Projet Cloud

Vous travaillez pour une entreprise qui souhaite proposer un système de Storage as a service (STaaS), permettant de mettre à disposition de ses clients un stockage de fichiers à distance. Vous êtes chargé d'élaborer une application web permettant aux clients de consulter et de gérer l'utilisation des disques qu'ils louent, et aux administrateurs de l'entreprise de visualiser les disques.

Résumé

Projet initial à utiliser	• Projet Cloud v1.0 sur gitHub • La base de données est dans le dossier app/database du projet
Outils	• Bootstrap • Micro-framework
Principales fonctionnalités	• Module client • Module Administration
Livraison	• en continu sur gitHub jusqu'au jour de livaison fin octobre

Ressources

- Prise en main Micro-framework
- Documentation API Micro-framework et cloud
- Twig documentation

Règles de gestion

Les clients peuvent disposer d'espaces de stockage (nommés disques) permettant de stocker leurs données. Chaque disque est loué à un certain tarif par le client ; le tarif comprend :

- Un prix (mensuel)
- Un quota (capacité de stockage exprimée dans une unité (o, Ko, Mo, Go ou To)
- Une marge de dépassement, à ne pas dépasser sous peine d'être facturé d'un surcoût.

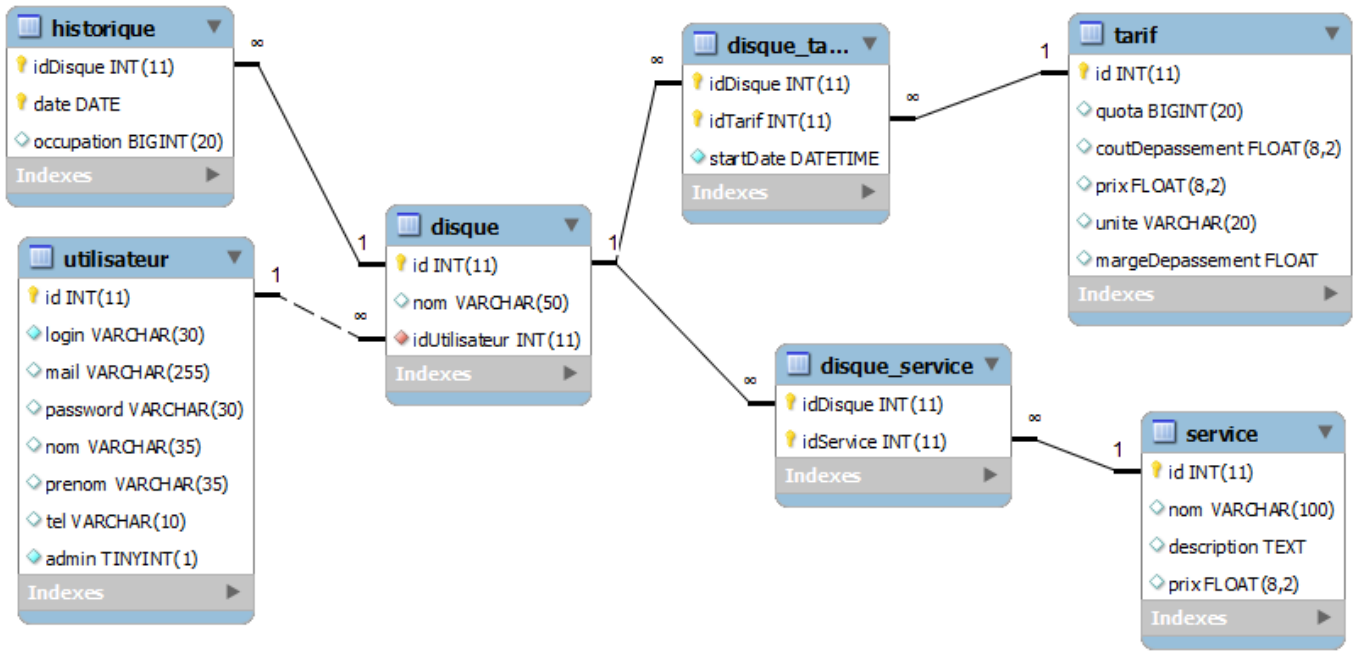
Le client a la possibilité de changer de tarif à tout moment pour chaque disque, de façon à diminuer ou augmenter son quota, en fonction de ses besoins. L'historique des tarifs associés à un disque est conservé.

Le tarif courant appliqué à un disque est celui correspondant à la date la plus récente.

Chaque location de disque peut être associée à des services (backup, loadBalancer...).

L'historique d'occupation des disques est obtenu de manière automatisée par un script PHP lancé de manière quotidienne par une tâche CRON.

Schéma de la base



Détail des fonctionnalités à mettre en place (sprint 1)

Les différentes pages devront gérer les droits, et n'être autorisées que pour les acteurs mentionnés.
Tenir compte du fait que l'administrateur a le droit d'accéder à tout ce qui est autorisé à l'utilisateur.

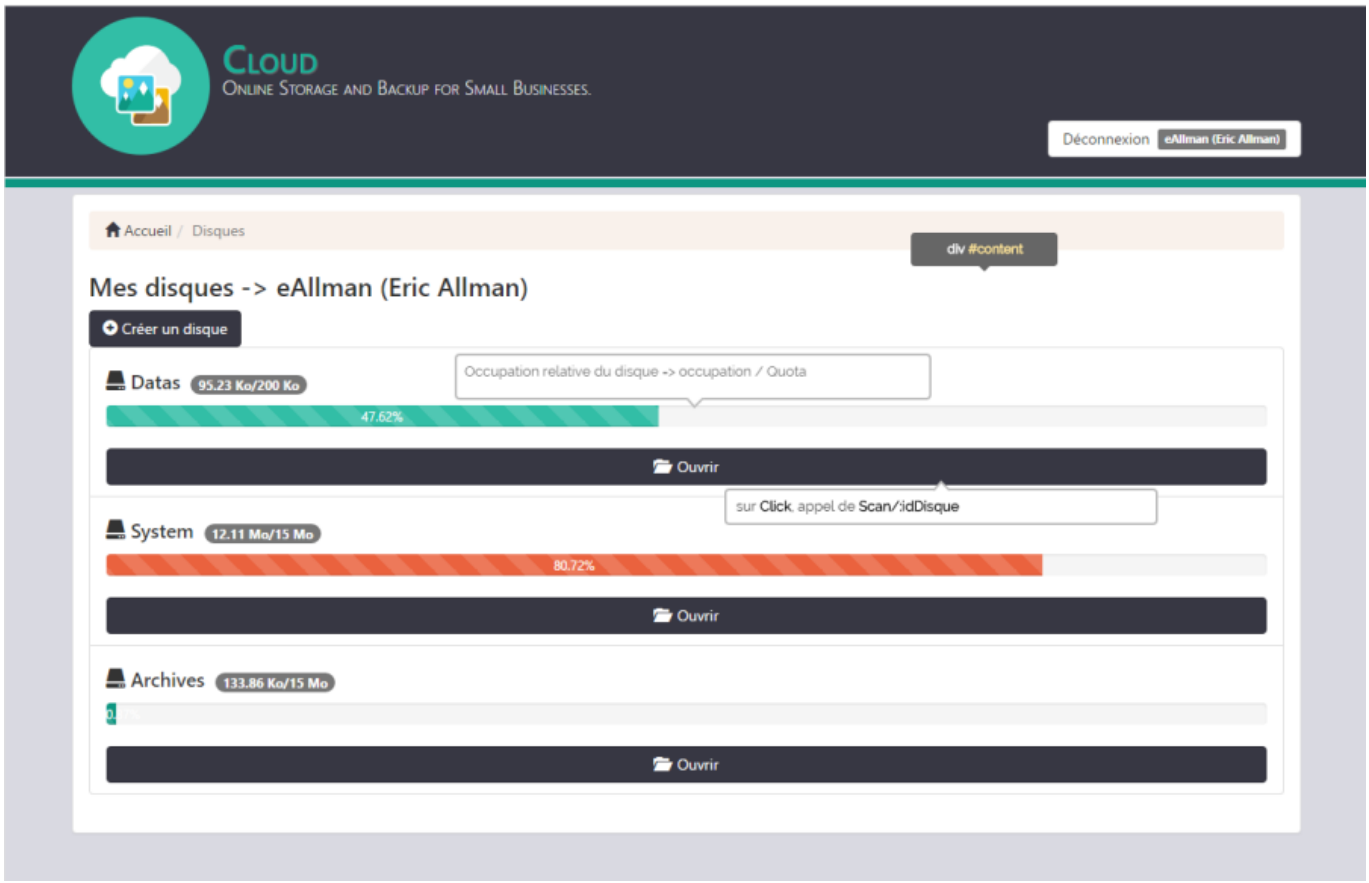
-- Module Client

-- Affichage des disques d'un client

//TODO 1.1

A l'adresse **MyDisques/index**, on souhaite afficher les disques de l'utilisateur actuellement connecté.

L'utilisateur connecté est obtenu par l'appel de la méthode **Auth::getUser()**.



Indications

Elément	**Indications**
Contrôleur	MyDisques
Action	index
Utilisateur connecté	L'utilisateur connecté est obtenu par l'appel de la méthode **Auth::getUser()**.
Occupation, Quota	le quota est obtenu sur le tarif actuel du disque, la classe **ModelUtils** permet de connaître l'occupation en cours du disque
Composants visuels Bootstrap	on pourra utiliser les composants bootstrap **Progressbar**, **GlyphButton**, **Listgroup**
Accès à la config du cloud	Les disques clients sont localisés dans le dossier **/files** de l'application, et les disques y sont localisés sous le nom : **$cloud→prefix-[disque.nom]**. Cette configuration est définie dans le fichier de **config.php** dans la variable **cloud**, accessible depuis les contrôleurs par **$GLOBALS[“config”]→cloud**
Tailles en octets	Vous pourrez utiliser la méthode **sizeConverter** de **ModelUtils**

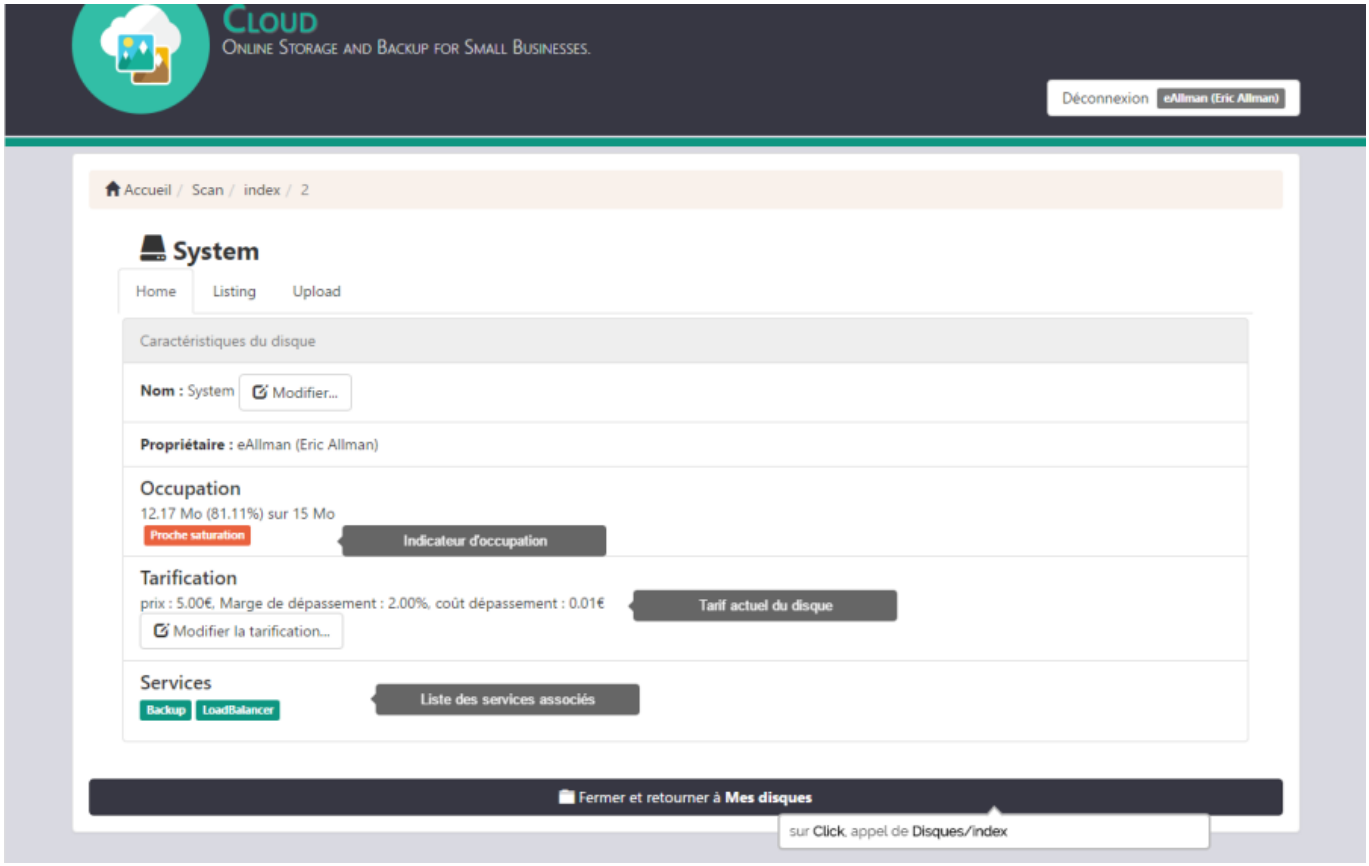
Le style des progressbars doit donner une indication sur le taux d'occupation en % :

Style	**Valeurs**
info	de 0 à 10%
success	de 10 à 50%
warning	de 50 à 80%
danger	plus de 80%

-- Affichage d'un disque

//TODO 1.2

A l'adresse **Scan/show/:idDisque**, afficher l'interface suivante :



Elément	**Indications**
Contrôleur	Scan
Action	show
Paramètre	idDisque
Utilisateur connecté	L'utilisateur connecté est obtenu par l'appel de la méthode **Auth::getUser($controller)** où **$controller** est le contrôleur actif.
Occupation, Quota	le quota est obtenu sur le tarif actuel du disque, la classe **ModelUtils** permet de connaître l'occupation en cours du disque
Composants visuels Bootstrap	on pourra utiliser les composants bootstrap **Label**, **GlyphButton**, **Listgroup**

L'indicateur d'occupation (**Label**) doit donner une indication sur le taux d'occupation en % :

Texte	**Style**	**Valeurs**
Peu occupé	info	de 0 à 10%
RAS	success	de 10 à 50%
Forte occupation	warning	de 50 à 80%
Proche saturation	danger	plus de 80%

-- Création d'un disque

//TODO 1.3

A partir de l'adresse **Disques/frm**

Afficher le formulaire de création d'un disque.

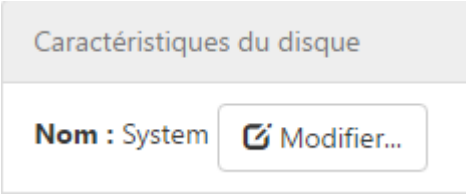
Soumettre le formulaire à l'adresse **Disques/update** pour créer le disque puis rediriger vers l'adresse **Scan/show/:idDisque**

Penser à le créer physiquement.

-- Renommage disque

//TODO 1.4

A partir de l'adresse **Scan/show/:idDisque**



Implémenter la modification du nom d'un Disque (avec renommage éventuel du dossier associé + vérification du nom)

-- Changement du tarif

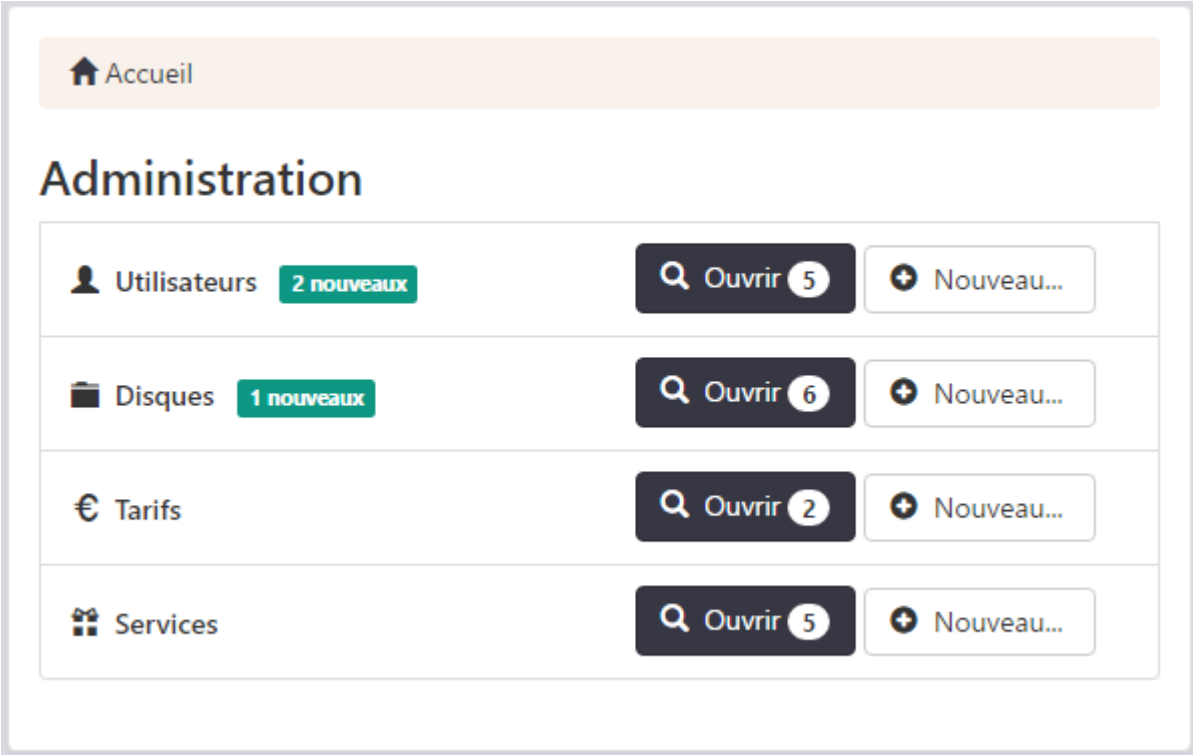
//TODO 1.5

Implémenter la modification du tarif par le client (avec vérification de non dépassement du quota)

-- Module administration

-- Page d'accueil

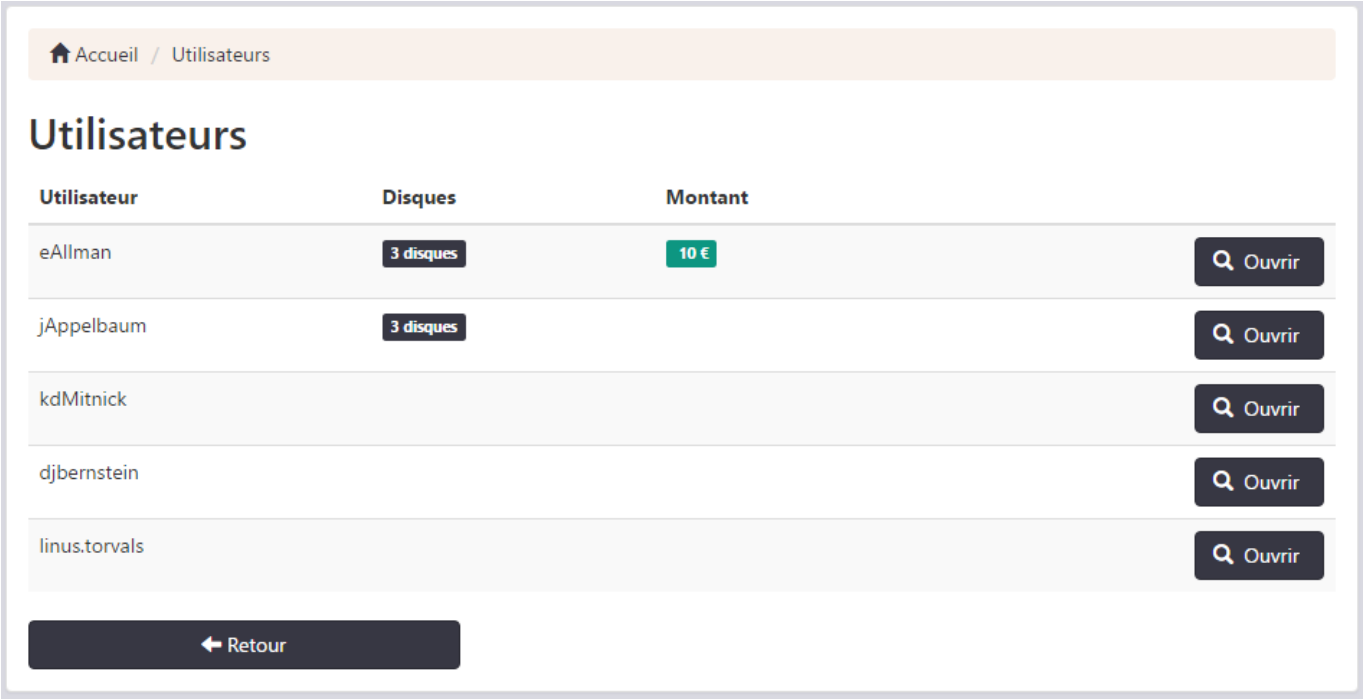
//TODO 2.1



Elément	Indications
Contrôleur	Admin
Action	index
Indication “nouveau”	Affiche le nombre d'utilisateurs ou de disques créés dans la journée.
Badge du Bouton ouvrir	indique le nombre d'instances dans les tables

-- Accès aux Utilisateurs

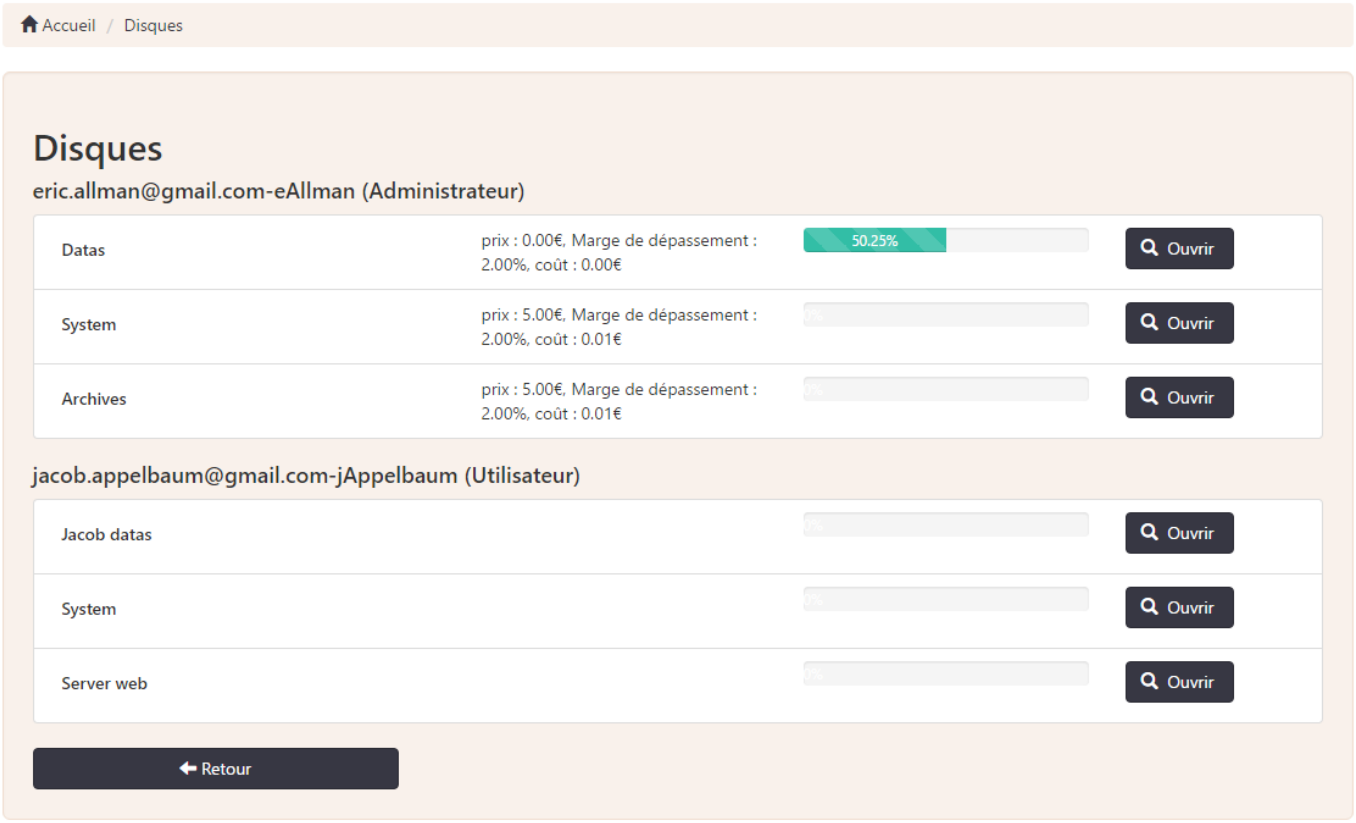
//TODO 2.2



Elément	Indications
Contrôleur	Admin
Action	users

-- Accès aux disques

//TODO 2.3



Elément	Indications
Contrôleur	Admin
Action	disques
paramètre	idUtilisateur(Facultatif : si précisé affiche les disques de l'utilisateur correspondant)

-- Module commun

-- Contrôle d'accès

//TODO 3.1

URL	Accès	Résultat/message
Admin/*	Accessible uniquement pour un Administrateur connecté	Afficher un message d'erreur “Accès à une ressource non autorisée”
Disques/index	Accessible uniquement pour un utilisateur connecté	Afficher message sur l'absence de connexion et proposer la connexion
Scan/index/:idDisque	Accessible uniquement pour un disque appartenant à l'utilisateur connecté	Afficher un message d'erreur “Accès à une ressource non autorisée”

Remarque : L'administrateur hérite des droits de l'utilisateur.

-- Fil d'ariane

//TODO 3.2

Le fil d'ariane (Breadcrumbs bootstrap) doit indiquer à tout moment la localisation de l'utilisateur à partir de sa page d'accueil :

Exemple :

Accueil / Disques

Contraintes techniques

- L'application sera développée en PHP objet, et utilisera un micro-framework facilitant les échanges avec la base de données.
- Elle respectera au mieux la séparation des couches (objets Métiers), classes techniques et vues (interfaces web de saisie et d'affichage).
- Elle utilisera la base de données Mysql fournie en annexe. Cette base pourra évoluer en fonction des besoins du développement.
- L'utilisation de scripts côté client (javascript et ajax) pourra compléter les validations côté serveur. L'utilisation du micro-framework est obligatoire, d'autres librairies ou frameworks peuvent le compléter..
- Bootstrap sera utilisé pour la partie présentation.

Fichiers

- Projet Github initial à cloner

Compléments

Bonnes pratiques

- Alimenter correctement la base de données en ajoutant des enregistrements valides et en nombre suffisant, mettant en valeur votre travail
- respecter la Normalisation HTML 5/Css 3
- Structurer les fichiers et dossiers de manière cohérente
- Nommer en respectant les normes et de manière significative (Contrôleurs, vues, méthodes, variables...)

A produire/livrer

- Analyse fonctionnelle (diag uses cases, descriptifs, maquettes)
- Application
- Documentation technique
- Tests (fonctionnels et unitaires)

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/slam4/cloud?rev=1493018604

Last update: **2019/08/31 14:38**