L’objet de l’application à concevoir est de permettre de créer, modifier, stocker, versionner et publier la documentation logicielle de la société éditrice de logiciel Océalia Informatique.

L’application doit gérer les différentes versions de la documentation.

Cette documentation existe, aujourd’hui chez l’éditeur, au format Word et se décline en documentation fonctionnelle, fiches pratiques et quelques vidéos.

Le logiciel qui sera conçu devra être capable de récupérer ces contenus.

Le contenu de l’outil de documentation sera géré de manière structurée en se basant sur un format XML de type DITA.

La mise en forme du texte dans l’application sera sommaire de manière à permettre de consulter un aperçu approximatif du contenu.   
Les feuilles de style seront ensuite appliquées en fonction du format de sortie attendu.

On distinguera dans les éléments qui composent la documentation :   
- la structure, ou map, de l’information : l’arborescence qui relie les informations (rubriques) entre elles   
- le contenu est rassemblé en rubriques qui peuvent être :   
a) un concept ou article (description d’une fonctionnalité),   
b) une tâche (description étape par étape de l’utilisation de la fonctionnalité),   
c) une référence (un lien vers des éléments d’information complémentaires)   
Pour plus d’information sur DITA, se reporter au référentiel : <https://www.ixiasoft.com/documentation/DITA_Style_Guide/fr/index.html>

A partir de la base documentaire, un composeur doit permettre de regrouper tout ou partie des différents types de contenus (texte, images, vidéos, questions/réponses) afin de produire, à la demande des contenus multiples.

Les formats de sortie attendus sont les suivants :   
1. De la documentation au format PDF.   
2. Une aide fonctionnelle de type web help qui pourra être utilisée par les applications auxquelles elle se rapporte. Le contenu de la documentation du web help doit pouvoir dépendre des fonctionnalités applicatives accessible à l’utilisateur en fonction de son profil.   
3. Du contenu pédagogique qui sera importable dans une plateforme de type Moodle. Ces contenus pourront être constitués, de texte, d’images, de vidéos et de questions/réponses.   
4. Du contenu destiné à alimenter une Intelligence Artificielle qui sera en charge, via un tchat bot, de traiter le premier niveau de support aux utilisateurs des applications de l’éditeur.

Les applications de l’éditeur Océalia Informatique sont :   
- Océalia Usager : logiciel de gestion du dossier unique de l’Usager (personne suivie par un établissement médical ou médical social)   
- Océalia Planning : logiciel de gestion des plannings des professionnels dans les domaines du médico-social ou des collectivités territoriales.   
Océalia Planning dispose de modules complémentaires :   
a) eOcéalia qui permet aux salariés de consulter et intervenir sur leurs plannings en ligne sur un site internet dédié. Ce module dispose de sa propre documentation.   
b) Océalia badgeuse : qui est une interface permettant de récupérer les badgeages, physiques ou non, pour les comparer par rapport aux plannings des professionnels dans Océalia Planning et gérer les écarts de temps de travail.

L’application à concevoir est un site internet hébergé chez l’éditeur.

Les accès sont restreints aux salariés d’Océalia Informatique qui peuvent être en charge :   
- de créer, modifier la documentation existante   
- de relire le contenu (et apporter des commentaires ou des modifications sur les articles)   
- de publier du contenu Il y aura donc une gestion des accès et des droits d’utilisation à définir.

La relecture et la modification des articles de la documentation sera tracée automatiquement grâce à l’utilisateur connecté.

L’application est à utiliser à partir d’un poste informatique mais peut être consultée à partir d’un téléphone mobile. Lors de l’usage sur un mobile, les fonctionnalités seront restreintes à la consultation de la documentation existante et à la modification d’articles en mode relecture.

Nous avons établi les éléments de base suivant :

A. Nous avons retenu Python pour le serveur qui sera utilisé sur un PC via Visual Studio Code.   
Le Gestionnaire de package pour Python retenu est pip.

B. La base de données est en PostgreSQL. V16.4.   
Elle est hébergée en local pour le développement. Nous verrons ensuite pour l'hébergement de cette dernière.   
La base se nomme : « documentationocealia »

C. Nous avons déterminé une arborescence de projet conçue de telle manière :

C:\DOCUMENTATION\_PROJECT

| .gitignore

| manage.py

| requirements.txt

+---.circleci

+---documentation

+---documentation\_frontend

| +---node\_modules

| +--- public

| +---src

+---documentation\_project

| +---users

| \_\_init\_\_.py

| asgi.py

| settings.py

| urls.py

| views.py

| wsgi.py

+---env

| +---Include

| +---Lib

| +---Scripts

| pyenv.cfg

+---logs

D. L'authentification se fait en interne par une solution développée (sans SSO...)   
Un journal des connexions a été mis en place.

E. Nous utilisons React.js pour le frontend. Les styles sont basés sur Tailwind CSS.

F. La navigation se fait sur PC.   
Le développement de l'environnement responsive se fera dans un deuxième temps. Il est à inclure dans le projet mais non prioritaire.

G. Le déploiement initial est en local. Il n'y a pas de choix arrêté pour le déploiement en version de production.

H. La gestion du code source et du versioning s’effectue sur GitHub.

I. Le testing s’effectue sur CircleCi. La plateforme est connectée à GitHub et fonctionnelle. Nous avons déjà testé un essai de création d’un compte utilisateur en base.

J. Il n'est pas nécessaire de penser à du chiffrement de données. Il n'y a pas de confidentialité critique dans ces informations.

H. La journalisation des actions est un élément à prendre en compte dans le projet. Elle doit pouvoir s'activer une fois la base documentaire initiale constituée.

Je te propose que nous abordions maintenant la structure de base de données dont nous avons besoin pour ce projet.

En dehors des tables liées aux utilisateurs de l’application qui sont déjà créées (auth\_group, auth\_group\_permissions, auth\_user…)

La documentation est composée de projets.

Chaque projet correspond à un produit chez l’éditeur.

Un produit peut contenir des modules. Les modules sont des composantes complémentaires du logiciel principal.  
Par exemple : le logiciel Océalia Planning comporte un module principal : le logiciel, en lui-même. Il possèdes des modules complémentaires : eOcéalia et Océalia Badgeuses.

Les modules sont composés de rubriques (ou articles). Une rubrique correspond à une fonctionnalité du logiciel. Cette rubrique peut être différents types :  
- concept : c’est la définition de la fonctionnalité  
- tâche : c’est la réponse à « comment faire »… comment utiliser cette fonctionnalité  
- référence : il s’agit d’exemples ou de liens vers une autre rubrique ou une autre tâche  
- questions : il s’agit d’un type spécifique à la constitution d’un questionnaire de vérification d’acquisition de compétence dans la conception d’une formation. Une rubrique peut être associée à une ou plusieurs questions, qui vont avoir plusieurs réponses, dont seulement une partie sera exacte.

Une rubrique va être « chargée » avec des attributs ou propriétés : comme par exemple, l’auteur, la date de création ou de mise à jour, l’audience à qui il est destiné, un numéro de révision et un ID Fonctionnalité qui fera le lien – en mode webhelp – avec les fonctionnalité du profil de l’utilisateur de l’application afin de lui proposer une aide contextuelle.Une image contenant texte, nombre, logiciel, ligne

Description générée automatiquement

Les rubriques sont assemblées par une structure qu’on appelle « carte » ou « map ».  
Elles permettent de :  
- définir les rubriques à livrer dans le document final. (Comment le document est constitué)  
- définir une hiérarchie des rubriques  
- créer un lien entre les rubriques

Il existera une carte « maître » pour tous les Produits.

Il sera ensuite possible de constituer des cartes « filles » en fonction de différents modèles  :  
- documentation  
- aide en ligne  
- fiche pratique  
- cours

Mais tout ce ceci étant du XML, je ne sais pas trop si il est nécessaire de constituer des tables spécifiques avec cette information ( ?)

Les concepts ou les tâches peuvent être agrémentées d’images ou de vidéos qui devront être référencées dans la table. Les images, en plus d’avoir un identifiant propre seront renommées selon une nomenclature qui permettra de les retrouver facilement (par exemple : code de 3 lettres par rapport à la fonctionnalité de rattachement, le mot bouton si c’est un bouton, plus un nom clair).

Voici les tables que nous avons créé :

-- Table Projet

CREATE TABLE projet (

id SERIAL PRIMARY KEY,

nom VARCHAR(255) NOT NULL,

description TEXT,

date\_creation TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

date\_mise\_a\_jour TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Table Module

CREATE TABLE module (

id SERIAL PRIMARY KEY,

projet\_id INT REFERENCES projet(id) ON DELETE CASCADE,

nom VARCHAR(255) NOT NULL,

description TEXT,

date\_creation TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

date\_mise\_a\_jour TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Table Rubrique

CREATE TABLE rubrique (

id SERIAL PRIMARY KEY,

module\_id INT REFERENCES module(id) ON DELETE CASCADE,

type\_rubrique VARCHAR(50) CHECK (type\_rubrique IN ('concept', 'tâche', 'référence', 'question')),

titre VARCHAR(255) NOT NULL,

contenu\_xml XML,

auteur VARCHAR(100),

date\_creation TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

date\_mise\_a\_jour TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

id\_fonctionnalite VARCHAR(50),

audience VARCHAR(100),

revision\_numero INT DEFAULT 1

);

-- Table Map

CREATE TABLE map (

id SERIAL PRIMARY KEY,

nom VARCHAR(255) NOT NULL,

type\_map VARCHAR(50) CHECK (type\_map IN ('maître', 'fille', 'documentation', 'aide en ligne', 'fiche pratique', 'cours')),

projet\_id INT REFERENCES projet(id) ON DELETE SET NULL,

structure\_xml XML

);

-- Table Relation Carte-Rubrique

CREATE TABLE carte\_rubrique (

id SERIAL PRIMARY KEY,

map\_id INT REFERENCES map(id) ON DELETE CASCADE,

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

ordre INT NOT NULL,

relation VARCHAR(100)

);

-- Table Média

CREATE TABLE media (

id SERIAL PRIMARY KEY,

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

type\_media VARCHAR(50) CHECK (type\_media IN ('image', 'vidéo')),

nom\_fichier VARCHAR(255) NOT NULL,

description TEXT,

chemin\_acces TEXT NOT NULL

);

-- Table Question-Réponse

CREATE TABLE question\_reponse (

id SERIAL PRIMARY KEY,

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

question TEXT NOT NULL,

reponse TEXT NOT NULL,

correcte BOOLEAN DEFAULT FALSE

);

-- Table Historique des Modifications

CREATE TABLE historique\_modification (

id SERIAL PRIMARY KEY,

rubrique\_id INT REFERENCES rubrique(id) ON DELETE SET NULL,

utilisateur VARCHAR(100) NOT NULL,

action VARCHAR(50) CHECK (action IN ('création', 'modification', 'suppression')),

date\_action TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Création d'index pour optimiser les recherches courantes

CREATE INDEX idx\_projet\_nom ON projet (nom);

CREATE INDEX idx\_module\_nom ON module (nom);

CREATE INDEX idx\_rubrique\_titre ON rubrique (titre);

CREATE INDEX idx\_map\_nom ON map (nom);

-- Suivi des versions du projet  
CREATE TABLE version\_projet (

id SERIAL PRIMARY KEY,

projet\_id INT REFERENCES projet(id) ON DELETE CASCADE,

version\_numero VARCHAR(50) NOT NULL,

date\_lancement TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

notes\_version TEXT

);

-- Annuaire des rubriques

CREATE TABLE tag (

id SERIAL PRIMARY KEY,

nom VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE rubrique\_tag (

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

tag\_id INT REFERENCES tag(id) ON DELETE CASCADE,

PRIMARY KEY (rubrique\_id, tag\_id)

);

-- Gestion des profils utilisateurs

CREATE TABLE profil\_utilisateur (

id SERIAL PRIMARY KEY,

nom\_profil VARCHAR(50) NOT NULL UNIQUE,

description TEXT

);

CREATE TABLE rubrique\_profil (

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

profil\_utilisateur\_id INT REFERENCES profil\_utilisateur(id) ON DELETE CASCADE,

PRIMARY KEY (rubrique\_id, profil\_utilisateur\_id)

);

-- Trace des historiques des exports

CREATE TABLE historique\_export (

id SERIAL PRIMARY KEY,

map\_id INT REFERENCES map(id) ON DELETE SET NULL,

format\_export VARCHAR(50) CHECK (format\_export IN ('PDF', 'web help', 'Moodle', 'chatbot')) NOT NULL,

date\_export TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

utilisateur VARCHAR(100) NOT NULL,

chemin\_export TEXT

);

-- Suivi des relectures

CREATE TABLE audit\_action\_utilisateur (

id SERIAL PRIMARY KEY,

utilisateur VARCHAR(100) NOT NULL,

action VARCHAR(50) NOT NULL,

cible\_id INT, -- ID de l'élément cible (rubrique, map, etc.)

cible\_type VARCHAR(50), -- Type de la cible (par ex., 'rubrique', 'map')

description TEXT,

date\_action TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Références externes

CREATE TABLE reference\_externe (

id SERIAL PRIMARY KEY,

rubrique\_id INT REFERENCES rubrique(id) ON DELETE CASCADE,

url TEXT NOT NULL,

description TEXT

);