PROJET N°3
ANNUAIRE COLLABORATIF DE SITES D'INFORMATIONS

DELAIN ARTHUR
BETHOULE YOHANN
CARDOSO ALEX
PERROT YANNIS
POINT JONATHAN

Encadré par SAMIR CHAFIK

IUT de Clermont-Ferrand, Département Informatique, Année 2017-2018

Nous autorisons la diffusion de notre rapport sur l’intranet de l’IUT.

Remerciements

Merci à M. Chafik Samir pour avoir encadré notre projet et avoir prit du temps pour nous.
Merci à Mme. Geneviève Castanet pour son aide dans la gestion de notre projet.

Sommaire

1. Introduction

Notre projet « Wib » est une application web comprenant un site et une extension de navigateur, dont l'idée originelle revient à Arthur Delain. Ce projet a pour but d'être un annuaire de sites d'informations, auquel les utilisateurs pourront activement participer. L'équipe de développement est constituée d'Arthur Delain -chef de projet-, Alex Cardoso, Yohann Bethoule, Yannis Perrot et Jonathan Point, et nous sommes encadrés par Chafik Samir.

Le site internet du journal « Le Monde » publie depuis 2017 un moteur de recherche de sites d'informations appelé le Décodex. Or, il nous semble regrettable de juger de la qualité d'un journal par l'avis d'un groupe restreint de rédacteurs, qui plus est employés par un groupe concurrent appartenant à un parti privé. La problématique de notre projet était donc de recréer le même concept d'annuaire de sites d'informations, mais avec une dimension collaborative, à la manière d'un wiki.

Ainsi, chaque visiteur de notre site internet pourra consulter la moyenne des notes données à un site d'informations par les autres utilisateurs ainsi que leurs commentaires. L'objectif de cette participation active des utilisateurs est de pouvoir présenter des indices de qualité et de fiabilité des médias du web qui soient les plus neutres et objectifs possibles.

Néanmoins, notre expérience personnelle nous a laissé penser qu'il serait peu probable qu'une personne surfant sur internet et ouvrant un article sur un site qu'elle ne connaît pas prenne la peine de rechercher notre site, puis de rechercher le site d'informations concerné dans notre base de données, avant de fermer ce nouvel onglet pour revenir à sa navigation. Nous avons donc décidé de développer une extension de navigateur en plus de notre site web, afin de permettre un gain de temps significatif dans l'utilisation de notre application. Son but est de présenter un aperçu rapide des informations disponibles à propos du site sur lequel l'utilisateur se trouve, ainsi que lui donner la possibilité de noter ou commenter lui-même ce site, directement depuis la bulle d'extension, et ce sans jamais quitter l'onglet courant.

2. Présentation du sujet

2.1 Cahier des charges

L'EXTENSION DE NAVIGATEUR – ou add-on – est un petit logiciel intégré à un navigateur web, et qui permet de personnaliser l'expérience de l'internaute. Elle se base sur les technologies standards du web : HTML, CSS et JavaScript. De manière générale, une extension doit suivre un seul but bien défini. L'interface utilisateur d'un add-on peut prendre diverses formes selon l'objectif recherché : un bouton dans la barre d'extension, une fenêtre pop-up, ou même remplacer complètement la page active. (dans un premier temps pour Mozilla Firefox, à déployer ensuite sur les autres navigateurs..).

Notre extension sera matérialisée par une vignette dans la barre des boutons du navigateur. Cette vignette répondra à un code couleur (rouge, orange, vert) pour indiquer en un seul coup d'œil à l'utilisateur la qualité du site qu'il visite. Ce code couleur est défini en fonction des moyennes des notes des utilisateurs sur le site.

Au clic sur la vignette, une petite fenêtre s'ouvre sous la vignette, qui affichera la fiche de l'article ou du site sur lequel se trouve l'internaute. Si le site n'est pas référencé, un bouton permettra à l'utilisateur de l'ajouter à notre base de données. Si l'utilisateur n'est pas connecté, il pourra se connecter et ainsi donner lui-même son avis.

Les deux champs de note seront représentés par 5 étoiles. Lorsque l'utilisateur passe sa souris sur une barre d'étoile, celle-ci s'adapte dynamiquement et permet ainsi à l'utilisateur de donner sa note concernant le critère jugé.

LE SITE WEB permet :
- de parcourir les sites internet ou articles de notre base de données, et de rechercher parmi ceux-ci
- d'afficher la fiche d'un site ou un article
- de donner son avis concernant un site ou un article, en modifiant directement la fiche de celui-ci

Afin de stocker les sites web, les articles, et toutes les données en rapport, nous devrons mettre en place une base de données et les requêtes adaptées à nos besoins.

Pour faciliter la compréhension entre les utilisateurs, nous avons choisi d'organiser les informations concernant un article dans une fiche qui contient les informations suivantes : nom de domaine du site, courte description, une note de fiabilité et une note de cohérence sur 5, et une zone de commentaires. N'importe quel internaute pourra accéder à cette fiche sur notre site, et si il est connecté, donner lui-même son avis. Il aura pour cela plusieurs moyens : la description du site pourra être éditée par n'importe quel utilisateur connecté, à la manière d'un wiki ; il pourra noter lui-même les critères de qualité et de fiabilité, sa note venant s'ajouter à toutes les autres pour donner la moyenne du site sur chaque critère ; enfin, il pourra ajouter un commentaire que les autres utilisateurs pourront ensuite consulter.

2.2. Choix techniques

Node.js

Wib étant une application web en ligne, nous avons d’abord dû choisir comment implémenter notre serveur afin de gérer l’accès et l’interaction des utilisateurs aux données. Nous avons choisi pour cela Node.js, un environnement de développement bas-niveau basé sur le langage Javascript.

WebExtensions

Afin de développer notre extension de navigateur, nous nous sommes tout d’abord tournés vers le navigateur Mozilla Firefox, car c’est un navigateur très ouvert et adapté au développement, avec de nombreux outils mis à disposition des programmeurs. Ainsi, nous avons choisi d’utiliser le nouveau système proposée par Mozilla, les WebExtensions. Ce format d’add-on permet la portabilité sur Google Chrome, Opera et Edge, avec dans le pire des cas de minimes changements à effectuer.

MySQL

Pour gérer les données que nous aurons à utiliser et dans un soucis de performance & de libre utilisation, nous avons choisi d'utiliser MySQL comme système de gestion de base de données. MySQL est puissant, libre, gratuit et facile à utiliser ; il convenait parfaitement aux besoins de notre projet. Il est également très complet et très clair dans son fonctionnement. MySQL s'intègre de manière rapide et facile avec la technologie Node.js et il nous permet de gérer les données du projet de manière simple.

3. Analyse et gestion de projet

3.1. Analyse conceptuelle

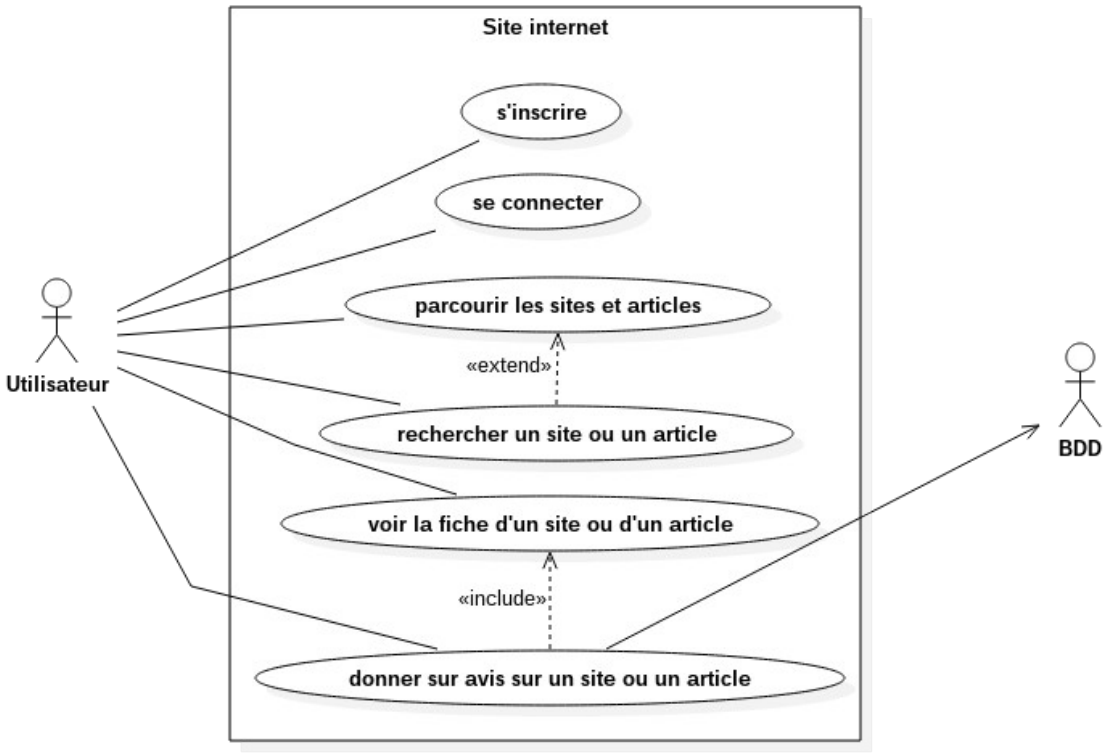


Figure 1: diagramme de cas d'utilisation

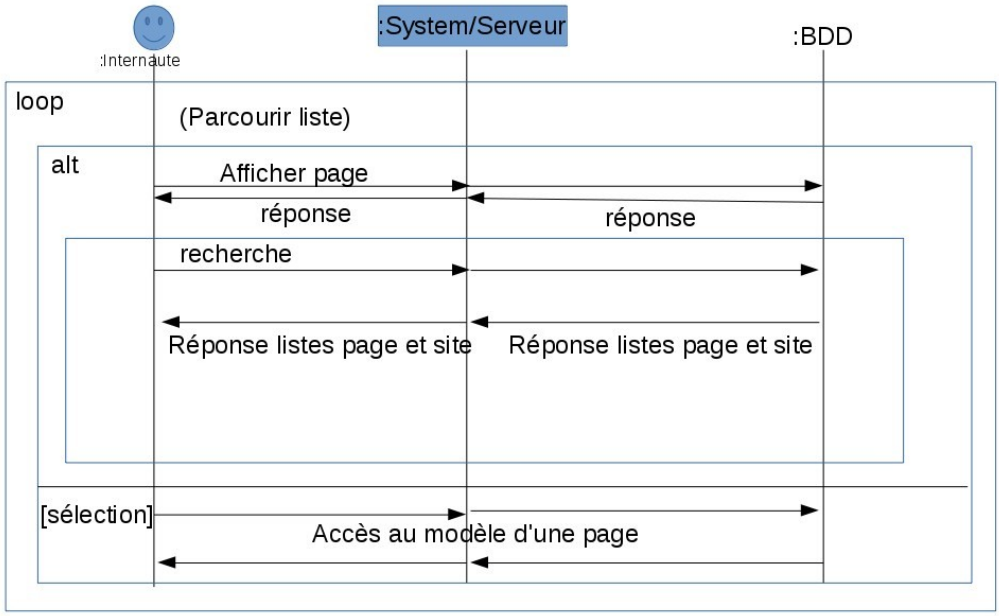


Figure 2: Diagramme de séquence-système de la page 'Recherche'

Le diagramme ci-dessus montrant que nous pouvons parcourir la liste des pages et des sites enregistrés et ainsi voir leur notes. Ensuite, nous pouvons, effectuer une recherche et ainsi le serveur nous renverra de nouvelles listes à parcourir, ou bien sélectionner une page et ainsi être redirigés vers celle ci.

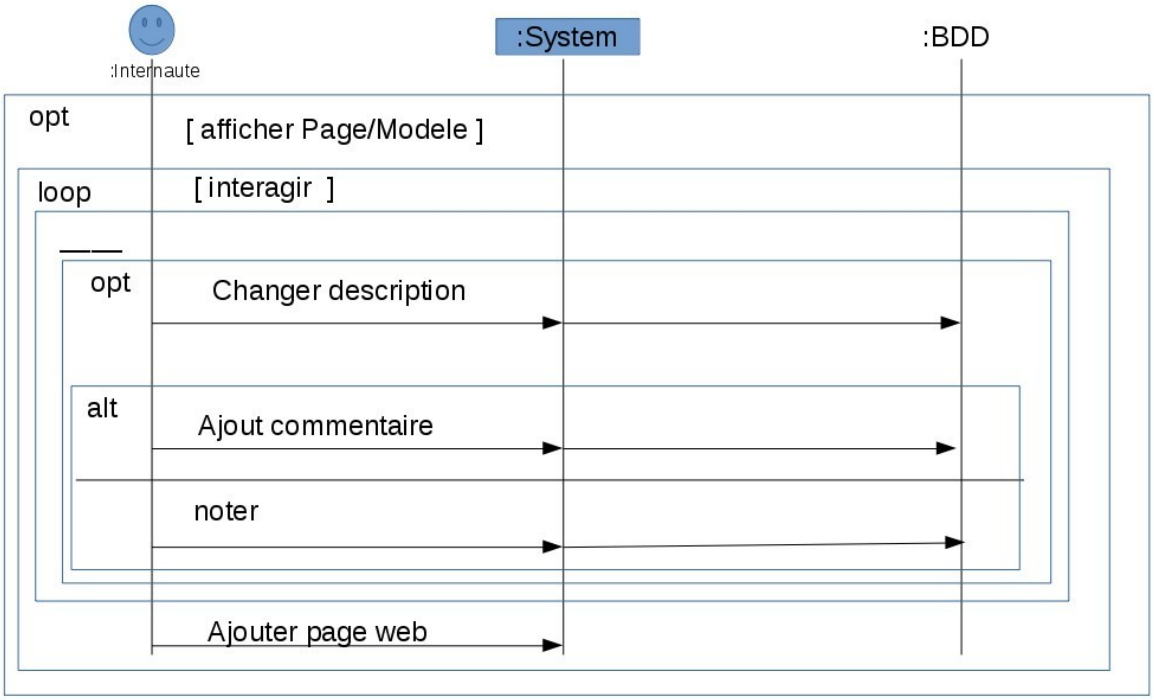


Figure 3: Diagramme de séquence-système de la page d'un article

Ce diagramme montre qu'un utilisateur connecté peut modifier la description ou bien modifier ses notes personnelles sur la page donnée (nos notes personnelles correspondant à celles de notre fiche sur la dite page, c'est celle-la qui est modifiée). Si il n'est pas connecté, il est renvoyé vers la page de connexion.

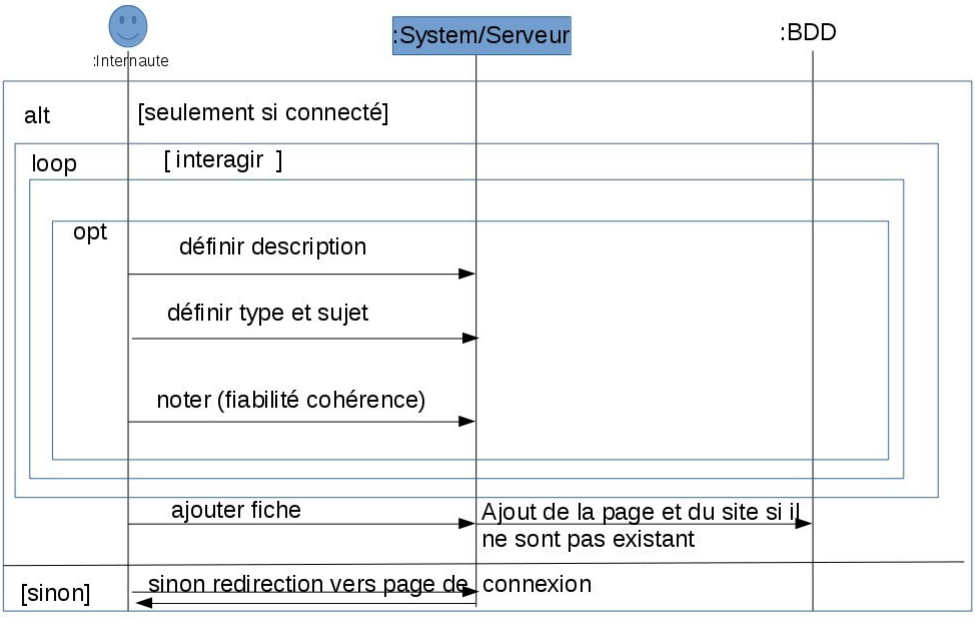


Figure 4: Diagramme de séquence-système de la page 'Fiche'

Ce diagramme montre que nous pouvons, si nous somme connectés, ajouter une fiche sur une page grâce à une URL. Nous pouvons aussi ajouter une description personnelle, ainsi qu'un type, un sujet, une note de cohérence, et une note de fiabilité. Si la page et le site correspondant ne sont pas répertoriés alors ils sont ajoutés. Si nous ne sommes pas connectés lors d'une interaction, nous sommes directement redirigés vers la page de connexion.

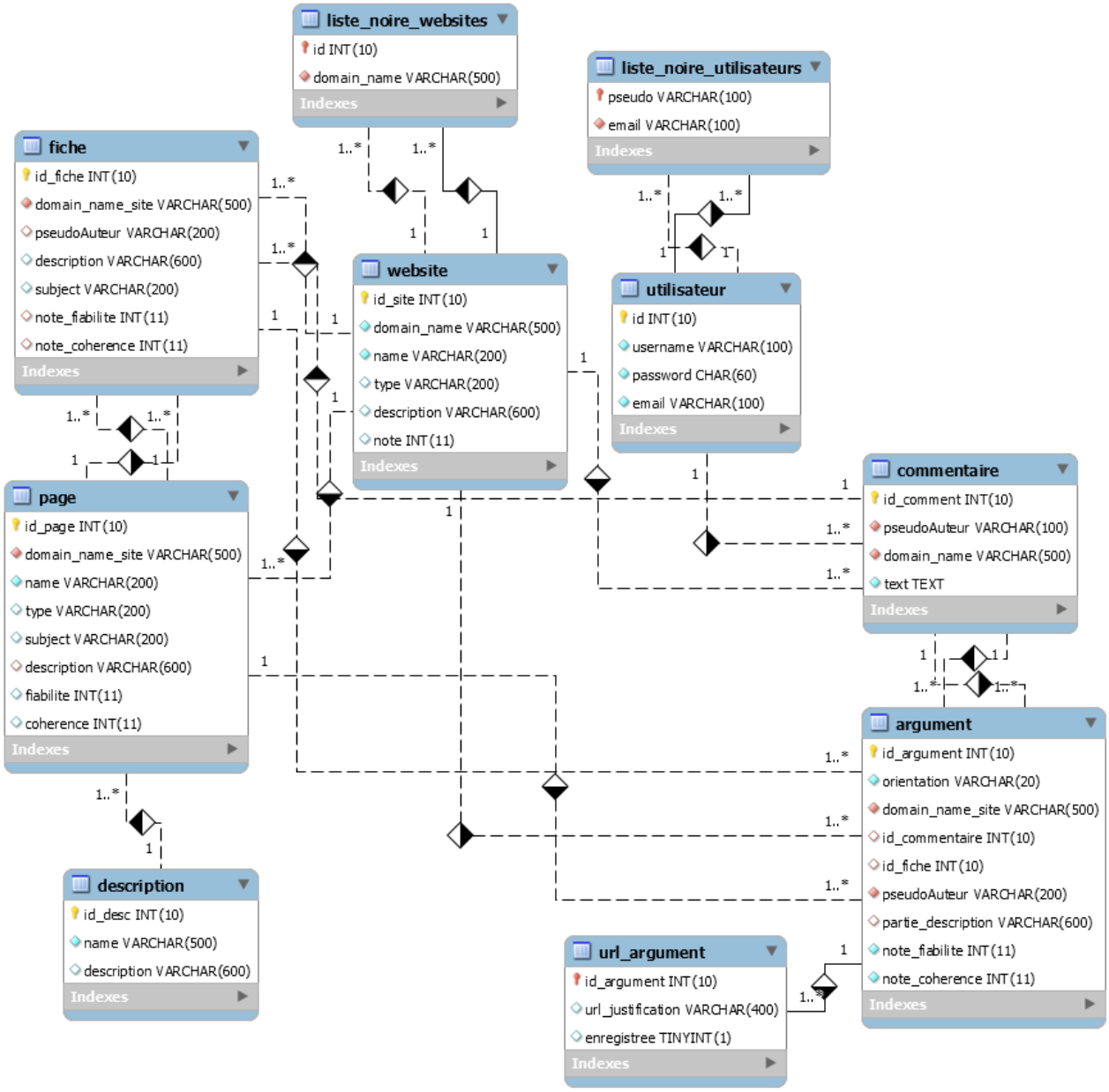


Figure 5: Modèle de la base de données

- La table WEBSITE regroupe tous les sites internets que nous enregistrons et que les utilisateurs pourront noter & commenter.
- La table UTILISATEUR regroupe tous les utilisateurs du site : ils ont un nom d'utilisateur, un mail et un mot de passe.
- La table FICHE regroupe les fiches d'informations d'un site internet.
- La table PAGE regroupe les informations d'un article en particulier d'un site.
- La table DESCRIPTION correspond à la description d'une page.
- La table COMMENTAIRE regroupe tous les commentaires issus d'utilisateurs.

- La table ARGUMENT concerne les commentaires des utilisateurs et permet de montrer l'orientation globale d'un avis.
- La table URL_ARGUMENT regroupe les liens des arguments.
- Il y a également 2 tables de listes noires concernant les utilisateurs et les sites internet : une fois ajouté à cette table, l'utilisateur ou le site concerné ne pourront plus être ajouté aux tables classiques d'utilisation.

3.2. Gestion de projet

Dates : 30 octobre 2017 - 13 mars 2018.

Nombres de Taches 95.

Gantt Réel Collectif 19 mars 2018

Tâches 2

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
Description projet	06/11/17	06/11/17	2	Création projet	Groupe entier
Description termes & objectifs	06/11/17	06/11/17	1	Création projet	Groupe entier
Réunion périodique	06/11/17	06/11/17	0.5	Réunion périodique	Groupe entier + tuteur
Répartition tâches	06/11/17	06/11/17	0.5	Projection/gestion projet	Groupe entier
Mise en place arborescence	07/11/17	13/11/17	1	Mise en place projet	A. Delain
Diagramme de classes	07/11/17	13/11/17	1	Mise en place projet	J. Point
Commencer cahier des charges	07/11/17	13/11/17	1	Gestion Projet	Y. Bethoule
Design & maquettes	07/11/17	13/11/17	3 (Yannis) + 4 (Alex)	Design	Y. Perrot, A. Cardoso
Mise en place base serveur Node.js	07/11/17	13/11/17	4	Mise en place projet	A. Delain
Rassemblement travail effectué & discussions	13/11/17	13/11/17	1	Gestion projet	Groupe entier
Choix design	13/11/17	13/11/17	0.5	Design	Groupe entier
Choix IDE & Logiciels	13/11/17	13/11/17	1	Design	Groupe entier
Réunion périodique	13/11/17	13/11/17	20 min	Réunion périodique	Groupe entier + tuteur
Répartition tâches	13/11/17	13/11/17	0.5	Gestion projet	Groupe entier
Digrammes d'activités	14/11/17	20/11/17	1	Mise en place projet	J. Point

Gantt Réel Collectif 19 mars 2018

Tâches 3

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
Transférer diagrammes sur STARUML	14/11/17	20/11/17	0.5	Clareté projet	J. Point
Commencement site web	14/11/17	20/11/17	2,5 (yannis) + 3 (Alex)	Mise en place site web	Y. Perrot, A. Cardoso
Commencement GANTT collectifs	14/11/17	20/11/17	1	Gestion projet	Y. Bethoule, J. Point
Commencement codage de la bulle d'extension	14/11/17	20/11/17	3.5	Application	Y. Bethoule, A. Delain
Rassemblement travail effectué	20/11/17	20/11/17	1.5	Gestion projet	Groupe entier
Discussions autour des rapports d'activités & des technologies utulisées	20/11/17	20/11/17	0.5	Gestion projet	Groupe entier
Réunion périodique	20/11/17	20/11/17	0.5	Réunion périodique	Groupe entier + tuteur
Répartition tâches	20/11/17	20/11/17	0.5	Gestion projet	Groupe entier
Démarrage serveur NodeJS	21/11/17	27/11/17	1.5	Application	A. Delain
Ajout librairies Node.js	21/11/17	27/11/17	1	Application	A. Delain
Documentation droits (législation)	21/11/17	27/11/17	1.5	Projet	A. Delain
Ajout nouvelles pages HTML	21/11/17	27/11/17	3	Site web	A. Cardoso
Recherche & choix des librairies à utiliser pour les graphiques	21/11/17	27/11/17	1	Application	Y. Perrot
Commencement MCD & MLD	21/11/17	27/11/17	2	Base de données	Y. Bethoule
Cahier des charges	21/11/17	27/11/17	1	Gestion projet	Y. Bethoule

Tâches

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
Poursuite diagrammes	21/11/17	27/11/17	2	Projet	J. Point
Rassemblement travail effectué, discussions	27/11/17	27/11/17	2	Gestion projet	Groupe entier
Discussions droits d'utilisation	27/11/17	27/11/17	0.5	Projet	Groupe entier
Réunion périodique	27/11/17	27/11/17	0.5	Réunion périodique	Groupe entier + Tuteur
Répartition des tâches	27/11/17	27/11/17	0.5	Gestion projet	Groupe entier
Recherches & tests BD MySQL	28/11/17	04/12/17	2	Bases de données	Y. Perrot
Fin cahier des charges	28/11/17	04/12/17	2	Gestion projet	A. Delain, Y. Bethoule
Design extension	28/11/17	04/12/17	2	Design	J. Point
Poursuite serveur Node.js	28/11/17	04/12/17	3	Serveur	A. Delain
Poursuite site web	28/11/17	04/12/17	2.5	Site web	A. Cardoso
Rassemblement travail effectué, discussions diverses	04/12/17	04/12/17	2.5	Gestion projet	Groupe entier
Répartition tâches	04/12/17	04/12/17	0.5	Gestion projet	Groupe entier
Poursuite site web	05/12/17	11/12/17	2	Site web	A. Cardoso
Poursuite serveur Node.js	05/12/17	11/12/17	3	Serveur	A. Delain
Apprentissage Node.js	05/12/17	11/12/17	1.5	Projet	Y. Perrot

Tâches

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
Poursuite MCD & BD	05/12/17	11/12/17	1.5	Bases de données	Y. Bethoule
Poursuite diagrammes	05/12/17	11/12/17	1	Gestion projet	J. Point
Rassemblement du travail effectué	11/12/17	11/12/17	1	Gestion projet	Groupe entier
Réunion périodique	11/12/17	11/12/17	0.5	Réunion périodique	Groupe entier + Tuteur
Répartition des tâches	11/12/17	11/12/17	0.5	Gestion projet	Groupe entier
Finission GANTT	12/12/17	18/12/17	2	Gestion projet	Y. Perrot
Préparation oral entrainement	12/12/17	18/12/17	1.5	Projet (oral)	Y. Bethoule
Préparation diaporama oral d'entrainement	12/12/17	18/12/17	1.5	Projet (oral)	J. Point
Poursuite serveur Node.js	12/12/17	18/12/17	2	Serveur	A. Delain
Poursuite site web	12/12/17	18/12/17	1.5	Site web	A. Cardoso
Rassemblement travail effectué	18/12/17	18/12/17	1	Gestion projet	Groupe entier
Oral d'entrainement	18/12/17	18/12/17	1	Projet (oral)	Groupe entier + Tuteur
Répartition des tâches	18/12/17	18/12/17	0.5	Gestion projet	Groupe entier
creation pre BDD, creation interaction utilisateur, affichage de donnée	15/01/18	15/01/18	4	Site web	A. Delain
poursuite site web (design)	15/01/18	15/01/18	4	Site Web	A. Cardoso

Gantt Réel Collectif

19 mars 2018

6

Tâches

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
Analyse node js	15/01/18	15/01/18	4	Site Web & BDD	Y. Perrot
poursuite création extension	15/01/18	15/01/18	4	Extension navigateur	Y. Bethoule
création objet inetermediaire et restructuration BDD	22/01/18	22/01/18	4	Site Web	A. Delain
Analyse node js et base de donnée	22/01/18	22/01/18	4	Site Web	Y. Perrot
Analyse node js	22/01/18	22/01/18	4	Site Web	J. Point
extension poursuite	22/01/18	22/01/18	4	Extension	Y. Bethoule
design et autre	30/10/17	30/10/17	4	Site Web	A.Cardoso
remise en forme des chemin & continue construction site	29/01/18	29/01/18	4	Site Web	A. Delain
reconception Base De Donnee	29/01/18	29/01/18	4	Base de Donnee	Y. Perrot
design , ajout slide , script javascript et résoudre problème passage html to jade	29/01/18	29/01/18	4	Site Web	A. Cardoso
extension poursuite	29/01/18	29/01/18	4	Extension	Y. Bethoule
rapport	29/01/18	29/01/18	4	gestion de projet	Y. Bethoule
création d'objet intermédiaire à la base de donné et continue création interactions utilisateurs	05/02/18	05/02/18	4	Site Web	A. Delain
création requete base de donnée	05/02/18	05/02/18	4	BDD	Y. Perrot
extension poursuite	05/02/18	05/02/18	4	Extension	Y. Bethoule

Gantt Réel Collectif

19 mars 2018

7

Tâches

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
poursuite construction page web et design	05/02/18	05/02/18	4	Site Web	A. Cardoso
interactions utilisateurs , écriture objet , mise en place inscription / connexion	12/02/18	12/02/18	4	Site Web	A. Delain
poursuite extension	12/02/18	12/02/18	4	Extension	Y. Bethoule
poursuite design	12/02/18	12/02/18	4	Site Web	A. Cardoso
réparation base de donnée	12/02/18	12/02/18	4	BDD	Y. Perrot
continuer écriture des calculs et changement des données , mise en place inscription / connexion , réparation	26/02/18	26/02/18	4	Site Web	A. Delain
poursuite création requete base de donnée	26/02/18	26/02/18	4	BDD	Y. Perrot
poursuite design	26/02/18	26/02/18	4	Site Web	A. Cardoso
exetension requete et affichage	26/02/18	26/02/18	4	Extension	Y. Bethoule
réparation et nettoyage du code + test +interactions utilisateurs	05/03/18	05/03/18	4	Site Web	A. Delain
réparation et construction BDD	05/03/18	05/03/18	4	BDD	Y. Perrot
extension poursuite	05/03/18	05/03/18	4	Extension	Y. Bethoule
integration diverse et poursuite design	05/03/18	05/03/18	4	Site Web	A. Cardoso
nettoyage du code , réglage de problèmes , mise en place image	12/03/18	12/03/18	4	Site Web	A. Delain
ajout et integration dans site web	12/03/18	12/03/18	4	Site Web & BDD	Y. Perrot

Gantt Réel Collectif

19 mars 2018

8

Tâches

Nom	Date de début	Date de fin	Durée (en h)	Justification	Ressources
finition extension	12/03/18	12/03/18	4	Extension	Y. Bethoule
finition design	12/03/18	12/03/18	4	Site Web	A. Cardoso
diagramme de gantt	12/03/18	12/03/18	4	Projet	Groupe entier
rapport	12/03/18	12/03/18	4	Projet	Groupe entier
orale preparation	12/03/18	12/03/18	4	Projet	Groupe entier

4. Développement

4.4 Design du site web et navigation

Le design et l'aspect du site a été effectué par Alex Cardoso.
Les technologie HTML, CSS, JavaScript, PHP et Jade ont été utilisé pour le développement du front-end.

Dans un premier temps des croquis et des maquettes de l'aspect du site furent réalisés. Après concertation, un design épuré avec comme couleurs principales le noire et le blanc.
Dans un second temps Alex Cardoso a fourni un site statique simplement en HTML/CSS/JS. Ce site a ensuite évolué en lien avec le développement côté serveur.

Problèmes rencontrés :
- Les convertisseurs Jade en ligne ne fournissent pas une bonne syntaxe lorsque les balises on a la fois une id et une classe.
- Le site devait être responsive. Certains éléments comme le carrousel ou la barre de navigation ont dû être modifié.

Finalement le site se divise en 5 parties accessibles depuis n'importe quelle page grâce à une barre de navigation.

Page d'accueil

Figure 6: Capture d'écran de la page d'accueil

La page principale est la page d'accueil, qui présente l'application et propose un bouton pour télécharger l'extension.

On peut voir un cadre de connexion en haut à droite du site : ce cadre est présent sur toutes les pages du site.

Recherche

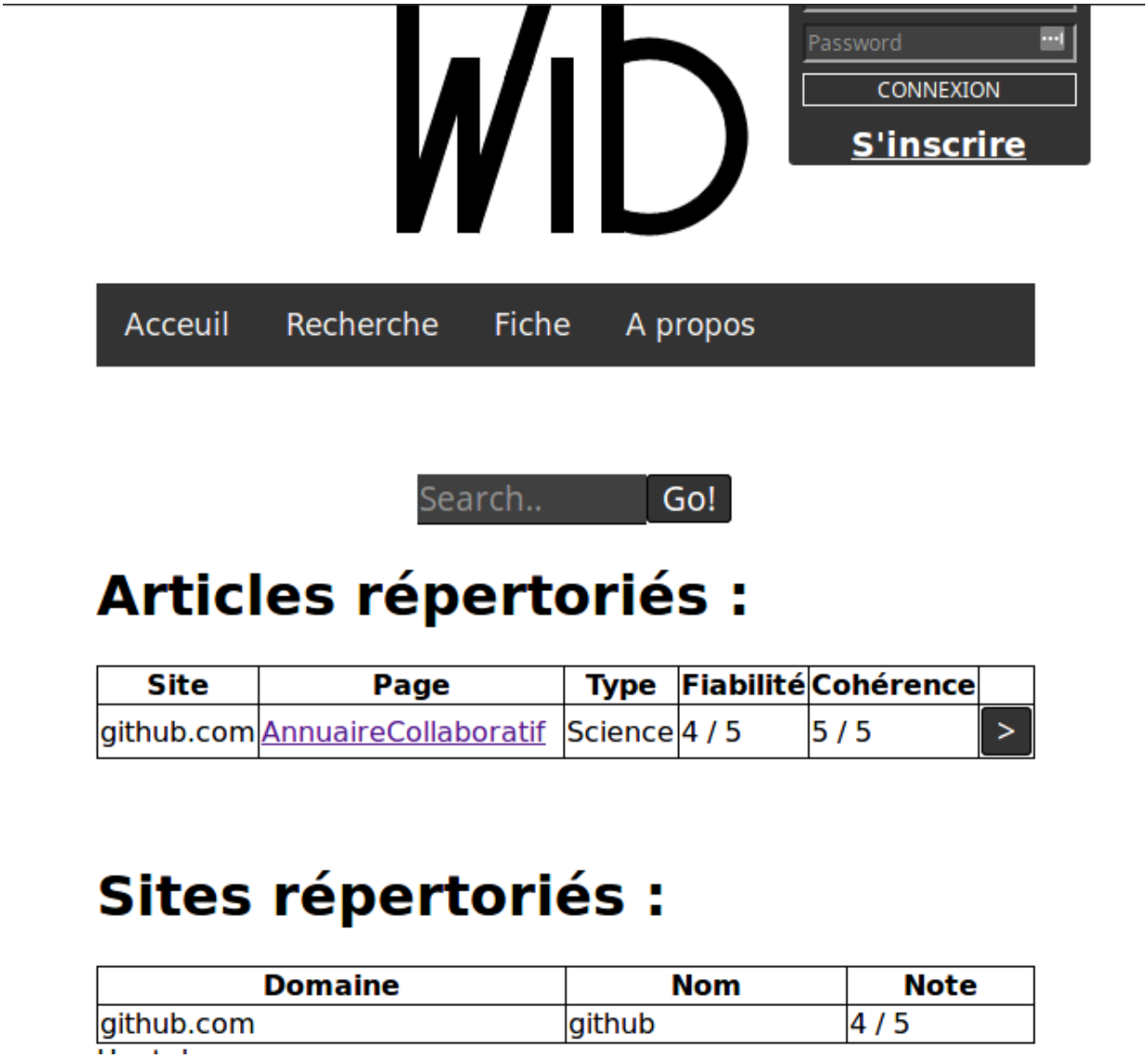


Figure 7: Capture d'écran de la page de recherche

L'utilisateur peut parcourir les articles et les sites recensés sur notre application, ainsi qu'effectuer une recherche. En cliquant sur le bouton à droite d'un article, il peut se rendre sur la fiche de celui-ci.

Fiche

Figure 8: Capture d'écran de la page fiche

Sur la page 'Fiche', un utilisateur connecté peut créer une fiche concernant un article, que celui-ci existe déjà dans la base de données ou pas. Il lui suffit de copie-coller l'URL de l'article en question dans le champ 'Lien', puis de renseigner les autres champs comme il le souhaite.

Inscription

La page 'Inscription', accessible depuis le lien 'S'inscrire' qui se trouve dans le cadre de connexion, permet à un visiteur de créer un compte en renseignant 3 champs : pseudonyme, mot de passe et adresse email. Cette dernière est ensuite vérifiée grâce à l'envoi d'un mail contenant un mot de passe de confirmation.

A propos

Une page contenant un petit texte expliquant qui nous sommes, développeurs de l'application, et quelle est notre démarche.

Le site est entièrement responsive grâce à l'utilisation des Flex Box, mode de mise en page CSS prévoyant une disposition des éléments d'une page de telle sorte que ces éléments possèdent un comportement prévisible lorsqu'ils doivent s'accommoder de différentes tailles d'écrans/appareils. Le site est donc utilisable sur mobile et tablette.

Figure 9: Capture d'écran du site adapté sur un écran de type smartphone

4.2. Programmation côté serveur

-Mise en place et choix des modules :

Tout d'abord, après analyse des différentes façons d'utiliser node js, et cela par rapport à des projets existants sur github ainsi que des tutoriels trouvés sur internet, nous avons du choisir les différents modules d'utilisations, ou en tout cas les principaux afin de démarrer la création du serveur web. L'ajout de nouveaux modules se fait grâce à *npm* qui est le gestionnaire de paquets officiel de Node.js et qui fonctionne en ligne de commande.

Après plusieurs essais sur des projets tests, nous avons décidé d'utiliser l'IDE PhpStorm qui intègre facilement de nouveaux modules et qui nous permet, entre autres, de générer directement un serveur http (dans bin/www et cela avec le module du même nom). Celui-ci crée aussi le package.json où tous les modules utilisés sont répertoriés, le fichier app qui s'occupe d'initialiser les modules utilisés et de rendre possible l'accès aux pages web ainsi qu'aux dossiers principaux, et donc à la répartition des différentes parties du serveur.

Ensuite, il fallait mettre en place le système de « routage » afin d'accéder aux différentes pages du site internet, pour cela nous avons utilisé le module express et son routeur. Situé dans routes/routes.js, le routeur permet, selon une requête donnée à une certaine adresse, de répondre en conséquence. Nous pouvons donc rediriger les requêtes GET, POST, etc, vers des fonctions

d'autres pages ou bien envoyer directement une page web en retour ,générée ou bien se trouvant dans view/.

Après lecture de nombreux articles, et d'après même PhpStorm, l'utilisation de jade est fortement répandue pour les projets node js/express d'où notre utilisation de ce moteur de template qui est en fait comme une page HTML mais simplifiée, qui crée la page html seulement a l'affichage de celle-ci.

Nous somme ensuite passés par différentes étapes :

-Les middleware :

Ce sont des fonctions qui peuvent accéder à l'objet Request et response (respectivement req et res) et elles servent entre autres à exécuter les différentes parties du code, nous avons donc dû apprendre à utiliser ces différents attributs.

-Affichage données Json :

Après avoir eu accès aux pages, nous avons dû afficher des données et donc les envoyer côté serveur. L'envoi se fait en JSON, grâce au module, body-parser, et nous pouvons directement envoyer les « objets » en les nommant. L'affichage se fait après du côté vue.

-Mise en place des contrôleur :

Le projet avançant et les fonctionnalités augmentant pour certaines pages, il a fallu mettre en place un système de contrôleur pour chacune d'entre elles. Le dossier routes/controller fut donc créé et il fallut juste ajouter un contrôleur pour chaque page complexe (modele,recherche,…).

-Mise en place BDD Mysql :

Il fallut ensuite mettre en place une base de données, nous avons choisi mysql. Le module du même nom nous a donc permis de faire des requêtes sur notre base de donnée, et aussi à la générer si elle n'existe pas sur la machine utilisée.

-Ajout / suppression / modification d'information :

Il fallut ensuite faire les ponts entre la base de donnée et les interactions sur les données. Il fallut donc relier les objets Json aux requêtes de la base de données.

-Création de pseudo classe :

Pour pouvoir, améliorer la visibilité du projet nous avons ensuite mis en place des pseudo-classes. Celles-ci permettent d'un coté de faire les requêtes vers la base de donnée et de l'autre de mettre en place différentes méthodes pour des actions précises, comme des traitements avec les informations des requêtes.

L'utilisation du module squeelize fut d'abord envisagée. Celui-ci sert « d'interprète » objet venant de bases de données, il permet donc de changer de base de données en simulant des objets. A la place de simples fonctions «simulant » des pseudo-objets ont été utilisées.

Ainsi nous avons crée un système afin que ce soient les utilisateurs qui déterminent les appréciations (note fiabilité et cohérence sur les pages et donc leur fiches, et note simple sur les site). Et donc plusieurs fonctionnalités ont été conçues:

-lorsqu'un utilisateur souhaite ajouter une fiche descriptive d'une page, la page correspondante est ajoutée si elle n'a pas été préalablement enregistrée,ainsi que le site correspondant,

-les notes des pages, fiabilité et cohérence, se font grâce à la moyenne des notes correspondantes sur l'ensemble de leurs fiches et leur type et sujet sont ceux les plus sélectionnés sur l'ensemble de leurs fiches

-les notes des sites se font sur la moyennes des notes de leurs pages.

-Inscription et connexion :

Le système d'ajout/suppression/modification des données commençant à se concrétiser, nous avons mis en place des utilisateurs. Ceux-ci pourront donc s'inscrire puis se connecter, ce qui leur permettra d' interagir avec les données (ajout,suppression,..).

Cela se fait avec le module passport, qui gère les requêtes à la base de données sur les utilisateurs. Il gère aussi la connexion « active » (la session) au serveur node js avec le module nommé session, cela permettant de sauvegarder le « statut de connexion » de l'utilisateur.

Nous avons en plus mis en place une confirmation d'inscription par email avec le module nodemailer , se qui permet de vérifier l'email inscrit par l'utilisateur et aussi d'éviter les inscriptions abusives.

Enfin, il fallut donc avec le système de routage, demander si l'utilisateur était bien connecté avant de permettre certaines interactions (ajout commentaires, notes,….).

Modules principaux :

express : le module principal, c'est lui qui permet l'accès aux différentes pages du site web, de réceptionner les différentes requêtes utilisateur et de vérifier leur connexion si la requête est restreinte que pour les utilisateur.

http : requêtes http de base et création de la base du serveur.

session & passport : passport est un middleware qui permet l'inscription et la connexion des différents utilisateurs, quand un utilisateur est connecté, il est disponible grâce à session. Il sert donc à vérifier si l'utilisateur est bien connecté (avec la méthode isAuthentificate dans routes/routes.js).

nodemailer : envoi de mail, sert pour la vérification de demande d'inscription (envoi grâce a gmail).

mysql : module d'utilisation de Mysql pour créer la base de données et pour faire les différentes requêtes utiles.

body-Parser : permet de faire passer des objets Json du serveur au client et inversement.

Path : permet l'accès à /view pour les fichiers html et jade et à /public pour le css, les images ...

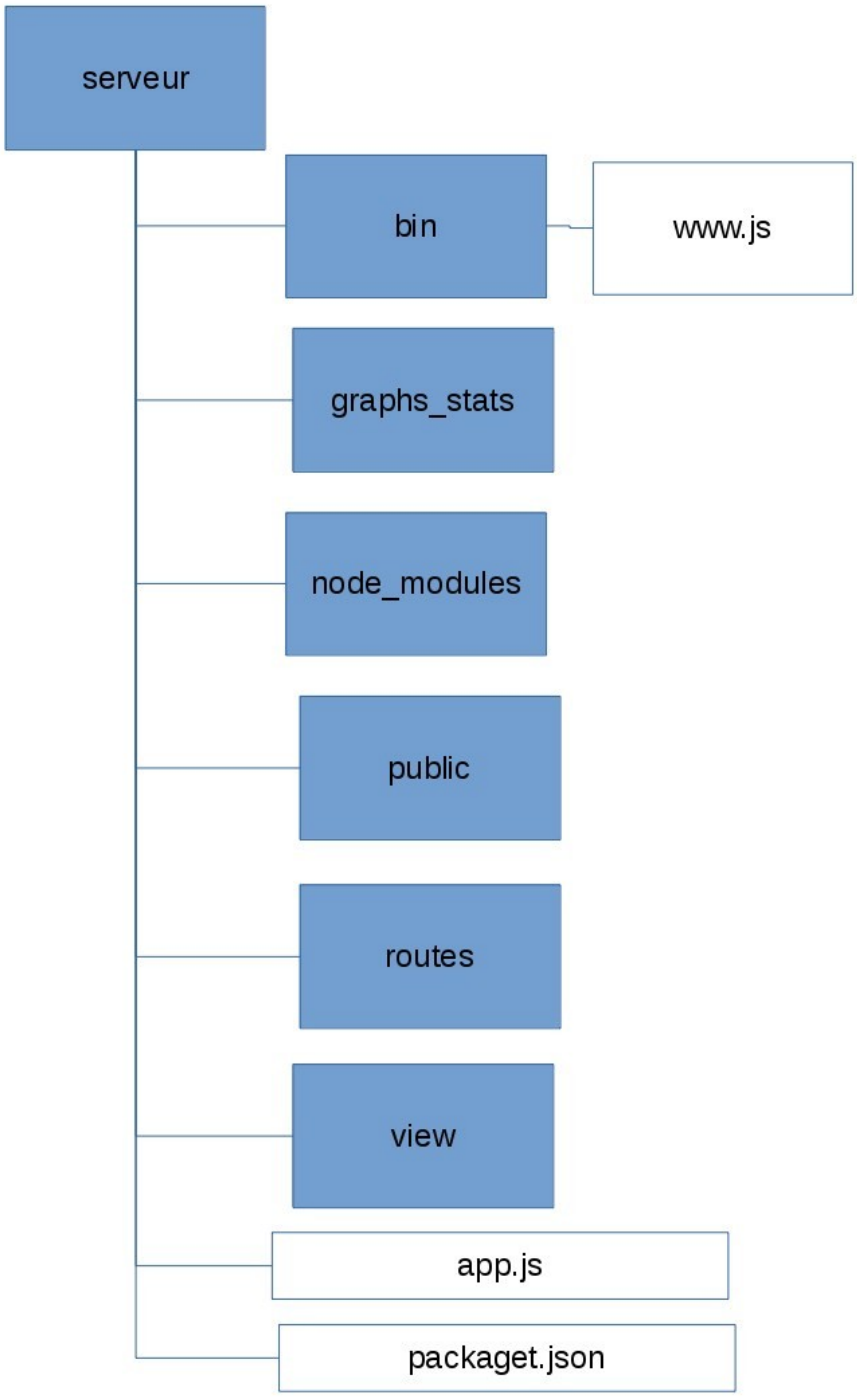
Passport : pour l'authentification , l'inscription ...

problèmes rencontrés :

-choix des liens url pour l'accès aux différentes pages : plusieurs changements après des url non cohérents. Dans un site internet les chemins sont les plus optimaux, et les choix précédents tendaient vers la complexité.

-liaison view et public : de nombreux problèmes sur le passage de html/javascript a jade, l'utilisation de convertisseur ayant de nombreuses erreurs, il fallut modifier ces dites pages à la main afin de trouver les incohérences.

-la simulation des Classes : l'utilisation de midellware gênant certaines exécutions avec l'attente de retour de donnée de la base de donnée,il fallut donc pour éviter les méthodes sur des objets « undefined », refaire des require dans les méthodes de retour (« callback«).



-Contient le serveur http, ouvert sur 3001

-non mis en place

-Contient l'ensemble des modules installés

-Dossier disponible depuis les page de view, permettant entre autre le css, les images...

-contient le système de routage les interactions BDD,

-Contient toutes les pages web

-lance les modules , routes/routes.js et /routes/users.js

-Package contenant les différents modules et leur version

Figure 10: Représentation de l'Arborescence

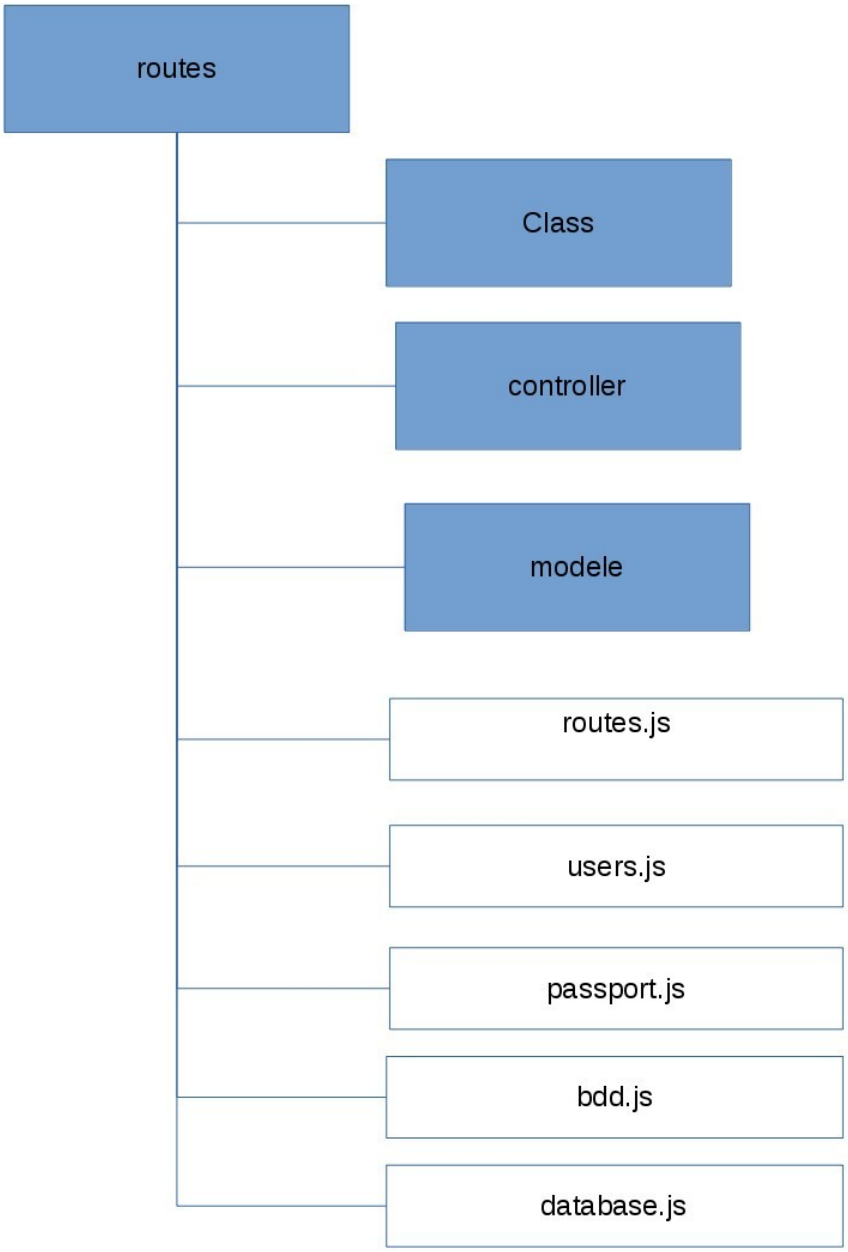
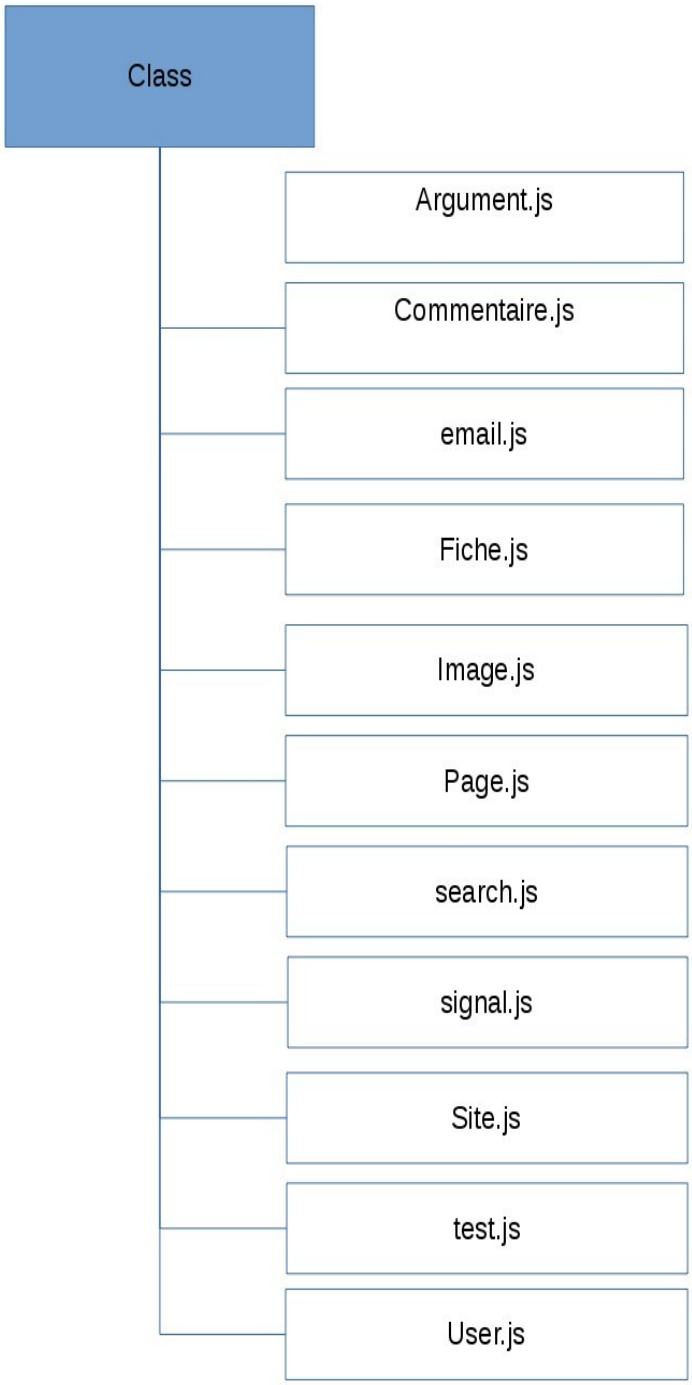


Figure 11: Représentation de *routes*, le répertoire principale du serveur



-Contient principalement des pseudo-classes/fonction, qui possèdent des méthodes, des traitements et des requêtes sql

-pseudo classe des Arguments (pour ou contre une page). Non intégrée

-pseudo classe Commentaire de Page

-permet l'envoi de mail lors de l'inscription, ainsi que la création d'un code secret de confirmation

-pseudo classe Fiche, pour juger les différentes pages(fiche descriptive)

-pseudo classe Image. Non intégrée.

-pseudo classe Page, pour les différents articles ajoutés

-pour faire des recherches sur les attributs de page et site grâce a des requêtes sql

-permet de signaler une page ou un site. Non intégrée

-pseudo classe Site

-permet de faire des tests d'exécution

-pseudo classe User

Figure 12: Représentation de *Class*, le répertoire des pseudo-classes et des différentes interactions.

4.3. Système de gestion de base de données

La gestion de la base de données a été gérée en grande partie par Yannis PERROT.

La version de MySQL que nous avons utilisé est la 5,7,21. C'est la version la plus récente et la plus stable de MySQL.

Tout d'abord, il a fallu inclure le module MySQL dans le serveur, c'est-à-dire l'inclure dans Node.js. Une fois fait, dans les fichiers concernés, il a fallu indiquer les identifiants, les requêtes qui nous serviront (SELECT, UPDATE, etc.) ainsi que la fonction de création des tables.

- Le fichier « users.js » comprends un utilisateur unique « wib_bdd » qui nous sert d'admin et que tous les développeurs utilisent.
- Le fichier « bdd.js » comprends les variables des tables à utiliser dans toute l'architecture du serveur.
- Le fichier « database.js » comprends les requêtes de création de table.
- Le dossier « modele » comprends plusieurs fichiers, et dans chacun de ces fichiers, il y a des requêtes concernant chaque table.

Les requêtes sont contenues dans des fonctions Node.js que l'on peut appeler à n'importe quel moment. Cela constitue une fluidité dans l'utilisation de la base de données et cela nous permet de récupérer les données par exemple pour les injecter dans les vues quand on en a besoin.

4.4. Extension de navigateur

L'extension de navigateur a été développée en majeure partie par Yohann Bethoule.

Le premier objectif que nous avons cherché à atteindre dans le développement de l'extension a été d'obtenir une vignette dans la barre des boutons du navigateur Firefox, qui affiche une fenêtre pop-up quand on clique dessus.

La première étape pour cela est de créer un dossier, et de placer dans celui-ci un fichier '*manifest.json*', un document JSON décrivant notre module.

```
{
  "manifest_version": 2,
  "name": "Wib",
  "version": "1.0",

  "description": "Adds a browser action icon to the toolbar. Click the button to open a popup and show more informations displayed by Wib.",

  "icons": {
    "48": "icons/icon_ext48p48.png"
  },

  "permissions": [
    "activeTab",
    "<all_urls>",
  ],

  "browser_action": {
    "default_icon": "icons/icon_ext32.png",
    "default_title": "Wib",
    "default_popup": "popup/wib.html"
  }
}
```

Code 1 : Manifest.json

Ce fichier correspond à l'en-tête de notre extension. Détaillons les informations qu'il contient :

- « **manifest_version** » : indique la version du manifest (ce qui représente en outre la version de WebExtension utilisée). La version actuellement usitée est la version 2, comme inscrit dans le fichier ci-dessus.

- « **name** » : le nom de l'extension, qui permet de l'identifier. Il est préférable que ce nom soit court (moins de 45 caractères) pour des raisons d'affichage dans les navigateurs.

- « **version** » : la version de l'extension (renseigné par le développeur).

Ces trois premières entrées doivent impérativement figurer dans le manifest.

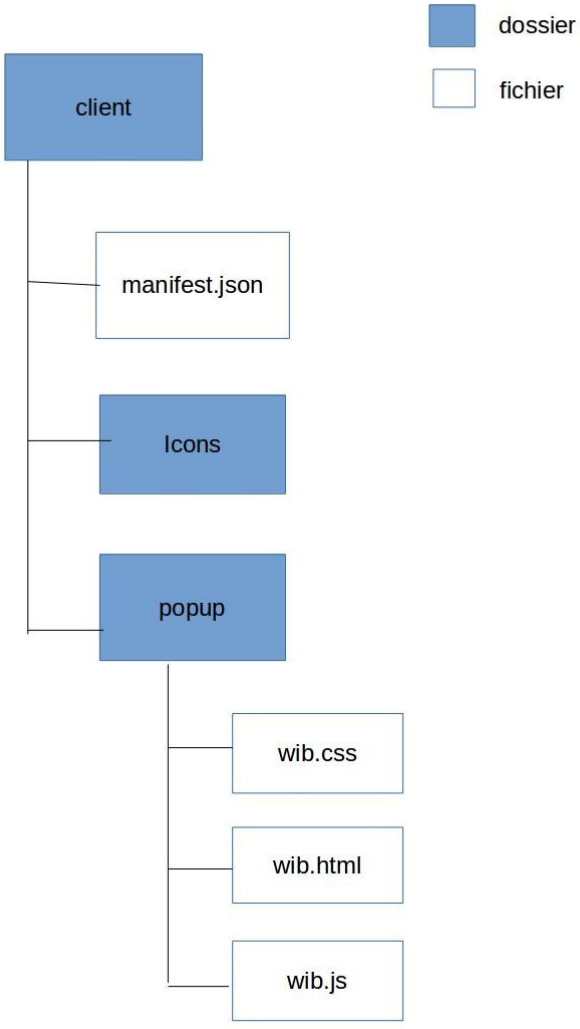
- « **description** » : une courte description de l'utilité de l'extension. Cette description est affichée dans le gestionnaire de modules du navigateur.

- « **icons** » : permet de spécifier l'icône qui apparaîtra dans le gestionnaire de modules.

- « **permissions** » : les permissions requises par l'extension. La permission *activeTab* permet d'accéder et de modifier l'onglet actif. <*all_urls*> autorise les connexions avec n'importe quel autre nom de domaine ; Cela nous permet d'envoyer des requêtes HTTP vers notre serveur.

- « **browser action** » : spécifie la fenêtre pop-up : son icône dans la barre des tâches, son titre et la page HTML associée.

Voilà pour l'en-tête de notre extension.



La figure ci-contre nous permet de visualiser l'organisation des fichiers de notre extension. On trouve le fichier '*manifest.json*' à la racine, ainsi qu'un dossier '*Icons*' et un dossier '*popup*'. '*Icons*' contient les icônes de notre extension (celle affichée dans le gestionnaire de module et celle de la barre des tâches). '*Popup*' contient les documents qui forment notre fenêtre pop-up : un document HTML, sa page de style CSS et un document de scripts JavaScript.

Figure : arborescence simplifiée des fichiers de l'extension

Ainsi, lorsque l'utilisateur clique sur l'icône de notre add-on dans la barre des tâches de son navigateur, c'est la page '*wib.html*' qui est affichée, et qui charge les fichier '*wib.css*' et '*wib.js*'. La taille de la fenêtre pop-up s'adapte automatiquement à la taille du corps de la page HTML.

Afin de pouvoir faire le lien entre le site que la page sur laquelle l'utilisateur se trouve et notre base de données, l'extension doit pouvoir récupérer l'URL de l'onglet actif. Nous avons suivi pour cela les exemples de la documentation MDN.

```
function updateActiveTab(tabs) {

  function isSupportedProtocol(urlString) {
    var supportedProtocols = ["https:", "http:", "ftp:"];
    var url = document.createElement('a');
    url.href = urlString;
    return supportedProtocols.indexOf(url.protocol) != -1;
  }

  function updateTab(tabs) {
    if (tabs[0]) {
      currentTab = tabs[0];
      if (isSupportedProtocol(currentTab.url)) {
            var domainName=urlToDomainName(currentTab.url);
                  document.getElementById("currentURL").innerHTML="<a
                        href=http://"+domainName+">"+domainName+"";
      }
      else {
        return "<p> Le format web n'est pas supporté par l'extension. </p>"
      }
    }
  }

  var gettingActiveTab = browser.tabs.query({active: true, currentWindow: true});
  gettingActiveTab.then(updateTab);
}

// listen to URL changes
browser.tabs.onUpdated.addListener(updateActiveTab);

//listen to tab switching
browser.tabs.onActivated.addListener(updateActiveTab);

//listen to window switching
browser.windows.onFocusChanged.addListener(updateActiveTab);

// update when the extension loads initially
updateActiveTab();
```

Code : extrait de *wib.js (1)*

Pour résumer ces lignes de code, le script écoute les changements dans les onglets (lorsque l'URL de l'onglet actif change, ou lorsque l'utilisateur change d'onglet ou de fenêtre). Lorsqu'un changement survient, la fonction *updateActiveTab* est appelée. Celle-ci s'occupe de vérifier que le protocole utilisé de l'URL est correct, puis modifie dynamiquement le contenu de la fenêtre pop-up pour l'adapter aux données disponibles sur la fenêtre actuelle.

Dans un second temps, il a fallu faire la liaison entre l'extension et les données de notre serveur. Pour cela, le script javascript du module devait envoyer une requête http vers notre serveur, pour récupérer ensuite les données en format JSON avant de les afficher dans la bulle d'extension.

```
function getInfos(url){
    var xhr = new XMLHttpRequest();
    if(url.startsWith("http") || url.startsWith("https")){
      if(url.startsWith("https")){
        url = url.slice(8,-1);
      }else{
        url = url.slice(7,-1);
      }
    }
    var requestedURL = "http://localhost:3001/extension/site/"+ encodeURIComponent(url);
    console.log(requestedURL);
    xhr.open('GET', requestedURL, true);
    xhr.addEventListener("load", function() {
      if(xhr.readyState == 4 && xhr.status == 200){
        //traitement des données
      }
    });
    xhr.send() ;
  }
```

Code : extrait de *wib.js (2)*

L'extrait ci-dessus est une version simplifiée de la fonction *getInfos(url)*, qui permet de demander au serveur les informations concernant le site internet sur lequel l'utilisateur se trouve. Elle reçoit en paramètre l'URL du site, que nous avons récupérée dans l'exemple précédent, puis crée et envoie une requête HTTP vers le serveur. Ce dernier répond à la requête en renvoyant la fiche du site au fomat JSON.

Cet extrait est un exemple simplifié, le script doit aussi récupérer les informations concernant l'article, ainsi que modifier l'affichage de la bulle d'extension en conséquence. Il est suffisant pour illustrer la liaison entre l'extension et le serveur car celle-ci se résume à l'envoi de requêtes HTTP et à la mise en forme de données au format JSON.

5. Bilan technique

1. État final du projet

A la date où nous rendons le projet, nous avons réussi à créer une application web fonctionnelle. Le site internet permet une inscription sécurisée, avec un email de confirmation et un mot de passe hachés avant d'être enregistré dans la base de données, ainsi qu'une connexion. Les principales fonctionnalités de notre application sont correctement implémentées : parcours et recherches dans notre base de données, possibilité d'ajouter une nouvelle fiche concernant un article, affichage des informations concernant un site ou un article et modification de celles-ci pour donner son avis.

L'extension de navigateur fonctionne elle aussi correctement. La bulle d'extension se met dynamiquement à jour lorsque l'onglet actif change, et affiche correctement les informations lorsqu'elles existent.

2. Améliorations possibles

Certains points que nous avions inscrits au cahier des charges au début du projet n'ont pas été réalisés.

Concernant le site internet, nous souhaitons rajouter:

- d'autre statistiques disponibles concernant un article ou un site web : évolution de la fréquentation, évolution des notes

- la possibilité de proposer une image qui illustre un site ou une page

- la possibilité de donner un argument en faveur ou contre un article. Un argument est un commentaire argumenté : un texte entré par l'utilisateur qui explique le sujet de l'argument, associé à un lien source qui vienne appuyer cette argument, par exemple un autre article.

Concernant l'extension de navigateur, seule la partie affichage des informations est disponible. Pour se connecter et donner son avis, l'utilisateur doit pour le moment suivre le lien vers la fiche de la page sur laquelle il se trouve et se connecter sur notre site internet.

De plus, l'aspect graphique générale de notre application pourrait être amélioré.

6. Conclusion

Notre projet tutoré nous a donc beaucoup apporté, tant en terme technique que humain. Après que l'idée principal du projet fut discutée et claire, l'annuaire collaboratif de site d'information ou en des termes plus compréhensible « répertoire de jugement d'utilisateur sur des sites d'informations », nous avons pu passé à la réalisation.

Tout d'abord, techniquement, nous avons dû beaucoup apprendre par nous même. N'ayant des cours de javascript seulement pendant cette période, nous avons donc appris grâce à des tutoriels et d'autres projets libres d'accès et aussi en nous indiquant entre nous les démarches à suivre. Il nous fallut donc, faire divers choix, suivre ou non des techniques utilisées et adapter notre projet en conséquence.

Cela nous a aussi fait relativiser sur la quantité et la nature des fonctionnalités voulues, puisque nous en avons modifiées ou supprimées certaines durant l'écriture de notre projet. Nous ne voulions ni surcharger certaines pages, ni ajouter des surplus, ce qui aurait nui à la visibilité et la compréhension du fonctionnement du site.

De plus, certains problèmes venant de la conception parasitant la réalisation, nous avons donc recréé et redéfini certaines parties dont la base de donnée, cela occasionnant une perte de temps.

Nous avons aussi appris à travailler en commun, pour la cohérence de ce projet. Il y eu donc quelques problèmes de version et de liaison entre parties. La répartition des tâches imposées quelques fois des intégrations sur des parties des différentes personnes, d’où ces problèmes. De plus, un membre du groupe a quitté celui-ci en pleine écriture du projet, ce qui à complexifié les tâches à accomplir pour chacun et nous a obligé à redéfinir certaines parties.

Il y eu aussi des problèmes persistants dans certaines parties ce qui en ralentissaient d'autres, nous nous sommes donc adaptés en conséquence.

Pour finir, ce projet nous a permis de nous rapprocher d'une expérience professionnelle, et nous a bien montré les dérives, techniques et humaines que cela pouvait amener. Cela nous a aussi montré l'importance d'avoir une conception sûre et stable afin que cela ne se répercute pas tout au long de la réalisation.

7. English summary

« Wib » is an online application, made of a website and a browser add-on. Its object is to be a news websites collaborative directory, which means that any user can participate to it. The main goal is to collect our user's opinions about news websites and article, so our application can display an objective indication of one website/article quality.

We chose to develop a browser add-on so people who install it will be able to get a quick summary of the informations we have about the website and article he is on, without interrupting their surfing. To develop a lasting and stable add-on, we used Mozilla WebExtension, an API

In order to program the back-end of our website, we chose to use Node.JS, so we had to learn a new way of developping for the web, event programming. On the website, anyone can consult the card of a website or a page, which displays its title, a short description, and ratings out of 5. Rating objects for an article are the redaction consistency and the quality of the arguments, and the displayed values are the mean of all users ratings, and a website mark is the mean of the marks of all its articles. A connected user can modify the description of a website or article, and rate it.

The add-on takes shape of a vignette in the browser toolbar, which opens a pop-up windown when clicked. The pop-up displays the mark of the website, and the two marks of the article, if they exist in our database. It does so by listening on changes of the current tab opened in the browser, and sending a HTTP request to our Node.JS server, which return the needed datas in JSON format.

Developing this project led us to learn a lot about web technologies. First, we didn't know how to program in javascript, Node.JS, or even how to make a browser add-on. We had to learn most of it by ourselves, by finding tutorials and documentation on internet. We also got to experience the technical and human problems that can happen in a professional projet.

Bibliographie

-*Mozilla Developer Network* (MDN) :
https://developer.mozilla.org/fr/Add-ons/WebExtensions
-Chrome extensions :
https://developer.chrome.com/extension
-Tutoriel Node JS :
https://openclassrooms.com/courses/des-applications-ultra-rapides-avec-node-js
https://nodejs.developpez.com/tutoriels/javascript/redecouvrir-javascript-avec-nodejs/#LIII-C-2
https://www.w3schools.com/nodejs/
https://www.supinfo.com/articles/single/2179-nodejs-expressjs-authentification-avec-passportjs
-express
http://expressjs.com/fr/starter/installing.html http://www.hacksparrow.com/express-js-tutorial.html
-mysql :
http://www.developper-jeux-video.com/category/node-js/page/2/
https://code.tutsplus.com/tutorials/using-passport-with-sequelize-and-mysql--cms-27537
-exemple :
https://github.com/manjeshpv/node-express-passport-mysql
https://github.com/tedeh/jayson/tree/master/examples
https://github.com/expressjs/express/tree/master/examples
https://github.com/sequelize/express-example