

# UEL313 : Bibliothèques logicielles - GROUPE 6 - S3

## Membres du groupe

Etudiant.e	Alias
Mathilde C.	Cloudy23
Kamo G.	Spaghette5
Mathieu L.	mathleys
Filippos K.	filkat34

## Dépôt public

Le projet est hébergé sur Github : <https://github.com/filkat34/UEL313-Groupe6-S3>

## Objectifs

- Assurer la maintenance corrective et évolutive d'une application existante
- Savoir utiliser le client git et la plateforme Github en vue de collaborer au sein d'une équipe de développement.

## Environnement de développement

- Cloner ce dépôt GIT sur votre machine.
- Suivre le guide d'installation de l'environnement docker (pdf fourni dans les ressources de l'UE) en remplaçant lors de l'étape "Run a new container" le "Host path" par le chemin vers le dossier cloné du dépôt sur votre machine.

## Principe général de collaboration

### Répartition du travail

Flux RSS	Page Links	Refonte UI
Filippos	Mathilde & Mathieu	Kamo & Mathieu

### Calendrier

Une réunion d'équipe est prévue à chaque fin d'échéance.

Echéance	Objectif
11/12	Phase de documentation et de réflexion sur la façon d'implémenter la fonctionnalité. Décrire le choix d'implémentation retenu ci-dessous.
13/12	Phase de développement de chaque fonctionnalité sur une branche distincte.

Echéance	Objectif
14/12	Relecture des branches, fusion et tests manuels fonctionnels.

## Phase de documentation et de réflexion

Ci-dessous sont explicitées les implémentations choisies pour chaque intervention évolutive sur l'application.

### Flux RSS

Les cadriels fournissent souvent des modules spécifiques pour la génération de flux RSS comme `sfeed` pour Symfony. Vu la simplicité du fonctionnement de cette application, nous avons décidé de ne pas avoir recours à l'un de ces modules mais de mettre en place nous-mêmes le flux RSS en suivant le protocole d'implémentation suivant :

- Création d'une nouvelle route pour le flux RSS `/feed` qui servira le flux RSS.
- Ajout d'une méthode DAO pour récupérer les 15 derniers liens dans `LinkDAO.php`
- Création d'un nouveau contrôleur `RssFeedController.php` qui récupère les 15 derniers liens grâce à la méthode DAO précédemment implémentée et qui génère le fichier xml du flux à partir des liens récupérés.

Idéalement, il aurait fallu également modifier la table des données de l'applications pour ajouter un champ `cratedat` car, en l'état actuel, les derniers liens sont récupérés en fonction de leur clé primaire qui est automatiquement incrémentée à chaque ajout de lien dans la base.

### Page de liens

Pour la pagination du back-office, on a choisi une approche côté serveur (PHP/SQL) plutôt qu'en JavaScript avec architecture MVC (DAO pour les données, contrôleur pour la logique métier, vue pour le rendu HTML avec Twig).

**Objectif :** Limiter la quantité de données chargées (éviter le chargement de toute la table `tl_liens`) et respecter la contrainte de 15 liens/page directement au niveau de la BDD.

- Côté DAO (`LinkDAO`) : Ajout de la méthode `countAll()` pour renvoyer le nombre total de liens présents + Ajout de la méthode `findByPage()` qui calcule un offset (décalage) selon le numéro de page, exécute une requête avec tri descendant et qui transforme les lignes SQL en objets via la méthode qui existe déjà `buildDomainObject()`.
- Côté contrôleur (`AdminController::indexAction`) : Ajout de l'objet `Request` pour pouvoir lire le paramètre `?page=` dans l'URL + Récupération du numéro de page + Appel de `countAll()` pour compter le nombre total de liens et donc du nombre total de pages + Vérification que page demandée ne dépasse pas la dernière page (sinon on donne la dernière page) + Remplacement de `findAll()` par `findByPage($page, $limit)` pour récupérer que les 15 liens de la page courante + Passage à la view Twig des variables `links`, `page` et `totalPages`.
- Côté vue (`admin.html.twig`) : Réutilisation du tableau pour afficher les liens avec ajout de la class `pagination` de Bootstrap pour afficher proprement les liens vers les pages 1 jusqu'à la dernière, pour indiquer la page courante comme active, pour désactiver les boutons "précédent" et "suivant" si on est sur la page 1 ou la dernière page et pour générer les URLs avec pour rester cohérent avec route `/admin`.

## Refonte UI

# Phase de développement

Plusieurs issues ont été identifiées en fonction des fonctionnalités à implémenter :

1. Chaque membre de l'équipe s'assigne une issue en fonction de son choix dans la répartition du travail.
2. Il crée une branche sur laquelle il travaille sur l'issue choisie en lui donnant un nom correspondant à ce qu'il implémente. Exemples : **feature/fluxRSS**, **feature/pagelinks**, etc.
3. Une fois son travail fini, il fait une demande de tirage et dans la description, ne pas oublier de lier la demande à une issue en mettant "Fixes #[numéro de l'issue concernée]" (par exemple : "Fixes #11"). Github se chargera de fermer l'issue en question une fois la fusion de la demande faite.

## Tests manuels fonctionnels

### Pagination des liens

Pour la page de liens, nous avons ajouté un système de pagination dans l'espace d'administration, limitant l'affichage à 15 liens par page. Cela permet de fluidifier la navigation et d'éviter l'affichage d'une liste trop longue.

## Administration

Links

Users

+ Add link

Title	Author	Description	
MySQL	admin	Système de gestion de base de données open source.	 
PHP	admin	Site officiel du langage PHP.	 
Symfony	admin	Framework PHP robuste et modulaire.	 
Laravel	admin	Framework PHP moderne pour le développement web.	 
Express	admin	Framework web minimaliste pour Node.js.	 
Node.js	admin	Environnement d'exécution JavaScript côté serveur.	 
Webpack	admin	Module bundler pour JavaScript.	 
Sass	admin	Langage de préprocesseur CSS.	 
Tailwind CSS	admin	Framework CSS utilitaire moderne.	 
Bootstrap	admin	Framework CSS populaire pour le développement responsive.	 
Angular	admin	Documentation officielle du framework Angular.	 
Vue.js	admin	Documentation officielle du framework Vue.js.	 
React	admin	Documentation officielle de la bibliothèque React.	 
JavaScript.com	admin	Ressources et guides pour JavaScript.	 
Frontend Mentor	admin	Défis de développement front-end réels.	 

Précédent 1 2 3 Suivant

## Administration

Links

Users

+ Add link

Title	Author	Description	
HackerRank	admin	Plateforme de pratique de code et de défis.	 
SitePoint	admin	Ressources et livres pour développeurs web.	 
Pluralsight	admin	Formations techniques pour développeurs.	 
Coursera	admin	Cours universitaires en ligne, y compris le développement web.	 
Udemy	admin	Cours en ligne sur le développement web et plus.	 
Codecademy	admin	Plateforme interactive pour apprendre à coder.	 
TutorialsPoint	admin	Tutoriels sur de nombreux langages et frameworks.	 
Smashing Magazine	admin	Articles et ressources pour designers et développeurs web.	 
W3Schools	admin	Tutoriels et références pour les technologies web.	 
freeCodeCamp	admin	Apprendre à coder gratuitement avec des projets pratiques.	 
CodePen	admin	Environnement de développement front-end en ligne.	 
GitHub	admin	Plateforme d'hébergement et de gestion de code source.	 
Stack Overflow	admin	Communauté de questions/réponses pour développeurs.	 
CSS-Tricks	admin	Astuces, articles et guides sur le CSS et le web design.	 
MDN Web Docs	admin	Documentation complète pour HTML, CSS, JavaScript et plus.	 

Précédent 1 2 3 Suivant

## Flux RSS

Pour tester le bon fonctionnement du flux RSS, nous avons d'abord consulté le fichier *xml* disponible sur [/feed](#).

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<rss version="2.0">
  <channel>
    <title>Watson web app</title>
    <link>http://localhost:1234</link>
    <description>Derniers liens ajoutés</description>
    <item>
      <title>MySQL</title>
      <link>https://mysql.com/</link>
      <description>Système de gestion de base de données open source.</description>
    </item>
    <item>
      <title>PHP</title>
      <link>https://www.php.net/</link>
      <description>Site officiel du langage PHP.</description>
    </item>
    <item>
      <title>Symfony</title>
      <link>https://symfony.com/</link>
      <description>Framework PHP robuste et modulaire.</description>
    </item>
    <item>
      <title>Laravel</title>
      <link>https://laravel.com/</link>
      <description>Framework PHP moderne pour le développement web.</description>
    </item>
    <item>
      <title>Express</title>
      <link>https://expressjs.com/</link>
      <description>Framework web minimaliste pour Node.js.</description>
    </item>
    <item>
      <title>Node.js</title>
      <link>https://nodejs.org/</link>
      <description>Environnement d'exécution JavaScript côté serveur.</description>
    </item>
    <item>
      <title>Webpack</title>
      <link>https://webpack.js.org/</link>
      <description>Module bundler pour JavaScript.</description>
    </item>
  </channel>
</rss>
```

Ensuite nous avons installé l'extension *Feeder* sur le navigateur *Chrome* et nous nous sommes rendus sur la même URL. Les 15 derniers liens ajoutés y apparaissaient correctement sur l'interface de l'application.

The screenshot shows the Watson web application interface. On the left is a sidebar with various icons: a house (Home), a search bar (Recherche...), a list icon (Tous les posts), a mail icon (Non lu), a star icon (Articles étoilés), a pen icon (Notes), a plus sign (Watson web a...), a gear (Configuration), a question mark (Aide), and a filter icon (Filtrer les flux...). The main area has a header "Tous les posts". Below it is a "Filtres" section with a dropdown menu set to "Affichage". A vertical scroll bar is visible on the right side of the main content area. The content area lists posts under categories: MySQL, PHP, Symfony, Laravel, Express, Node.js, Webpack, Sass, and Tailwind CSS. Each category has one post listed, all from "Watson web app" and published "14 mins" ago. A message at the bottom right says "Aucun post sélectionné" and "Ouvrez un article dans la barre à gauche pour commencer à consulter les flux".

Nous avons ajouté ensuite un nouveau lien "test" dans l'application Watson pour vérifier si l'affichage était bien dynamique : une notification concernant la publication de ce nouveau lien est correctement apparue.

The screenshot shows a dark-themed interface for a social media or news feed application. On the left is a sidebar with various icons: Home, Search (Recherche...), Tous les posts, Non lu (1), Articles étoilés, Notes, Watson web a... (1), and a Plus icon. Below these are settings and help icons. A search bar at the top right contains the placeholder "Recherche...". The main area is titled "Tous les posts" and includes filters for "Filtres" and "Affichage". The feed lists several posts:

- test**: Watson web app · 2 mins ago
- MySQL**: Watson web app · 4 mins ago
- PHP**: Watson web app · 4 mins ago
- Symfony**: Watson web app · 4 mins ago
- Laravel**: Watson web app · 4 mins ago
- Express**: Watson web app · 4 mins ago
- Node.js**: Watson web app · 4 mins ago
- Webpack**: Watson web app · 4 mins ago
- Sass**

On the right side, there is a detailed view for the "PHP" post, which includes a thumbnail, the title "PHP", a timestamp ("Watson web app · il y a 5 mins"), a link to "Site officiel du langage PHP.", and a "Lire plus" button. A small orange circular icon with a white speech mark is located in the bottom right corner of the main content area.