Objectifs principaux à atteindre (dans un monde idéal)

Interface graphique

De manière générale : l'identification des utilisateurs devra apparaître en haut à droite. Si la personne n'est pas loggée il n'y a rien d'afficher sur le site sauf l'invitation à se logger (cf base de données).
A l'avenir, on pourra envisager la distinction de profils "experts" et "assistants", sachant que seuls les experts ont la possibilité de consulter l'activité des autres, d'insérer des articles, de faire changer le statut en "traité" ou de réattribuer un article qui ne leur appartient pas. Le rôle des assistants se limiterait donc à travailler sur des "indéfinis" pour les passer en "à traiter" par eux et donc à les traiter les articles (prises de notes/annotation des pdf) pour au final les repasser à un "expert" pour finalisation.

Page principale

Aspect général de la page

- Cinq onglets/pages permettent d'accéder à l'un des cinq tableaux en fonction du statut des articles et de l'utilisateur (cf partie base de données). Un article n'apparaît donc que dans un seul tableau/onglet.
- Ces tableaux ont une structure commune, décrite dans la partie tableau d'articles
- Leurs ordre et spécificité sont décrits dans la partie spécificités des tableaux

Tableau d'articles

Dans ces tableaux, chaque ligne correspond à un article et est cliquable pour ouvrir une page article. Les tableaux devront être triables alphabétiquement et filtrables par expressions régulières depuis une ligne d'en-tête persistante en cas de *scroll down*.
Les champs (de la table article; cf partie base de données) à afficher sont :

- infos de l'article : titre et auteurs (1er ... dernier) avec un lien vers l'article en ligne (e.g. PMC). Le survol permet d'afficher l'abstract dans un encadré.
- les notes prises par les curateurs : affichage réduit, mais survol affiche la totalité
- le statut de l'article, modifiable, le changement entraînant automatiquement son ré-assignement au bon tableau ainsi que dans le cas particulier où il passe de "à traiter" on déclenche alors la récupération de(s) accession(s), l'identification de(s) l'entrée(s) CAZy et le téléchargement du pdf (cf parties récupération des accessions/base et liens CAZy et récupération du pdf)
- l'attribution à une personne unique (*autocomplete user list*), modifiable uniquement si "non-attribué" ou si l'on en est "propriétaire". Il faudrait qu'en un clic "large" on puisse se le réserver et surtout que le statut soit actualisé immédiatement chez les autres utilisateurs.
- dans un futur proche, des scores
    - le score de pertinence à partir du *Text Mining*
    - la moyenne d'un "score de sentiment" des curateurs basé sur l'appréciation de l'article, de la qualité des données, ...

Spécificités des tableaux

1. Tableau “à traiter” par l’utilisateur (donc spécifique à l’utilisateur loggué)
2. Tableau “à traiter” mais non-attribués à un utilisateur
   ⇒ Les articles dans ce tableau n’ont pas encore été réservés par/assignés à un utilisateur, ici c’est donc le marché ! On se réserve l’article en se l’attribuant, ce que les autres doivent voir rapidment !
3. Tableau “à traiter” par les autres utilisateurs
4. Tableau d’articles avec un statut “indéfini”
   ⇒ Possibilité de demander la récupération du pdf via un bouton pour décider
   ⇒ Ce tableau a la particularité d’être celui alimenté par les scripts automatiques décrits partie script d’insertions récurrentes
   ⇒ ne permettent pas d’ouvrir de pages articles mais la page “pubmed”
5. Tableau des articles “traités” : listing des 10 derniers articles
6. Tableau des articles “rejetés” : listing des 10 derniers articles

Page article

La page article doit permettre de visualiser à la fois l'article et les attributs et note "synthétique". L'aspect ergonomique n'a pas encore été décidé, entre une visualisation gauche-droite, haut-bas, ou en très grand avec un menu persistant permettant de monter descendre de l'un à l'autre (pas seulement par scroll down).

Accès à l'article

- dans un premier temps, le pdf a été récupéré, si possible, via PubMedCentral, sinon CNRS (click and Read).
- on extrait le texte/HTML du pdf et on l'intègre dans un GoogleDocs ou autre solution collaborative permettant de mettre en forme : surtout de surligner les passages importants et commenter si besoin d'aide des autres.
- on sauvegarde le docs/html et on mets seulement le lien dans la base

Les attributs/notes

On doit pouvoir actualiser, mais pas de manière concurrente, l'ensemble de ces attributs et la notes synthétique de l'article. Pour cela, dans un premier temps on n'autorise la modification qu'à la personne qui s'est attribuée l'article. Si personne ne se l'est attribué, une solution est de récupérer l'ensemble des infos **initiales** (lors du chargement de la page) et de les stocker en mémoire. Puis une récupération de toutes les infos a lieu toutes les XY secondes dans le but de sauvegarder :

- **si et seulement si** l'état dans la base est toujours identique à l'état initial stocké en mémoire. A ce moment-là, on sauvegarde et modifie l'état initial/le stockage mémoire.
- Sinon on affiche un warning disant qu'il y a eu accès concurrent et qu'il faut garder/recopier ses notes ailleurs.

Les informations à prendre en compte (cf partie Tables) :

- le statut
- la personne en charge (permettre de s'en libérer ou se l'attribuer si "non-attribué", l'assigner à quelqu'un d'autres mais si on se libère, il ne faut pas effacer les notes, juste actualisation de la colonne "personne en charge")
- la note synthétique, e.g. un champ WYSIWYG avec :
    - enregistrement fréquent : pour ne pas perdre les infos en cours
    - débute ou se termine par [l'utilisateur:timestamp] : pour transmettre au suivant/garder une trace

- Spécifique CAZy : tableau dans lequel marquera l'avancée de la curation de l'article
    - les accessions des protéines avec liens web (on doit pouvoir les ajouter si impossible de les extraire automatiquement, cf partie récupération des accessions/base et liens CAZy, ou les enlever si trop de choses rapatriées par ce script)
    - les entrées CAZy, une fois créés/trouvées, affichées par leur modularité simplifiée avec lien web (privé)
    - les fonctions (champ libre avec autocompletion)
    - Plus tard, des vérifications (marquée d'un ✓) sur CAZy pour attester que :
        - le PMID a bien été entré ;

- une fonction a bien été associée ;
- le lien entre la fonction et le PMID a bien été établi.

Page d'insertion

Plusieurs stratégies peuvent être définies :

- l'insertion directe dans la base d'un(e liste de) article;
- la consultation d'un résultat de requêtes par mot-clés (non insérées dans la base) afin d'y calculer un score de pertinence (projet futur);

Dans ces deux cas, ainsi que dans les requêtes automatiques (cf script d'insertions récurrentes), les articles déjà présents dans CAZy ou cette base ne devront pas apparaître.

Insertion directe

Pour commencer, un simple champ doit permettre à l'utilisateur d'indiquer un ou plusieurs articles (séparés par des sauts de lignes) par leur PMID/doi. Par exemple, l'idée en début de stage est de rentrer dans un premier temps les articles du "suivi curation" (GoogleDocs CAZy - onglet Fonctions).
Il a alors le choix de les affecter à quelqu'un, et du statut : "indéfini" ou "à traiter"

Requêtes

Dans un deuxième temps, un champ libre pour l'utilisateur permettra d'aller chercher des articles sur une base (NCBI, Scholar, Tweeter) à partir de mots-clefs. Par exemple : nom d'auteur, nom de famille (GH5 par exemple), articles cités/citants un PMID d'intérêt.
Ici un *input* permettra à l'utilisateur d'indiquer la requête (cf partie récupération des infos d'articles) dont le résultat ne sera pas directement inséré dans la base (aucun statut, même pas "indéfini") mais d'abord listés temporairement à l'utilisateur dans un tableau.
Les articles seront sélectionnables (checkbox) pour que l'utilisateur puisse choisir ceux qu'il souhaite intégrer dans la base en spécifiant le statut et l'utilisateur. Toute insertion dans la base fait disparaître l'article de la liste temporaire.

Champ recherche

Il doit être possible de lancer des recherches sur le contenu de la base, sur les auteurs, titres, etc. pour retrouver des articles anciens, de différents statuts.

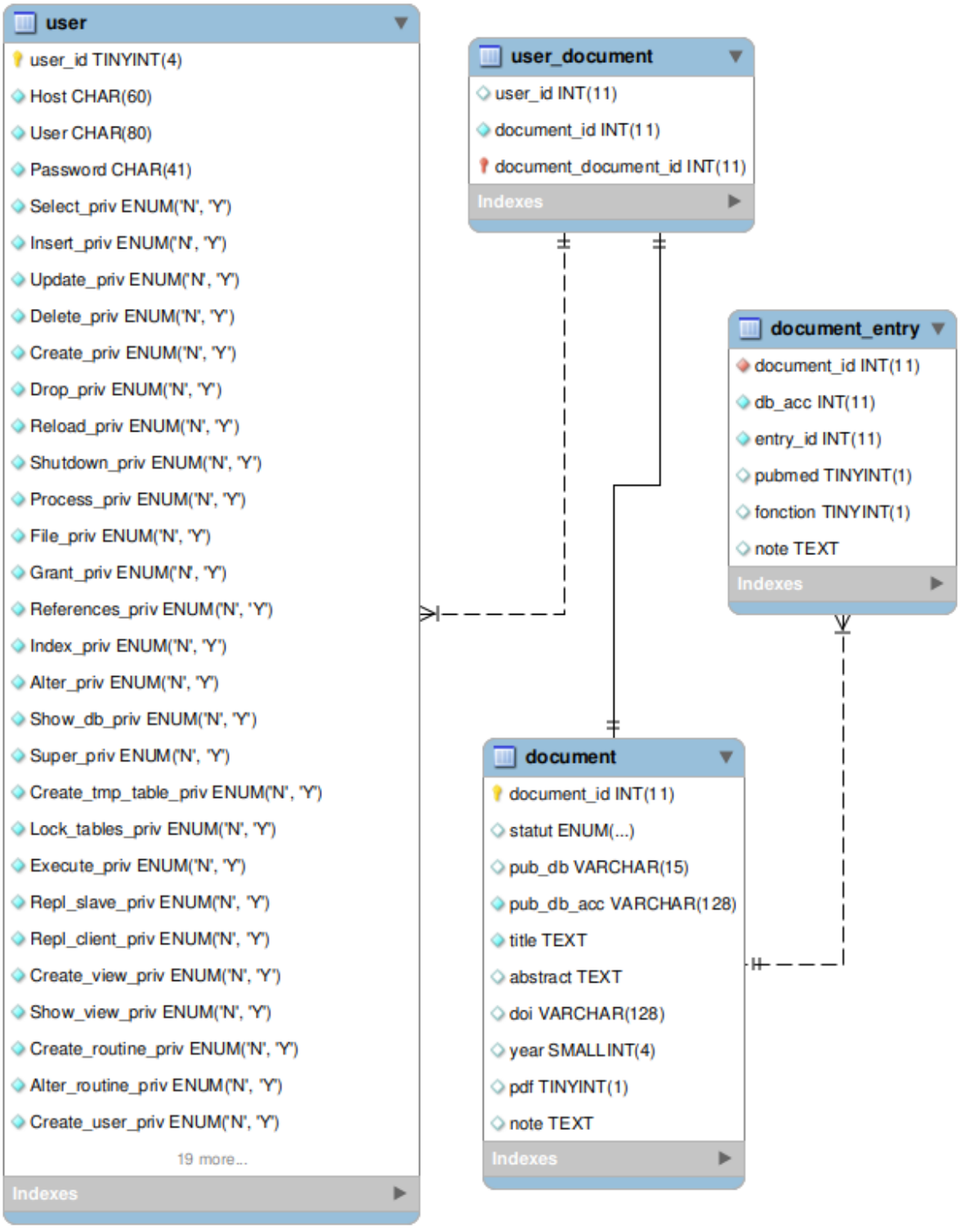
Base de données

Objectifs

- contenir les informations de bases sur les articles;
- lier un article à un utilisateur pour une gestion plus efficace de qui est en train de traiter quels articles
- y ajouter les données “d’annotation” de ces articles entrées par les utilisateurs

Tables

- user : la table user de mysql de CAZy pourra être utilisée
- article : sur le modèle de la structure de la table pub_document de la base extern_db de CAZy, étendue avec
    - un booléen qui nous dit si le pdf a été téléchargé
    - les notes prises par les annotateurs
    - le statut de l’article (indéfini, à traiter, traités, rejetés)
- relation user-article : est-il possible de faire le lien entre qui est en train de lire quoi, pour éviter les accès concurrents ? Car champ “Note” sensible...
- relation article-entries : les accessions, les entry_id (si dans CAZy) (si extraites automatiquement ou vérifiées/créées par les utilisateurs), l’intégration du PUBMED dans l’entrée, et une fonction (écrite par le user dans son champ libre) et la présence d’une fonction pour cette entrée.


user
user_id TINYINT(4)
Host CHAR(60)
User CHAR(80)
Password CHAR(41)
Select_priv ENUM('N', 'Y')
Insert_priv ENUM('N', 'Y')
Update_priv ENUM('N', 'Y')
Delete_priv ENUM('N', 'Y')
Create_priv ENUM('N', 'Y')
Drop_priv ENUM('N', 'Y')
Reload_priv ENUM('N', 'Y')
Shutdown_priv ENUM('N', 'Y')
Process_priv ENUM('N', 'Y')
File_priv ENUM('N', 'Y')
Grant_priv ENUM('N', 'Y')
References_priv ENUM('N', 'Y')
Index_priv ENUM('N', 'Y')
Alter_priv ENUM('N', 'Y')
Show_db_priv ENUM('N', 'Y')
Super_priv ENUM('N', 'Y')
Create_tmp_table_priv ENUM('N', 'Y')
Lock_tables_priv ENUM('N', 'Y')
Execute_priv ENUM('N', 'Y')
Repl_slave_priv ENUM('N', 'Y')
Repl_client_priv ENUM('N', 'Y')
Create_view_priv ENUM('N', 'Y')
Show_view_priv ENUM('N', 'Y')
Create_routine_priv ENUM('N', 'Y')
Alter_routine_priv ENUM('N', 'Y')
Create_user_priv ENUM('N', 'Y')
19 more...
Indexes
user_document
user_id INT(11)
document_id INT(11)
document_document_id INT(11)
Indexes
document_entry
document_id INT(11)
db_acc INT(11)
entry_id INT(11)
pubmed TINYINT(1)
fonction TINYINT(1)
note TEXT
Indexes
document
document_id INT(11)
statut ENUM(...)
pub_db VARCHAR(15)
pub_db_acc VARCHAR(128)
title TEXT
abstract TEXT
doi VARCHAR(128)
year SMALLINT(4)
pdf TINYINT(1)
note TEXT
Indexes

Outils sous-jacents à la partie “interface graphique”

Récupération des infos d’articles

Les informations de base concernant un article (qui n’est ni dans CAZy ni déjà dans cette base) doivent être récupérées (à PUBMED initialement, mais des alternatives pourront être proposées) soit lorsqu’il est inséré dans la base (cf script d’insertions récurrentes), soit lors d’une recherche “consultative” (cf page insertion).
Code existant en perl avec API NCBI, parsing du xml et récupération des infos minimales.
Attention de nombreux cas de réponse / formats XML renvoyés par le NCBI sont à gérer (cf RunParse.pm : sub *query_pubmed* dans le cas d’un PMID et sub *query_crossref* dans le cas d’un DOI.

Récupération des accessions/base et liens CAZy

Si un article a été jugé pertinent (passant de “indéfini” à “à traiter”), il faudrait essayer de récupérer l’accession de la protéine (identifiant/base) pour faire le lien avec CAZy
Via l’API NCBI il est possible de lister les accessions qui sont liées à un article :
https://www.ncbi.nlm.nih.gov/books/NBK25498/#_chapter3_ESearch__ELink__ESummaryEFetch_
Si cette accession existe déjà dans CAZy alors il est possible de la retrouver (attribut db_acc de la table annotation) qui permettra d’en déduire l(es) entrée(s) CAZy concernée(s).
Sinon il y aura un bouton qui enverra vers le script pour créer une nouvelle entrée à partir de l’accession (quid si l’accession vient d’Uniprot et pas de Genbank), et il faudrait avoir un bouton pour insérer directement l’article dans l’entrée CAZy.
Sinon il rentrera les accessions fera à la main.

Récupération du pdf

Télécharger les pdf sur l’espace commun dès qu’il passe en “à traiter” ou sur demande lorsque “indéfini”. Cela demande de gérer les questions d'accessibilité du pdf de l’article : soit disponible via PMC soit via outil “click and read” du CNRS.
Extraire le texte du PDF.

Recherche d’articles à partir de mots-clés

via PubMed Central (PMC) APIs
“PMC provides several APIs that provide programmatic access to various services that deal with PMC literature content, including file validation tools, Open Access web services, and an ID convertor that interconverts PMCID's, PMID's, Manuscript ID's, and DOI's.”
Il y a de nombreux exemples de l’API PMC ici
https://www.ncbi.nlm.nih.gov/pmc/tools/get-pmcids/ qui utilise principalement esearch.cgi

Script d’insertions récurrentes

A partir d’un résultat de recherche/requête (décrit ci-dessus), il devrait être possible de mettre en place/pérenniser une requête dans un script qui s’effectuera alors régulièrement (par exemple articles citant Henrissat ou mentionnant CAZy).
Les requêtes récurrentes sont à **insérer directement dans la base en statut “indéfini”**.

Perspectives (stage TextMining de Ghassan)

Apprentissage d'un champ lexical (et d'auteurs récurrents) à partir de l'ensemble des articles liés à des fonctions dans CAZy (positifs), par comparaison à un univers d'articles (négatifs) .
Par exemple univers = autres articles citant CAZy ou issus d'une recherche "vaste" du domaine Glycobiologie mais pas dans CAZy
Calcul d'un score en fonction des mots clés ou de l'auteur, affiché pour les "indéfini"
Coloration des mots-clefs ou des auteurs pertinents dans le pdf télécharger