

DOSSIER DE PROJET

Yves Rodrigue EWOMBA-JOCTANE

Titre visé : Développeur Web et Web mobile

Studi

Application Web BiblioTIPPEE

Remerciements

Je tiens à exprimer ma profonde gratitude envers les personnes qui ont contribué de manière significative à la réalisation de mon projet d'études au titre de Développeur Web et Web mobile et à l'élaboration de ce rapport. Leur soutien inestimable et leur contribution ont été des facteurs déterminants dans la réussite de cette expérience professionnelle.

Je tiens à remercier tout particulièrement, Monsieur **Jean Pierre BOUSSOUGOU**, Secrétaire Permanent par Intérim du Secrétariat Permanent de la Commission Nationale des TIPPEE, pour son autorisation de réaliser cette formation au-delà de mes responsabilités de Chargé de Projet à la Commission, ce qui a été une opportunité précieuse dont je suis extrêmement reconnaissant.

Je voudrais également exprimer ma reconnaissance envers Monsieur **Sosthène MAMBANDI**, Responsable Informatique de la Commission, pour son encadrement attentif et ses conseils avisés tout au long du projet. Sa disponibilité et son expertise ont été d'une grande importance pour moi.

J'adresse également mes remerciements à l'ensemble du personnel de la Commission pour leur encouragement constant et leur soutien moral. Leur bienveillance et leur soutien ont été des sources d'inspiration et de motivation tout au long de mon immersion au service informatique, malgré mes nombreuses charges professionnelles.

Je souhaite également exprimer ma gratitude envers tous les apprenants de mon école et de ma promotion. Leurs aides multiformes ont joué un rôle crucial dans la résolution des difficultés rencontrées lors de l'avancement du projet. Leur soutien et leur collaboration ont été précieux.

Enfin, je tiens à remercier tous les enseignants qui m'ont prodigué leurs conseils techniques tout au long de la réalisation de cette application. Leurs connaissances et leur expertise ont été d'une grande valeur pour moi et ont contribué à mon développement professionnel.

Je tiens à remercier chaleureusement toutes ces personnes pour leur contribution, leur soutien et leur confiance. Leur engagement et leur appui ont été essentiels pour la réussite de ce projet d'études.

SOMMAIRE

Remerciements	2
Sommaire	3
Introduction	4
1. Compétences du référentiel couvertes par le projet	4
2. Résumé du projet.....	5
3. Environnement humain et technique.....	6
Définition du projet.....	8
1. Le cahier des charges	8
2. Technologies Requises	9
3. Les spécifications techniques	11
4. Eléments conceptuels	13
La réalisation du projet	15
1. Conception de la base de données.....	15
2. Développement de la partie front-end.....	23
Fonctionnalités significatives	24
1. L'authentification	24
2. Le changement de mot de passe ou mot de passe oublié	24
Conclusion	25

Introduction

1. Compétences du référentiel couvertes par le projet

a. 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.

L'application web BibioTIPPE est fonctionnelle sur tous les navigateurs connus aussi bien sur des écrans au format desktop comme sur les écrans au format mobile. Ultérieurement, selon le calendrier de livraison de l'application, une version mobile pour la consultation des documents sera réalisée. Notons, que le front fait

b. 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.
- Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce.

J'ai donc opté pour l'utilisation du framework Symfony 7.0 pour la partie back-end de l'application BibioTIPPEE pour bénéficier des fonctionnalités de sécurité avancées qu'offre ce dernier. En effet, il met un accent particulier sur la sécurité et propose des composants dédiés à la protection des applications web, tels que la gestion sécurisée des mots de passe, le chiffrement des données sensibles et la protection contre les attaques CSRF (Cross-Site Request Forgery). De plus, Symfony 7.0 présente des avantages tels que la modularité, la flexibilité, la productivité accrue et de non-moindre le soutien d'une communauté active. Ainsi, pour la création des tables de la base de données liées aux entités PHP, j'ai utilisé Doctrine, l'ORM (Object Relation Mapping) de Symfony 7.0 qui présente des avantages clés lors de la création

des tables, notamment : (i) la génération automatique du schéma de base de données ; (ii) la productivité accrue ; (iii) la correspondance directe entités-tables, (iv) la migration automatisée ; etc.

2. Résumé du projet

Le client est la Commission Nationale des Travaux d'Intérêt Publics pour la Promotion de l'Entrepreneuriat et de l'Emploi (CNTIPPEE) qui est une entité publique du Ministère Gabonais de l'Economie et des Participations. Elle cherche à optimiser la gestion de ses documents.

C'est pourquoi, elle a besoin de mettre en place une solution numérique pour l'archivage de ses documents. En effet, elle génère un grand nombre de documents PDF qui doivent être archivés de manière efficace, efficiente et sécurisée.

Ainsi, elle désire faire développer une application ayant des fonctionnalités comprenant la possibilité d'uploader des documents sous format PDF, de les classer par catégories, par années, par projets, par natures (financiers, techniques, sauvegardes environnementales et sociales, passation de marchés, etc.), de les rechercher par mots-clés, de les consulter en ligne et/ou de les télécharger. L'application doit également garantir la sécurité et l'inviolabilité des documents archivés.

Pour ce faire, les technologies à employer pour le développement de cette application seront les langages de programmation Node.JS et/ou PHP pour le backend, le framework JavaScript React pour le frontend, et la base de données POSTGRESQL pour le stockage des documents. Le client propose optionnellement la possibilité de réaliser entièrement cette application en PHP/Symfony v7.0 pour des questions de diligence.

A noter que, le contexte du développement est un environnement de travail agile, avec un sprint toutes les deux semaines et des réunions hebdomadaires de suivi avec le point focal en charge des archives physiques.

Enfin, la durée du développement est estimée à quatre (4) mois, en tenant compte des phases de conception, de développement, des tests d'acceptances et de déploiement.

3. Environnement humain et technique

a. Environnement humain

Dans le cadre de ma formation de Développeur Web - Web Mobile, je réalise à temps partiel une immersion au sein du service informatique de la Commission Nationale des TIPPEE, qui est également mon employeur dans le domaine de la gestion fiduciaire des projets sur financement extérieur.

L'idée d'élaborer une application Web, voire également Mobile, qui permettrait d'une part l'archivage par « upload » des documents au format PDF par l'administrateur, et d'autre part, le "téléchargement" par les utilisateurs de l'application, est issue d'un besoin identifié dans le fonctionnement de la Commission.

Depuis le mois de février 2024, je suis donc en immersion partielle au service informatique, travaillant en collaboration avec le responsable de ce service, Monsieur Sosthène MAMBANDI. Notre objectif premier a été d'identifier et d'analyser les besoins en fonctionnalités pour répondre aux différents cas d'usage.

N'ayant pas de UX designer dédié, j'ai dû acquérir cette compétence par moi-même en suivant des tutoriels et en lisant des ressources afin de pouvoir développer des interfaces conviviales et adaptatives pour tous les types d'écrans. Cet apprentissage de l'UX m'a permis de prendre conscience du rôle et de l'importance d'un UX/UI designer dans la phase conceptuelle du projet BiblioTIPPEE.

De plus, les retours de mes collègues de bureau ont été essentiels pour évaluer la qualité du design de l'application.

En somme, je suis convaincu que les contributions en termes de design et les fonctionnalités prioritaires retenues par mon groupe interne de testeurs ont été déterminantes pour la réalisation des premières fonctionnalités. Leur apport a été précieux pour garantir la pertinence et l'utilité de cette application.

b. Environnement Technique

Dans le but de réaliser l'application BiblioTIPPEE, je travaille sur un ordinateur portable afin de me permettre d'être mobile, c'est-à-dire de pouvoir travailler en bureau mais aussi à la maison pour pouvoir tenir les délais impartis pour obtenir une première version en pré-déploiement.

La Commission Nationale de TIPPEE dispose d'un environnement technique favorable car au regard de mes fonctions en son sein, je dispose d'un large bureau équipé en moyen informatique et d'une connexion internet très haut débit.

Je dispose aussi de deux écrans de plus de 24 pouces pour me permettre de travailler sur plusieurs écrans à la fois me facilitant ainsi l'apprentissage des différentes technologies par les biais des différentes plateformes d'apprentissage de code y compris celle de Studi, mon école de formation.

Par ailleurs, le réseau informatique de la Commission étant également superviser par l'Agence Nationale des Infrastructures Numériques du Gabon, il m'est arrivé au cours de leur passage de supervision de nos serveurs, d'échanger avec ces derniers pour obtenir leur ressenti sur la qualité technique de mon code et quelques conseils d'experts avisés. Je peux affirmer que ces exercices de questionnement m'ont grandement aidé à mieux organiser mon code en plus des recommandations d'usage que l'on retrouve dans les forums spécialisés.

A noter que la Commission est dotée d'un serveur de fichiers de grande capacité sur quoi l'application BiblioTIPPEE s'appuie pour obtenir un jeu de données pour les phases de tests.

Enfin, l'environnement technique est également complété par :

- Langages de programmation : J'utilise PHP pour le back-end et JavaScript pour certaines parties du front-end de l'application BiblioTIPPEE.
- Framework : Pour faciliter le développement, j'utilise le framework Symfony pour le back-end et React pour le front-end. A noter qu'une combinaison des deux langages est envisagée également pour limiter les rechargements des pages en partie front pour une meilleure expérience utilisateur.
- Base de données : J'utilise PostgreSQL comme système de gestion de base de données pour stocker les informations de l'application.
- Outils de gestion de documents PDF : Pour manipuler les documents PDF, j'utilise des bibliothèques et des outils tels que PDF.js et pdf-lib.
- Système d'exploitation : L'application est déployée sur un serveur Linux hébergé sur la plateforme Horoku.

- Outils de développement : Pour le développement, le débogage et le test de l'application, j'utilise l'IDE VS CODE, GIT pour le versioning et les outils de test de Symfony.

De plus, l'hébergement et les services d'équilibrage de charge sont administrés par la plateforme Heroku.

Définition du projet

1. Le cahier des charges

Le projet de développement d'une application de gestion de documents au format PDF est une initiative visant à créer un logiciel qui facilite la manipulation, l'organisation, le stockage et la récupération de documents PDF. Cette application n'existe pas encore, donc nous partirons de zéro mais une fois réalisée, elle offrira des fonctionnalités telles que :

a. Importation de documents PDF

Les utilisateurs pourront importer des documents PDF dans l'application depuis leur appareil local ou depuis des sources en ligne.

b. Organisation des documents

L'application permettra aux utilisateurs autorisés de classer et d'organiser leurs documents PDF en fonction de divers critères, tels que la date, le sujet, l'auteur, etc.

c. Recherche et récupération de documents

Les utilisateurs pourront rechercher des documents spécifiques en utilisant des mots-clés ou des filtres, et récupérer ces documents pour consultation ou modification.

d. Partage de documents

Les utilisateurs pourront partager des documents PDF avec d'autres utilisateurs de l'application, ou les exporter pour les partager en dehors de l'application.

e. Sécurité des documents

L'application assurera la sécurité des documents PDF en utilisant des protocoles de cryptage et en offrant des options par exemple pour la protection par mot de passe pour l'accès à un certain type des documents.

2. Technologies Requises

a. Frontend

- Utilisation de React.js intégrée dans du code PHP pour la création d'une interface utilisateur interactive et réactive sans rechargement de page.
- Intégration éventuelle de Redux pour la gestion de l'état global de l'application.
- Utilisation de HTML5, CSS3 et JavaScript pour le développement des composants frontend.
- Utilisation de bibliothèques telles que Material-UI pour des composants d'interface utilisateur.

b. Backend :

- Utilisation du framework Symfony 7.0 pour le développement du serveur backend.
- Utilisation de PHP pour la logique métier et l'interaction avec la base de données.
- Intégration de Doctrine ORM pour faciliter l'accès et la manipulation des données dans la base de données.

c. Consultation en Ligne

- Intégration de bibliothèques telles que PDF.js pour la visualisation et la manipulation de documents PDF dans le navigateur.

d. Gestion des Documents

- Utilisation de Symfony 7.0 et Doctrine ORM pour le stockage des documents et des métadonnées associées.
- Intégration de fonctionnalités de téléchargement pour permettre aux utilisateurs de télécharger des documents dans différents formats.

e. Authentification et Autorisation

- Utilisation de Symfony qui fournit tous types de fonctions d'authentification et d'autorisation nécessaires pour sécuriser mon application.
- Utilisation de requêtes HTTP liés aux outils de sécurité, comme les cookies de session sécurisés et la protection CSRF est assurée par défaut par Symfony à travers le bundle « SecurityBundle.
- Utilisation d'un formulaire de connexion pour l'authentification des utilisateurs à l'aide d'un identifiant composé d'une adresse e-mail et d'un mot de passe.
- Mise en place de stratégies d'autorisation basées sur les rôles pour contrôler l'accès aux fonctionnalités de l'application.

f. Sécurité de l'Application et de la Base de Données

- Mise en place de mesures de sécurité telles que la validation des entrées utilisateur pour prévenir les attaques par injection de code malveillant.
- Utilisation de protocoles de cryptage tels que HTTPS pour sécuriser les communications par API entre le frontend et le backend.
- Mise en place de pare-feu et de mécanismes de sécurité au niveau du serveur pour protéger l'application contre les attaques.
- Utilisation de techniques de hachage pour sécuriser les mots de passe des utilisateurs dans la base de données.
- Mise en œuvre de sauvegardes régulières de la base de données pour prévenir la perte de données en cas de problème.

3. Méthodologie de Développement

Utilisation de la méthode Agile pour la gestion du projet, avec des itérations courtes et une planification flexible basée sur les épics et les « users stories » (voir mon TRELLO en annexe 1).

En conclusion, ce cahier des charges vise à définir les technologies et les approches de développement nécessaires pour la réalisation de l'application BiblioTIPPEE en tenant compte des fonctionnalités requises et des exigences des utilisateurs.

4. Les spécifications techniques

Pour rappel, en intégrant des mesures de sécurité solides à la fois pour l'application et la base de données, je peux garantir la confidentialité, l'intégrité et la disponibilité des données des utilisateurs.

En effet, tout ce dispositif m'a permis de créer une application web conviviale, offrant des fonctionnalités de consultation en ligne, de gestion des documents et d'authentification sécurisée, tout en suivant les principes de développement Agile pour une flexibilité et une réactivité maximale.

Aussi, les spécifications techniques ci-dessous pour l'application de gestion de documents au format PDF (BiblioTIPPEE) ont été discutées et approuvées par le client.

a. Environnement de travail

L'application BiblioTIPPEE sera développée sur un ordinateur portable, permettant une mobilité pour travailler à la fois en bureau et à domicile.

Un bureau équipé de moyens informatiques et d'une connexion internet très haut débit sera disponible pour faciliter le développement.

Deux écrans de plus de 24 pouces seront utilisés pour faciliter le travail sur plusieurs écrans simultanément.

L'outil de versionning Git, qui me permet de développer les fonctionnalités sur une branche sera utilisé et en plus le code de mon application sera déposé dans un dossier de Github après chaque modification majeure par un commit.

Le logiciel Postman me permet d'effectuer des tests de fonctionnement de mes API durant leur développement.

Des échanges avec les experts de l'Agence Gabonaise d'Infrastructure Numérique pour obtenir des conseils et des retours sur la qualité du code seront effectués.

b. Frontend :

L'interface utilisateur de l'application sera développée en utilisant les technologies web modernes telles que HTML, CSS, Bootstrap, Sass et JavaScript.

La framework React sera utilisés pour faciliter le développement de l'interface utilisateur sans rechargement de pages.

Les fonctionnalités d'affichage, de recherche et de manipulation de documents PDF ont été implémentées dans le frontend.

Une attention particulière sera accordée à la convivialité de l'interface utilisateur pour assurer une expérience utilisateur intuitive et agréable.

c. Backend :

Le backend de l'application sera développé en utilisant le langage de programmation PHP et son Framework Symfony 7.0.

Une base de données PostgreSQL sera utilisée pour stocker les métadonnées des documents PDF et faciliter les opérations de recherche.

Des API développé avec Symfony ou des bibliothèques spécifiques pour la manipulation des fichiers PDF, telles que PyPDF2 ou iText, pourront être utilisées.

Des mécanismes de sécurité Symfony seront mis en place nativement lors de l'initialisation du projet pour protéger les documents et assurer la confidentialité des données.

L'ORM (Object Relational Mapping) Doctrine sera utilisé pour faciliter la création et la gestion des tables devant contenir les données de l'application.

Des fonctionnalités telles que l'ajout, la suppression et la modification de documents seront implémentées dans le backend.

5. Éléments conceptuels

a. La charte graphique, les fonts et les maquettes

L'application BiblioTIPPEE de gestion de documents PDF nécessite une charte graphique, des fonts et des éléments visuels attrayantes et cohérentes pour offrir une expérience utilisateur agréable et professionnelle.

- Charte graphique :

Headline Text
Font : Sofia-Pro

Primary

code : 3E3E3F

Text content
Font : Work sans

Secondary

code : F5F8FF

Ceci est un bouton

code : 28A745

[Ceci est un lien](#)

code : 5A49F8

bg-paire

code : F3EFEF

- Typographie :

Comme typographie, j'ai utilisé :

- Pour les entêtes, la font : Sofia-Pro
- Pour le corps, la font : Work sans

- Logo et icônes :

Voir charte graphique ci-dessus.

- Mise en page :

Voir maquettes en version desktop et mobile en annexe 4.

b. Les EPICs et Users Stories

Initiative de développement / Commande Client

Elaboration d'une application Web ou Web mobile dont l'objectif est la consultation en ligne et/ou le téléchargement de documents de différentes natures au format PDF contenus dans une base de données hébergée dans un serveur de fichiers

N° Ordre	EPIC	Observations
1	Épic 1 : Gestion des utilisateurs et des autorisations	Cette fonctionnalité permettrait de gérer les utilisateurs de l'application, d'attribuer des autorisations spécifiques pour la consultation ou le téléchargement de documents, et de garantir la sécurité des données.
2	Épic 2 : Interface utilisateur conviviale	Cette fonctionnalité se traduit par la conception d'une interface utilisateur intuitive et conviviale pour faciliter la navigation et l'utilisation de l'application. Elle inclura des fonctionnalités telles que la recherche de documents, la catégorisation des documents, et la possibilité de les trier ou de les filtrer.
3	Épic 3 : Gestion des documents	Cette fonctionnalité permettrait de gérer les documents stockés dans la base de données PostgreSQL. Cela pourrait inclure des fonctionnalités telles que l'ajout de nouveaux documents, la modification des métadonnées des documents, et la suppression des documents obsolètes.
4	Épic 4 : Fonctionnalités de consultation en ligne	Cette fonctionnalité permettrait aux utilisateurs de consulter les documents en ligne, sans avoir besoin de les télécharger. Cela pourrait inclure des fonctionnalités telles que la visualisation des documents dans le navigateur, la possibilité de zoomer ou de faire défiler les pages. Mais aussi une fois authentifié, permettre la prise de commentaires et la notation de la qualité de la fonctionnalité.

5	Épic 5 : Fonctionnalités de téléchargement	Cette fonctionnalité permettrait aux utilisateurs de télécharger les documents sur leur appareil local. Cela pourrait inclure des fonctionnalités telles que le choix du format de téléchargement, la compression des fichiers, et la possibilité de télécharger plusieurs documents simultanément.
---	--	---

La réalisation du projet

1. Conception de la base de données

Dès l'acceptation par le Client du cahier de charges et des spécifications techniques, j'ai donc procédé au démarrage des phases de collecte d'informations, de conception et de planification des activités de développement de l'application BiblioTIPPEE.

Les questions posées ont été sans être exhaustif : quelles sont les entités à créées pour l'application ? Quelles sont les acteurs du système ? Quels sont les types de documents à administrer et archiver dans la base de données ?

En réponse à ces questions, j'ai donc eu à réaliser dans un premier temps un diagramme des cas d'utilisation (cf. annexe 2) qui m'a permis de réaliser l'analyse complète des interactions entre les acteurs externes et internes du système que constitue l'application BiblioTIPPEE.

Ainsi, j'ai donc utilisé la méthode UML, méthode d'analyse et de conception de systèmes d'informations similaire à la méthode MERISE pour la conception des bases de données.

J'ai ensuite réalisé le diagramme des Class (cf. annexe 3) en définissant les relations entre les entités en précisant les cardinalités. Notons que pour les relations ayant des cardinalités $*...1$ — $*...n$, la clé primaire de la table côté $*...n$ devient une clé étrangère de la table côté $*...1$ créant ainsi la relation entre les tables. S'agissant des relations "Many to Many", représentées par des cardinalités $*...n$ — $*...n$, elles génèrent une nouvelle table qui comportera des champs correspondant aux id de chacune des 2 entités.

Etant donné, que j'ai fait le choix d'utiliser Symfony qui dispose de **Doctrine ORM** et **Doctrine Migration**, c'est cette dernière qui va générer tous les scripts SQL de création des tables représentants chaque entité après avoir effectué les migrations nécessaires de chaque entité. En effet, une *migration* est une classe qui décrit les changements nécessaires pour mettre à jour un

schéma de base de données (voir annexe 5), de son état actuel vers le nouveau, en fonction des attributs de l'entité.

a. Mon diagramme de Cas d'utilisation

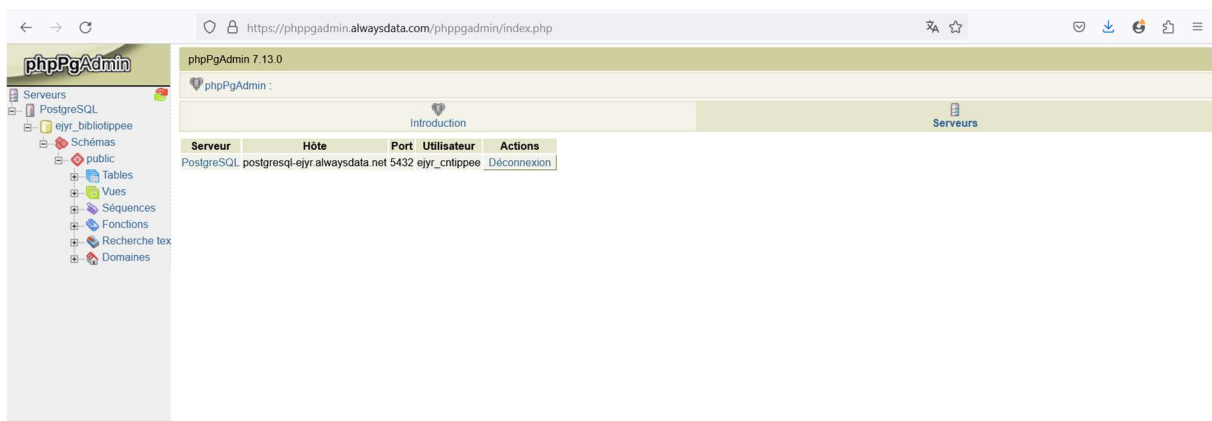
Voir document en annexe 2.

b. Mon diagramme de Class

Voir document en annexe 3.

c. Installation et initialisation de la base de données PostgreSQL

En première action, je me suis appuyé de la plateforme <https://www.alwaysdata.com>, qui est un service en ligne entre autres d'hébergement de base de données, pour créer la base de données au nom de « ejyr_bibliotippee » puis s'en est suivi la création d'un utilisateur principal pour accéder à cette base de données et toujours depuis la même plateforme.



Cette dernière recevra toutes les tables qui seront créées après migrations par Doctrine en fonction de mes entités identifiées dans mon diagramme de Class.

Pour créer le lien entre mon application BiblioTIPPEE et ma base de donnée, j'ai modifié le contenu du fichier « .env » de mon application en y précisant les informations de connexion nécessaires. Soit :

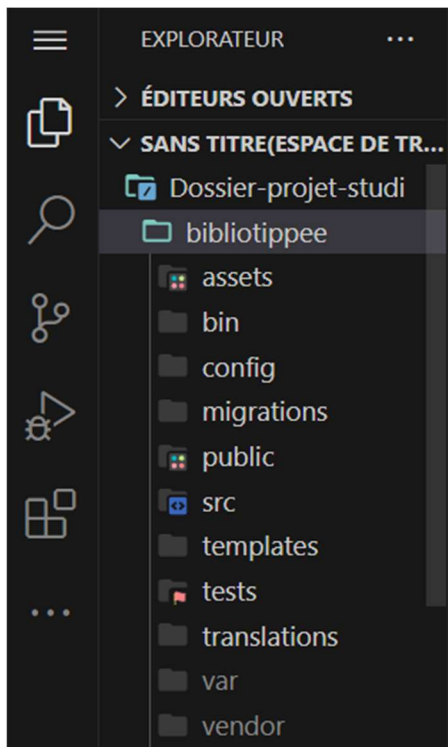

```
DATABASE_URL="postgresql://ejyr_cntippee:!ChangeMe@127.0.0.1:5432/postgresql-ejyr.alwaysdata.net?serverVersion=14&charset=utf8"
```

d. Initialisation du projet BiblioTIPPEE

Je crée un nouveau projet Symfony avec la commande symfony ci-dessous après avoir installé le framework Symfony :

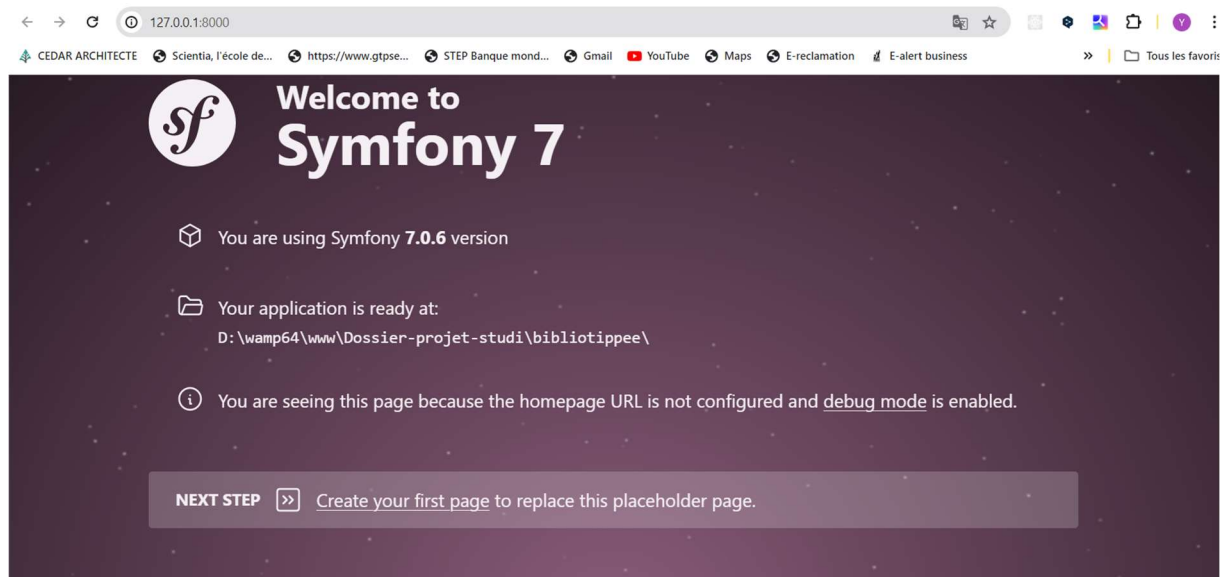
```
>> symfony new bibliotippee --version=7.0 --php=8.2 --webapp
```

Ci-dessous la structure des répertoires qui ont été créés :



À partir du répertoire du projet, je démarre le serveur web en arrière-plan (option -d) :

```
>> symfony server:start -d
```



Le serveur a démarré sur le port 8000. La page d'accueil de Symfont 7 ci-dessus m'indique que l'initialisation de mon projet est correcte et sans bug. Je réalise ensuite mon premier commit en réalisant les commandes :

```
>> git commit -m "Initialisation de mon application"
```

Notons, que j'ai choisi Git et Github comme système de versionning et d'archivage de mon code applicatif. Et, afin de pouvoir accéder à ce dernier, il vous faudra clonez le référentiel du projet avec la commande :

```
>> git clone : https://github.com/Jery2022/bibliotippee.git
```

e. Mise en production

Il est vrai que nous n'avons même pas encore de page HTML pour accueillir convenablement nos internautes, mais voir la petite image "en construction" sur le serveur de production a été une grande satisfaction. Aussi, j'ai fait le choix d'hébergement l'application BiblioTIPPEE sur la plateforme Heroku qui fournit tout ce dont nous avons besoin à partir d'un forfait gratuit.

J'ai donc créé une application « bibliotippee » sur la plateforme Heroku et connecté un pipeline avec mon IDE en utilisant la commande :

```
>> heroku git:remote -a bibliotippee
```

Aussi comme méthode de déploiement, j'ai retenu le « Déploiement avec Git » en suivant pas-à-pas la procédure indiquée pour connaître davantage sur les

possibilités de déploiement dans la plateforme Heroku. Aussi, j'ai fait mon premier déploiement avec la commande :

```
>> git push heroku main
```

Ci-après le lien pour accéder à mon application en préproduction :

<https://bibliotippee-e7f5c196e9b3.herokuapp.com/>

f. Développement de la base de données

Suivant l'analyse de mon diagramme de Class en UML, les relations entre les tables sont les suivantes :

- La table **Utilisateur (User)** a une relation un-à-plusieurs avec les tables **Document (Document)**, **Téléchargement (Download)**, **Chargement (Upload)**, **Favori (Favori)**, **Commentaire (Comment)** et **Recherche (Search)**.
- La table **Favori** a une relation plusieurs-à-un avec la table **Document**.
- La table **Téléchargement** a une relation plusieurs-à-un avec la table **Document**.
- La table **Document** a une relation un-à-plusieurs avec les tables **Chargement**, **Commentaire** et **Recherche**.
- La table **Recherche (Search)** a une relation plusieurs-à-plusieurs avec les tables **Mots-clés (WordSearch)** et **Période-recherchée (PeriodSearch)**.

Ces tables et relations permettront de stocker les informations relatives aux utilisateurs, aux documents, aux téléchargements, aux chargements, aux favoris et aux recherches, en assurant la cohérence des données et en facilitant la mise en œuvre des fonctionnalités requises.

Ainsi, pour interagir avec la base de données depuis PHP, nous allons nous appuyer sur Doctrine ORM (une librairie pour manipuler le contenu de notre base de données en utilisant des objets PHP) et Doctrine Migrations pour la création des tables dans la base de données.

Auparavant, j'ai été amené à créer toutes mes entités PHP et définir les relations entre elles. J'ai par exemple pour l'entité « User » utilisé la commande :

```
>> symfony console make:entity User
```

C'est donc grâce au **Maker Bundle** que j'ai généré la « class User » ou « Entity class » qui va représenter les utilisateurs.

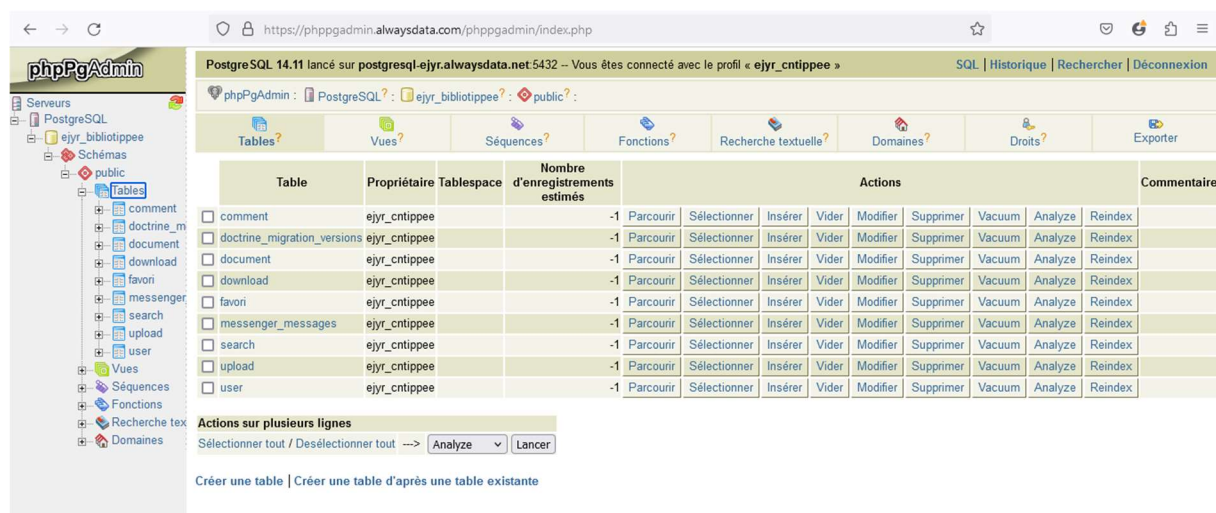
Après avoir réalisé la création de toutes mes entités, j'ai donc procédé à la création des relations entre elles (voir en annexe 3). Par exemple, entre les tables « User » et « Favori » par le biais de la commande précédente.

Toutes mes « Class » étant décrites, grâce à **Doctrine ORM**, j'ai pu ainsi générer le fichier de migration avec **Doctrine Migration** à travers la commande :

```
>> symfony console make:migration
```

Tous mes fichiers de migration étant mise en place, grâce à nouveau à **Doctrine Migration**, j'ai pu mettre à jour le schéma de la base de données BiblioTIPPEE pour qu'elle soit prête à stocker les données en exécutant la commande :

```
>> symfony console doctrine:migrations:migrate
```



The screenshot shows the phpPgAdmin web interface for a PostgreSQL database. The left sidebar displays a tree view of the database structure, including servers, databases, schemas, and tables. The main panel shows a list of tables with columns for Table, Propriétaire, Tablespace, Nombre d'enregistrements estimés, Actions, and Commentaire. The tables listed are comment, doctrine_migration_versions, document, download, favori, messenger_messages, search, upload, and user. Each table has a set of actions: Parcourir, Sélectionner, Insérer, Vider, Modifier, Supprimer, Vacuum, Analyze, and Reindex. Below the table list, there is a section for 'Actions sur plusieurs lignes' with a dropdown menu set to 'Analyze' and a 'Lancer' button.

Table	Propriétaire	Tablespace	Nombre d'enregistrements estimés	Actions	Commentaire
<input type="checkbox"/> comment	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> doctrine_migration_versions	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> document	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> download	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> favori	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> messenger_messages	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> search	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> upload	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	
<input type="checkbox"/> user	ejyr_cntippee		-1	Parcourir Sélectionner Insérer Vider Modifier Supprimer Vacuum Analyze Reindex	

Afin de toujours disposer d'un jeu de données qui soient fictives et à jour pour créer un environnement non vierge lors du développement de l'application Symfony BiblioTIPPEE et pour aussi pouvoir réaliser des tests, j'ai créé et mis en place des « fixtures » à l'aide de Symfony par le biais de la commande :

```
>> composer require --dev orm-fixtures
```

Et pour effectuer le chargement des données fictives, j'ai exécuté sur dans terminal la commande :

```
>> php bin/console doctrine:fixtures:load
```

Elles m'ont permis d'initialiser les données de la base de données afin de pouvoir manipuler cette dernière.

g. Développement des routers et controllers

En rappel, un « controller » est une fonction qui crée la réponse HTTP à la suite d'une requête HTTP et elles sont implémentés en tant que classes PHP.

Tandis qu'une route est la liaison entre le chemin de demande et une fonction appellable en PHP.

Donc, Symfony nous met à disposition un Bundle qui m'a aidé à générer beaucoup de classes différentes.

Pour créer mon premier contrôleur via la commande « Make », j'ai utilisé la commande ci-après par exemple sur l'entité « User » :

```
>> symfony console make:controller UserController
```

Une fois tous les contrôleurs en place, j'ai créé une interface d'administration de ma base de données avec le bundle « **easyAdmin** » pour créer toutes les dépendances liées à mon projet. J'ai donc exécuté la commande suivante :

```
>> symfony composer req "easycorp/easyadmin-bundle:4.x-dev"
```

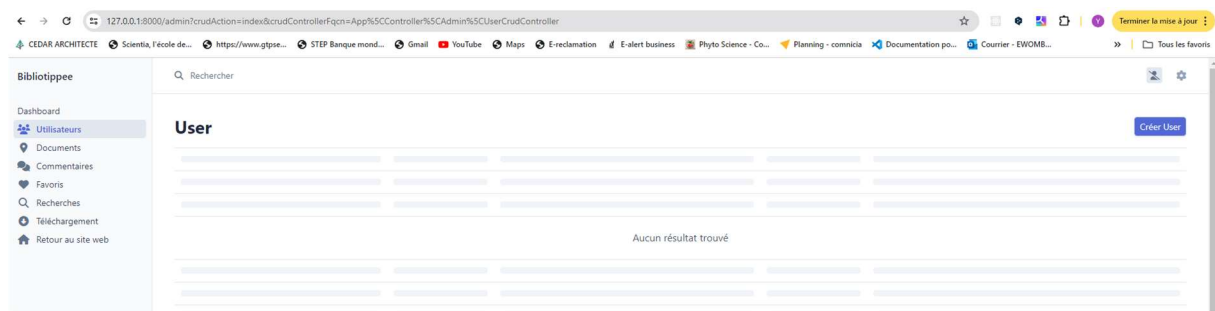
Puis, j'ai commencé avec easyAdmin a généré le « **Dashboard** » qui sera le principal point d'entrée pour gérer les données du site web :

```
>> symfony console make:admin:dashboard
```

Cela m'a permis de générer les opérations de base : création, lecture, mise-à-jour et effacement (CRUD) des données de toutes mes entités par le biais de la commande :

```
>> symfony console make:admin:crud
```

Par exemple, ci-après, la console d'administration de l'entité User :



Remarque, pour les entités « Downloader » et « Uploader », j'ai installé et configuré le bundle « vich/uploader-bundle » en utilisant la commande :

```
>> composer require vich/uploader-bundle
```

Une fois tous les CRUD mis en place, j'ai poursuivi pour créer la première version de l'interface utilisateur du site.

Pour des questions de performances en matière de sécurité, j'ai fait le choix d'utiliser Twig pour gérer les sorties et les différents échappements nécessaires. En effet, selon la documentation de Twig, ce dernier apporte beaucoup de fonctionnalités intéressantes que j'ai exploité dans le cadre de ce projet à l'exemple de l'héritage de modèles.

Voici un exemple d'interface d'administration incluant des images uploadées :

h. Sécurisation de l'interface d'administration

Notons que l'interface d'arrière-plan d'administration ne devrait être accessible que par des personnes autorisées. Aussi, j'ai sécurisé cette zone à l'aide de la composante « Symfony Security ».

La première étape a consisté à définir une entité Utilisateur spécifique que j'ai nommé « Admin ». Grâce au système d'authentification de la sécurité Symfony, ce dernier a des exigences spécifiques. Par exemple, il a besoin d'un mot de passe sécurisé. Pour ce faire, j'ai utilisé la commande :

```
>> symfony console make:user Admin
```

Outre la génération de l'entité « Admin », la commande a également mis à jour la configuration de sécurité pour câbler l'entité avec le système d'authentification et fais le réglage du contrôle d'accès à l'interface d'administration (fichier : security.yaml).

Puis, avec la commande ci-après, j'ai généré le hachage du mot de passe :

```
>> symfony console security:hash-password
```

J'ai porté mon choix sur l'authentification par mot de passe et email conformément à ma maquette validée par mon client. Ci-après l'écran de connexion par formulaire d'authentification (pas encore aligné à la charte et design du maquettiste).

The screenshot shows a web browser window displaying the login page of 'BiblioTIPPEE'. The browser's address bar shows 'localhost:8000/login'. The page features a header with the site logo, navigation links, a search bar, and buttons for 'Se connecter' and 'S'inscrire'. The main content area is titled 'Please sign in' and contains input fields for 'Email' and 'Password', a 'Remember me' checkbox, and a 'Sign in' button. The footer includes links for 'Consulter notre :', 'Retrouvez-nous sur :', and 'Nous vous accompagnons :', along with a copyright notice for 2024 Commission Nationale des TIPPEE, Inc.

2. Développement de la partie front-end

a. Mise en place de l'environnement de développement

Pour la partie front-end, j'avais préalablement installé :

- Bootstrap version 5.3.3
- Sass

On note comme dossiers Principaux :

- src/ : Ce dossier contiendra tous les fichiers sources de l'application.
- assets/ : Ce dossier contiendra les ressources statiques telles que les images, les polices, etc.

b. La navigation au sein de l'application

J'ai utilisé des modèles de base des modèles proposés par Bootstrap que j'ai modifié pour tenir compte de ma charte graphique.

Des améliorations des interfaces en front sont encore en cours pour améliorer l'expérience utilisateur dans la prochaine version de l'application.

c. Les contextes

Le contexte de développement de la partie front-end n'est totalement pas différents de celui du développement de la partie back-end. Les outils et la technologie reste les mêmes. J'ai donc développé la partie front-end en PHP Symfony.

Dans une prochaine version, je compte faire l'intégration du framework React JS pour réduire les rechargements des pages et obtenir plus d'interactivités.

Fonctionnalités significatives

1. L'authentification

Pas encore abouti. Quelques bugs à corriger.

2. Le changement de mot de passe ou mot de passe oublié

Pas encore réalisé.

3. L'affichage et la visualisation des PDF

Pas encore réalisé.

Conclusion

Dans le cadre du développement de notre application web, nous avons rencontré quelques difficultés liées à des bugs qui ont affecté la disponibilité de certaines fonctionnalités. Malgré cela, nous avons réalisé des progrès significatifs dans le développement de l'application et avons pu mettre en place plusieurs fonctionnalités clés.

Au cours de notre projet, nous avons identifié les bugs et avons travaillé activement sur leur résolution. Malheureusement, en raison de contraintes de temps et de ressources, nous n'avons pas été en mesure de corriger tous les bugs et de rendre toutes les fonctionnalités disponibles dans le cadre de cette version.

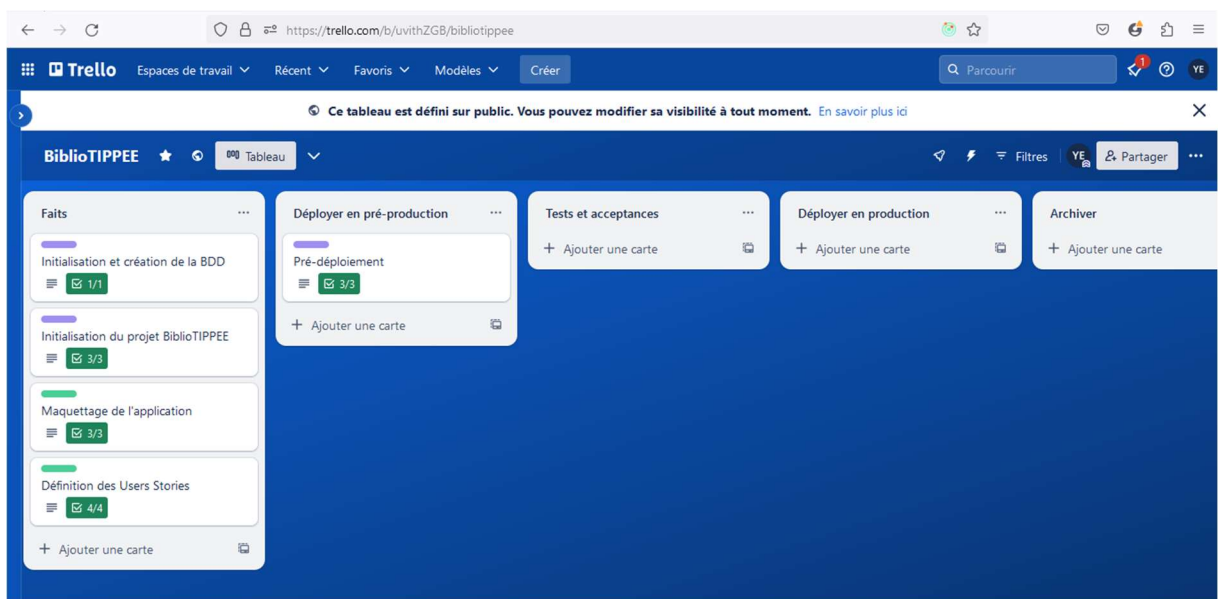
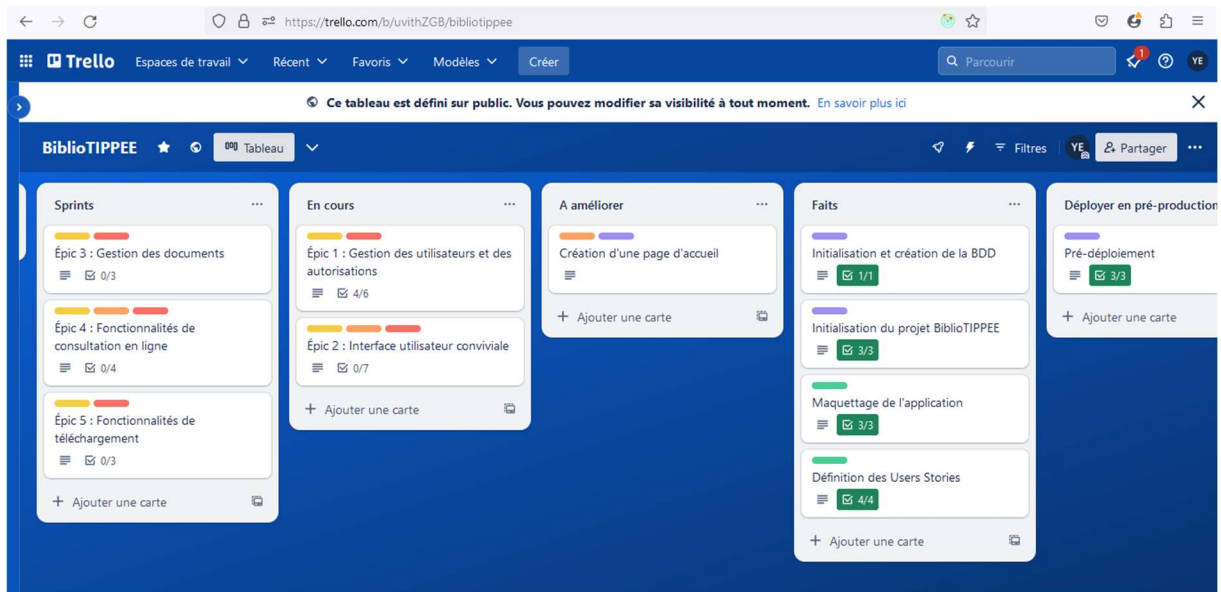
Cependant, nous sommes conscients de l'importance de ces fonctionnalités pour l'expérience utilisateur et nous nous engageons à continuer à travailler sur la résolution des bugs restants. Nous prévoyons de lancer des mises à jour régulières pour améliorer l'application et rendre toutes les fonctionnalités disponibles dès que possible.

Malgré les difficultés rencontrées, nous sommes fiers des réalisations que nous avons accomplies jusqu'à présent.

Nous tenons à remercier tous les membres de la CNTIPPEE, ainsi que les enseignants, pour leur soutien et leur compréhension tout au long de ce projet.

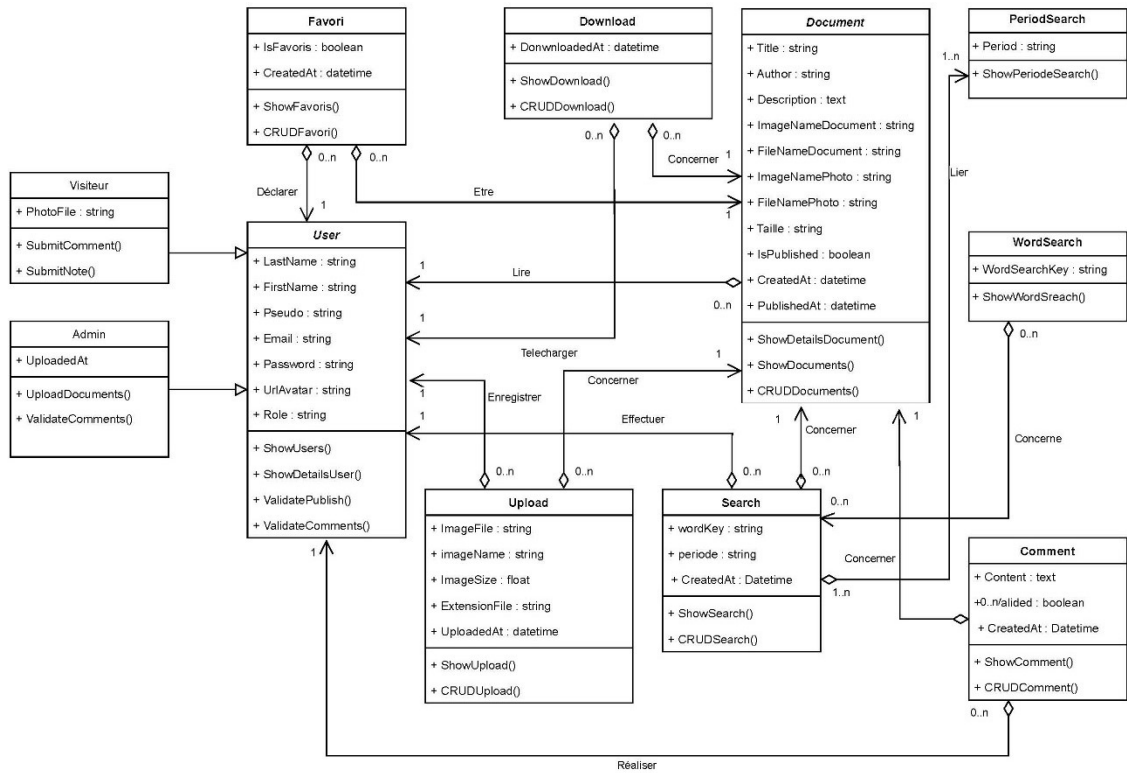
ANNEXES

ANNEXE 1 : Gestion de projet – Mon TRELLO



ANNEXE 2 : Cas d'utilisation

ANNEXE 3 : Diagramme de Class



ANNEXE 4 : Maquettes en version desktop et mobile



Desktop - Connexion



Saisissez vos identifiants pour vous connecter

Email:

Mot de passe:

[Mot de passe oublié ?](#)

Desktop - Inscription



Inscrivez-vous pour vous connecter

Nom:

Prenom:

Email:

Mot de passe:

ANNEXE 6 : README.md

Application pour la gestion des documents numériques de la Commission Nationale des TIPPEE créé avec Symfony 7.0

1 Comment accéder en local à mon code de développement

1- Clonez le référentiel du projet avec git clone :

<https://github.com/Jery2022/bibliotippee.git>

2- Installez les dépendances en exécutant « npm » dans le dossier du projet

Il s'agit de pour l'essentiel :

- Installer composer
 - <https://getcomposer.org/download/>
- Installer symfony cli
 - <https://symfony.com/download>
- Vérifier les requirements
 - symfony check:requirement
- Pour compiler en dev
 - npm run dev
- Pour compiler en prod
 - npm run build
- lancer composer require symfony/webpack-encore-bundle
- lancer npm install file-loader --save-dev
- lancer l'installation du bundle easyadmin
- Lancer un serveur
 - symfony server:start
- bootstrap

2 Comment accéder en ligne à mon code en production dès la levé du bug Apache

Il faut cliquer sur le lien :

<https://bibliotippee-e7f5c196e9b3.herokuapp.com/>

Scripts disponibles

Dans le dossier projet, lancer la commande suivante pour faire tourner l'application en local après avoir installé toutes les dépendances indiquées plus haut :

`symfony`

Lancer l'application en mode développement

ouvrir avec le lien : [<https://localhost:8000/>](<http://localhost:8000>)

ou `symfony open:local`

afin de voir le site à partir d'un navigateur en local.

Près requis techniques

Serveur :

- Heroku.com
- Version PHP 8.2
- PostgreSQL (version 14)

Les langages et/ou frameworks cités ci-dessous ont été employés pour le développement du site.

Pour le front :

- HTML 5
- CSS 3
- Bootstrap
- Sass

- JavaScript

Pour le back :

- PHP 8.2 sous PDO
- Symfony 7.0
- PostgreSQL (version 14)

Sites inspirations :

- <https://www.editions-eni.fr/>
- <https://www.editions-eyrolles.com/>

Apprendre plus sur Symfony

Vous pouvez en apprendre plus en consultant la documentation officielle à travers le lien : <https://symfony.com/>.

Apprendre plus sur Heroku

<https://devcenter.heroku.com/start>

Informations utiles sur le front-end

TO DO :

- Mise en conformité du design des interfaces par rapport aux différentes maquettes
- Résolution du bug de la page d'authentification
- Création de la page d'inscription
- Correction et finalisation des Fixtures
- Création de la page de contact et soumission
- Finalisation de la soumission des commentaires sur une document numérique
- Finalisation des filtres et recherche en front-end

- Mise en place de la soumission du status favori

Deployment

Le site est déployé sur la plateforme Heroku. Mais, difficulté d'affichage de l'application (Erreur 404 : Forbitten) liée à Appache sur le serveur de production. Traitement du bug en cours.

Enjoy You !