UNIVERSITE D'ORLEANS

POLYTECH ORLÉANS

Rapport de Stage

JEREMIE PASSERAT

MAITRE DE STAGE : M. RODOPLHE WEBER
ENSEIGNANT TUTEUR : M. MATHIEU EXBRAYAT

SMA0ST03 – UE34 Stage | 21/09/2021 | version 1.0

Table des matières

Remerciements

Je tiens à remercier sur cette page toutes les personnes qui ont contribuées au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

En premier lieu, je tiens à remercier M. Rodolphe Weber et la fondation Polytech pour avoir mis en place ce stage et pour m'avoir fait confiance et laissé une grande liberté d'action pour définir les contours de la mission de stage qui m'a été confiée.

Je joins des remerciements à l'attention de la direction de Polytech Orléans, M. Christophe LEGER, pour m'avoir accueilli dans son établissement ; et également à l'attention du personnel du laboratoire PRISME que j'ai pu côtoyer pendant six mois.

Je tiens également à remercier vivement M. Rodolphe WEBER, mon maître de stage pendant toute la durée du stage, pour m'avoir accompagné au quotidien dans les diverses étapes de la réalisation de ma mission de stage.

Je tiens à inclure M. Albert CAU, de la fondation Polytech, pour nous avoir aiguillé à diverses étapes du développement du projet, mais aussi le personnel de la fondation Polytech, dont Mme Alice DALMENESCHE, pour la gestion administrative de mon stage.

Je remercie M. Rodolphe WEBER et David PASSERAT pour leur relecture attentive de ce rapport.

Enfin, je tiens à remercier M. Matthieu EXBRAYAT pour son accompagnement durant ce stage et également le personnel enseignant du département informatique de l'université d'Orléans pour m'avoir encadré pendant ces deux années de Master.

Introduction

Dans le cadre de la deuxième année de Master, un stage de mise en application de tous les enseignements théoriques vus est compris dans le semestre 4. D'une durée de quatre à six mois, il correspond à un premier aperçu de la réalité concrète du métier de développeur informatique.

La mission

La mission proposée par le stage consiste en la conception et la réalisation d'une solution de création, partage, maintenance et gestion des quizz (terminologie ensuite modifiée en contenus pédagogiques) dans un cadre multi utilisateurs.

Le but de cet outil est donc de proposer une solution palliant certaines limitations de Moodle (et d'autres plateformes analogues) qui ne proposent qu'une création, visualisation, édition des contenus (du point de vue de l'enseignant).

Qui est concerné (durant le développement – la finalité)

La finalité de ce produit est d'être proposé, aux écoles du réseau Polytech.

Dans le cadre de ce stage le site a été transmis à plusieurs enseignants de la fondation Polytech pour avis & retours, qui ont pu découvrir de manière anticipée la solution OpenQuizz.

Ce qui a été réalisé

A l'heure ou ces lignes sont écrites l'objectif initial d'OpenQuizz (avant qu'il évolue de manière plus universaliste) est réalisé : les quizz de Moodle sont importables, il est possible de les prévisualiser, de les modifier, de les mettre à jour et de les réexporter sur Moodle.

Des investigations seront menées auprès des partenaires du réseau Polytech pour étudier la faisabilité et la viabilité de la poursuite du projet.

CONTEXTE DU STAGE

LE RESEAU POLYTECH

Parmi les 204 écoles d'Ingénieurs accréditées en France, le réseau Polytech regroupe les 15 écoles d'ingénieurs publiques présentes en France, ainsi que quatre écoles associées

Créé en 2004 pour accroitre la visibilité des écoles d'ingénieur publiques, elle est présente dans la majorité des grandes villes françaises (Paris, Lyon, Lille, Marseille, …)

Proposant plus de 100 spécialités issues de 12 domaines scientifiques variés (Informatique, Electronique, Mécanique, …), le réseau Polytech accueille 17000 étudiants aussi bien en formation initiale qu'en alternance, mais également plus de 1300 doctorants grâce aux nombreux laboratoires ou équipes de recherche associés.

La fondation Polytech, créée en 2017, est une structure qui a pour objectif : « *d'accompagner les écoles Polytech dans leurs missions nationales et internationales, tout en restant en cohérence avec les politiques de site définies par les universités d'appartenance* »

L'ECOLE POLYTECHNIQUE DE L'UNIVERSITE D'ORLEANS

Polytech Orléans, fondée en 2002 via la fusion de l'école supérieure de l'énergie et des matériaux et de l'école supérieure des procédés électroniques et optiques, accueille 1350 étudiants (classiques ou alternants) et une centaine d'enseignants.

L'école propose sept spécialités pour le cycle d'ingénieurs (Bac + 3 à Bac + 5) ainsi qu'un parcours préparatoire intégré (Bac à Bac +2) permettant de préparer les futurs élèves à intégrer le cycle d'ingénieurs. Il est possible de préparer un *master scientifique ou d'administration des entreprises* en parallèle de la formation d'ingénieur, et également de poursuivre, après le diplôme, avec un diplôme de Mastère (Bac + 6)

Les enseignements sont dispensés sur deux sites, tous deux présents sur le campus de l'Université d'Orléans (Vinci et Galilée), et, pour une spécialité, dans une autre ville : la formation Ingénieur·e Génie industriel pour la cosmétique, la pharmacie et l'agro-alimentaire est dispensée à Chartres.

Forte de son fort taux d'enseignants-chercheurs (75% du personnel enseignant), l'école s'appuie sur huit laboratoires de recherche (dont le laboratoire PRISME, présenté dans la partie ci-dessous), lui permettant l'accueil de doctorants, de proposer une immersion à ses élèves concernant les problématiques de recherche en ingénierie mais aussi des recherches professionnelles en collaboration avec les professionnels de l'industrie (locaux ou internationaux).

LE LABORATOIRE PRISME

Le laboratoire Pluridisciplinaire de Recherche en Ingénierie des Systèmes, Mécanique et Energétique, ou PRISME, est un laboratoire de recherche conjoint de l'Université d'Orléans et de l'INSA Centre Val de Loire (EA 4229).

Il travaille sur un large panel de champs disciplinaires, qui va de la combustion dans les moteurs à la robotique en passant par le traitement du signal et de l'image ou l'aérodynamique et la mécanique des fluides.

Il comprend 90 enseignants chercheurs et 52 doctorants, répartis sur 5 sites et 4 villes (Chartres, Orléans, Bourges et Châteauroux).

Mon maître de stage, Rodolphe Weber, étant partie prenante du laboratoire PRISME, j'ai pu effectuer la partie présentielle de mon stage au sein des locaux du laboratoire PRISME (même si ma mission n'était en aucune façon directement liée aux activités du laboratoire).

LE PROJET OPENING

Le projet OpenING est un projet commun du réseau des écoles Polytech visant à l'hybridation des formations, qui comporte en particulier un volet pédagogique appelé SPOC, qui consiste à présenter des savoirs universitaires via des petites vidéos (avec l'inspiration des cours en lignes de type MOOC).

Le projet OpenQuizz et la mission de stage qui m'ont été confiées ont pour objectif de s'intégrer dans les projets pédagogiques portés par le projet OpenING.

REALISATIONS

PRESENTATION SYNTHETIQUE DE LA MISSION

Dans un premier temps je vais présenter une vue synthétique des enjeux de la mission et des démarches effectuées pour y répondre.

LE CONTEXTE / L'EXISTANT

Les plateformes sont devenues (encore plus grâce à la crise sanitaire) des outils privilégiés pour mettre en place des cours ou des évaluations.

Polytech Orléans, comme les autres structures de l'université d'Orléans-Tours, utilise une version personnalisée de Moodle, appelée Celene.

Cette plateforme propose, dans les options d'administration de tous ses cours, une sous-catégorie appelée Banque de Question, dans laquelle il est possible de créer des questions de types différents, d'en voir un aperçu, de les affecter à un e catégorie particulière (liée au cours) ou de les supprimer.

Selon certains enseignants, dont Rodolphe WEBER, il serait appréciable de pouvoir disposer d'autres fonctions pour gérer ces cours, pour plusieurs raisons :

- Le partage : Pouvoir reprendre une question pertinente d'un confrère du réseau Polytech, a laquelle on n'aurait pas pensé ou qui semble pertinente pour un enseignant.
- La synergie : Si on dispose d'un outil communautaire ou il est possible de suggérer des modifications à un confrère, la compétence collective peut amener a des échanges et des contenus de grande qualité pédagogique. Le suivi des modifications se faisant via un système de versionnage.
- La centralisation : Pouvoir disposer d'un grand nombre de contenus au même endroit.
- La diversité des contenus (objectif de l'application) : Pouvoir disposer d'une plateforme pouvant accueillir un grand nombre de contenus différents en son sein, et pas seulement du Moodle.

LA DEMARCHE APPLIQUEE

La première partie de la mission a consistée à définir le cadre de la mission, les fonctionnalités que le produit devait apporter et comment réaliser ces fonctionnalités

Dans ce cadre, une analyste fonctionnelle et une analyse technique ont été rédigées.

Il a ensuite été décidé de réaliser une version logicielle pour avoir un aperçu fonctionnel de l'application en python, qui servirait en particulier à observer comment implémenter la partie versionnage et quelles métadonnées conserver sur les contenus qu'on manipule.

Une fois satisfaits avec R.Weber, je me suis en suite attelé à imaginer a quoi pourrait ressembler les interfaces visuelles de l'application grâce à un outil de maquettage.

Une fois les maquettes validées, un essai visuel a été fait, sur le volet latéral de recherche, permettant de nous mettre d'accord sur la teneur visuelle à adopter pour OpenQuizz.

Ce volet réalisé, il a servi de base pour le site web complet. La version logicielle d'essai à servie de modèle pour la réalisation du web service.

Le web service, réalisé avec Flask, conserve une structure relativement simple :

- Un fichier de routage des adresses en entrées (qui permet également de lancer l'application)
- Quatre fichiers

Je vais ensuite entrer dans le détail de tous les points mentionnés ci-dessus.

L'ANALYSE FONCTIONNELLE

La première étape de ce stage a été d'établir une liste détaillée et précise des fonctionnalités offertes (dans le futur) par l'application.

Pour ce faire, nous sommes partis d'une ébauche de cahier des charges, rédigé par Rodolphe Weber, afin de déterminer comment axer nos recherches concernant les fonctionnalités à développer.

Version primitive

La première version de l'analyse fonctionnelle que nous avons réalisée était axée sur la manipulation de quizz. Le distinguo panier (quizz propriétaires d'un utilisateur) et banque (quizz publics des autres propriétaires) apparait, avec également la première version du processus de mise à jour d'un contenu.

Une des règles que nous nous sommes imposés pour la réflexion autour de ces fonctionnalités est de faire totalement abstraction de toute question technique (prévues ultérieurement). Par exemple, quand était discuté un système de versionnage des contenus, l'imaginer le plus générique possible et non directement inspiré de solutions existantes (git, svn) ou créables (solution « manuelle » de gestion des versions).

Suite à ça une première version rédigée (sous la forme d'un « PowerPoint ») de l'analyse fonctionnelle a été rédigée et présentée à M. CAU de la fondation Polytech. Cet échange, très constructif, a permis de nous conforter sur la vision globale que nous avions dessiné pour les fonctionnalités à donner à l'application mais aussi apporté quelques améliorations.

Une amélioration notable est d'abandonner la focalisation sur le fait que l'on manipule quz des quizz mais d'étendre le concept à tout fichier utile dans un cadre pédagogique (pour coller à l'objectif initial de la mission de stage) tout en gardant, dans un premier temps, un regard centré sur les quizz.

Version ‘définitive’

Suite à cette réunion et à la réflexion qu’elle a suscitée chez R. Weber et moi-même, une seconde version de l’analyse technique, cette fois ci rédigée sur un document texte, a vu le jour.

Ce document a ensuite été transmis à certaines personnes du réseau Polytech afin d’avoir un « œil extérieur » sur nos propositions. Cette transmission n’a cependant, malheureusement, rien permis de remonter à ce sujet.

Voici les principales fonctionnalités présentes dans cette version définitive (l’analyse fonctionnelle, plus détaillée, est disponible en **annexe 1**) :

1) L’utilisateur dispose de contenus dans son panier, de plusieurs sortes :

∗ Les contenus « personnels », qu’il a créé ou importé d’ailleurs. Si on considère le réseau des écoles Polytech, la source principale d’import est la plateforme Moodle. Mais l’application (nommée OpenQuizz) devra permettre l’accueil de n’importe quel type de fichier (on peut par exemple citer la solution h5p qui permet de créer des quizz avec beaucoup d’options, en particulier au niveau des ajouts de contenus multimédias associés au quizz).

L’import se fait en ‘uploadant’ un fichier associé à un contenu depuis un menu adapté au niveau du panier. Dans la même logique, il devra être possible d’exporter un fichier à partir d’OpenQuizz pour le réinsérer dans la plateforme pédagogique avec laquelle on travaille

Dans un futur proche il serait idéal que les imports / exports de contenu depuis d’autres plateformes puissent se faire de manière automatisée (tout en gardant la possibilité de les faire ‘à la main’, comme c’est le cas actuellement)

∗ Les contenus « tiers ». Ce sont des contenus créés ou importés par d’autres utilisateurs que l’utilisateur courant s’est approprié (via la banque).

∗ Les contenus en-cours. Ce sont des contenus que l’utilisateur est en train de modifier et, tant qu’aucune action particulière n’a été réalisée, ces contenus ne sont visibles par personne d’autres que leur auteur. Ils peuvent être de deux « types » :

Les contenus en-cours issus d’un contenu personnel -> l’utilisateur veut mettre à jour une de ses créations, il peut directement valider la transformation du contenu en cours comme étant la nouvelle version de son contenu.

Les contenus en cours issus d’un contenu tiers -> l’utilisateur veut modifier le travail d’un tiers et a la possibilité de soumettre à ce dernier les modifications qu’il a imaginées.

2) La banque permet de consulter tous les contenus personnels

* Chaque fois qu'un utilisateur « créée » un nouveau contenu, il est automatiquement disponible pour tous les autres utilisateurs.

* Chacun des contenus présents dans la banque peut être « mis en tiers [1]» par un utilisateur.

* Si un utilisateur s'est approprié un contenu dans son panier, celui-ci n'apparait plus dans sa banque.

3) D'autres mécanismes particuliers

* La gestion d'un versionnage pour chaque contenu, qui commence en version 1 et qui peut évoluer au fur et à mesure de sa « vie »

* Une gestion pour permettre la gestion des demandes de publication (accepter / refuser des mises à jour).

* Une gestion de tags (à plusieurs niveaux) associés à un contenu.

* La possibilité d'associer des contenus à une évaluation, permettant de conserver l'information sur un contenu ayant déjà été proposé aux élèves. Cette fonctionnalité n'est actuellement pas implantée dans OpenQuizz)

* Des possibilités de recherche pour restreindre les filtres selon certains critères

Ces points correspondent à notre base de travail pour la suite de la mission, mais ne sont pas figés dans le marbre. L'évolution du développement du projet est en effet en mesure de faire varier certains aspects d'OpenQuizz si des améliorations ou changements pertinents nous viennent en tête en cours de route.

[1] La « mise en tiers » correspond à l'action consistant, à partir de la banque, à s'approprier un contenu dans son panier

L'ANALYSE TECHNIQUE

Une fois les fonctionnalités validées, nous nous sommes penchés sur les outils techniques qui permettront de réaliser l'application

Cette réflexion s'est décidée selon cinq axes (non classés par ordre d'importance) :

1) Le framework pour faire l'API REST

Pour déterminer la solution logicielle à retenir, plusieurs points ont été vérifiés, soit la présence : d'un plugin de sécurité (JWT, OAUTH), de gestion relationnelle avec une base de données (ORM, ODM), d'un plugin de test, assez gros pour disposer d'une communauté et d'un support assez conséquent et idéalement pourvu d'un système de templating (dans un premier temps nous avons eu l'idée de créer les pages web avec, idée finalement abandonnée au profil de VueJS (voir point 3 ci-dessous)).

Ayant arbitrairement décidé de partir sur le langage Python, il ne restait plus qu'a choisir parmi les deux frameworks majeurs proposés par le langage, que sont Django et Flask.

L'aspect très simplifié de Flask a fait la différence (ayant juste les fonctionnalités de base pour réaliser une API REST, le reste étant construit à la demande via l'ajout de plugins adaptés).

2) Le système de gestion de données

Plusieurs éléments ayant été comparés (rapidité, pertinence de garder les contraintes relationnelles de manière sécurisée, …), la solution de type NoSQL (MongoDB dans notre cas) a été préférée aux SGBD relationnels car le côté orienté document de MongoDB nous a paru adapté à notre problématique (le panier d'un utilisateur et la banque n'étant au final que des collections de contenu).

De plus, si OpenQuizz devait grandir, MongoDb devrait être plus performant sur des très gros volumes de données.

Les problématiques d'intégrité relationnelle seront reproduites avec MongoDb (détaillées plus bas dans ce rapport), par exemple pour gérer le fait qu'un contenu doit (sauf à l'exception du contenu 'renié', voir **annexe 1** pour plus de détail) appartenir obligatoirement à un et un seul utilisateur.

3) Le framework pour l'ihm

(Pour être exact, le choix du framework n'a pas été décidé lors de l'analyse technique mais lors de la réalisation du prototype du volet latéral (avec Jquery)).

Trois solutions majeures sont présentes sur le « marché » pour répondre à la problématique de proposer des interfaces web : Angular, React et Jquery.

Ayant déjà vu Angular dans le cadre du deuxième semestre du Master 1, j'ai préféré écarter cette solution. VueJs a ensuite été préféré à React car considéré comme un framework plus léger et facile à appréhender.

4) Le système pour gérer le versionnage

Pour régler la gestion du versionnage d'un contenu, plusieurs solutions ont été envisagées, dont deux principales :

- Utiliser une solution logicielle déjà existante

  Deux noms nous sont alors venus en tête : git et svn. Après renseignements, svn a été écartée car de fonctionnement très similaire à git (ce dernier étant un produit déjà connu donc plus rapide à mettre en place). Nous avons donc choisi de partir sur git en termes de solution logicielle (python offrant un plugin adapté), tout en gardant en tête la potentielle « lourdeur » de cette solution.

- Une solution artisanale

  L'avantage de cette solution est un fonctionnement plus facile que git (pas de plugin). Cependant l'inconvénient est que créer un système de gestion de version pourrait donner l'impression de vouloir réinventer la roue, une très mauvaise pratique en informatique

Après avoir essayé de manière empirique les performances des deux solutions il a été décidé de partir avec git pour gérer la problématique de la gestion de version dans l'application

5) L’architecture du projet

J’ai eu pour consigne de me renseigner sur l’intérêt et la faisabilité de la mise en place d’une structure de type microservices. Cependant, après investigation et avis de M. Exbrayat et M. Moal, il en a résulté que cette solution ne correspondait pas aux besoins de la mission et cette problématique a donc été abandonnée.

LE DEVELOPPEMENT – « PREAMBULE »

LE SQUELETTE (BASE DE TRAVAIL POUR LE WEB SERVICE)

Il a été décidé, pour observer le comportement possible de l'application, de réaliser un petit programme en python (appelé squelette) simulant les fonctions basiques de l'application sans IHM dédiée.

Les fonctionnalités attendues de ce squelette étaient :

- Permettre la création & modification de contenu (simule le panier)
- Permettre la consultation des contenus d'autres auteurs (simule la banque)
- Permettre de mettre à jour (et de conserver l'historique) d'un contenu
- Interagir avec la base de données pour conserver les métadonnées
- Faire un système de demande de publication (acceptation de maj proposée par d'autres auteurs)
- Faire un système de maj.
- Faire un système basique d'association de contenu à une évaluation.

Pour cette application sera considérée comme étant un contenu un simple fichier texte avec la phrase suivante : « Bonjour, je suis un contenu dans sa version 1 » (le dernier numéro de la phrase évoluant si le contenu est mis à jour)

L'affichage est prévu via le terminal.

Voici à quoi ressemble le produit développé :

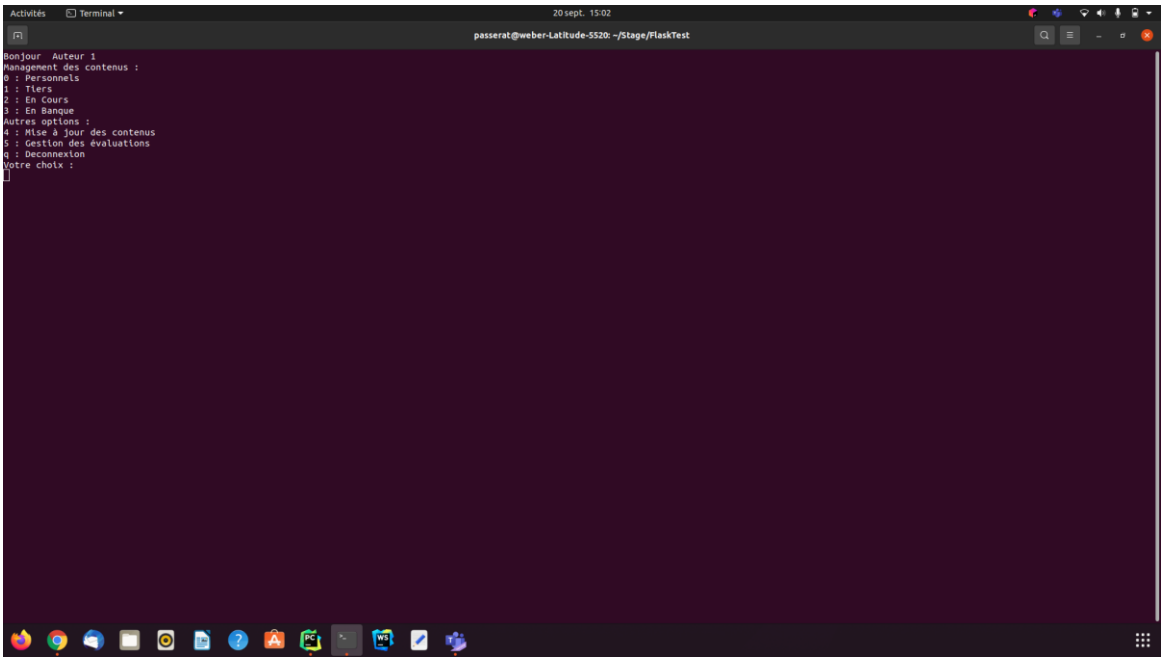


Image 1 - Application Primitive

La réalisation de ce squelette a eu deux avantages pour la suite :

- Des parties de code seront réutilisables dans le web service (en particulier les parties sur les interactions avec mongoDB et avec git)

- Notre approche pour réaliser l'application a été éclaircie sur certains points. Pour illustrer avec un exemple :

L'utilisateur 1 veut accéder au contenu n°1 dans sa version 2.

L'utilisateur 2 veut ensuite accéder au contenu n°1 dans sa version 1.

L'utilisateur 2 va perturber la lecture de l'utilisateur 1 car il modifiera la version présente dans le répertoire ou le contenu est stocké (le contenu n'existera plus dans la version 2 mais dans sa version 1).

Pour solutionner cela, il a été décidé qu'aucun des deux ne travaillerait directement sur le fichier mais que pour chaque demande d'accès à un fichier, le web service renvoie le texte constitutif du contenu, qui est ensuite utilisé par l'application web. De ce fait aucune histoire d'accès concurrentiel au fichier n'existe, aucun besoin de duplication du fichier non plus.

Ce produit a validé un axe de développement souhaité par Rodolphe Weber quand nous avons défini les grandes lignes de la mission : ne pas dupliquer les fichiers. Il est en effet possible de travailler en attribuant à chaque contenu (concept extensible aux évaluations) un identifiant unique qui permet d'y faire référence à divers endroits (par exemple pour gérer les 'tiers' d'un utilisateur).

LE VOLET LATERAL AVEC JQUERY (BASE DE TRAVAIL POUR LE CÔTE VUEJS)

Avant de réaliser la moindre ligne de code concernant les IHM, une étape de maquettage a été réalisée, grâce à l'outil figma.

Ces maquettes nous ont permis de dessiner une première vision des choses sur l'aspect que devait avoir l'application. Comme pour la phase de création de l'analyse fonctionnelle, une concertation avec, M. CAU, de la fondation Polytech, a pu valider notre vision des choses et enrichir notre réflexion créatrice.

Il a ensuite été décidé de réaliser un « échantillon d'IHM », pour voir ce que cela donnerait. La partie sélectionnée, le volet latéral, a été choisie car étant la partie la plus technique du projet.

Je suis parti, dans un premier temps, sur JQuery, une technologie que je connais déjà, donc rapide à utiliser (l'idée de partir sur un framework plus « gros » pour la suite étant déjà actée à ce moment)

Suite à la finalisation de ce volet, son portage et la continuation de l'application web directement sur VueJs ont été décidés, abandonnant de ce fait Jquery comme outil principal (pour des raisons pratiques ce plugin a été gardé (cf. partie concernant l'application web)).

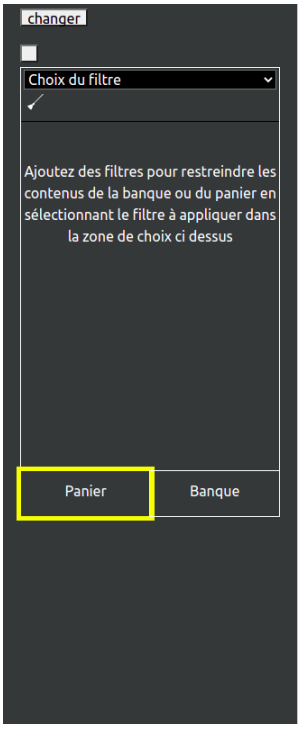


Image 2 - Le volet latéral (avec le mode sombre activé – version VueJS)

LE WEB SERVICE

POURQUOI UN WEB SERVICE

Le web service développé pour l'application OpenQuizz aura deux objectifs :

- Dans un premier temps, faire le lien avec la base de données (Mongo). L'application VueJs ne permet aucune modification possible sur la base mais doit faire tous ses échanges grâce à des appels au « module » Flask.

- Permettre de manipuler les fonctionnalités de l'application avec d'autres clients que celui mentionné dans ce rapport. En effet dans un futur idéaliste il serait souhaité d'ouvrir certaines fonctionnalités à tous (comme l'import d'un contenu, qui serait intégrable n'importe où).

REPRISE DES ELEMENTS DU SQUELETTE (et compléments)

Le squelette précédemment mentionné a été réutilisé dans une certaine mesure pour certaines parties :

- La base de données mongo a été reprise (et légèrement modifié)
- Les fonctions manipulant git et une partie des fonctions CRUD ont également été reprises, en particulier :
    - Création d'un contenu
    - Mise à jour d'un contenu
    - Suppression d'un contenu
    - Création d'un utilisateur
    - Suppression d'un utilisateur

Le squelette a donc servi de base pour créer et étendre les fonctionnalités nécessaires à l'application OpenQuizz

Nous allons ensuite découvrir les différentes entrées du web service (présentées dans un tableau), puis le détail des fonctionnalités (en lien avec le code)

LES ENTREES DU WEB SERVICE

Le web service a été divisé en cinq sous domaines :

urlduWS/utilisateurs*:*

Contient toutes les fonctions liées à l'utilisateur (création, suppression, changement de mot de passe)

urlduWS/contenus :

Partie majeure du web service, elle offre toutes les fonctionnalités permettant la manipulation d'un contenu (création, évolution, suppression, mise en tiers), mais aussi d'autres fonctionnalités plus spécifiques comme le parsage des xml moodle, la récupération du texte d'un contenu, de sa note ou l'association / dissociation d'un tag à un contenu.

urlduWS/en_cours :

Permet de créer, supprimer et consulter les contenus en cours pour un utilisateur. Des fonctions permettent également de récupérer le texte ou l'auteur d'un contenu en cours

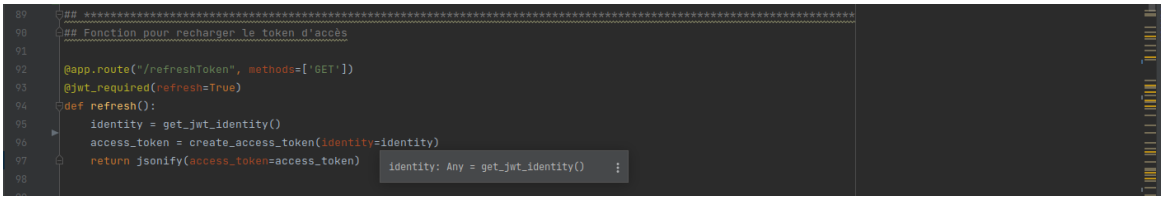
urlduWS/demande_pub :

Permet de créer, supprimer et consulter les demandes de publication pour un utilisateur. L'acceptation / refus d'une demande de publication est géré dans la partie contenus.

urlduWS/evaluations :

Permet de créer, supprimer et consulter les évaluations pour un utilisateur

A noter qu'une entrée spécifique, non associée à un sous domaine, permet de rafraichir le token permettant l'accès aux fonctionnalités du web service (grâce à un second token, spécialement utilisé dans ce cas, obtenu à la connexion) :



Code 1 - Actualisation du token d'accès

Bien qu'il existe des outils permettant de réaliser le mappage direct avec MongoDB, appelés ODM (Object Document Mapper) ; il a cependant été décidé de ne pas travailler avec cet objet car les interactions ont été reprises du squelette, qui utilisait le plugin PyMongo et pas d'ODM.

Comme Mongo est utilisé directement, sans outil spécifique de mappage, il n'a pas été nécessaire de créer des classes pour récupérer les informations de la base de données. En effet, les interactions via PyMongo offrent en retour des dictionnaires, faciles à manipuler et à transmettre via l'api

Par exemple, getAllContenus permet de récupérer directement un tableau de dictionnaires, chacun des dictionnaires étant un contenu.

Le modèle a été séparé en quatres parties (pas de facade mais envisagé dans le futur).

La sous partie principale (la plus importante regroupe toutes les fonctionnalités permettant la gestion des contenus personnels et tiers (création, modification, suppression, récupération d'un sous élément comme la note ou les tags) et des utilisateurs.

Trois autres documents servent à gérer les contenus en cours, les demandes de publication ou les évaluations.

Deux fichiers spéciaux, l'un intitulé Utils.py et regroupe des fonctionnalités majoritairement liées au texte des contenus (la lecture des fichiers présents sur le disque) et l'autre, intitulé GitUtils.py, permet les interactions liées au versionning (pour des facilités d'utilisation chaque contenu est inséré dans un répertoire qui lui est exclusivement dédié et on git est initialisé systématiquement).

Un fichier spécifique pour la configuration d'apache, appelé app.wsgi, apparait aussi dans le projet

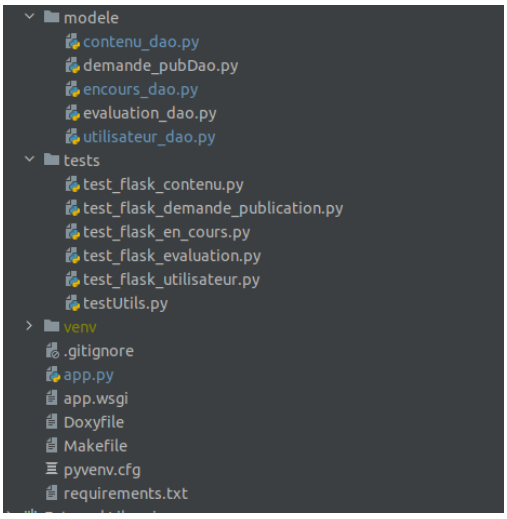


Image 3 - Fichiers de code du web service (PyCharm)

LA SECURITE

La sécurité dans un web service sert à limiter les accès aux fonctionnalités pour restreindre des adresses critiques (comme les fonctions de suppression), entres autres.

Deux solutions ont été envisagées pour résoudre cette problématique :

- La solution Oauth2, la plus performante des deux. Cette solution a temporairement été écartée (pour la durée du stage) car plus longue à mettre en place. Elle pourra cependant être mise en place plus tard dans la vie du projet.

- Les tokens JWT (JSON Web Token). Une chaine de caractères « humainement incompréhensible » est générée quand un utiilsateur se connecte et ce dernier doit passer cette chaine lors de chaque appel au web service.

Cette chaine est infalsifiable et offre une sécurité d'accès aux fonctionnalités du web service. La seule fonctionnalité pour laquelle la vérification du web service n'est pas requise est le login.

Deux tokens différents sont renvoyés lors de la connexion d'un utilisateur : un token d'authentification, utile pour réaliser toutes les opérations courantes ; et un token de rafraichissement, permettant de récupérer un nouveau token d'authentification quand nécessaire.

Https

Pour des raisons de planification le HTTPS n'est pas implémenté dans le web service mais sera mis dans un futur proche.

LA DOCUMENTATION BIPARTIE

Deux outils ont été utilisés pour générer la documentation :

Doxygen

Pour documenter tout le modèle, l'outil Doxygen a été utilisé (sur demande directe du client).

La documentation doxygen (en python) se fait sur ce modèle :

```python
## ********************************************************************************************************************
## Entrées liées à la gestion utilisateur
@users.route('/newUser')
@users.doc(params={'login': "le nom de l'utilisateur à créer"})
@users.doc(description="Création d'un nouvel utilisateur")
class CreerUtilisateur(Resource):
    @users.response(201, 'Utilisateur créé')
    @users.response(409, 'Utilisateur existant')
    ## Création d'un nouvel utilisateur
    def post(self):
        if utilisateur_dao.presence_utilisateur(request.json["login"]) == 0:
            utilisateur_dao.ajouter_utilisateur(request.json["login"])
            return "Utilisateur créé", 201
        else:
            return "Utilisateur existant", 409
```

Code 2 - Exemple de documentation Doxygen

Doxygen a pour but d'éclairer toute personne qui lit la documentation sur toutes les fonctions du modèle, a quoi elles servent et avec quoi elles fonctionnent.

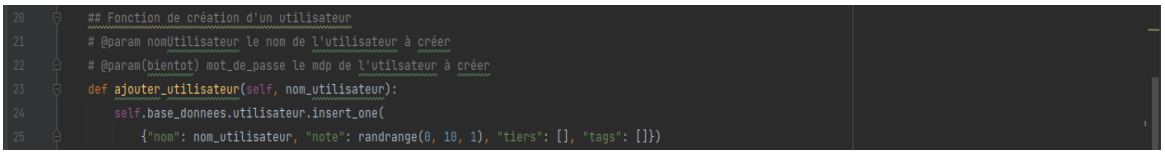
Pour générer la documentation l'outil doxywizard sera installée sur le serveur ou OpenQuizz est déployée (et conseillée si l'application doit être migrée ailleurs).

L'avantage de doxywizard est qu'elle permet d'enrichir la lecture de la documentation en ajoutant, par exemple, un volet latéral de navigation dans la page html générée.

Swagger / OpenApi

Ce plugin est utilisé pour documenter les entrées http du web service.

La documentation est de la forme :



Code 3 - Exemple de documentation OpenApi

LES TESTS

Les tests pour le web service ont été réalisés avec le plugin pytest.

Les tests créés visent à simuler des scénarios basiques d'utilisation de la plateforme pour s'assurer que tout se déroule bien.

L'APPLICATION WEB

Pour que l'utilisateur puisse accéder à toutes les options de manipulation des contenus, il a donc été décidé de développer une solution grâce à l'outil VueJs, particulièrement adapté pour les solutions de type *Single Page Application*.

Les prochaines pages permettront de découvrir de manière succincte les différents éléments de cette application web.

L'AUTHENTIFICATION

La première page sue laquelle un utilisateur tombe est un classique formulaire de connexion, avec la place pour un login et un mot de passe.

Dans l'état actuel des choses la vérification du mot de passe n'est pas effective (pour des raisons effectives, mais dans un futur proche il sera mis, à la place du formulaire artisanal, des solutions d'authentification plus performantes (les plugins permettant l'identification de toute personne affiliée à une université, comme on peut retrouver sur les divers espaces ENT, pourraient être envisagés).

Une fois qu'une connexion a été réussie, l'utilisateur arrive dans l'espace comprenant le volet latéral, le tableau des contenus et le menu des autres options, qui seront détaillés ci-dessous, ainsi que les actions possibles :

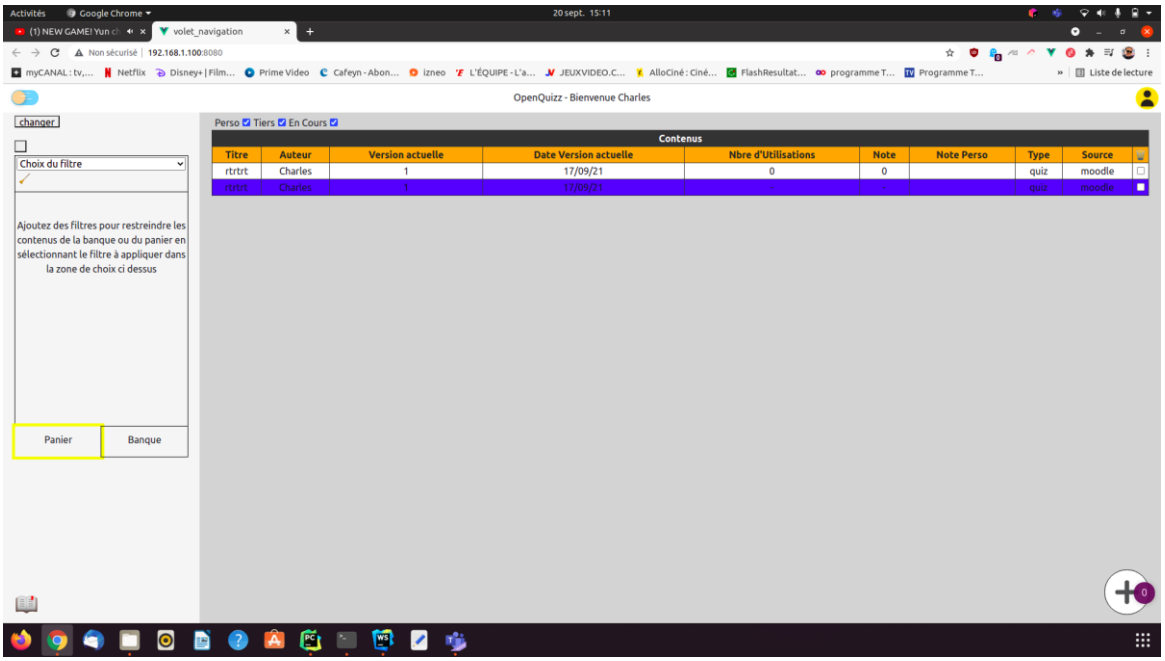


Image 4 - Vue principale d'OpenQuizz

LE VOLET LATERAL

Imaginé dans un premier temps grâce à l'outil Jquery, le volet latéral a été ensuite porté et enrichi pour permettre de remplir ses deux objectifs majeurs, que voici :

Les filtres

Dans la zone supérieure du volet se situe un menu déroulant permettant de choisir un filtre (au nombre de 7). Un filtre a pour objectif de restreindre les contenus présents dans le tableau des contenus situé à droite du volet latéral.

Les différents filtres disponibles pour l'utilisateur sont :

- Les tags officiels. (Voir ci-dessous la partie concernant la création / l'import de contenus).
- Les tags libres. Sur un modèle similaire aux tags officiels, obligatoirement définis lors de la création / l'import d'un contenu, il est possible pour un utilisateur de définir sa propre architecture de tags. Utilisable seulement en mode panier.
- Le filtre auteur. Permet de filtrer un ou plusieurs auteurs de contenu. En mode panier, il est possible qu'un utilisateur choisisse son propre nom dans la liste (ce qui permet de garder tous les contenus dont il est à l'origine).
- Le filtre type / source. Le type correspond à la nature du contenu (même si l'application n'est, pour le moment, récipiendaire que de quizz elle pourra bientôt accueillir d'autres types de contenu, comme des td/cours). La source correspond à l'endroit d'origine des quizz (Moodle, créé directement sur OpenQuizz, autres).
- Le filtre date. Sur le modèle de ce que propose le plus célèbre moteur de recherche du marché, il est possible de restreindre les contenus par date de création (dernière semaine / mois en cours / année en cours).
- Le filtre note : Permet, soit via un seuil de note maximal, un seuil de note minimal ou une fourchette de notes, de restreindre les contenus.
- Le filtre recherche. Dispose de trois options : la recherche sur le titre, la recherche sur le texte du contenu ou la recherche sur le titre et le texte du contenu. Pour le moment la recherche est monoterme mais ce point est amené à évoluer.

Juste en dessous de ce menu latéral se situe la zone d'affichage des filtres choisis. Chaque filtre choisi peut ainsi être supprimé via une petite croix à côté de son nom.

Encore en dessous de cette zone vient la zone de détail de chaque filtre (principalement des zones de texte, des menus déroulants ou des arborescences).

A noter que le bouton juste au-dessus du volet permet de le masquer, ce qui a pour effet de donner plus de place au tableau des contenus, que nous allons ensuite découvrir.

La variante du menu de bascule

Tout en bas de ce menu latéral nous avons un menu spécial permettant de faire basculer l'application du mode panier au mode banque). Une bascule via ce menu affecte le tableau des contenus mais aussi certains filtres (comme vu ci-dessus).

Il est possible, via le bouton juste au-dessus celui qui permet de masquer le volet, de faire basculer ce menu au niveau du titre si l'utilisateur le désire

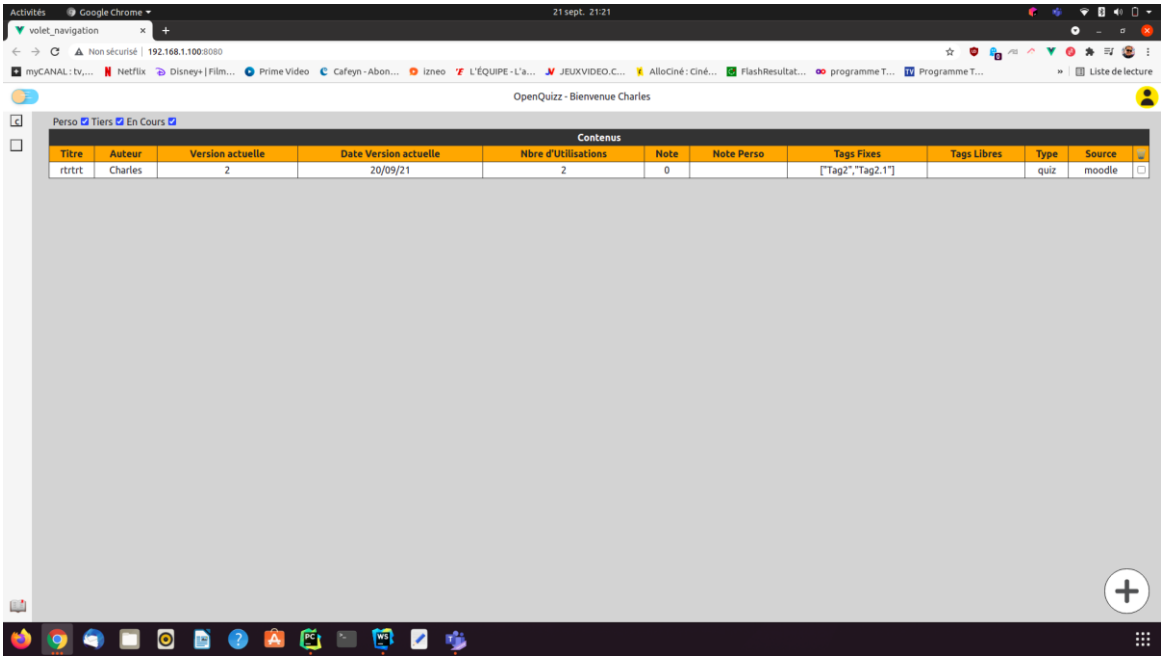


Image 5 - La vue principale avec le volet rétracté

LE TABLEAU DES CONTENUS

Le tableau visible sur l'espace de droite quand l'utilisateur est dans la partie principale de l'application regroupe tous les contenus qu'il possède (s'il est en mode panier) ou que d'autres ont mis à disposition de tout le monde et qu'il peut s'approprier (s'il est en mode banque).

Regardons dans le détail ce que nous pouvons trouver dans ce tableau

En mode panier

Tous les éléments humainement lisibles d'un contenu sont affichés dans le tableau. Dans l'ordre, nous avons : son titre, son auteur, sa version courante, la date de sa version courante, le nombre d'utilisations (multi-versions), sa note (liée à la version), la note qui lui a été attribuée (si on est avec un contenu de type « tiers »), son type et sa source.

Dans la dernière colonne nous retrouvons en tête de colonne une corbeille. Elle signifie que si on coche une ou plusieurs lignes et qu'on clique sur cette corbeille ces contenus seront supprimés.

Dans le cas où le volet latéral est masqué, deux colonnes supplémentaires apparaissent permettant de découvrir les tags fixes et les tags libres associés au contenu.

Un survol du titre à la souris permet d'avoir une prévisualisation de l'intérieur du contenu. Pour rentrer totalement dans le contenu, il faut cliquer sur ce même titre (cf. la partie LA CONSULTATION D'UN CONTENU plus bas)

Il existe également une interaction possible avec un des filtres, qui est le filtre « tags libres ». En effet si plusieurs lignes sont sélectionnées il est possible d'effectuer un glisser déposer vers un des tags de l'arborescence ce qui aura pour effet immédiat l'association entre le tag visé et tous les contenus sélectionnés.

Trois coches sont présentes au-dessus du tableau. Elles permettent de restreindre l'affichage en fonction des trois sources différentes de contenu dans le panier :

- Les contenus « persos », directement créés par l'utilisateur
- Les contenus « tiers », récupérés via la banque
- Les contenus en cours, que l'utilisateur est en train de modifier, et qui peuvent être soit d'origine « perso » soit d'origine « tiers »

En mode banque

En mode banque l'affichage est très similaire au mode panier. Les colonnes du tableau sont quasiment identiques et seules changent trois éléments :

- Les trois coches présentes au-dessus du tableau qui permettaient de filtrer les sous éléments du panier ont disparues
- Les tags libres, n'ayant aucune pertinence, disparaissent
- A la place de la corbeille dans la dernière colonne est dorénavant présente une étoile, qui permet de mettre en tiers tous les contenus cochés.

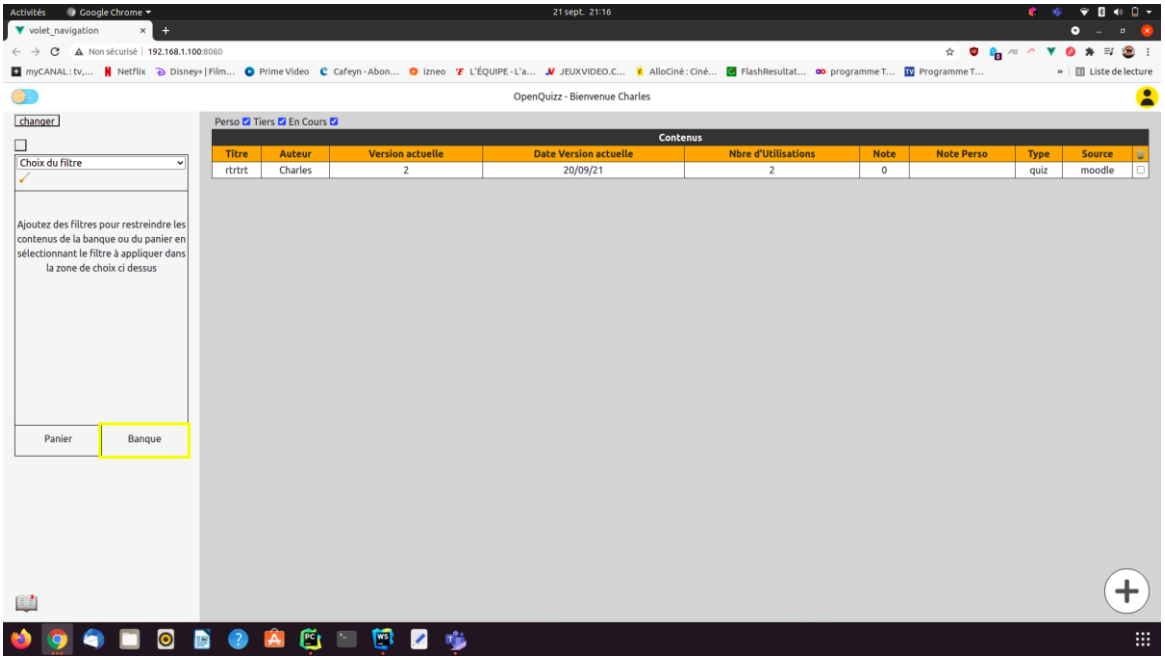


Image 6 - Vue principale d'OpenQuizz (en mode banque)

LA CREATION D'UN CONTENU

Pour le moment, deux méthodes existent pour qu'un utilisateur créée un nouveau contenu. Pour accéder au sous menu de création d'un contenu, il faut, en mode panier, cliquer sur le gros bouton ⊕, puis sur l'option désirée dans le sou menu qui s'ouvre immédiatement après

Création directe (markdown)

Le premier mode de création ouvre un plugin *(mavon-editor)* permettant de saisir directement un contenu et supportant la norme markdown. Dans la partie droite de l'écran il est possible de voir directement le contenu mis en forme.

Une zone située juste au-dessus de ce plugin de création permet de renseigner le titre du contenu.

Une fois validé, l'utilisateur est redirigé vers une page l'invitant à renseigner les tags fixes. Ces derniers correspondent à la classification pédagogique du contenu. Par exemple une question impliquant du langage C pourra être classée dans Sciences/Informatique).

Une fois les tags validés l'utilisateur revient sur le panier et peut découvrir une ligne supplémentaire dans ce dernier.

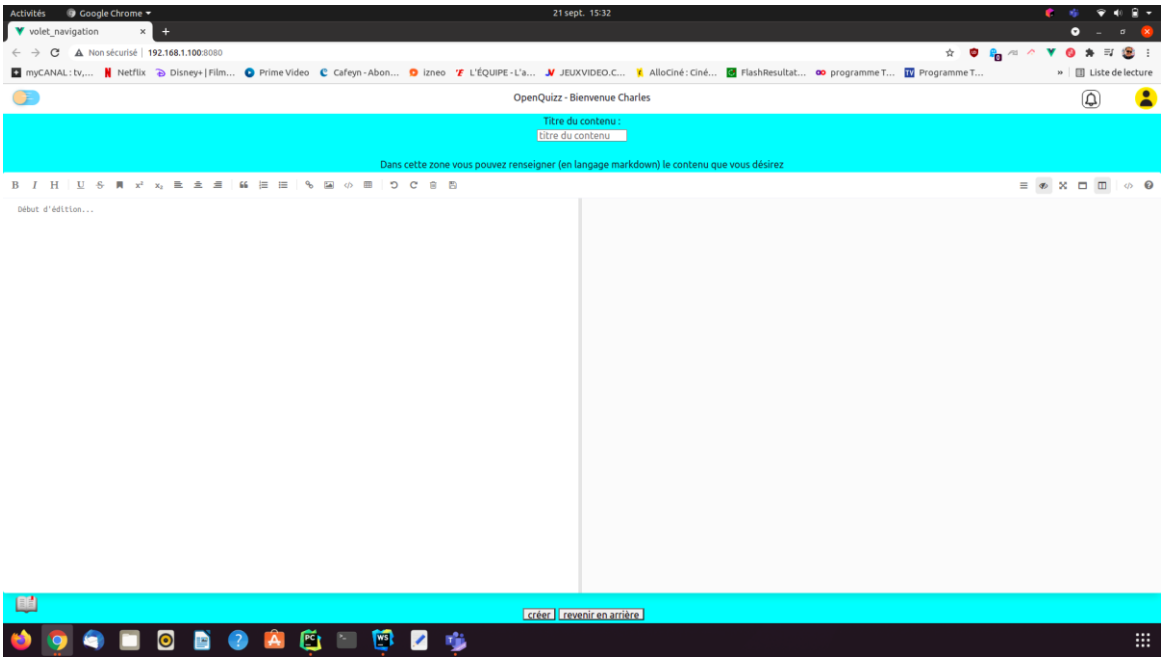


Image 7 - Création d'un contenu

Import de fichier

Le second mode de création invite l'utilisateur, via un formulaire à uploader un fichier contenant le contenu pédagogique. Pour le moment seuls les xml issus de Moodle sont pleinement pris en compte (au niveau de la prévisualisation et de l'édition) mais théoriquement tout type de fichier peut être uploadé.

Après validation, l'utilisateur est également redirigé vers la page de sélection des tags fixes puis, après seconde validation, vers son panier avec une ligne nouvellement créée.

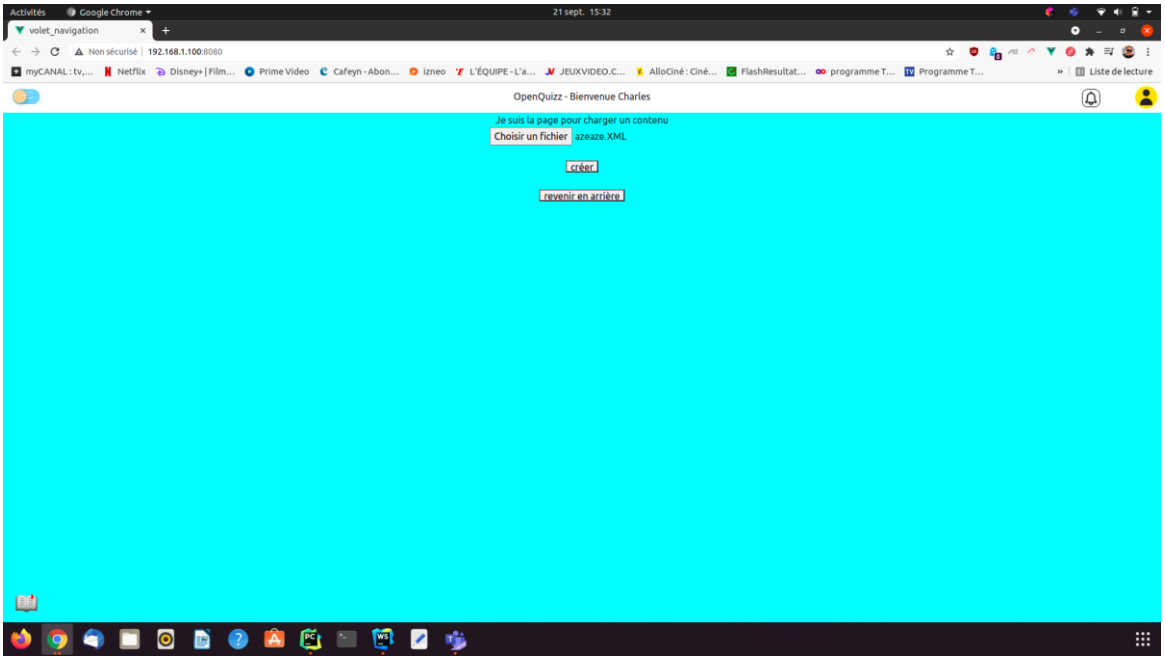


Image 8 - Import d'un Contenu

LA CONSULTATION D'UN CONTENU

Pour consulter un contenu il suffit de cliquer sur son nom, que ce soit en mode panier, en mode banque ou dans le menu des mises à jour.

L'affichage de la consultation d'un contenu se fait de manière identique en mode banque et en mode mise à jour :

Certaines informations caractéristiques du contenu sont présentes dans un tableau pour rappel, et le texte associé au contenu est également affiché. En mode banque une option apparait, permettant de sélectionner la version désirée du contenu (s'il a déjà été mis à jour) et sa mise en contenu tiers. En mode mise à jour ces options n'existent pas.

Quand on désire consulter un contenu « en mode panier » l'affichage est différent. Dans un bandeau on retrouve également les informations caractéristiques du contenu mais d'autres options apparaissent :

Il est dorénavant possible de consulter et de modifier les tags du contenu selon sa catégorie (un contenu perso tous les tags sont modifiables, seulement les tags libres pour un contenu tiers).

Pour un contenu tiers apparait, à droite de la zone pour modifier le titre, une option permettant de noter le contenu ou de consulter la note mise.

Une troisième zone d'options est présente à droite du bandeau regroupant des options pour supprimer le contenu (icone de poubelle) ou pour valider une modification/ proposer une publication (dans le cadre d'un contenu en-cours).

La majorité de l'écran est occupée pour afficher le texte associé au contenu, qui est ici modifiable. La partie concernée par la modification varie en fonction du type de contenu manipulé

LA MODIFICATION D'UN FICHIER

Comme vu ci-dessus, la consultation d'un contenu en cours permet la modification directe du texte du contenu.

Une fois modifié, un contenu en cours est créé. Cette modification correspond à la première étape du cycle de mise à jour d'un contenu. Deux cas sont distinguables :

- Si un auteur modifie un contenu perso, il peut directement valider sa modification. Le cas particulier de la modification d'un contenu tiers anonyme permet également la validation directe d'un contenu en cours.
- Si un auteur modifie un contenu tiers, une demande de publication est ouverte (voir la partie *Le menu des mises à jour* plus bas pour plus de détails)

LA SUPPRESSION D'UN CONTENU

Un contenu est supprimable via deux méthodes : de manière collective, en cochant les lignes du tableau ou de manière individuelle, en consultant le contenu.

Trois cas sont à distinguer concernant la suppression d'un contenu :

- Si c'est un contenu perso : si au moins un utilisateur possède ce contenu dans son propre panier, le contenu est renié -> son auteur passe à « Aucun » et tous ceux qui possèdent le contenu le gardent (et peuvent le prendre à leur compte s'ils font une démarche de modification du contenu).

Si personne ne possédait le contenu, celui-ci est définitivement supprimé.

- Si c'est un contenu tiers : le contenu est supprimé du panier mais peut être retrouvé dans la banque. Si l'auteur du contenu tiers est « Aucun » et que le contenu est l'unique exemplaire restant, il est définitivement supprimé.

- Si c'est un contenu en-cours, le contenu est définitivement supprimé. Un contenu en cours ayant fait l'objet d'une demande de publication ouverte n'est pas supprimable (il n'apparait pas dans les contenus visibles par l'utilisateur).

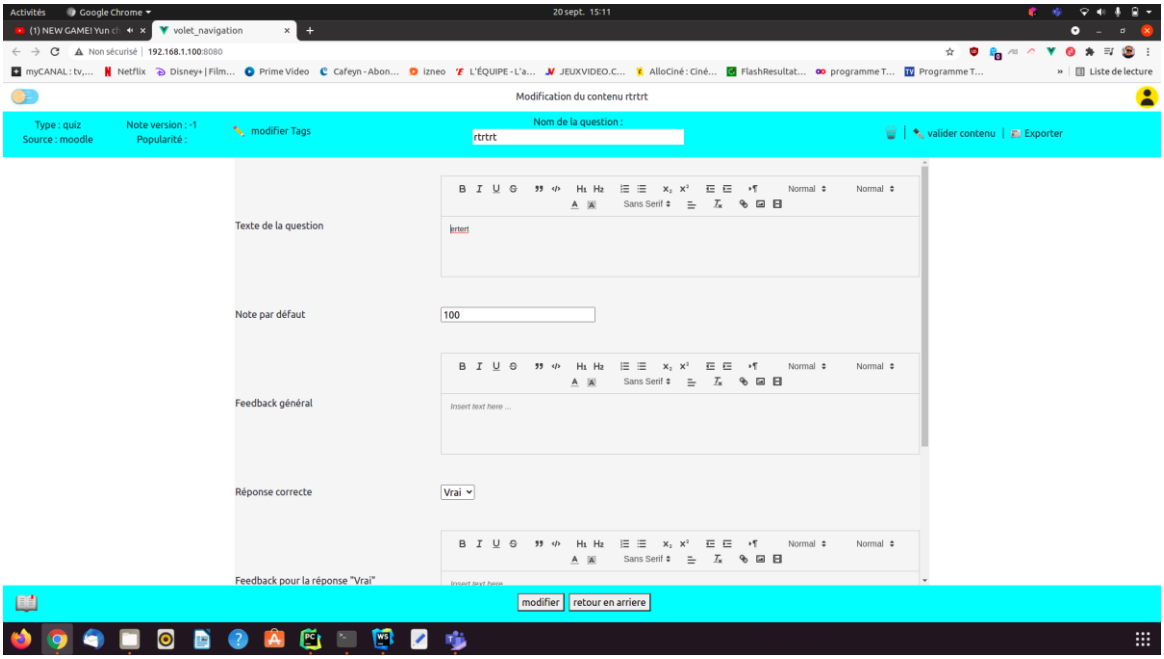


Image 9 - Vue de consultation / modification d'un contenu pédagogique

LE MENU DES AUTRES FONCTIONNALITES

Le menu des autres fonctionnalités est accessible par un clic sur l'icône jaune en haut à droite de l'application. Ses différents sous menus sont :

La consultation du profil

Permet de consulter son profil (identifiant, école) et offre un menu pour modifier son mot de passe (cette option pourrait disparaitre selon l'évolution du système d'authentification)

Le menu des mises à jour

Catégorie la plus importante de ce menu, elle propose deux tableaux permettant la gestion des mises à jour recules ou proposées.

Quand un utilisateur B propose une mise à jour à un utilisateur A, c'est ici que A vient dans un premier temps refuser ou accepter une mise à jour.

Si A accepte la mise la jour, le contenu concerné est mis à jour chez A et chez B (qui possède le contenu concerné dans son panier).

Si A refuse la mise à jour, B devra se rendre également dans ce menu et possèdera trois choix : supprimer sa demande de mise à jour et le contenu associé, conserver le contenu ayant fait l'objet d'une demande de publication ou créer un nouveau contenu (dans ce cas il apparaitra dans la banque des autres utilisateurs).

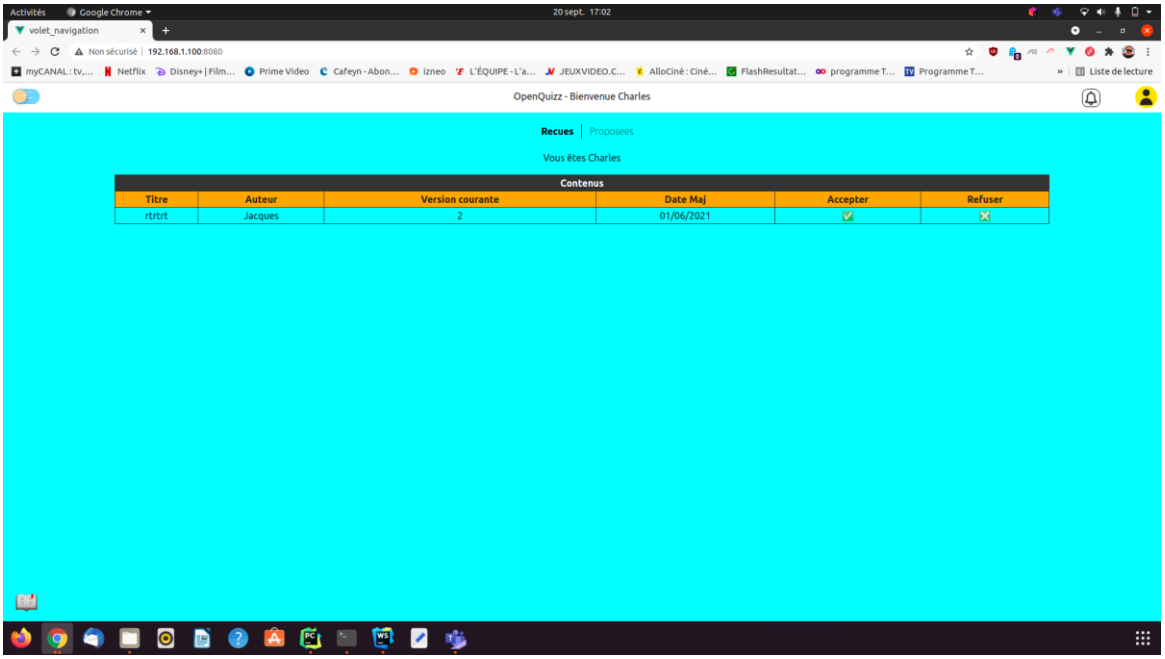


Image 10 - Menu des mises à jour en mode "reçues"

Le menu de gestion des tags

Deuxième sous menu majeur, il permet de créer et de supprimer des tags libres pour l'utilisateur connecté

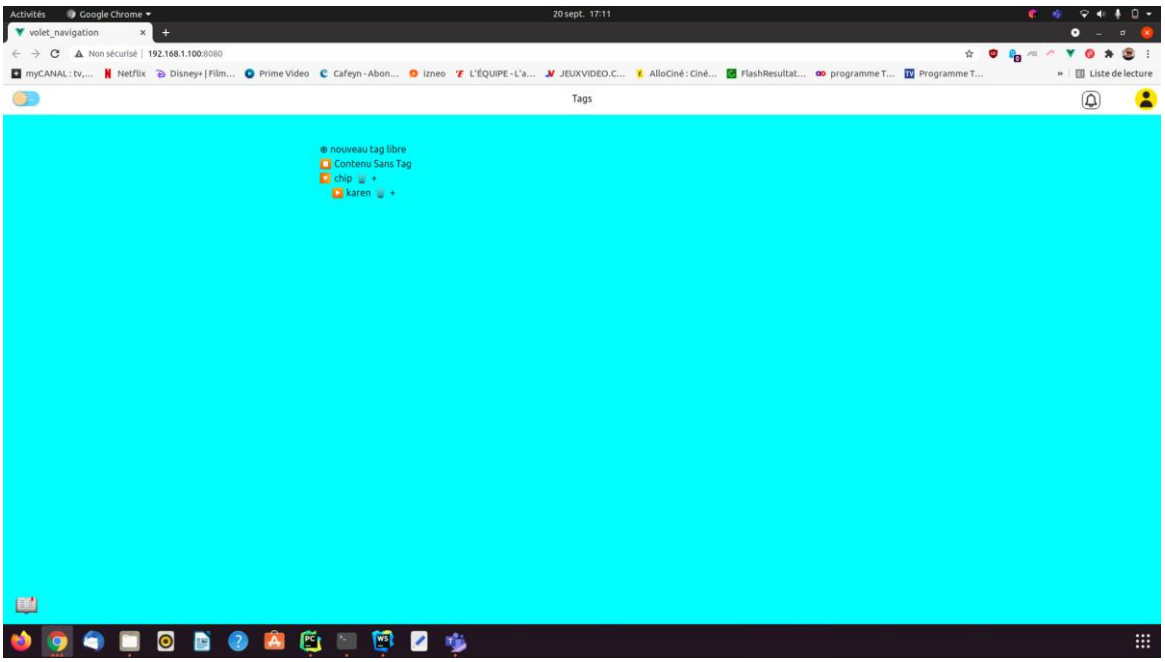


Image 11 - Menu de gestion des tags libres

Le menu de gestion des évaluations

Actuellement vide, permettra de gérer les évaluations. Une évaluation, dans le cadre d'OpenQuizz, est un marqueur sur un contenu indiquant qu'il a été proposé en évaluation à une certaine date pour une certaine classe.

MANUEL UTILISATEUR

Un manuel utilisateur, répertoriant les cas d'utilisation prévus de l'application, est disponible à tout moment (et en particulier sans avoir besoin de se connecter) via une icône en bas à gauche de l'application.

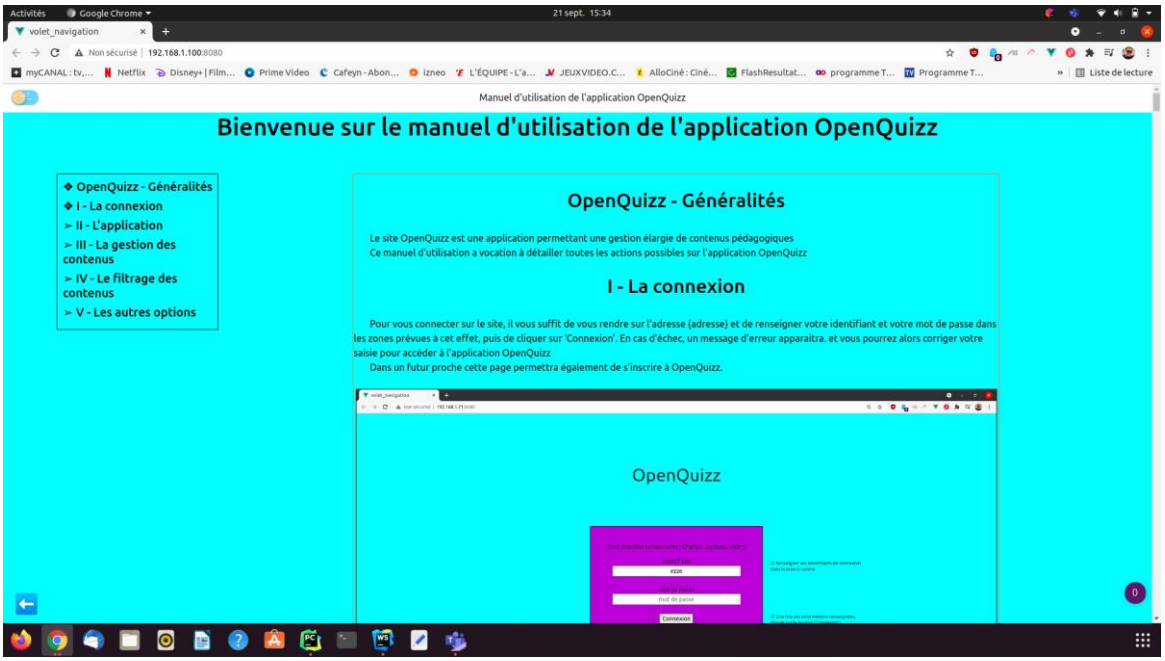
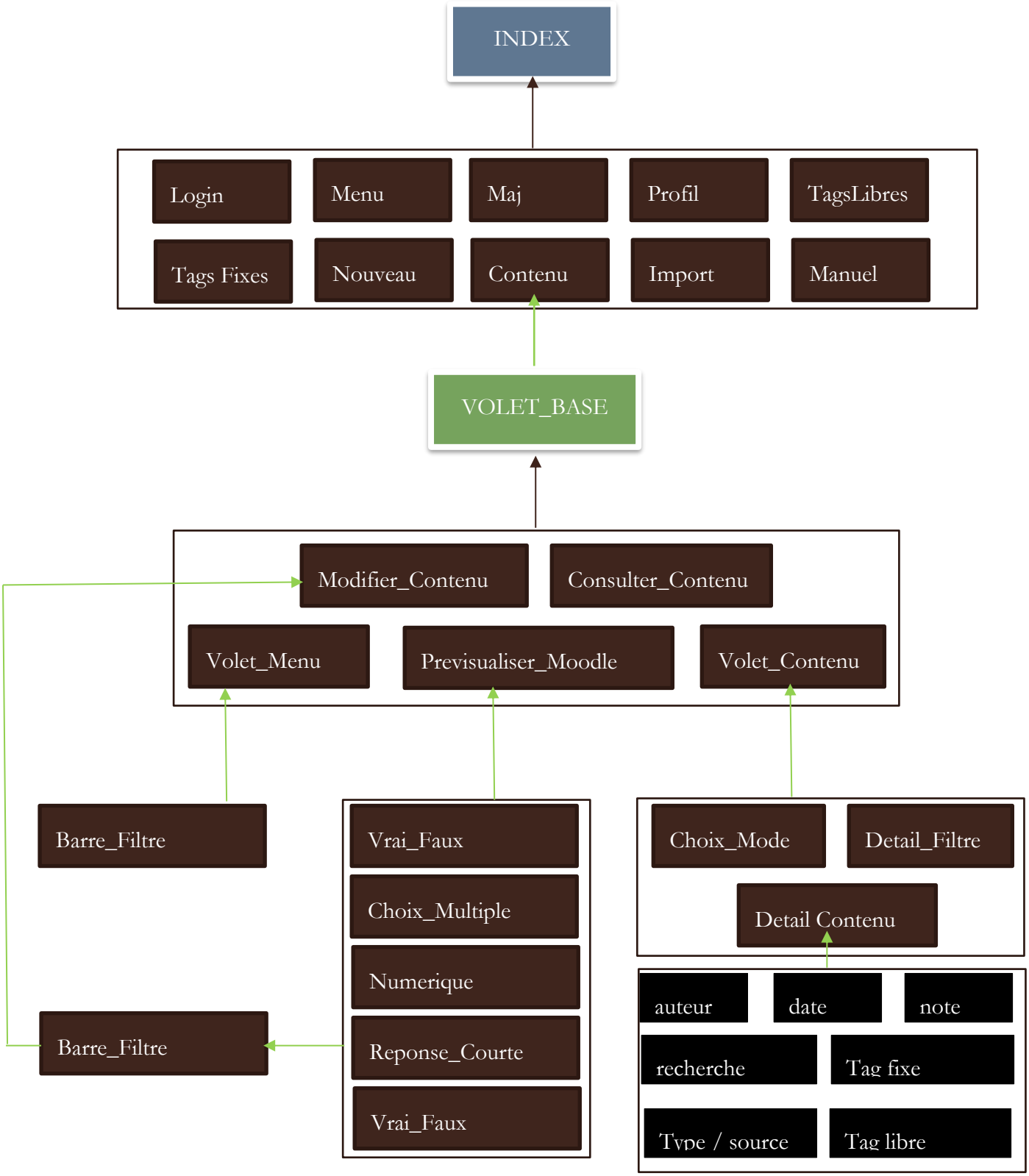


Image 12 - Manuel utilisateur d'OpenQuizz

ARCHITECTURE DU PROJET

Compte tenu de la structure « par composants » de VueJs, une des difficultés de cette partie a été de faire les bons découpages des différentes parties de l'IHM. Voici une vue schématique des composants créés pour OpenQuizz (page suivante) :



Pour les cinq types de fichier Moodle, un composant existe par type pour la prévisualisation et un pour l'édition (soit 10 en tout)

AUTRES FONCTIONNALITES

Les notifications de maj

Si l'utilisateur se connecte à son compte et qu'il y a du nouveau concernant la mise à jour d'un contenu (une nouvelle demande de publication, un changement de statut sur une demande en cours), une icône représentant une cloche apparaitra à côté de l'icône jaune de menu. Un clic sur cette icône de notification le renverra directement sur le menu des mises à jour.

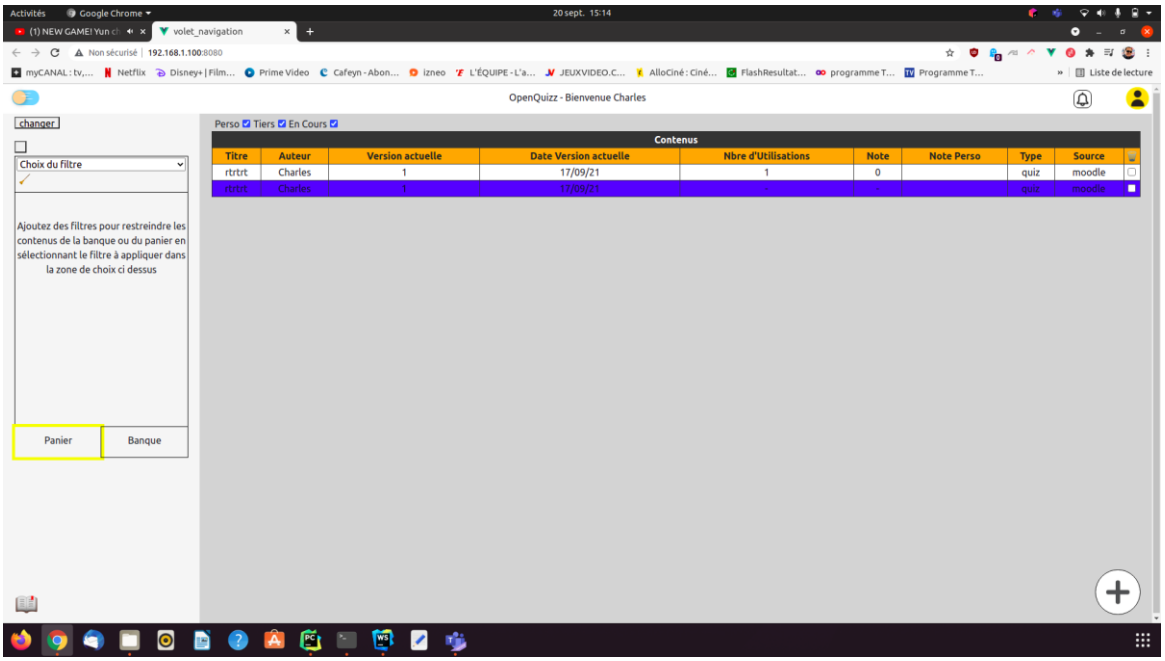


Image 13 - Vue principale avec icone de notification (en haut à droite)

Le mode sombre

Pour plus de confort visuel et selon la volonté de l'utilisateur il est possible d'activer le mode sombre partout sur le site. Ce mode d'affichage transformera presque tous les éléments dans des tons sombres.

LES TESTS

Comme pour la partie Flask des tests (sommaires dans l'état actuel des choses) ont été mis en place pour contrôler certains éléments des composants majeurs de l'application (certains $emit, certains boutons).

Le plugin utilisé pour ces tests est *vuese*.

POINT PARTICULIER : LA VISUALISATION / L'EDITION XML MOODLE

L'objectif de cette partie est de présenter une petite problématique technique rencontrée en fin de stage, que voici :

La visée première de l'application étant les contenus Moodle, ils doivent pouvoir disposer des fonctionnalités de prévisualisation et d'édition / de modification comme n'importe quel contenu manipulé sur le site.

Pour fournir ces deux fonctionnalités, deux solutions ont été envisagées :

- Récupérer le code de Moodle pour prévisualiser et éditer un contenu Moodle. Cette solution est celle privilégiée par le client pour la raison est qu'il serait sympathique de pouvoir retrouver la même charte graphique et les mêmes options que celles proposées sur Moodle.

- Créer à la main l'interface de prévisualisation et d'édition.

Mes recherches se sont donc portées sur des façons de résoudre cette problématique concernant la prévisualisation et l'affichage de contenus xml. Moodle met à disposition son code source sur GitHub (rédigé en PHP)

Le fichier contenant le code pour la prévisualisation des questions a été rapidement trouvé mais le très grand nombre d'inclusions d'autres fichiers et le fait que les questions ne « circulent pas en clair » dans l'application (les questions sont directement récupérées d'une base MySQL) ont freiné l'enthousiasme pour cette solution.

Pour confirmer ce sentiment, il a été décidé de demander conseil à des développeurs expérimentés ayant travaillés avec Moodle. Deux pistes ont été creusées :

- Contacter un référent Moodle de l'université d'Orléans. Cette piste s'est révélée stérile.
- Demander conseil sur le forum officiel des développeurs Moodle. Les personnes ayant répondues sur ce média ont confirmé les craintes concernant la difficulté d'extraire la prévisualisation de Moodle pour l'intégrer à OpenQuizz.

Suite à ça, il a donc été décidé de partir sur une solution « artisanale ». Pour aider dans ce chantier, j'ai pu disposer d'un précédent travail réalisé par un ancien étudiant de Rodolphe Weber, qui consistait, grâce à Python, à un logiciel permettant d'importer, de modifier et d'exporter un quizz Moodle, sous la forme XML.

Ce projet m'a permis de reprendre toute la partie permettant de parser le xml, que j'ai pu intégrer à Flask.

Concernant la partie visuelle, il a été décidé de s'inspirer de ce qui se fait sur Moodle, et plus particulièrement sur la déclinaison développée par l'université d'Orléans Tours, Celene.

Pour toutes les captures d'écran qui vont suivre, elles ont été réalisées par mes soins grâce à un accès enseignant donné par R. Weber sur un des cours proposés par Polytech et pour la durée du stage.

Celene propose beaucoup de formats de question lors de la création. Cinq d'entre elles, les plus fréquentes, ont été incluses dans OpenQuizz : les vrai:/faux, les questions à choix multiple, les questions à réponse courte, les questions à réponse numérique et les questions calculées simples.

La prévisualisation

La prévisualisation a été un peu modifiée.

En effet, sur Celene, la prévisualisation est interactive. Il est en effet possible de saisir une réponse sur le champ réservé à la question et de tester sa validité.

Sur OpenQuizz, la prévisualisation présente directement toutes les réponses possibles, tout en mettant en valeur (entre parenthèses) la bonne réponse (via le pourcentage de note alloué à la bonne réponse).

De plus, il a volontairement été décidé de se recentrer uniquement sur l'énoncé de la question et sur les réponses possibles et d'occulter les autres options que l'on aurait retrouvées sur Celene.

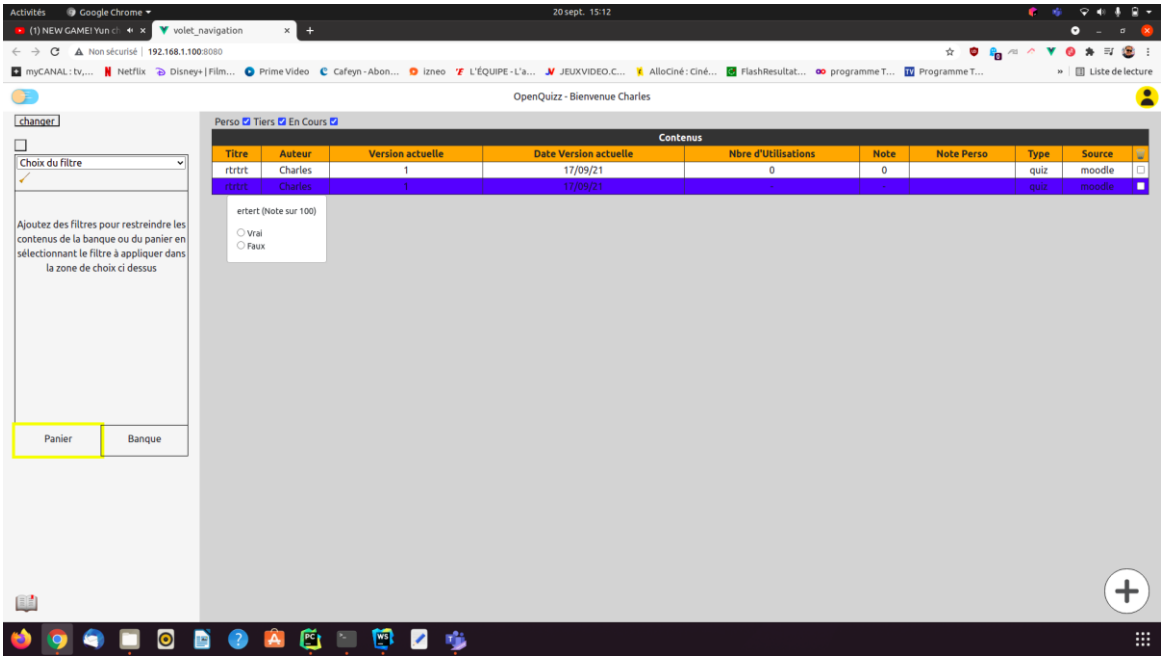


Image 14 - Prévisualisation d'un contenu Moodle de type vrai/faux

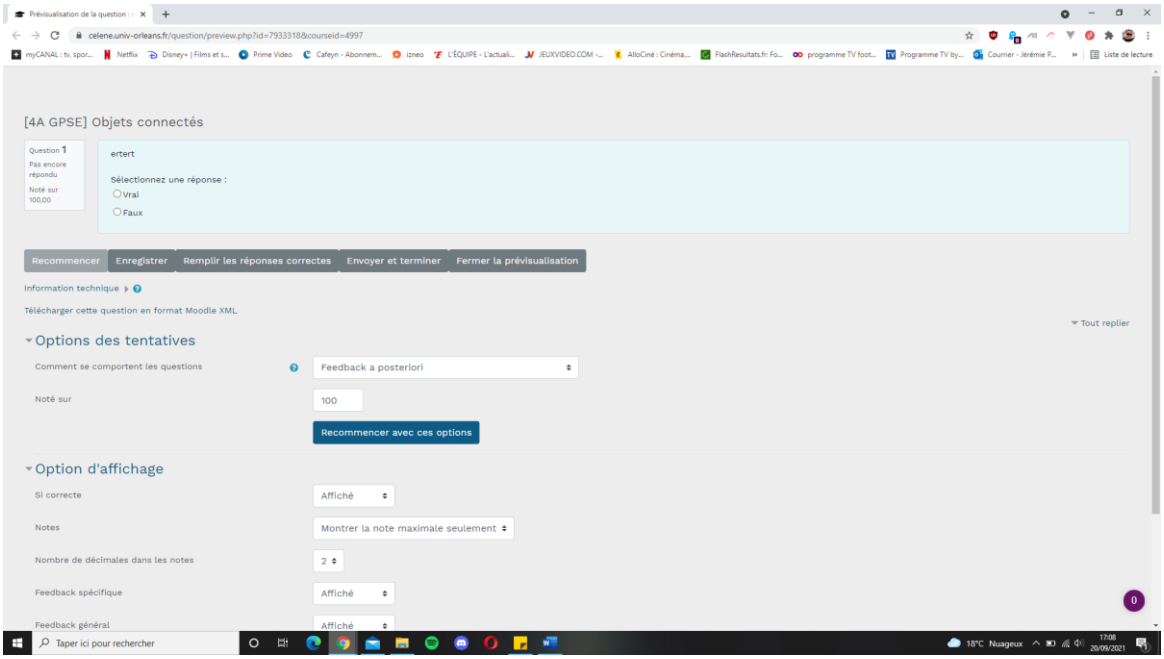


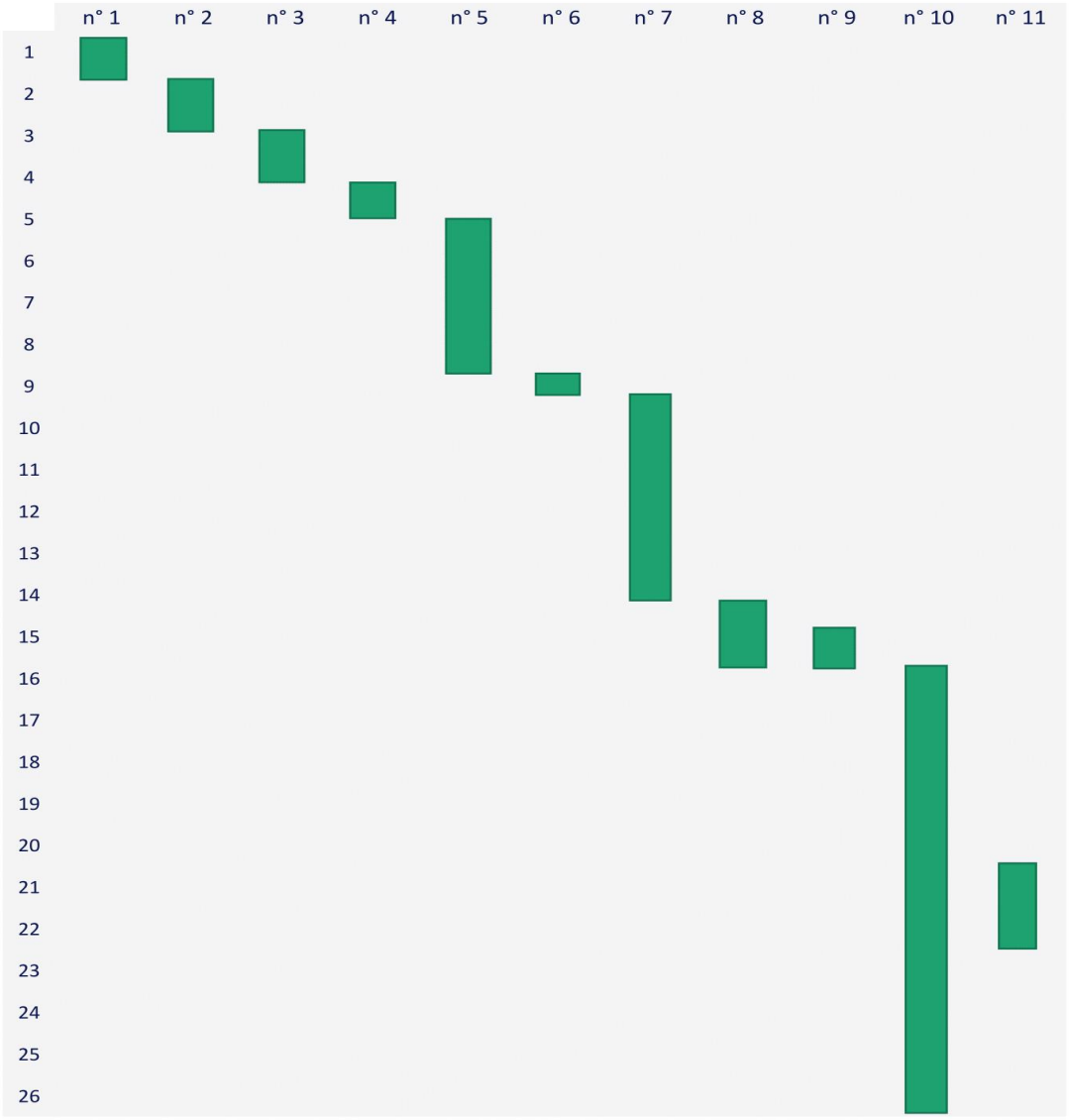
Image 15 - Prévisualisation (aperçu) du même contenu sous Celene (Moodle)

L'édition

Cette partie est visuellement plus conforme à ce que propose Celene. Il a cependant été décidé de se concentrer sur les options les plus utilisées par les enseignants lors de la création d'un quizz et d'occulter les autres.

GANTT – ORGANISATION DU STAGE

Un diagramme temporel sur la façon dont j'ai organisé mon stage



A gauche, les n° de semaine du stage. En haut, les n° des tâches effectuées (détaillées ci-dessous)

n°1	Analyse fonctionnelle	n°2	Analyse technique
n°3	Squelette	n°4	Maquettage
n°5	Creation VueJs	n°6	Creation Flask
n°7	Connexion Vue-Flask	n°8	Correction bugs VueJs
n°9	Redesign IHM	n°10	Améliorations VueJS
n°11	Recherches " XML Moodle"		

BILAN

BILAN DU STAGE

Mon impression finale sur le stage est très positive.

La mission s'est révélée totalement adaptée en termes de complexité attendue pour un stage de Master 2

Aucune difficulté majeure n'a pu être rencontrée durant ce stage (excepté peut être celle mentionnée plus haut), seulement quelques points techniques (principalement liés à VueJS) réglés par consultation de la documentation officielle ou des forums spécialisés.

J'ai pu travailler sur les deux versants du développement web (côté client et côté serveur) sur deux nouveaux Frameworks que je ne connaissais pas, mais aussi retravailler sur certains aspects adjacents du développement web (conception de projet, configuration d'un serveur apache).

Polytech Orléans (et le laboratoire PRISME) offrent un cadre de travail calme et sérieux dans lequel j'ai pu facilement évoluer au quotidien pour effectuer ma mission.

La collaboration et le suivi du stage avec M. Rodolphe Weber se sont révélés très productifs : des points réguliers étaient faits (sur place ou à distance) pour contrôler que la mission avance et dans le bon sens mais une grande liberté et autonomie m'a été laissée. Pour suivre les tâches auxquelles je m'attelais, un suivi via l'outil Issues de GitLab a été mis en place.

De plus, une grande flexibilité d'organisation a été possible pour l'avancée de ma mission car j'ai pu mettre en place du télétravail partiel pendant mon stage mais aussi deux semaines et demie de télétravail intégral en août et 4 semaines de télétravail en septembre.

Ces conditions de stage m'auront montré de manière importante la réalité de la conduite d'un projet à terme assez important et la difficulté de choix concernant la conduite du projet au jour le jour et à long terme.

J'aurai aimé finir totalement cette mission, mais la tâche était trop vaste pour être finir en six mois. OpenQuizz devra être poursuivi par un autre développeur pour pouvoir être concrètement utilisé par du personnel pédagogique.

BILAN DE LA FORMATION

Clôture des deux années de Master m'ayant apportées la connaissance, d'un côté, des Web Services, et d'un autre côté, des framewoks « frontend », j'ai entamé ce stage avec déjà « plusieurs flèches dans mon arc » pour répondre aux problématiques de la mission proposée par R. Weber.

Cependant la flèche la plus importante issue des deux ans de Master (et dans une échelle plus large de mes cinq années universitaires) n'est pas la connaissance de tel ou tel outil mais une méthodologie de « réflexion informatique » qui me permet de pouvoir appréhender une problématique informatique et d'y apporter des solutions appropriées.

De ce fait, j'ai pu me lancer dans cette mission avec l'aide de deux frameworks nouveaux pour moi ainsi qu'avec l'outil MongoDB, que je n'avais que très peu pratiqué.

Ce Master aura été enrichissant pour moi car il m'aura permis de progresser dans le développement web, mais également de voir d'autres « types » d'informatique, comme le Big Data et les Systèmes d'Information Géographiques.

Ce stage à renforcé mon choix de m'orienter professionnellement vers une carrière dans le développement web, idéalement du côté « backend »

ANNEXE

12/04/2021

Application de gestion des quiz

Analyse fonctionnelle

POLYTECH

Jérémie Passerat

SOUS LA SUPERVISION DE M. RODOLPHE WEBER

Table des matières

Introduction

L’enseignement universitaire en 2021, et plus en particulier depuis le début de la pandémie de Covid-19, est de plus en plus porté vers l’utilisation des outils numériques pour proposer du contenu aux étudiants, via des outils en ligne comme Moodle ou autres.

Ces plateformes offrent diverses possibilités pour interagir et proposer du contenu pédagogique, et en particulier un système de quiz notés, reprenant une série de questions choisie par un professeur.

Cependant ces outils peuvent être inadaptés sur certains points et assez rigides dans la gestion des quiz.

Le projet (introduit ci-dessous) exposé dans ce document s’insère dans ce cadre. Pour le réaliser, un étudiant en stage de fin d’année de Master 2 y a été affecté.

Les procédures mentionnées sur ce document seront axées sur le fonctionnement de Moodle, mais il est prévu un outil qui pourrait s’adapter à toutes les plateformes.

Présentation du projet

L'objectif du projet est la création d'une plateforme permettant la « création », **le partage, la maintenance et la gestion** de contenus pédagogiques (quiz, scénari, h5p, …)

Dans ce document, on se focalisera sur le contenu « quiz » sans que ce soit réducteur. Un quiz est considéré comme étant la combinaison d'une question et d'une réponse associée.

Un utilisateur de l'application devra pouvoir accéder à une banque de questions publiques, un espace unique ou toutes les quiz publiés des autres utilisateurs devront être disponibles. L'application doit donc permettre à plusieurs utilisateurs d'interagir en même temps.

Ces quiz devront pouvoir évoluer, en conservant les caractéristiques de chaque version, être exportables (pour les intégrer dans Moodle ou autres), être importables (si l'on créée une question sur Moodle / H5P par exemple, pouvoir l'insérer sur la plateforme doit être possible).

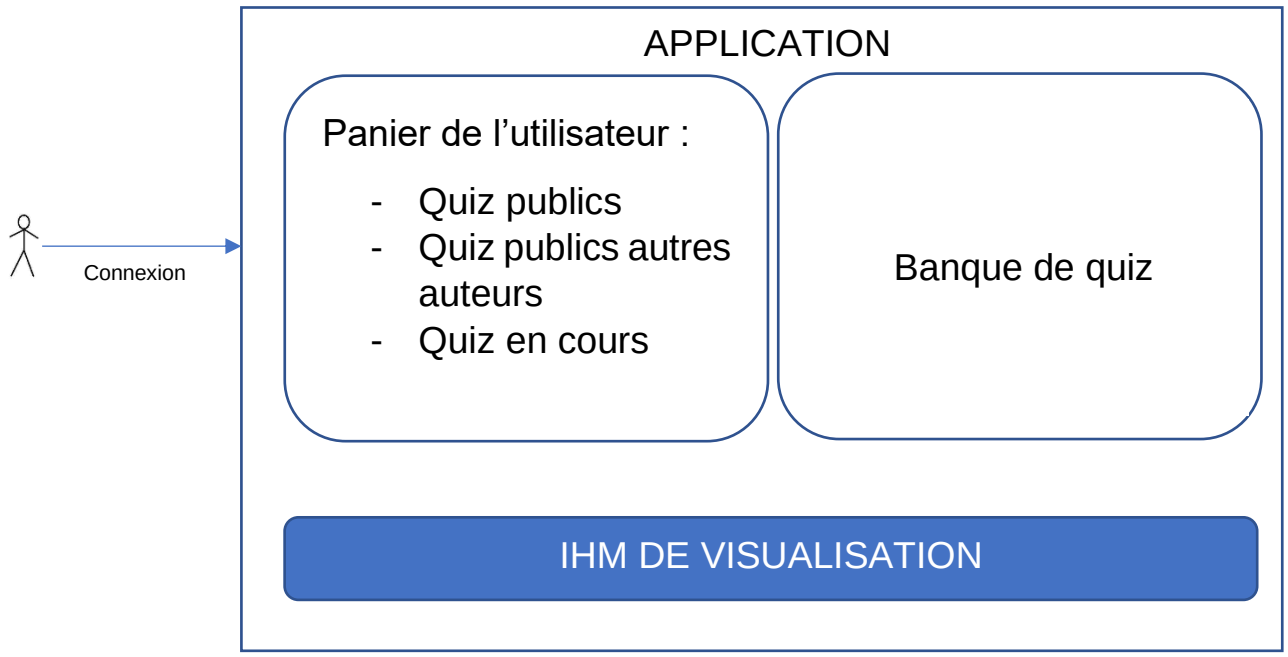
Le dernier point à prendre en compte est la possibilité d'évaluer la qualité et la popularité de chaque quiz.

Partie 1 : La « vie » des quiz

Présentation générale

Un utilisateur qui se connecte sur l'application arrive sur un espace ou deux parties majeures lui sont affichées :

- La première partie est son « panier », un espace où il dispose de : tous les quiz qu'il a créé et publié, tous les quiz d'autres auteurs qu'il a mis en « favori », tous les quiz en cours de modification, que ce soit les siennes ou celles d'auteurs tiers.
- Un espace public reprenant toutes les quiz publiés d'autres auteurs, appelé la « banque de quiz »
- A chaque clic sur un quiz, il pourra en voir le détail (historique des versions, « note », question et réponse, …)



Les scénarios

Pour interagir avec les questions les cinq scénarios suivants ont été imaginés :

- La « création »
- La modification d’un quiz personnelle
- La modification d’un quiz tierce (supprimée ou non)
- La suppression d’un quiz personnelle
- La « mise en favori » d’un quiz tierce

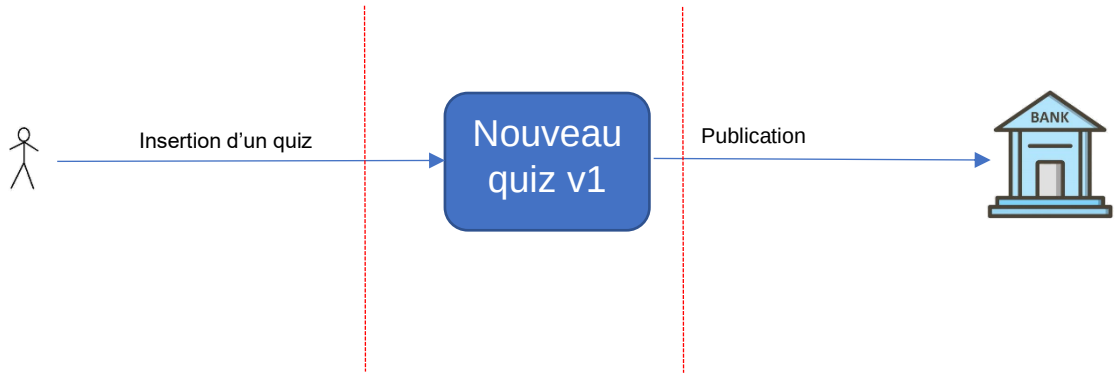
Chacun de ces scénarios sera détaillé dans les pages suivantes.

Scénario 1 : la création

Le terme de création n'est pas pleinement représentatif mais ce scénario s'intéresse aux moyens d'intégrer un quiz dans le cycle de gestion de l'application. La question n'a pas été intégralement tranchée mais l'idéal serait de permettre l'intégration de quiz créés par des plugins existants dans notre application pour ensuite les faire vivre. La création via cet outil est, pour le moment, une piste non privilégiée.

Dans le processus ci-dessous, une question est importée dans le logiciel. Elle est alors disponible dans les « quiz en cours » de l'auteur, qui devra la publier pour qu'elle apparaisse dans la banque.

La banque a vocation à répertorier les questions publiées dans leur version courante, ici la v1.



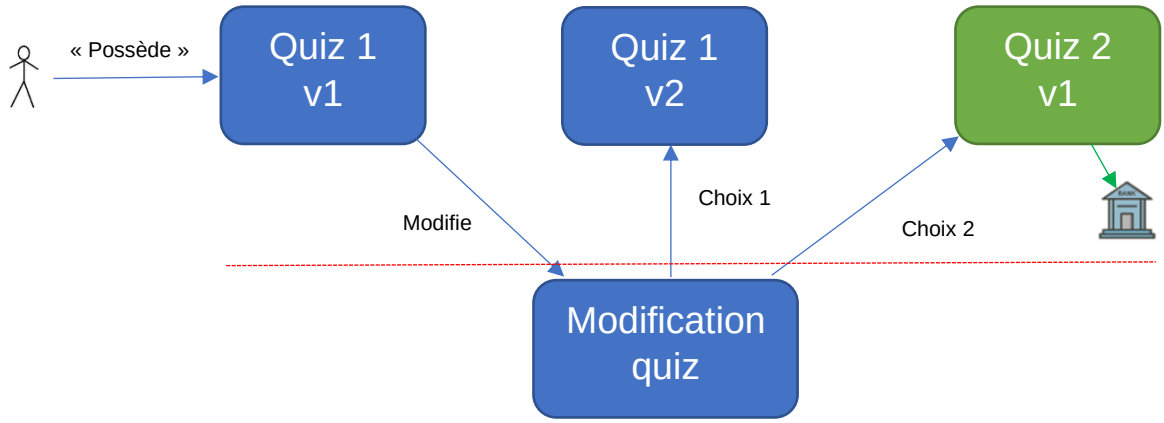
Entre ces deux lignes, le quiz est dans « quiz en cours », donc non publique.

Deux situations sont à distinguer concernant la modification d'un quiz existant.

La situation de départ est cependant tout le temps la même : un auteur 1 créée un quiz 1, qui est publié dans la banque de quiz dans sa version 1.

Si l'auteur 1 décide de modifier sa question 1, il a deux possibilités : il fait évoluer sa question 1 de la version 1 à la version 2 (ce qui modifiera la version présente en banque) ou alors il décide de garder son quiz 1 en version 1 et créée une question 2 en version 1 (et cela créée une nouvelle entrée dans la banque de question).

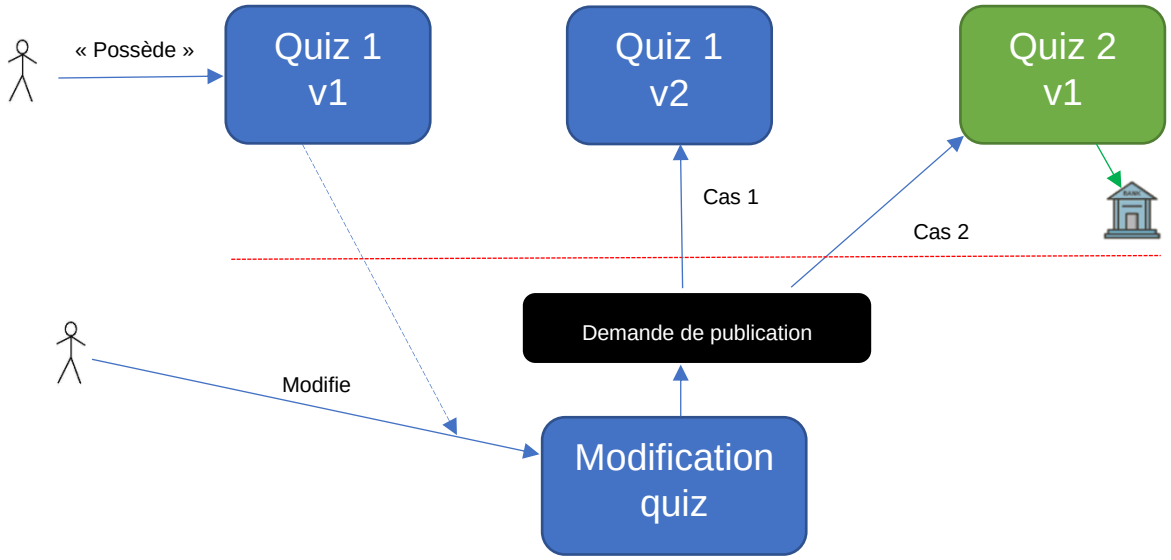
La suppression est une possibilité tant que l'utilisateur est en cours de modification



Choix 1 : La question modifiée met à jour la question initiale (et son indice courant)

Choix 2 : La question modifiée devient une nouvelle question indépendante (et une nouvelle entrée dans la banque)

Si un auteur 2 décide de modifier la question 1 de l'auteur 1, il créée une demande de publication auprès de l'auteur 1 dès que la modification est validée. Si l'auteur 1 l'accepte, sa question 1 passe en v2 ; sinon, l'auteur 2 peut s'approprier ses modifications et créer une question 1 v1 « à son nom ».



Cas 1 : L'auteur autorise la modification, sa question passe en v2 (indice courant changé dans la banque)

Cas 2 : La question modifiée devient une nouvelle question indépendante (et une nouvelle entrée en banque)

A noter qu'un quiz en cours de modification est visible uniquement dans le panier de l'auteur qui le modifie et nulle par ailleurs, tant que la modification n'a pas été validée. Il est également possible pour ce dernier de supprimer sa version de travail.

Scénario 4 : la suppression

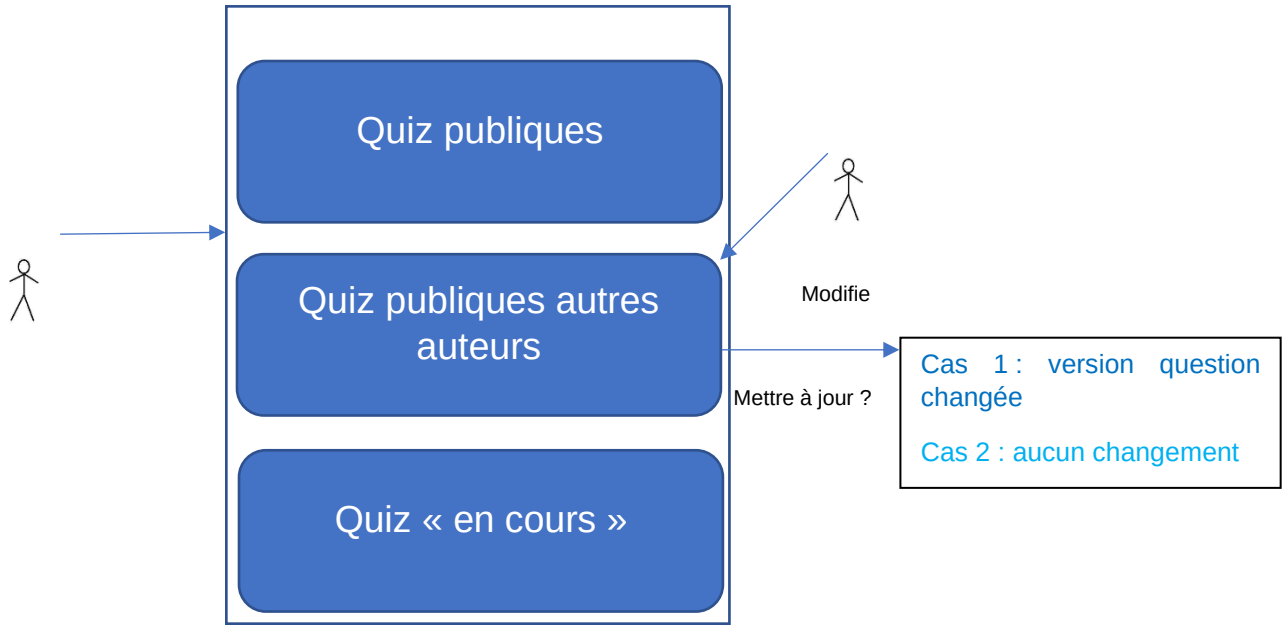
Un auteur peut décider de « renier » un de ses quiz. Cette dernière continue alors à exister mais devient sans auteur, sauf si elle n'a été mise en favori par personne, auquel cas le quiz est détruit

Si un auteur « tiers » décide de modifier un quiz « orphelin » cela lui permet de s'affranchir de la phase de demande de publication et la modification est automatiquement acceptée. L'auteur venant de créer la modification est directement propulsé en tant que nouvel auteur du quiz autrefois « orphelin ».

Scénario 5 : la mise en favori

La mise en favori consiste à « s'approprier » un quiz d'un autre auteur (la partie quiz public autres auteurs du panier) afin d'en conserver une trace. Pour ce faire, il faut aller dans la banque de quiz et sélectionner la version désirée du quiz (car les versions sont conservées au fur et à mesure).

Dans le cas ou un quiz en favori est sujet à une modification (que ce soit une modification directe ou la création d'une nouvelle question à partir de la modification), l'utilisateur possédant le favori se voit proposer de mettre à jour la version. S'il accepte, son favori change de version ; si il refuse, il garde son favori tel quel.



Partie 2 : Les Tags et la Recherche

Les Tags

L'idée derrière la mise en place d'un système de tag associé à chaque question est de permettre une catégorisation de chaque question.

Deux systèmes de tags ont été imaginés :

- Au niveau de la banque de quiz, un système de tags « généraux », prédéfinis à la création de l'application (et gérables par une personne de type administrateur), est associé à chaque question.

  L'idée globale est de mettre des tags organisés hiérarchiquement, du plus général au plus détaillé, comme ceci par exemple :

  Niveau 0 : Sciences / Management / …

  Niveau 1 (dans Sciences) : Informatique / Electronique / …

  Et on peut affiner si besoin.

- Au niveau du panier utilisateur, il est possible de définir des tags personnalisés, permettant une recherche facilitée et adaptée à ses envies

La recherche

La recherche (au niveau de la banque de quiz) est possible sur deux niveaux :

- Sur les tags « généraux »
- Sur le titre ou le contenu de chaque quiz.

Partie 3 : La « valorisation »

Deux systèmes parallèles ont été imaginés pour cette partie :

- Une évaluation de la qualité d'un quiz via un système de likes. Un utilisateur pourra aimer 1 fois chaque question. Ce système sera aussi mettable en place pour faire connaitre un auteur.

- Une évaluation de la popularité d'une version d'un quiz. Dès qu'une version d'un quiz est mise en favori, le compteur associé est incrémenté. Cela permet, quand on observe le détail d'un quiz, d'observer la popularité de chacune de ses versions.

Partie 4 : La création d’évaluations

La finalité d’un quiz est d’être groupé avec d’autres pour figurer dans une évaluation. Il serait donc appréciable de pouvoir profiter d’une fonctionnalité permettant de le faire dans cet outil (regrouper plusieurs questions sous une même catégorie pour Moodle).

L’objectif pour la création de ces quiz est également de conserver l’information sur les quiz intégrés à cette question (pour pouvoir les varier d’une année à l’autre, par exemple)

Pour ce faire l’évaluation sera mémorisée (avec les questions associées) et au niveau de chaque question les quiz associés (pour un utilisateur) seront mentionnés.

Partie 5 : L’import/export

Dans les conditions actuelles, ces deux activités se feraient à la main, en manipulant les fichiers. On créée un quiz sur Moodle, on l’exporte sur l’application web et on lui fait vivre le cycle de vie. Pour l’export dans l’autre sens, il faut télécharger le quiz et l’importer dans Moodle

Dans l’idéal il serait intéressant de pouvoir faire ces deux taches de manière automatisée. Pouvoir exporter vers Moodle et autres en cliquant sur un bouton et trouver une manière automatisée de profiter des outils de création de quiz existants. Dans ce cadre, garder les fonctionnalités mentionnées sur le paragraphe précédent pourrait rester intéressant.