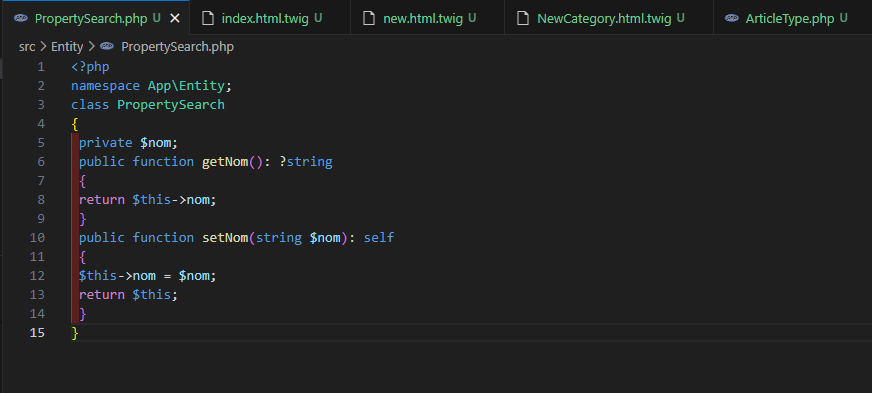
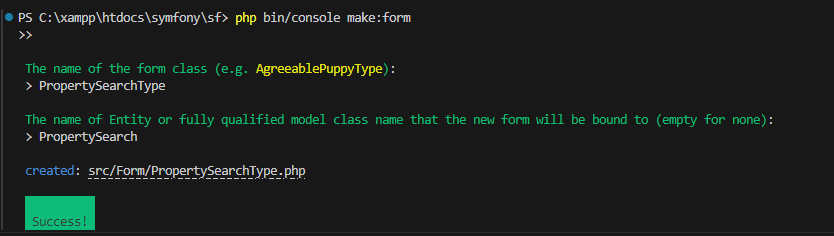
Compte rendu TP7

J'ai créé la classe PropertySearch dans le répertoire Entity de l'application Symfony. Cette classe a été mise en place sans recourir à la commande php bin/console make:entity. Elle sert à représenter un objet de recherche avec une seule propriété nom, qui permet de spécifier le nom d'un article à rechercher. La classe est conçue avec des méthodes getNom() et setNom() pour accéder et définir la valeur de cette propriété.

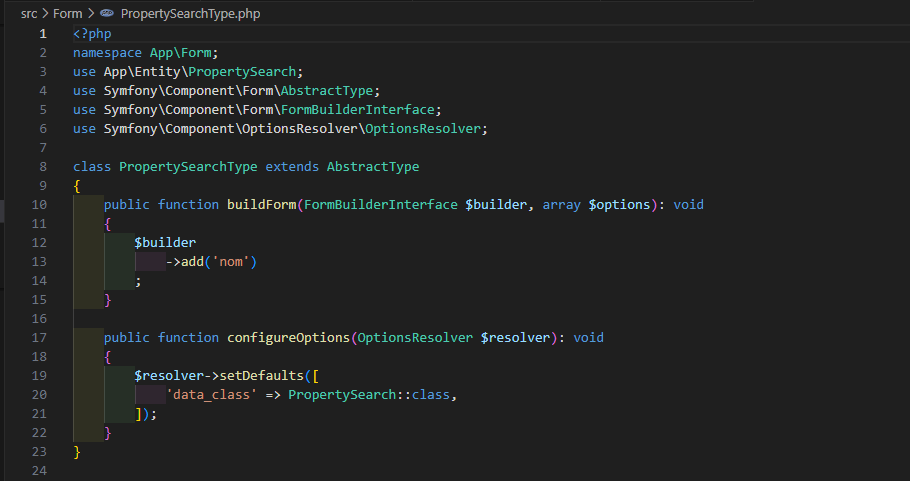


On Crée le formulaire PropertySearchType basé sur l’entité PropertySearch avec la commande

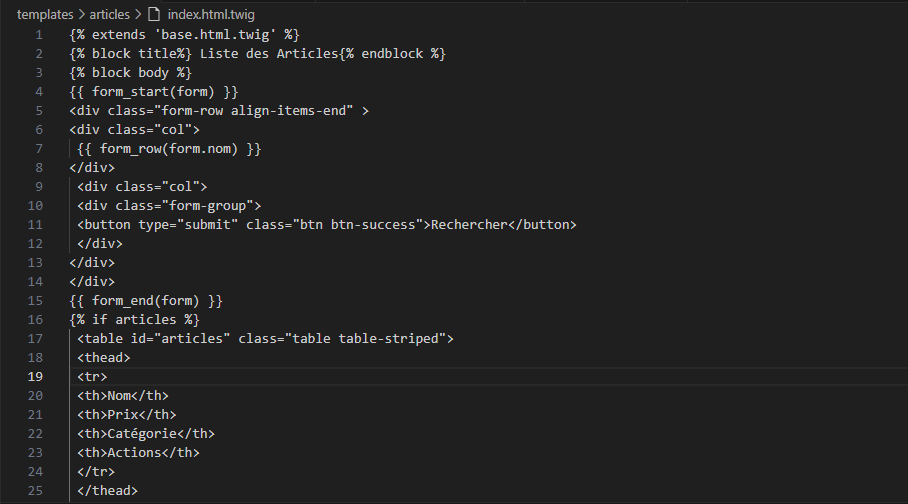
php bin/console make:form



La classe PropertySearchType est responsable de la création du formulaire associé à l'entité PropertySearch. Ce formulaire est conçu pour collecter les données de recherche, avec un champ unique pour le nom. En héritant de AbstractType de Symfony, la classe configure ce champ comme un champ de texte, permettant ainsi à l'utilisateur de saisir le nom de l'article qu'il souhaite rechercher. La liaison entre le formulaire et l'entité est établie grâce à OptionsResolver, garantissant que les données soumises seront correctement associées à l'entité PropertySearch.



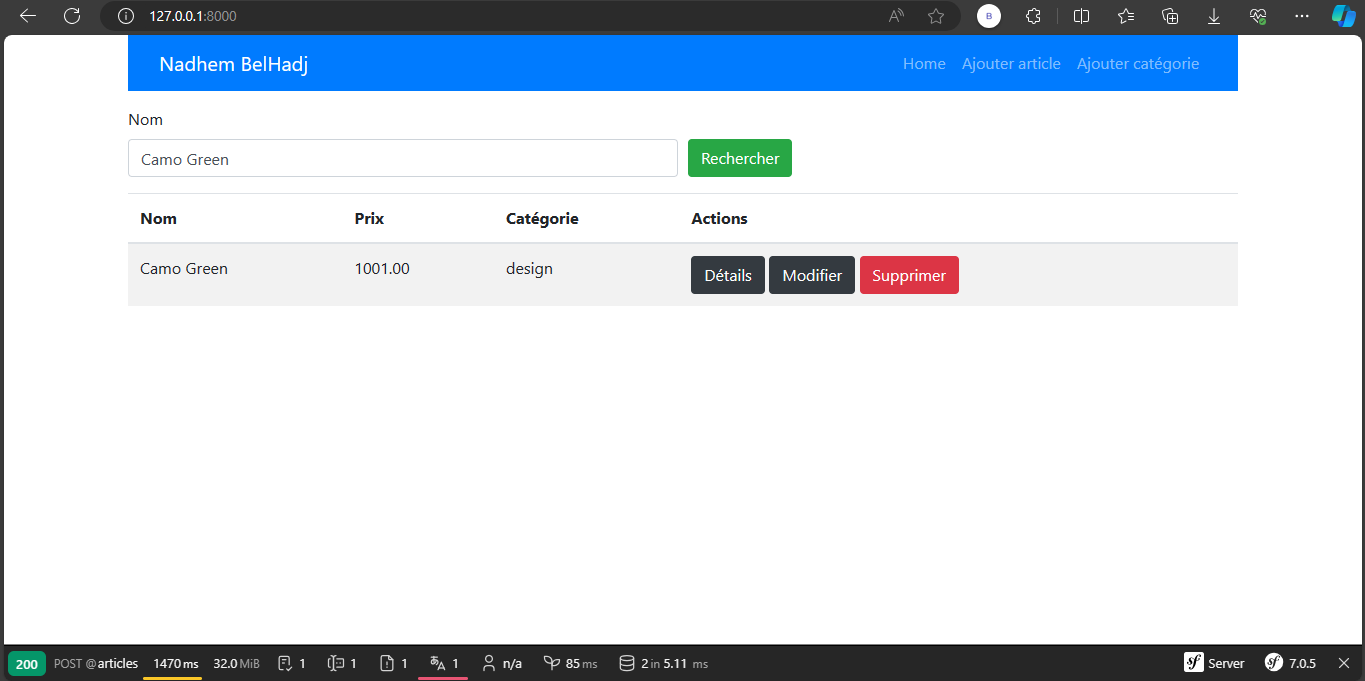
Le code ajouté au début du fichier **index.html.twig** permet de créer un formulaire de recherche d'articles par nom. Ce formulaire est rendu à l'aide de Twig en utilisant la variable **form** qui doit être fournie au template. Le formulaire comprend un champ de saisie pour le nom de l'article et un bouton de soumission "Rechercher". Lorsque l'utilisateur soumet le formulaire, il peut rechercher des articles en fonction du nom saisi.



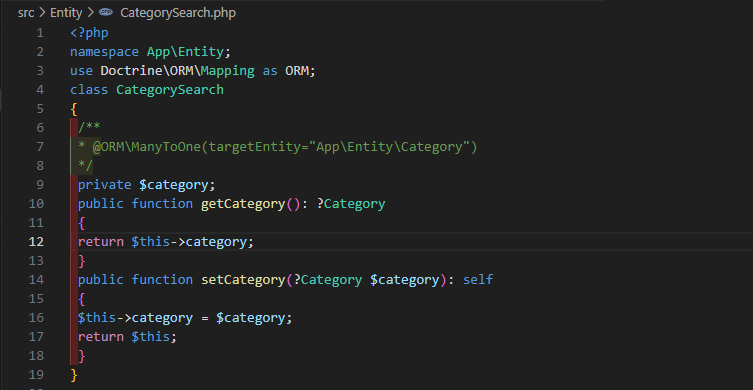
La fonction home() du contrôleur IndexController a été modifiée pour prendre en charge la recherche d'articles par nom. Elle utilise un formulaire de recherche basé sur l'entité PropertySearch pour collecter les critères de recherche. Lorsque le formulaire est soumis, la fonction vérifie s'il est valide et récupère les articles correspondants au nom saisi en utilisant la méthode findBy du gestionnaire d'entités Doctrine. Si aucun nom n'est fourni, elle récupère tous les articles à l'aide de la méthode findAll. Enfin, elle rend la vue articles/index.html.twig avec le formulaire et les articles récupérés.



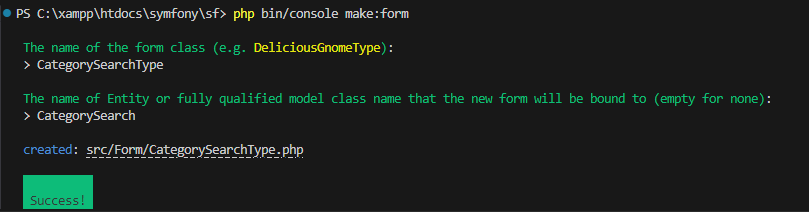
Si on veut chercher le nom du « Camo Green » il affiche :



La classe CategorySearch est une entité utilisée pour rechercher des articles par catégorie dans une application Symfony. Elle contient une seule propriété $category qui représente la catégorie à utiliser comme critère de recherche. Cette classe est utilisée pour créer des formulaires de recherche permettant aux utilisateurs de sélectionner une catégorie spécifique et de filtrer les articles en conséquence.



On crée le formulaire CategorySearchType basé sur l’entité CategorySearch en utilisant la commande php bin/console make:form



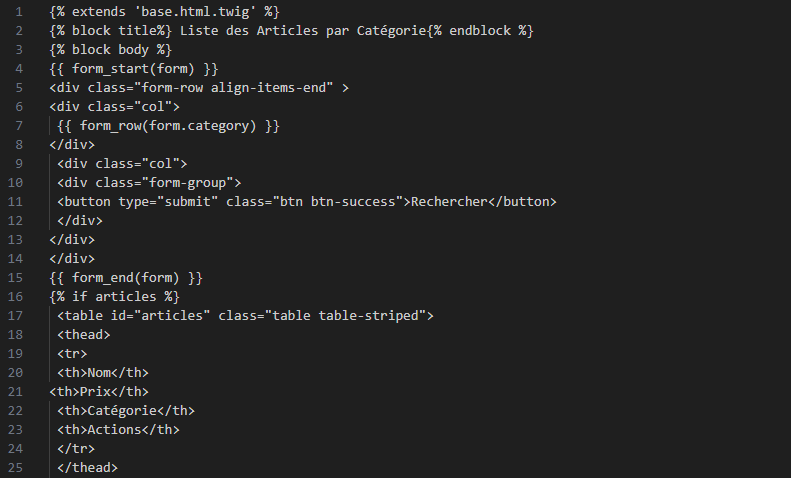
Le formulaire **CategorySearchType** est une classe Symfony permettant de créer un formulaire basé sur l'entité **CategorySearch**. Ce formulaire est utilisé pour rechercher des articles par catégorie. Il contient un champ de sélection pour choisir la catégorie désirée parmi une liste de catégories disponibles. Lorsque le formulaire est soumis, il envoie les données à traiter côté serveur pour effectuer la recherche d'articles correspondants à la catégorie sélectionnée.

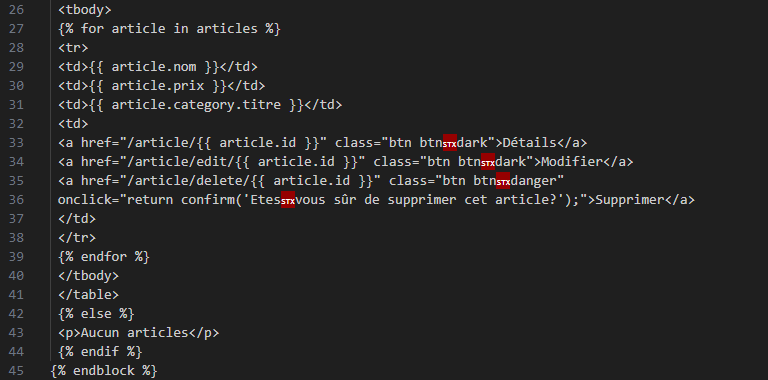


La fonction articlesParCategorie du contrôleur IndexController permet de rechercher et d'afficher des articles en fonction de la catégorie sélectionnée. Elle crée un formulaire de recherche basé sur l'entité CategorySearch en utilisant la classe CategorySearchType pour la construction du formulaire. Ensuite, elle gère la soumission du formulaire, récupère la catégorie sélectionnée et recherche les articles correspondants à cette catégorie. Si aucune catégorie n'est sélectionnée, tous les articles sont affichés. Enfin, elle renvoie le résultat vers le template articles/articlesParCategorie.html.twig pour affichage.

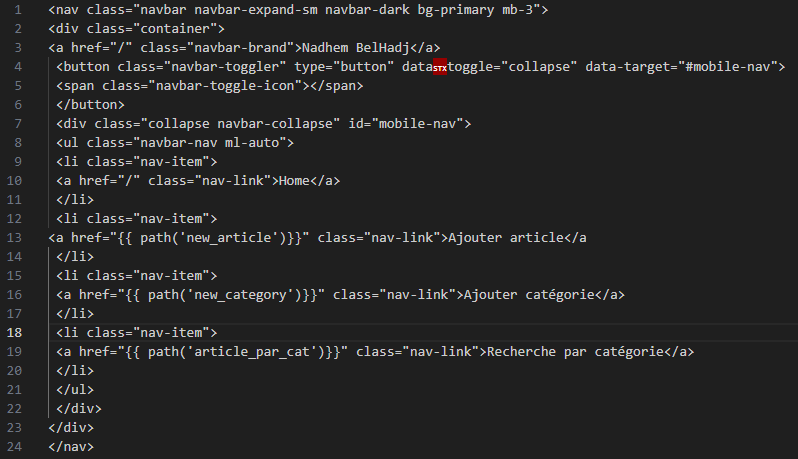


On va créer le fichier **articlesParCategorie.html.twig** qui est un modèle Twig qui affiche une liste d'articles filtrée par catégorie. Il comprend un formulaire pour sélectionner une catégorie, ainsi qu'un tableau présentant les articles correspondants à cette catégorie. Chaque article est affiché avec son nom, son prix et sa catégorie, et des liens sont fournis pour afficher les détails, modifier ou supprimer chaque article. Si aucun article n'est trouvé pour la catégorie sélectionnée, un message indiquant qu'aucun article n'est disponible est affiché.

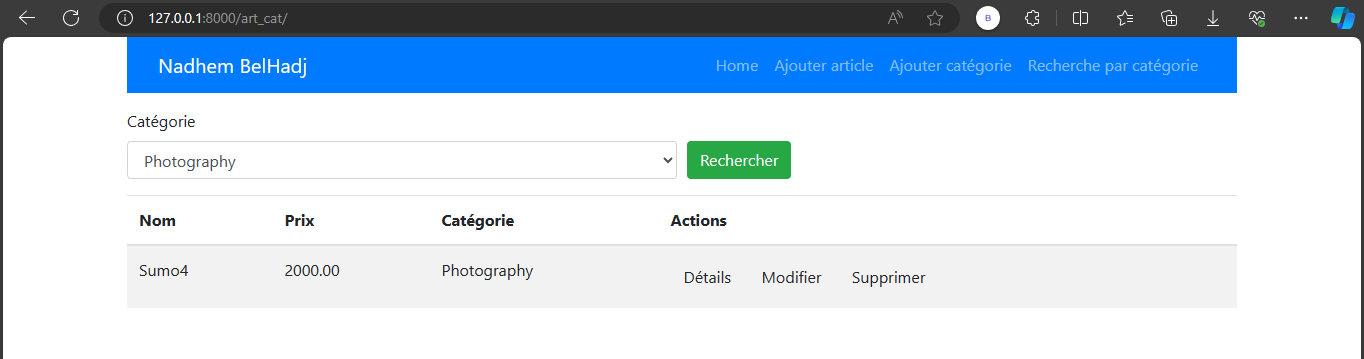




On ajouter un lien supplémentaire dans la barre de navigation (navbar) pour la recherche par catégorie. Ce lien permettra aux utilisateurs d'accéder facilement à la page de recherche des articles par catégorie. En cliquant sur ce lien ils pourront effectuer des recherches spécifiques d'articles en fonction de la catégorie choisie.

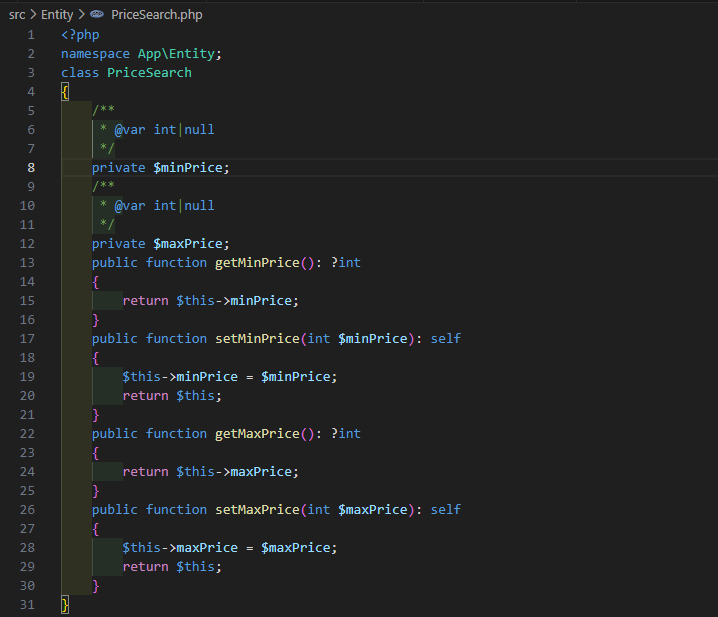


On vérifie par la recherche du catégorie ‘Photography’, il affiche :



On va Créer la classe PriceSearch dans Entity sans utiliser la commande make :entity .

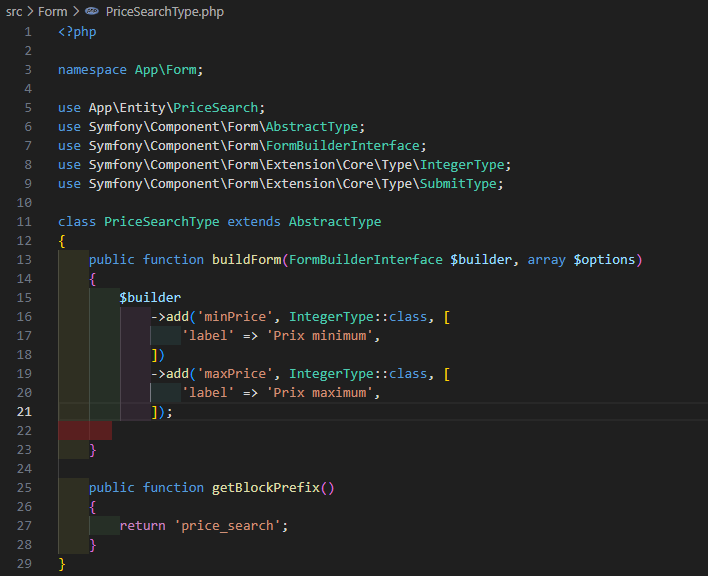
La classe PriceSearch est une entité PHP représentant un objet de recherche de prix. Elle contient deux attributs, minPrice et maxPrice, qui représentent respectivement le prix minimum et maximum des articles recherchés. Les méthodes getMinPrice() et getMaxPrice() permettent d'obtenir les valeurs de ces attributs, tandis que les méthodes setMinPrice() et setMaxPrice() permettent de définir ces valeurs. Cette classe est utilisée pour encapsuler les critères de recherche de prix dans le système d'information.



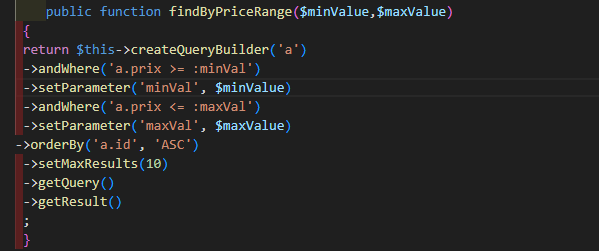
On va Créer le formulaire PriceSearchType basé sur l’entité PriceSearch en utilisant la commande

php bin/console make:form

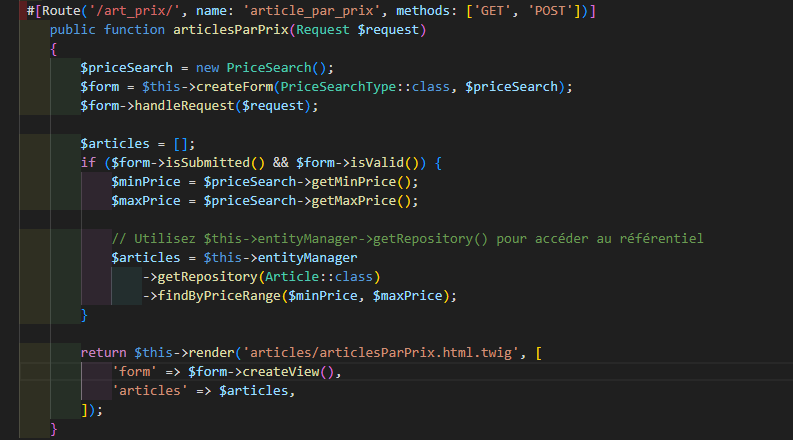
Le formulaire PriceSearchType est créé en se basant sur l'entité PriceSearch. Il comprend deux champs : minPrice et maxPrice, correspondant respectivement au prix minimum et maximum des articles à rechercher. Ces champs sont ajoutés au constructeur du formulaire à l'aide de la méthode add(). Enfin, les options par défaut du formulaire sont configurées pour utiliser la classe PriceSearch comme classe de données. Ce formulaire permettra aux utilisateurs d'entrer les critères de recherche de prix lors de la recherche d'articles.



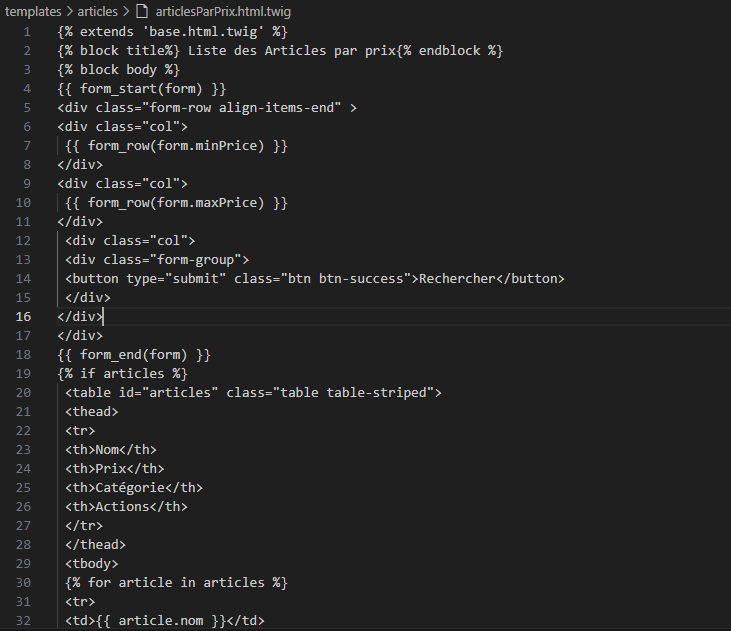
La fonction findByPriceRange est ajoutée à la classe ArticleRepository. Cette fonction permet de récupérer les articles dont le prix est compris entre deux valeurs spécifiées. Elle utilise QueryBuilder pour créer la requête en filtrant les articles en fonction des valeurs de prix minimum et maximum passées en paramètres. Ensuite, les valeurs des paramètres sont définies à l'aide de setParameter. Enfin, la méthode getResult() est utilisée pour exécuter la requête et retourner les résultats.

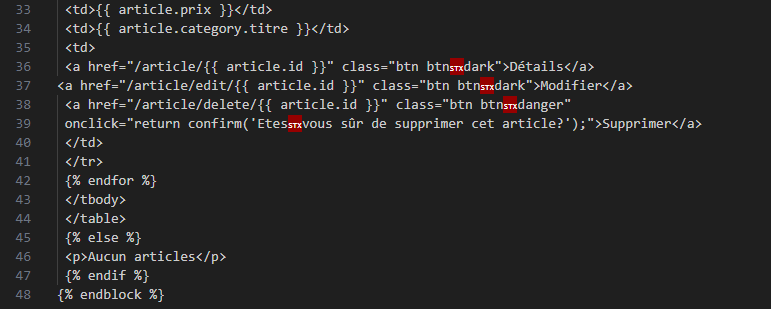


La fonction articlesParPrix est ajoutée à IndexController. Elle gère la route /art\_prix/ pour les méthodes GET et POST. Cette fonction utilise un formulaire de recherche de prix pour récupérer les articles correspondant à la plage de prix spécifiée par l'utilisateur. Ensuite, elle affiche les résultats dans la vue articlesParPrix.html.twig.

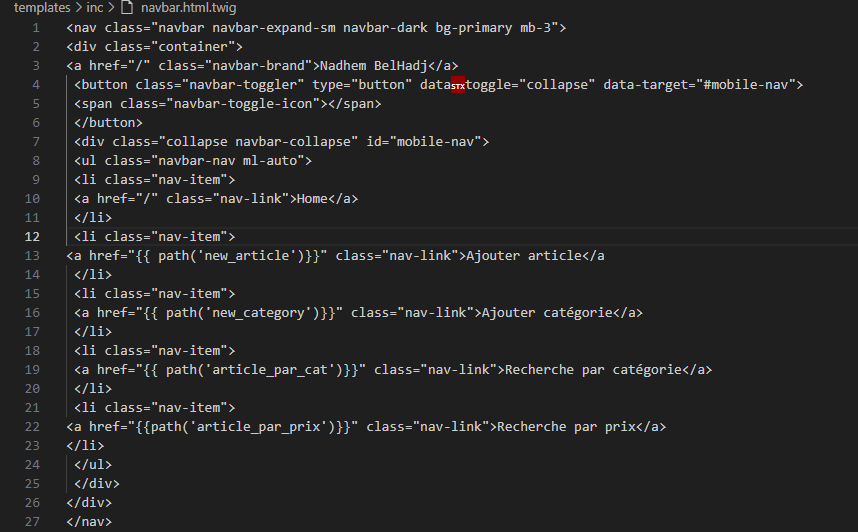


On crée le fichier articlesParPrix.html.twig dans articles qui est une vue qui permet aux utilisateurs de rechercher des articles par prix. Il affiche un formulaire pour spécifier une fourchette de prix, puis présente les articles correspondants dans un tableau. Chaque article est affiché avec son nom, son prix, sa catégorie et des options pour afficher les détails, le modifier ou le supprimer.





On ajoute un lien "Recherche par prix" dans le menu de navigation permet aux utilisateurs d'accéder facilement à la fonctionnalité de recherche par prix sur le site. En cliquant sur ce lien, les utilisateurs seront dirigés vers la page où ils pourront spécifier le prix pour rechercher des articles correspondants. Cela améliore l'expérience utilisateur en offrant une navigation fluide et intuitive. Voici le code complet du menu de la barre de navigation :



On vérifie notre application, on cherche les articles entre le prix 1000 et 2000.Il affiche :

