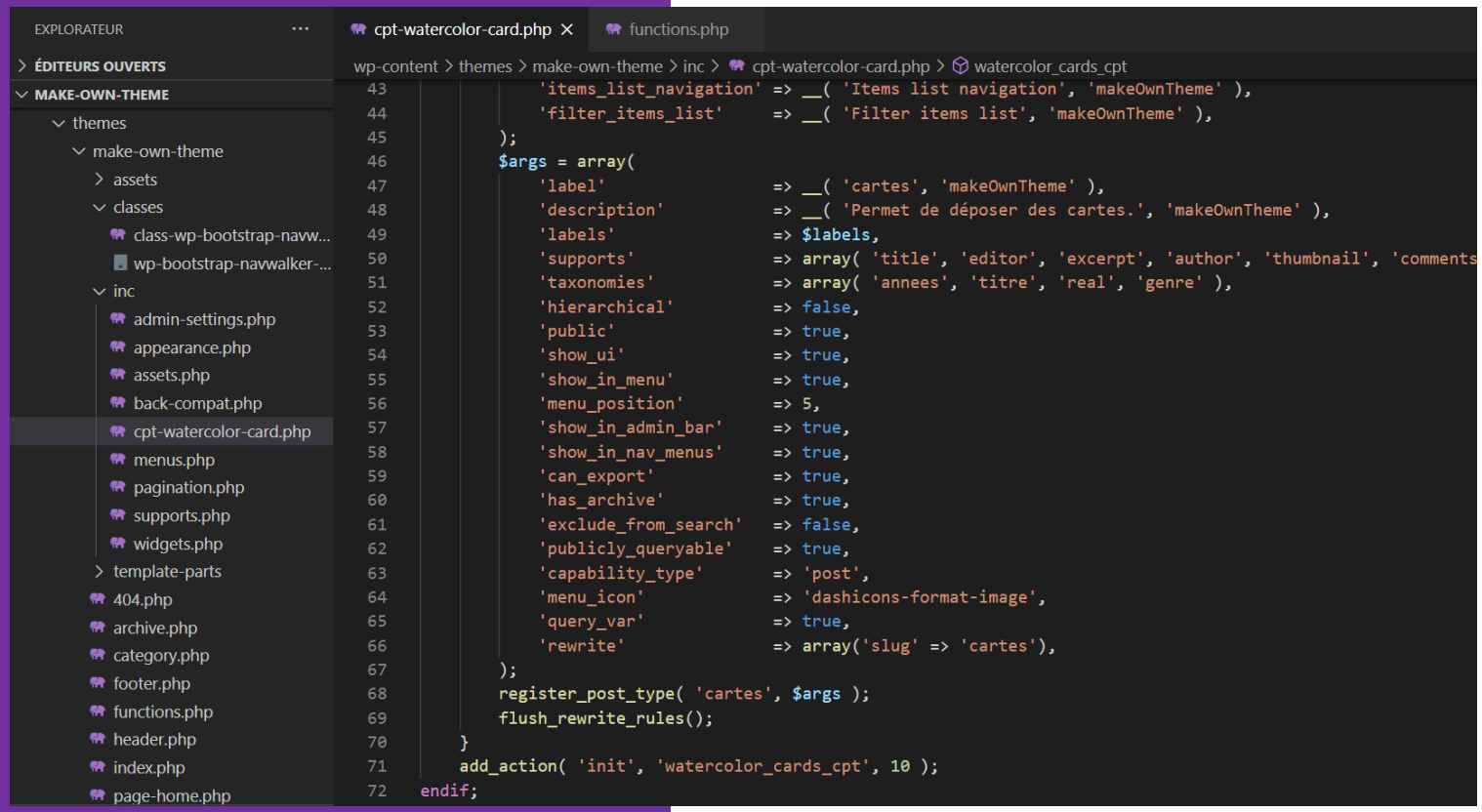
Ici, le « custom post type » est nommé « carte » mais il est bien sur possible de créer un tout autre en changeant le nom et les paramètres.

Le tableau « labels » contient tous les paramètres à mettre en place pour gérer le nom au singulier et au pluriel du « custom post type » .

Mais aussi tous les affichages des textes dans le backoffice.



Les arguments du custom post type



Après avoir mis en place le tableau « labels », il faut créer le tableau d'arguments pour le « custom post type ».

Ici, il faut déterminer le label, la description, les supports autorisés (titre, éditeur, image à la une) et les taxonomies utilisées.

Le paramètre « menu_position » permets de définir une position dans le menu du backoffice.

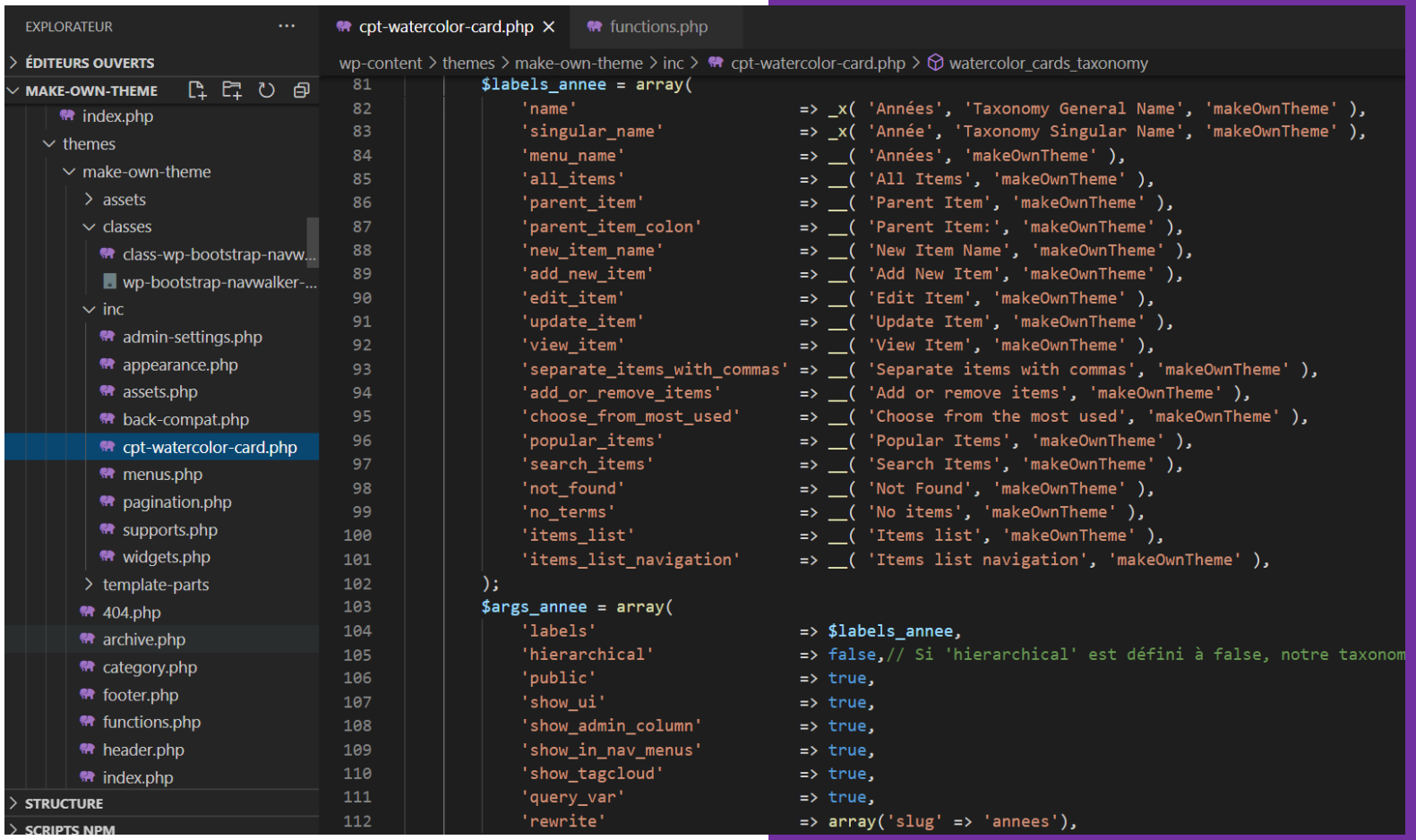
Il ne reste plus qu'a l'enregistrer grâce à la fonction « register_post_type ».

Taxonomie “année”

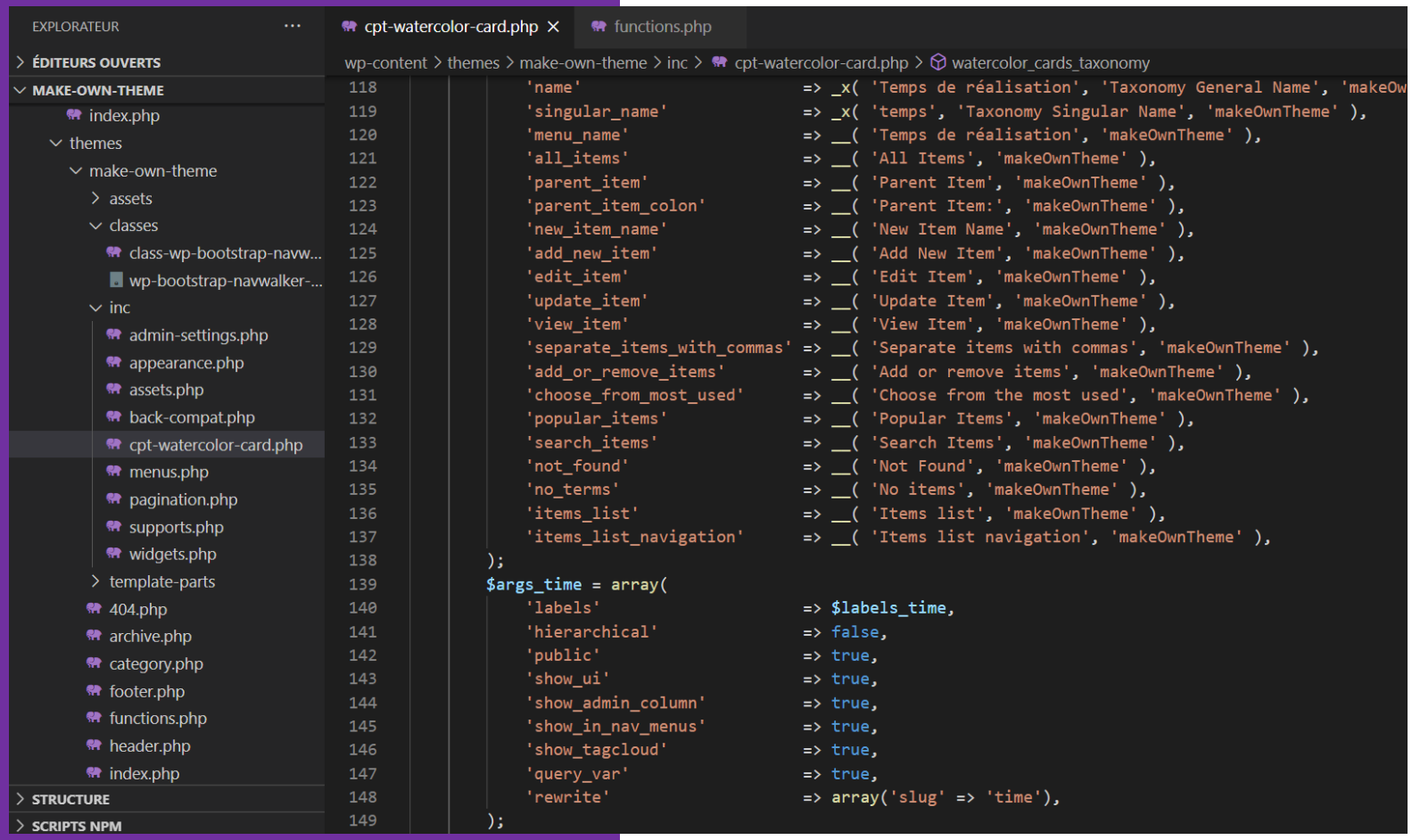
Comme pour le « custom post type », ses taxonomies doivent être enregistrée.

Il faut déclarer deux tableaux, un pour les libellés et un pour les arguments.

Pour la taxonomie « année », il faut déterminer les affichages pour les textes, sa hiérarchie, si son affichage est publique ou non, etc...



Taxonomie “temps”



Une fois la taxonomie « annee » déclarée, c’est au tour de la taxonomie « temps ».

Il faut déclarer deux tableaux, un pour les libellés et un pour les arguments.

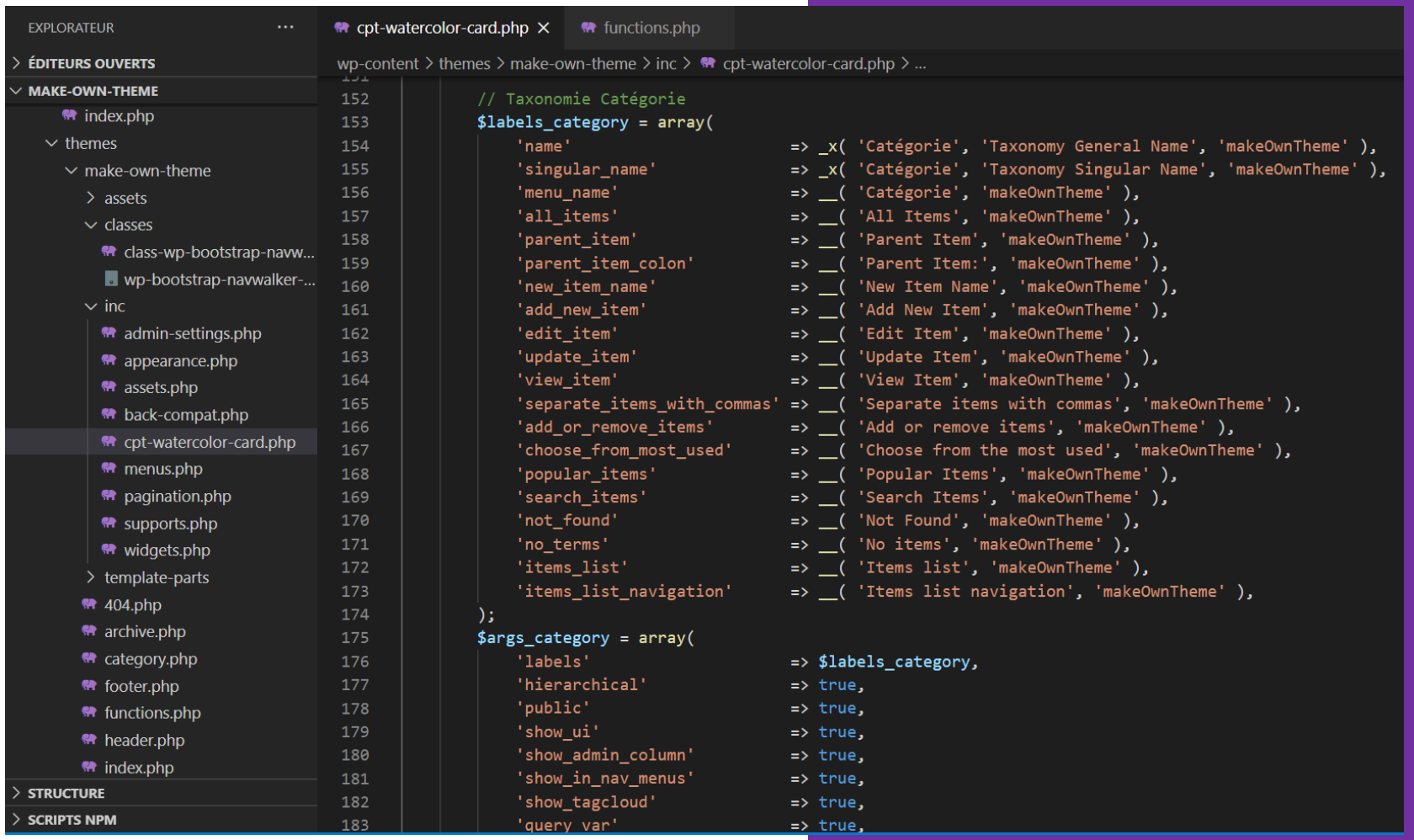
Pour la taxonomie « temps », il faut déterminer les affichages pour les textes, sa hiérarchie, si son affichage est publique ou non, etc...

Taxonomie “catégorie”

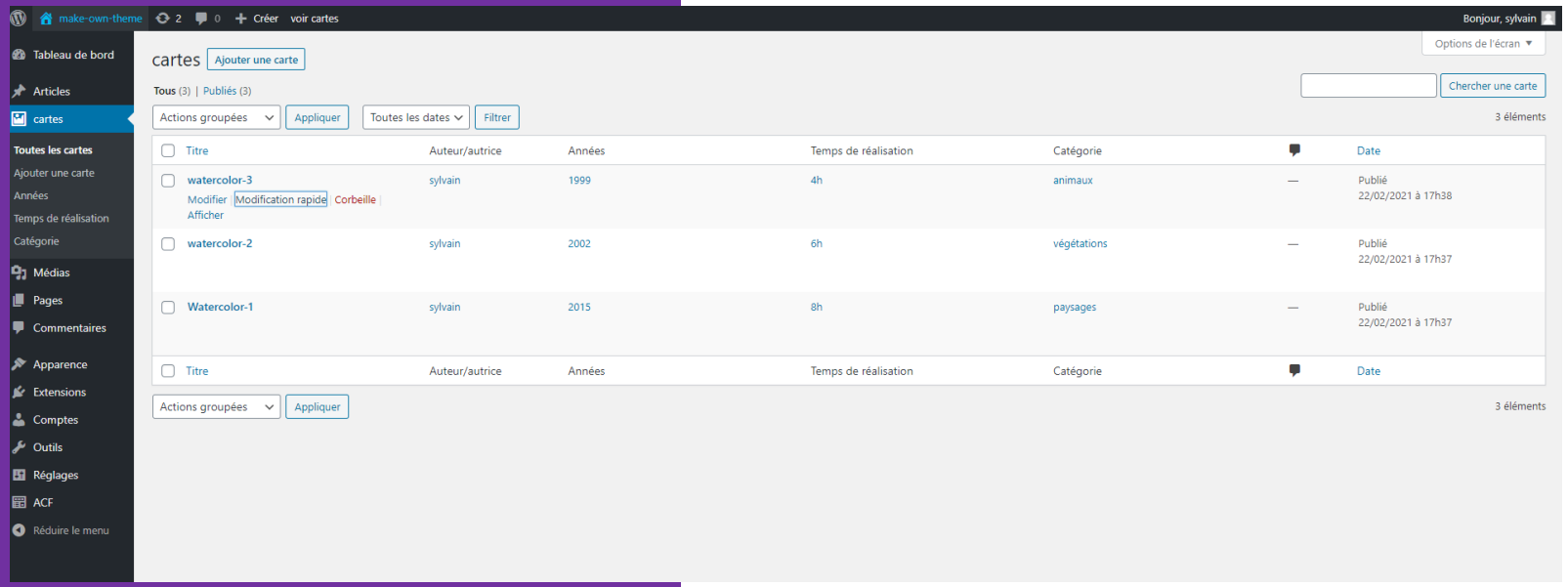
Une fois la taxonomie « temps » déclarée, c’est au tour de la taxonomie « catégorie ».

Il faut déclarer deux tableaux, un pour les libellés et un pour les arguments.

Pour la taxonomie « catégorie », il faut déterminer les affichages pour les textes, sa hiérarchie, si son affichage est publique ou non, etc...



Les “custom post type” dans le back office

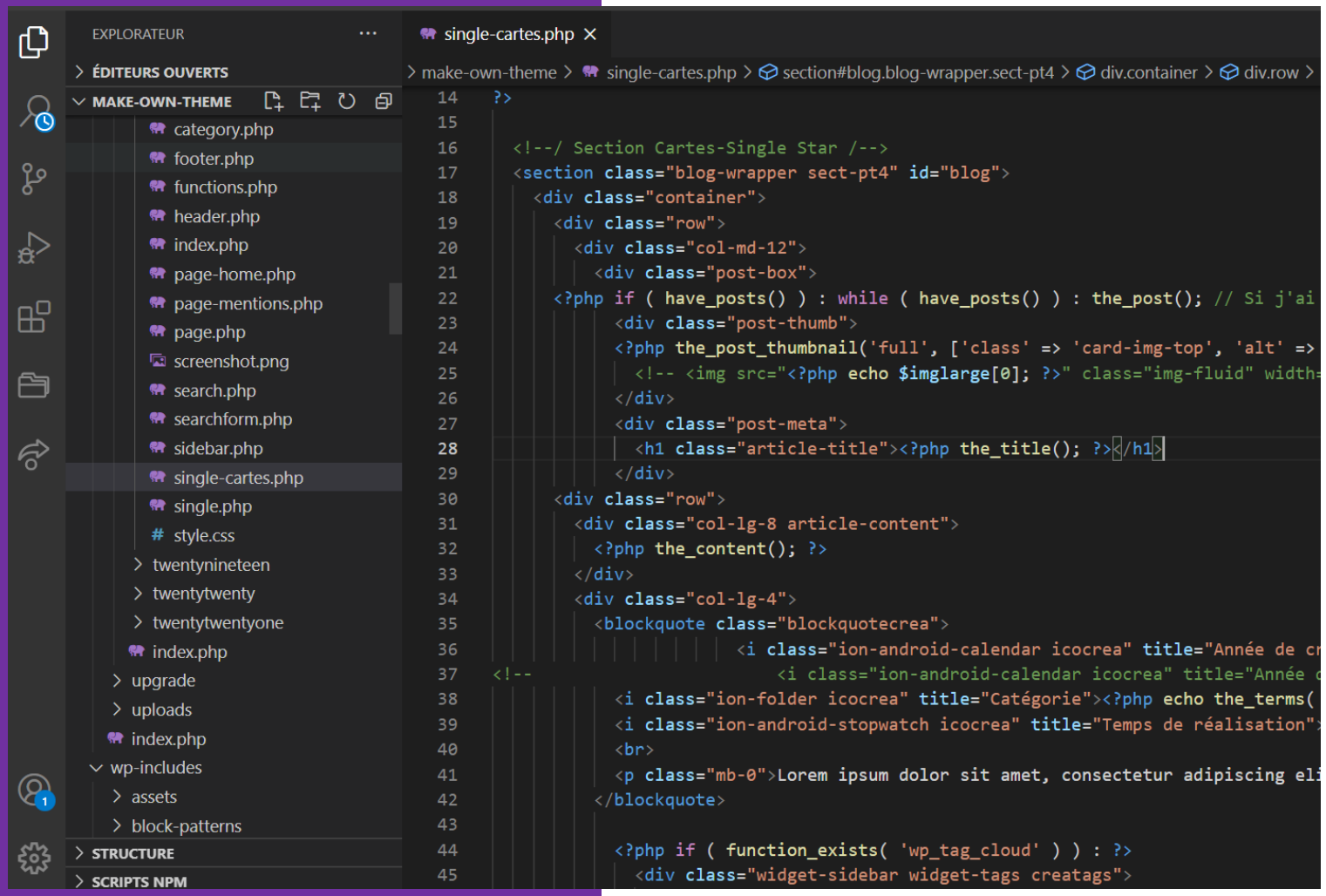


Quand le « custom post type » et ses taxonomies sont déclarés, vous devriez avoir dans votre backoffice un affichage similaire à cette capture.

Wordpress

Création de la page single-SLUG-DU-CPT.php pour l’affichage d’un item de ce post type

Page pour custom post type



Comme pour les articles de Wordpress qui ont un affichage spécifique pour chaque articles.

Les « custom post type » en ont un aussi.

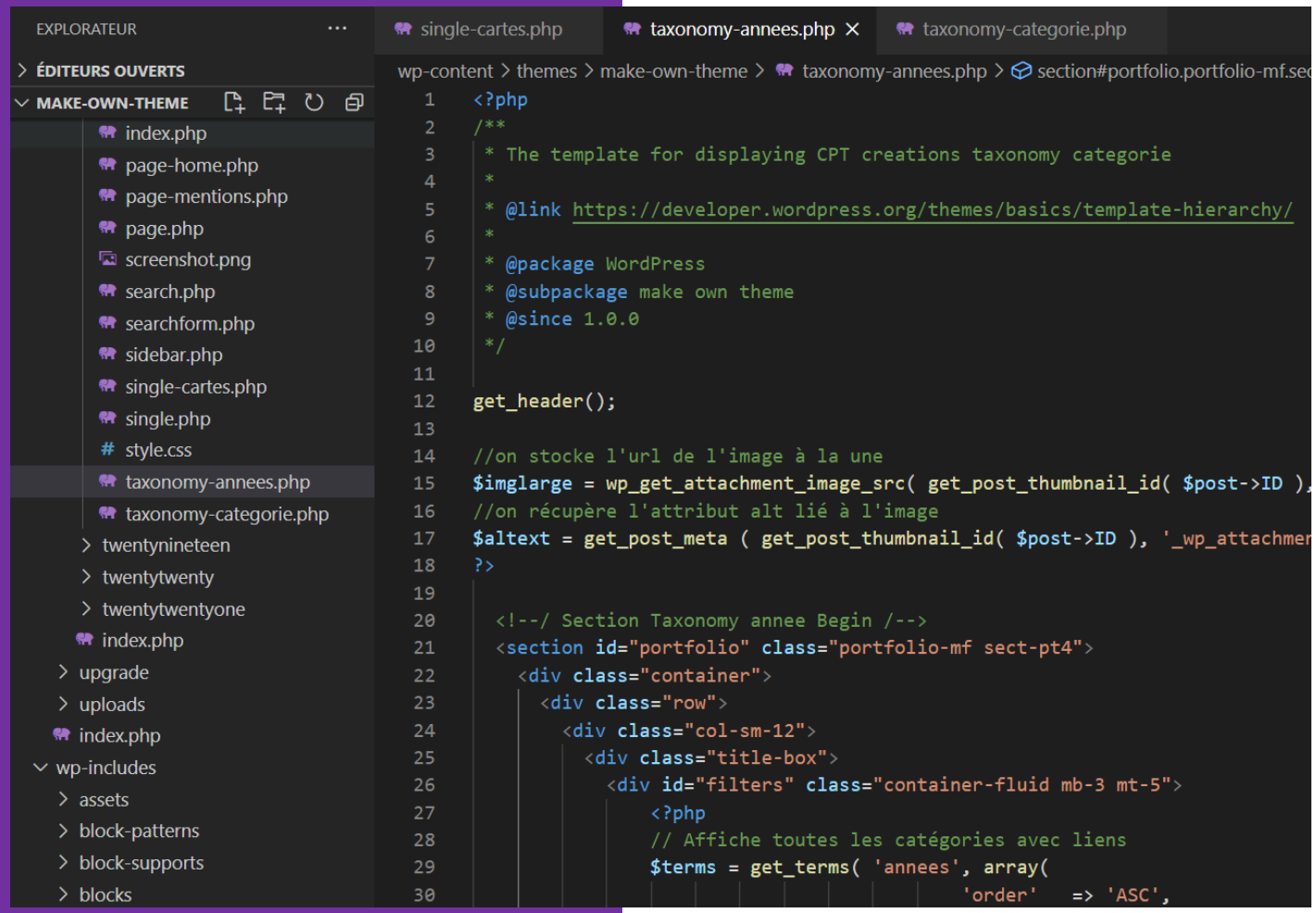
Il suffit de créer un fichier à la racine du théme en suivant se nommage « single-cpt.php ».

Le terme « cpt » sera a remplacer par le nom du « custom post type ».

Wordpress

Création de la page taxonomy-SLUG-TAXONOMY.php pour l'affichage des post type d'une taxonomie spécifique

Taxonomie “annees”



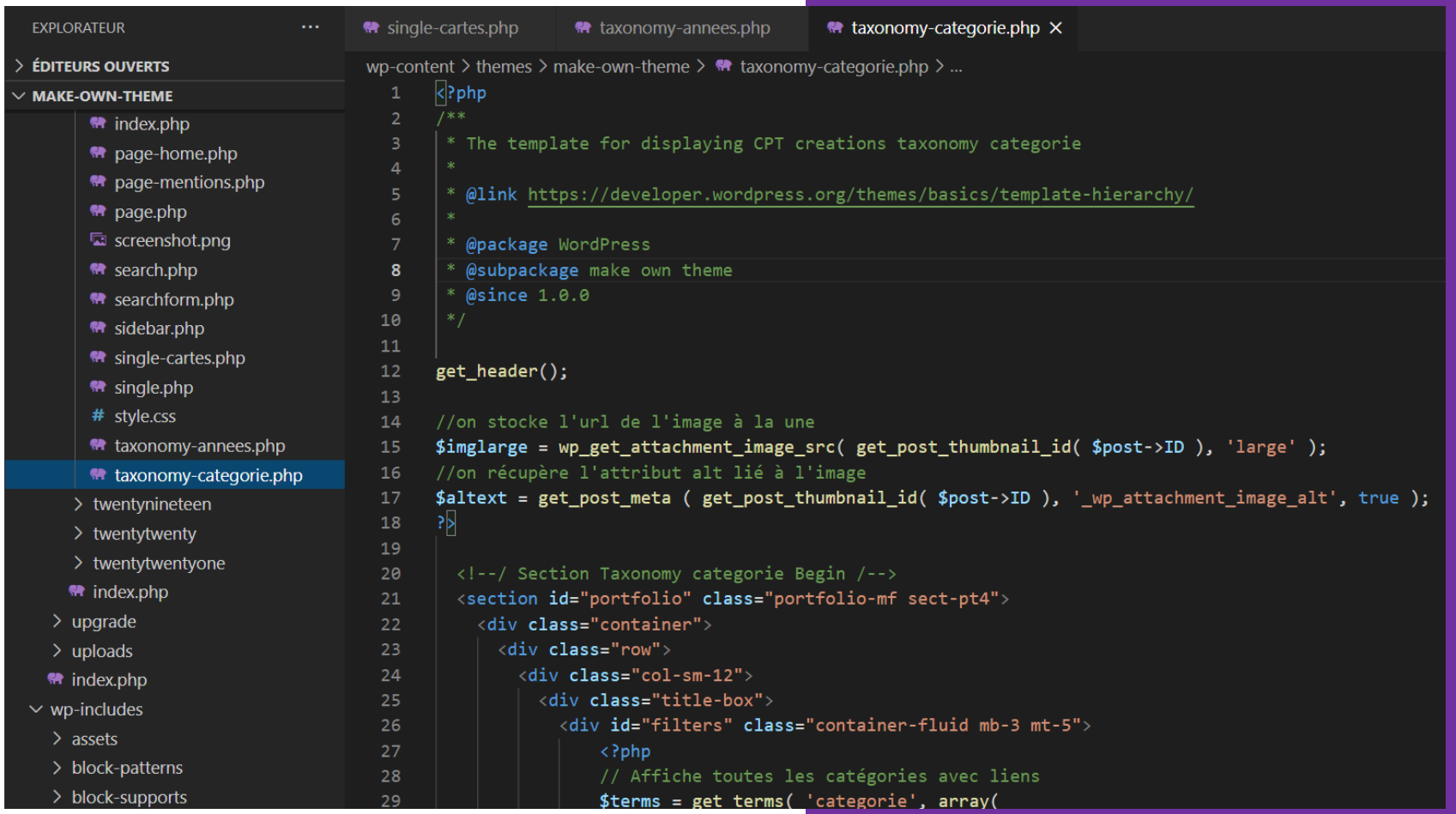
En mettant en place le fichier « taxonomy-annees.php », il est possible de gérer une page qui affichera une liste de « custom post type » qui ont une taxonomie « années » en particulier.

Taxonomie “categorie”

Tout comme la taxonomie « années », il est possible de créer une page dédiée à la taxonomie « catégorie ».

Il faut créer le fichier « taxonomy-categorie.php » pour y dispatcher l’affichage correspondant.

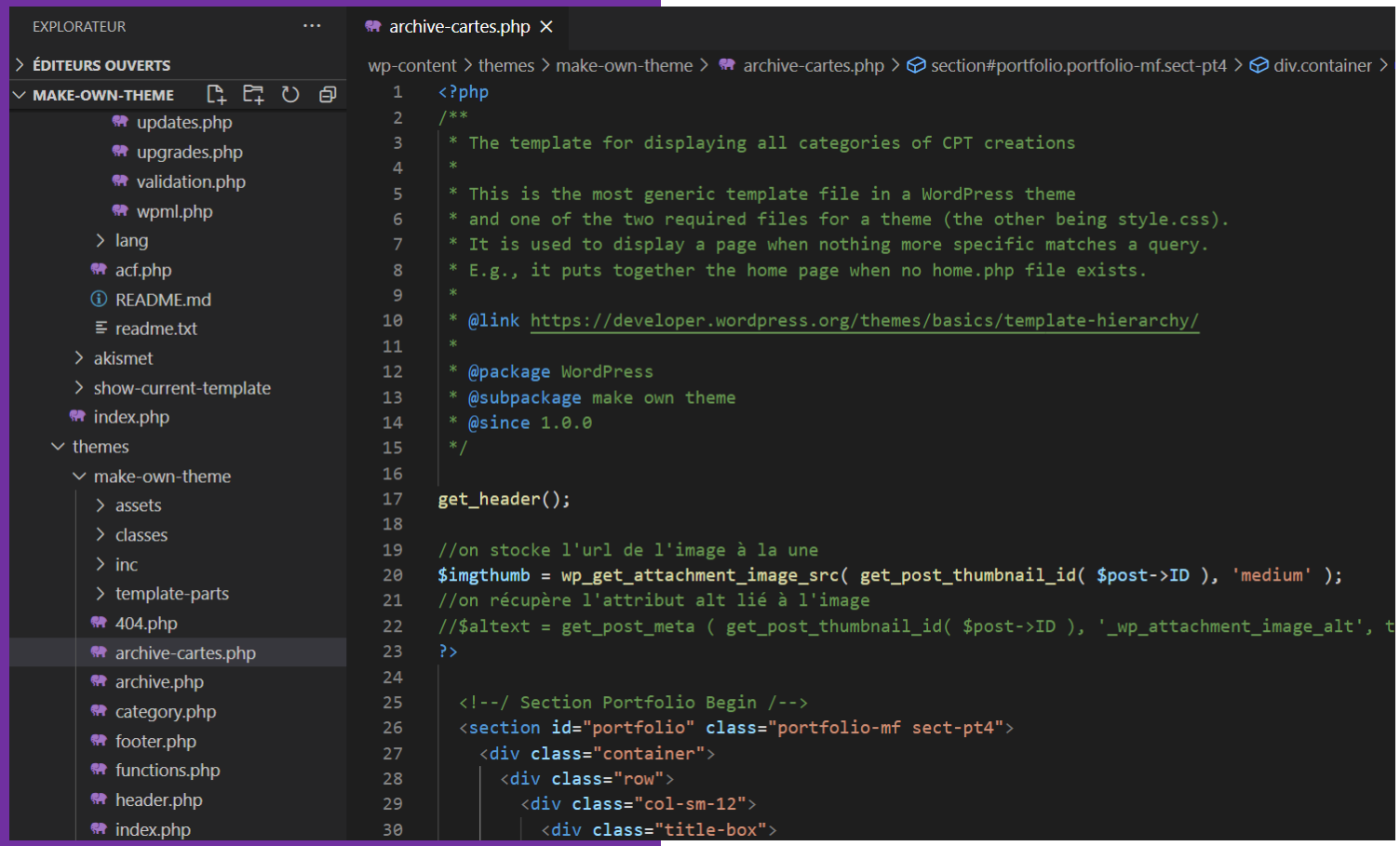
Dans la requête utilisée pour récupérer les articles correspondant à la taxonomie, il ne faut pas oublier de rajouter un paramètre pour celle-ci.



Wordpress

Création de la page archive-SLUG-CPT.php pour l’affichage de tous les items du post type

Page d’archive pour le custom post type



Pour mettre en place une page d’archives pour le « custom post type », il faut créer à la racine du thème le fichier « archive-cartes.php ».

Une fois mis en place, il ne faut pas oublier de mettre en place des filtres pour les différentes taxonomies.

Et pour finir, il faut utiliser la boucle de Wordpress pour afficher tous les articles concernés.

Wordpress

Mise en place d’une boucle spécifique permettant le filtrage des items du CPT en fonction des différents « Terms » d’une Taxonomie

Boucle filtrage custom posts en function d’une taxonomie

```php
$args = array(
'post_type'=> 'cartes',
'posts_per_page' => 5,
'orderby' => 'date',
'order' => 'DESC',
//'post_parent' => 0,
'paged' => $paged,
'tax_query' => array(
    array(
        'taxonomy' => 'annees',
        'field'    => 'slug',
        'terms'    => $catslug,
    ),
),
);
$loop = new WP_Query( $args );
```

Les « custom posts » peuvent être filtrer par taxonomie directement dans les arguments de la requête mise en place.

Il suffit de le préciser dans un tableau qui servira d’arguments.

Ici dans le « tax_query », c’est la taxonomie « annees » qui est appellée avec une catégorie définie en amont dans la variable « $catslug ».

Wordpress

Point sur la sécurité dans WordPress

Protéger “wp-admin”

Pour protéger au maximum la page de login vers l’administration de Wordpress, il est plus prudent de changer l’url d’accès vers celle-ci.

Grâce au plugin « change wp-admin login », la configuration se fait rapidement et efficacement.

Voici le lien vers cette ressource:

https://wordpress.org/plugins/change-wp-admin-login/

Réglages pour la sécurité

Depuis quelques versions maintenant, Wordpress propose un « état » de la sécurité sur le site. Il est bien sur essentiel de vérifier la sécurité rapidement.

Dans cette optique, il faut toujours vérifier si tous est à jour sur le site, surtout les plugins.

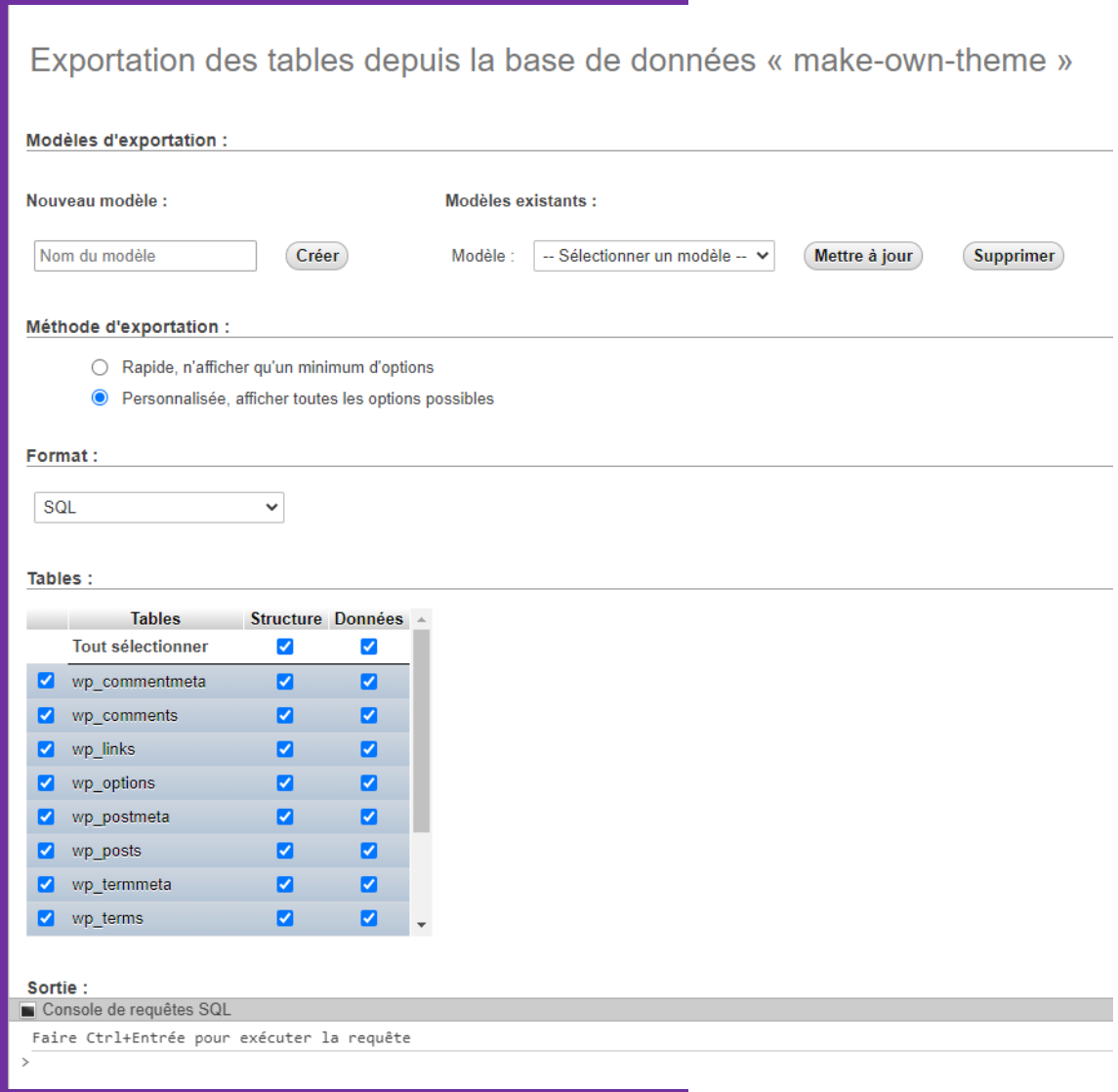
Lors du paramétrage de la base de données dans la configuration, il est important de préfixer celle-ci avec un autre préfixe que « wp ».

Il faut rester vigilant sur la création du mot de passe en préférant utiliser le générateur de mot de passe.

Wordpress

Préparation et migration du thème d’exercice de son environnement local vers un serveur de production via phpMyAdmin et FileZilla

Migration base de données



Avant de commencer la migration du site, il faut penser à exporter la base de données.

Allez sur la page suivante « localhost/phpmyadmin » sur votre navigateur puis sur la base de données concernée.

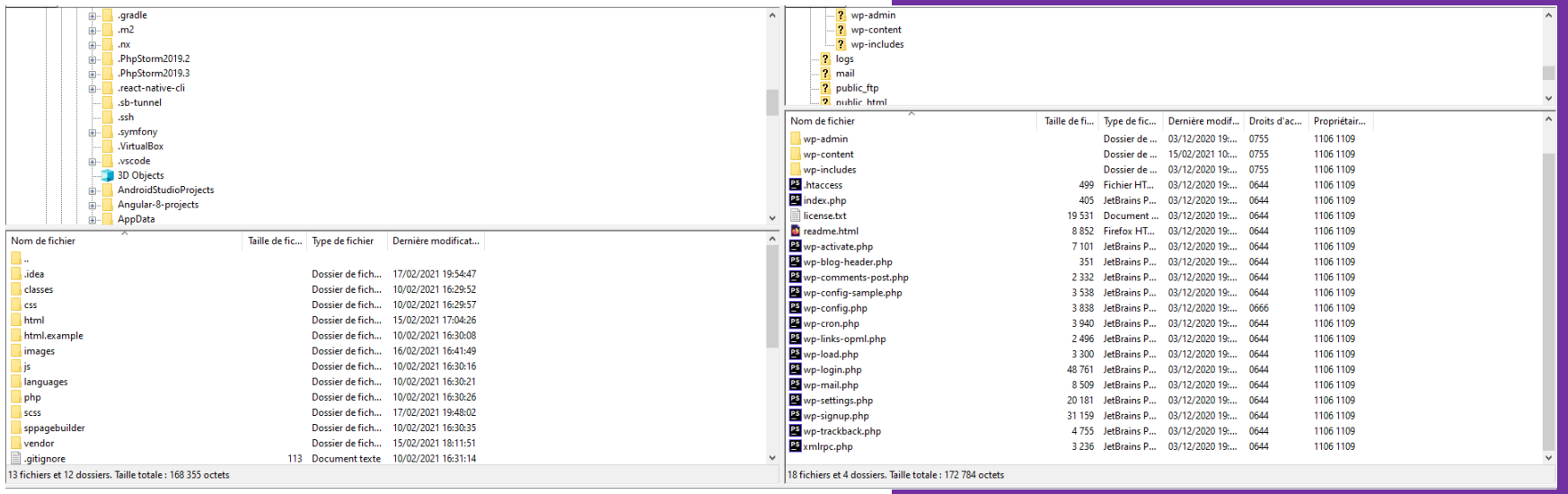
Il vous suffit ensuite de cliquer sur « exporter » pour avoir accès à l'exportation des tables depuis la base de données concernée.

Migration fichiers sources sur le serveur

Les fichiers contenu dans le dossier en local doivent être transférés en ligne dans le dossier qui correspond au site.

Le logiciel « Filezilla » permets de faire ses manipulations sur le serveur par un simple glisser/déposer.

Il ne faut pas oublier ses identifiants pour se connecter sur le serveur via ce logiciel.



Modifications de “wp-config.php” sur le serveur

```
/ ** Réglages MySQL - Votre hébergeur doit vous fournir ces informations. ** //
/** Nom de la base de données de WordPress. */
define( 'DB_NAME', 'votre_nom_de_bdd' );

/** Utilisateur de la base de données MySQL. */
define( 'DB_USER', 'votre_utilisateur_de_bdd' );

/** Mot de passe de la base de données MySQL. */
define( 'DB_PASSWORD', 'votre_mdp_de_bdd' );

/** Adresse de l’hébergement MySQL. */
define( 'DB_HOST', 'localhost' );

/** Jeu de caractères à utiliser par la base de données lors de la création des tables. */
define( 'DB_CHARSET', 'utf8' );
```

Il faut aussi changer les paramètres mis en place dans « wp-config.php » car ce sont ceux du site en local.

Pour les changer, il suffit de changer la deuxième valeur mise en paramètres dans les fonctions « define ».

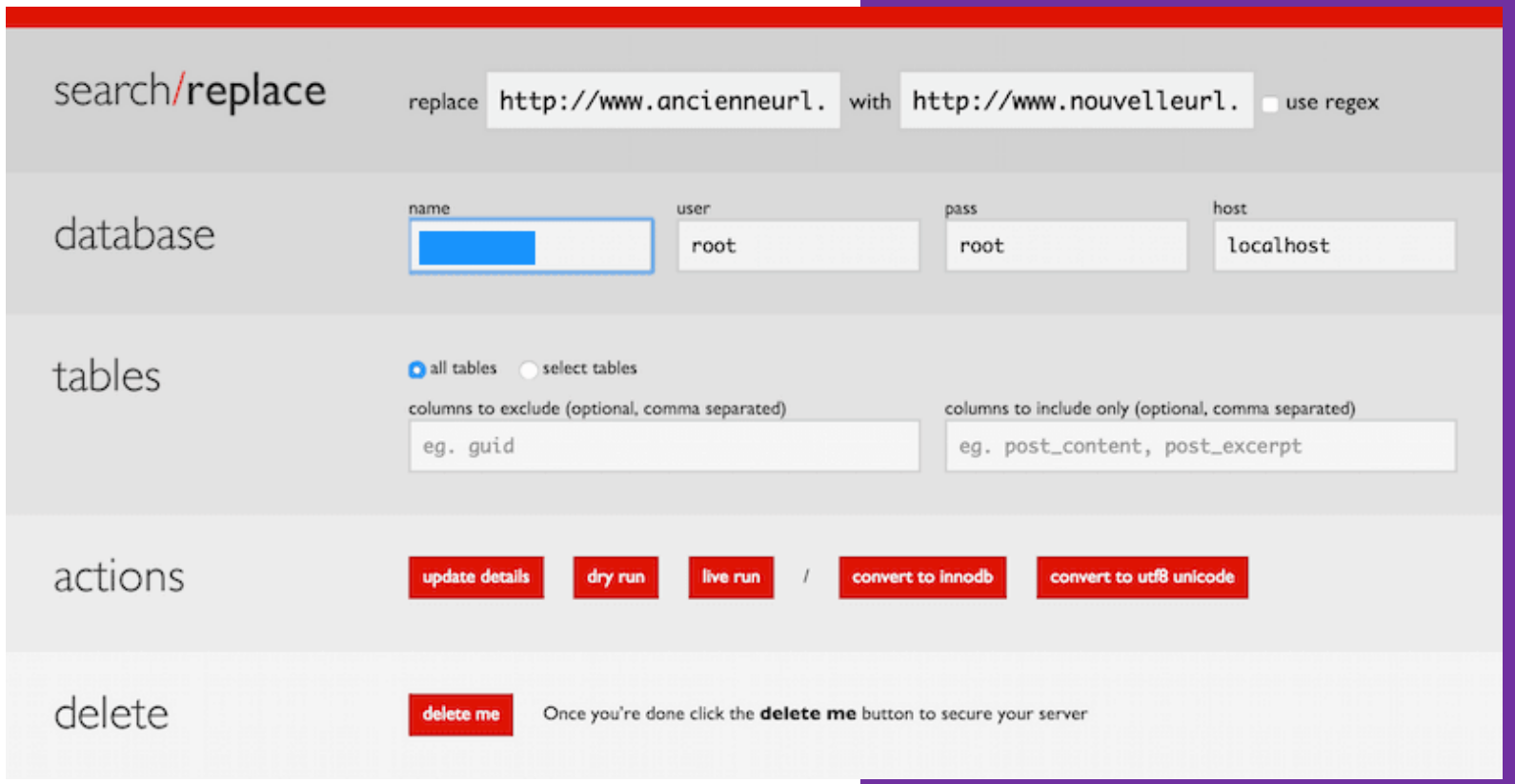
Vous trouverez ses valeurs chez votre hébergeur. Pour la base de données, il faudra la mettre en place sur l’hébergement et récupérer les identifiants lors de la création de celle-ci.

Mettre à jour les urls du site en ligne

Lorsqu'il faut enregistrer le changement des liens suite à un passage entre le site en local vers la production, c'est l'outil « srdb » qui est le plus utile.

Il permets de remplacer en base de données, les anciennes urls par la nouvelle correspondant au site en ligne.

Une fois que les changements sont faits, il faut le supprimer du serveur pour éviter des problèmes de sécurité.



Les derniers réglages

Maintenant que tout est mis en place sur le serveur, il faut scruter le site en ligne pour vérifier que tout marche correctement.

Il ne faut pas hésiter à tester tous les liens, les affichages, les pages, les ressources, etc...

Et pour terminer, faire un petit tour sur l'administration pour vérifier que tout est en régle comme l'état du site, les contenus déjà créés dans l'ancienne administration.