Stockage dans le cloud décentralisé

DA2

Introduction :

Ce document aura pour but de présenter les documentations liées au projet StoreIt. Il sera présenté en trois parties en commençant par la *documentation utilisateur* composé du manuel d'utilisateur, la *documentation technique* qui sera composé du manuel développeur et pour finir, *la documentation d'installation* qui détaillera la procédure d'installation du projet.

Description du document

Titre	DA2_StoreIt
E-Mail	Storeit_2017@labeip.epitech.eu
Auteurs	Adrien MOREL Kenny NGUYEN Alexandre FULGONI Louis MONDESIR Romain GJURA
Date	19 Juillet 2016
Version du modèle	1.0

Tableau des révisions

Date	Auteur	Section(s)	Commentaire
19/07/2016	Kenny NGUYEN	Toutes	Mise en page du document
15/07/2016	Adrien MOREL Alexandre FULGONI Louis MONDESIR Romain GJURA	Toutes	Rédaction des parties respectives

SOMMAIRE

I- Documentation utilisateur
1) Le serveur StoreIt page 5
a. Utilisation du serveur page 5
2) Web-App page 6
a. Login page 6
b. Explorateur page 6
c. Menu Contextuel page 7
3) Application iOS page 8
4) Application Android page 10

II- Documentation technique
1) Le serveur StoreIt page 13
a. Le protocole de communication réseau page 13
2) Contribuer au projet page 14
a. Obtenir le code page 14
b. Norme page 14
c. Les tests page 14
3) Application Android de StoreIt page 15
a. Liste des logiciels requis page 15
b. Procédure de mise en place du projet page 15
c. Architecture du projet page 15
d. Procédure de test page 15
4) Application iOS de StoreIt page 16
a. Conception du projet page 16
b. Test unitaires page 16
c. Norme page 16
5) Web-App page 17
a. Technologies page 17
b. Structure page 17
c. Développement page 17

III- Documentation d'installation
1) Le serveur page 19
a. Prérequis page 19
b. Installation du serveur page 19
c. Lancement du serveur page 19
2) Application iOS page 20
a. Installation de l'application page 20
3) application Android page 21
a. Installation de l'application page 21
4) Web-App page 21
a. Installation de l'application page 21

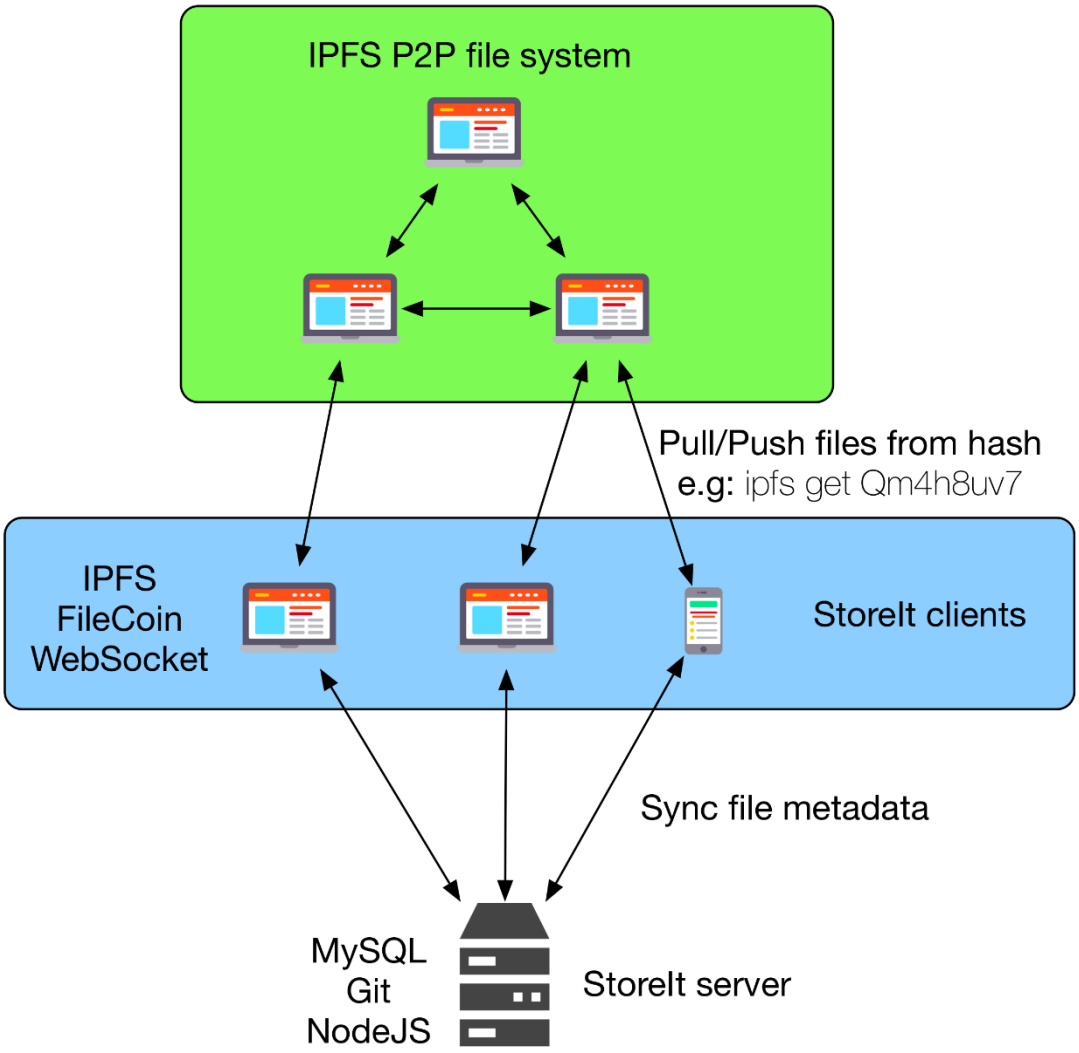
I

Documentation Utilisateur

1)Le serveur

a. Utilisation du serveur

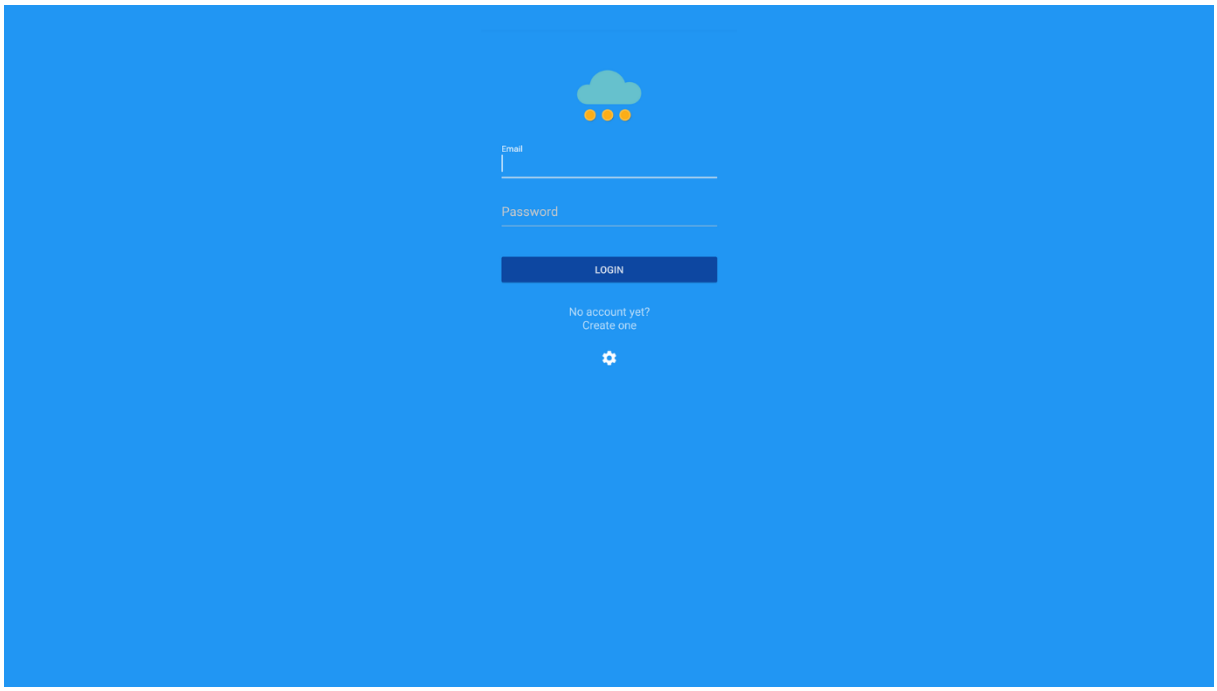
Le serveur est un programme en ligne de commande. Comme on peut le voir dans le schéma ci-dessous, il permet la synchronisation de tous les clients IPFS.



2) Web-App

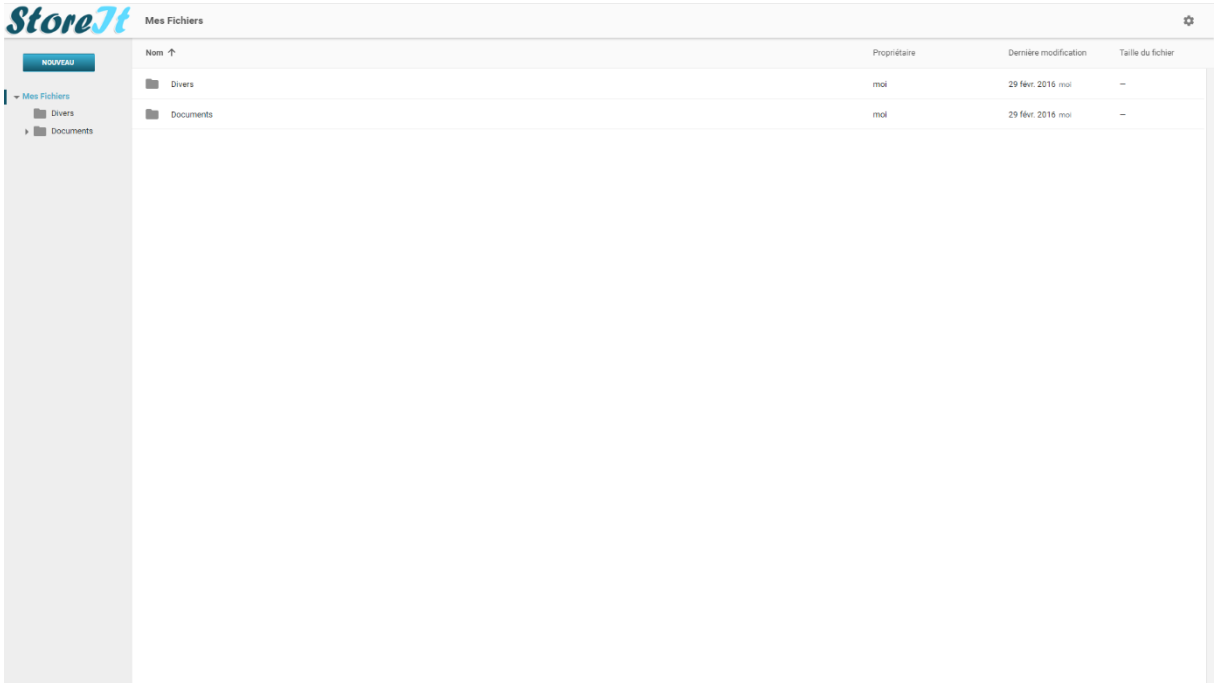
a. Login

L'utilisateur s'authentifie au serveur via son couple identifiant/mot-de-passe



b. Explorateur

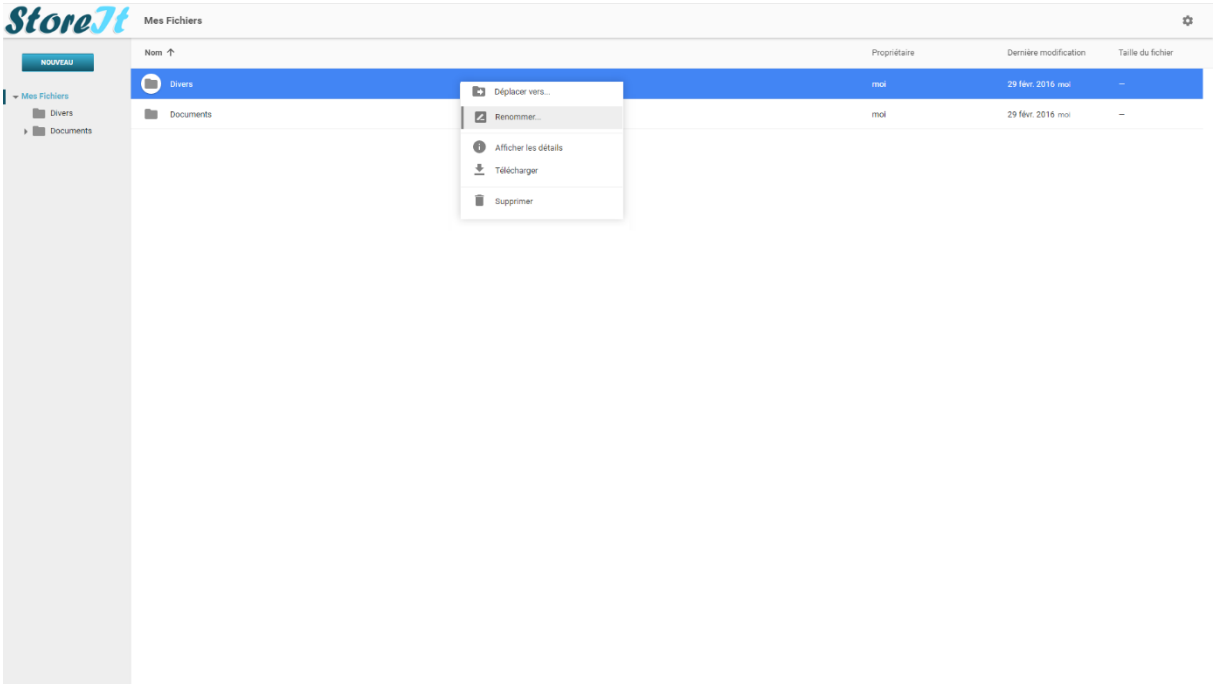
Une fois connecté, l'utilisateur est redirigé vers le gestionnaire de fichiers.



La sidebar présente une vue compacte de l'arborescence et le bouton Nouveau permet d'importer un nouveau fichier.

c. Menu contextuel

Le clic droit sur un fichier ouvre le menu contextuel.

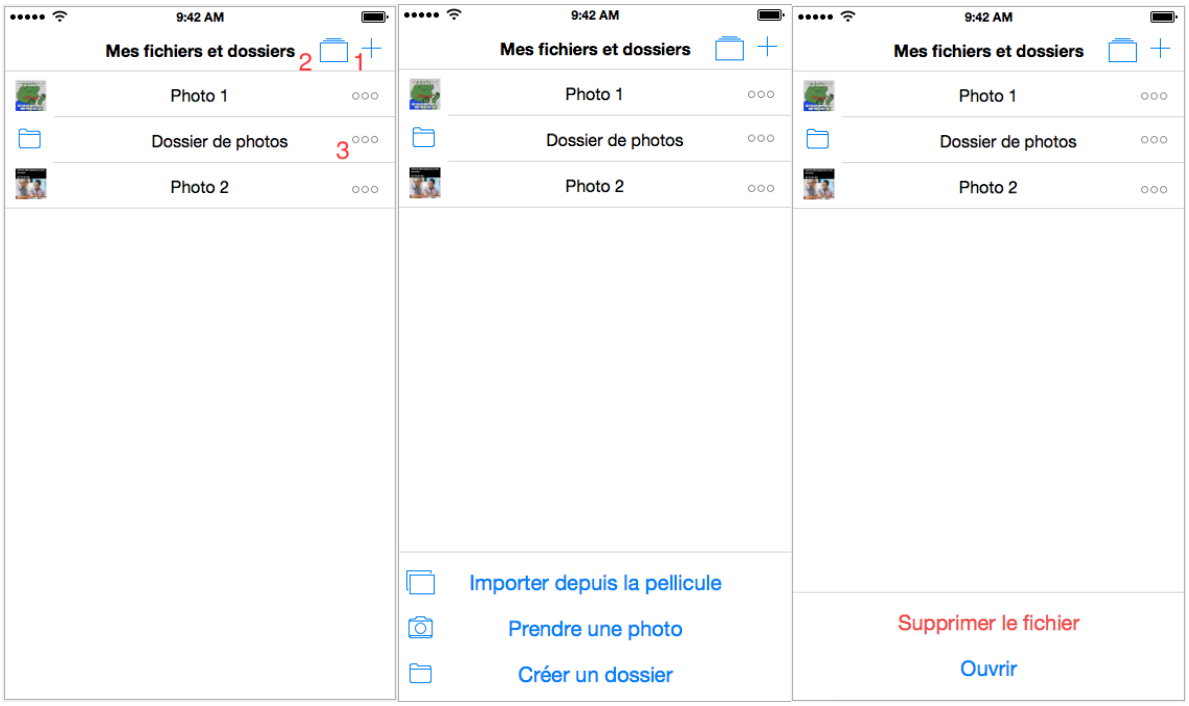


Il propose les options suivantes:

- Déplacer le fichier
- Renommer le fichier
- Afficher les informations sur le fichier
- Télécharger le fichier
- Supprimer le fichier

3) Application iOS

L'utilisateur dispose d'un écran de connexion pour accéder au contenu de l'application. Il peut se créer un compte s'il n'en possède pas. Il peut aussi changer son mot de passe en cas d'oubli.



Ceci est l'écran principal de l'application. Il permet de lister les fichiers et dossiers présents dans le dossier synchronisé StoreIt. L'utilisateur a alors accès à plusieurs fonctionnalités. L'affichage par défaut est en liste, mais l'utilisateur à la possibilité ses fichiers et dossiers en grille (2).

Grâce à l'icône (1), l'utilisateur peut :

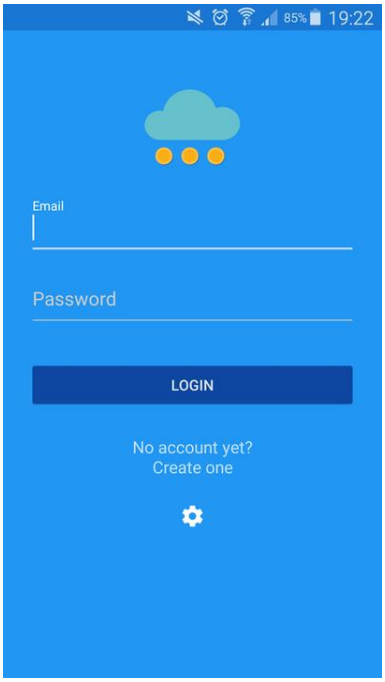
- Importer des fichiers dans le dossier synchronisé StoreIt
- Réorganiser le contenu du dossier synchronisé (créer des nouveaux dossiers, déplacer des fichiers et les supprimer)

Lorsque l'utilisateur utilise l'icône (3), un menu contextuel est alors ouvert et il peut effectuer des actions sur le dossier ou fichier concerné, comme :

- Le supprimer
- L'ouvrir
- Consulter des informations relatives à ce fichier ou dossier

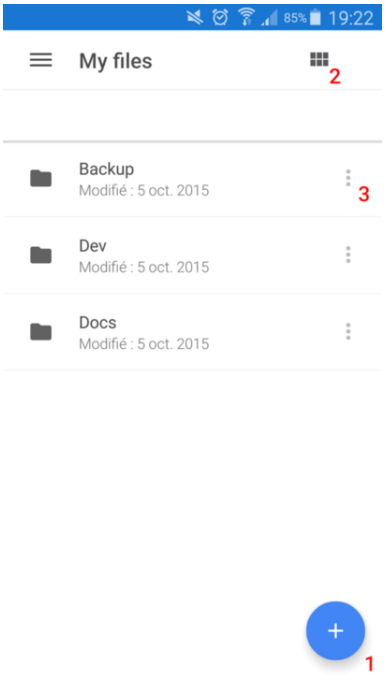
4) Application Android

- Ecran de connexion



L'utilisateur dispose d'un écran de connexion pour accéder au contenu de l'application. Il peut se créer un compte s'il n'en possède pas. Il peut aussi changer son mot de passe en cas d'oubli.

- Affichage du dossier synchronisé

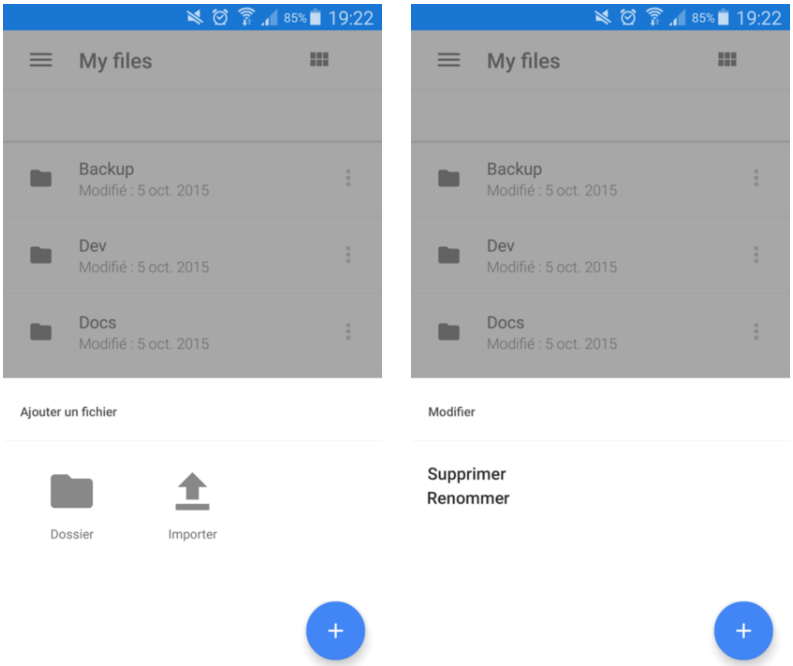


Ceci est l'écran principal de l'application. Il permet de lister les fichiers et dossiers présents dans le dossier synchronisé StoreIt. L'utilisateur a alors accès à plusieurs fonctionnalités.
L'affichage par défaut est en liste, mais l'utilisateur à la possibilité ses fichiers et dossiers en grille (2).

- Importer des fichiers et organiser le dossier synchronisé

Grâce à l'icône (1), l'utilisateur peut :

- Importer des fichiers dans le dossier synchronisé StoreIt
- Réorganiser le contenu du dossier synchronisé (créer des nouveaux dossiers, déplacer des fichiers et les supprimer)



- Menu contextuel d'un fichier ou dossier

Lorsque l'utilisateur utilise l'icône (3), un menu contextuel est alors ouvert et il peut effectuer des actions sur le dossier ou fichier concerné, comme :

- Le supprimer
- Le renommer

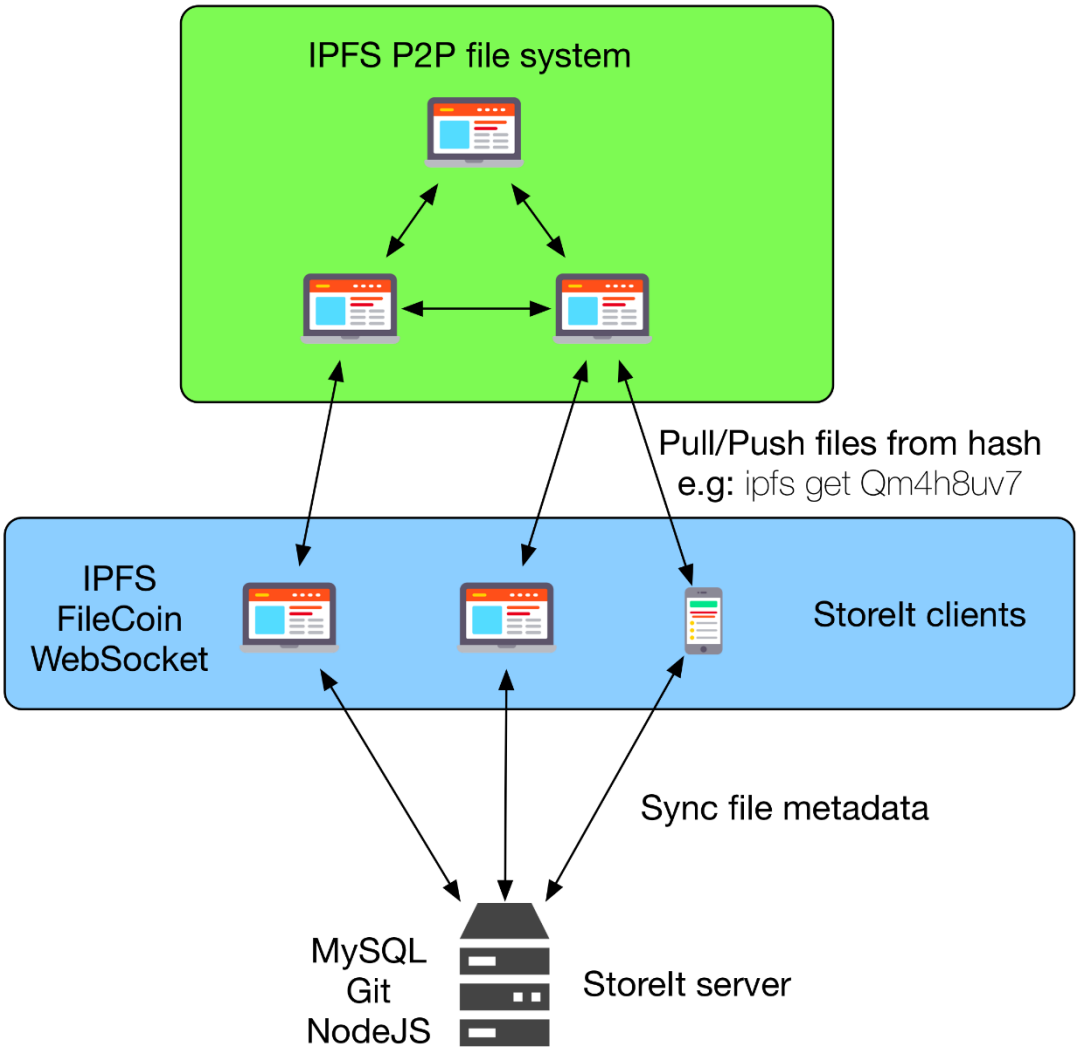
II
Documentation Technique

Ce document est une présentation technique du projet. Il couvre la conception et sert de guide à toute personne souhaitant contribuer au projet. Nous y évoquons les différents composants logiciels (client, serveur, appli Android, appli iOS, appli web) et décrivons l'organisation du code, des tests, du déploiement, etc.

1) Le serveur StoreIt

a) Le protocole de communication réseau

StoreIt est un service de cloud décentralisé. Du point de vue d'un utilisateur, il est possible de synchroniser un dossier dans le cloud. Il est aussi possible de louer un disque au réseau StoreIt. Dans cette situation, le client est considéré comme un hébergeur va stocker de manière automatisée les données des autres utilisateurs.



La communication entre les clients et le serveur se fait via un protocole ASCII qui fonctionne au-dessus du protocole WebSocket.

La documentation du protocole est disponible dans le dépôt (https://github.com/Sevauk/storeit/blob/master/document/protocol/protocol.pdf)

2) Contribuer au projet

a) Obtenir le code

Le code source du serveur est disponible sur github (https://github.com/Sevauk/storeit).

```
git clone https://github.com/Sevauk/storeit
```

Consultez le document sur l'installation pour installer et lancer le serveur.

b) Norme

Le code doit respecter les normes ECMAScript en utilisant ESLint.

c) Les tests

Les outils qui sont utilisés ou qui vont être mis en place prochainement sont les suivants :

travis CI + slack intégration avec des webhooks istanbul (test coverage) mocha (test framework) chai (test DSL) coffeescript (language) pour les tests unitaires.

3) Application Android de StoreIt

a) Liste des logiciels requis

- Android Studio 2.1
- Git

b) Procédure de mise en place du projet

- Cloner le dépôt git
- Depuis Android Studio ouvrir le projet situé dans le répertoire **/src/android-app/StoreIt**
- L'ensemble des bibliothèques nécessaires seront automatiquement téléchargées via une tâche gradle

c) Architecture du projet

Le projet est organisé autour de 3 grandes parties :

1. Un module de gestion réseau (websocket)
2. Un module de gestion des fichiers
3. L'ensemble du code lié à l'interface utilisateur

Chacun de ces modules se situe dans un package Java différent

d) Procédure de test

Des tests unitaires ont été mis en place. Ils reposent sur l'utilisation de JUnit dont l'utilisation est détaillée ici http://developer.android.com/tools/testing/testing_android.html

Pour l'instant seulement la fonctionnalité de création de hash utilise des tests unitaires.

4) Application iOS de StoreIt

a) Conception du projet

Le projet est constitué de trois parties principales :

Network

Cette partie contient toutes les classes relatives à la gestion du réseau, en l'occurrence les classes Server et Client, ainsi que plusieurs classes utilitaires, comme la classe RequestBuilder qui permet la construction des requêtes envoyées par les deux classes citées précédemment. Client permet de recevoir et d'envoyer des requêtes au serveur, alors que Server permet de recevoir et d'envoyer des requêtes à d'autres clients.

FileManagement

Cette partie contient toutes les classes relatives à la gestion de fichier, aussi bien la gestion du dossier synchronisé StoreIt que le dossier de stockage des chunks.

Views

Cette partie contient toutes les vues de l'application.

b) Test unitaires

Les tests unitaires se trouvent dans leurs dossiers respectifs Xcode : StoreItTests et StoreItUITests. Ces dossiers contiennent des sous-dossiers correspondants aux parties évoquées ci-dessus.

c) Norme

Il n'y a pas de norme de codage particulière, si ce n'est qu'il faut suivre au possible les conventions établies par Apple

Consulter ces conventions ici.

5) Web-App

a) Technologies

- Langage: JavaScript (ES2015 + Tc39 stage-3)
- Framework: AngularJS 1.5+
- Templates: Jade
- Stylesheets: PostCss
- Gestionnaire de dépendances: JSPM
- Automatiseur de tâches: Gulp
- Testing: Karma + Mocha + Chai + Istanbul

b) Structure

Les sources de l'application sont organisées en Modules Angular. L'application contient 2 types de modules:

1. **Components**: Route + Controlleur + Vue (Template et Style). Un component est intimement lié à ça vue, mais peut être composé à partir de component plus petits.

2. **Core**: Contient de la logique réutilisable (principalement des services et models)

Example de component:

```
home
|  home.js
|  home.jade
|  home.css
|
|__sidebar
   | sidebar.js
   | sidebar.jade
   | sidebar.css
```

c) Développement

Suivre les instructions d'installation, puis utilisez:

```
npm run dev
```

III

Documentation d'installation

1) Le serveur

a. Prérequis

- git (https://git-scm.com)
- node-js (https://nodejs.com)
- mysql (https://mysql.com)

b. Installation du serveur

```
$> git clone https://github.com/Sevauk/storeit
$> cd storeit/src/server
$> npm install # for local installation/development
$> npm install -g # for global installation
```

c. Lancement du serveur

```
$> node main.js # local installation
$> storeit-srv # global installation
```

2) Application iOS

a. Installation de l'application

- Posséder un Mac afin d'installer la dernière version Xcode
- Installer Cocoapods Tutoriel sur le Site de Cocoapods ici
- Cloner le dépôt git
- Depuis le dépôt, installer les dépendances pod install
- Ouvrir le projet dans Xcode open StoreIt.xcworkspace
- Compilier le projet avant de lancer l'émulateur !

3) Application Android

a. Installation de l'application

L'application StoreIt sera disponible sur le Google Play Store. Tout utilisateur ayant un compte google et un appareil compatible pourra installer l'application via le Play Store.

4) Web-App

a. Installation de l'application

Installer Node.js version 5 ou plus et npm version 3 ou plus.

Une fois le repository cloné, lancez la commande suivante depuis src/web-app:

```
npm install
```

Vous pouvez lancer l'application avec:

```
npm start
```

Vous pouvez tester l'application avec:

```
npm test
```