RTP-API3 / Rattrapage API3

Modalités

Dépôt	https://rendu-git.@username
Fichiers requis	Tous les fichiers nécessaires au lancement et au déploiement de l'API
Correction	À distance
Durée	1 RUN
Taille de groupe	Seul

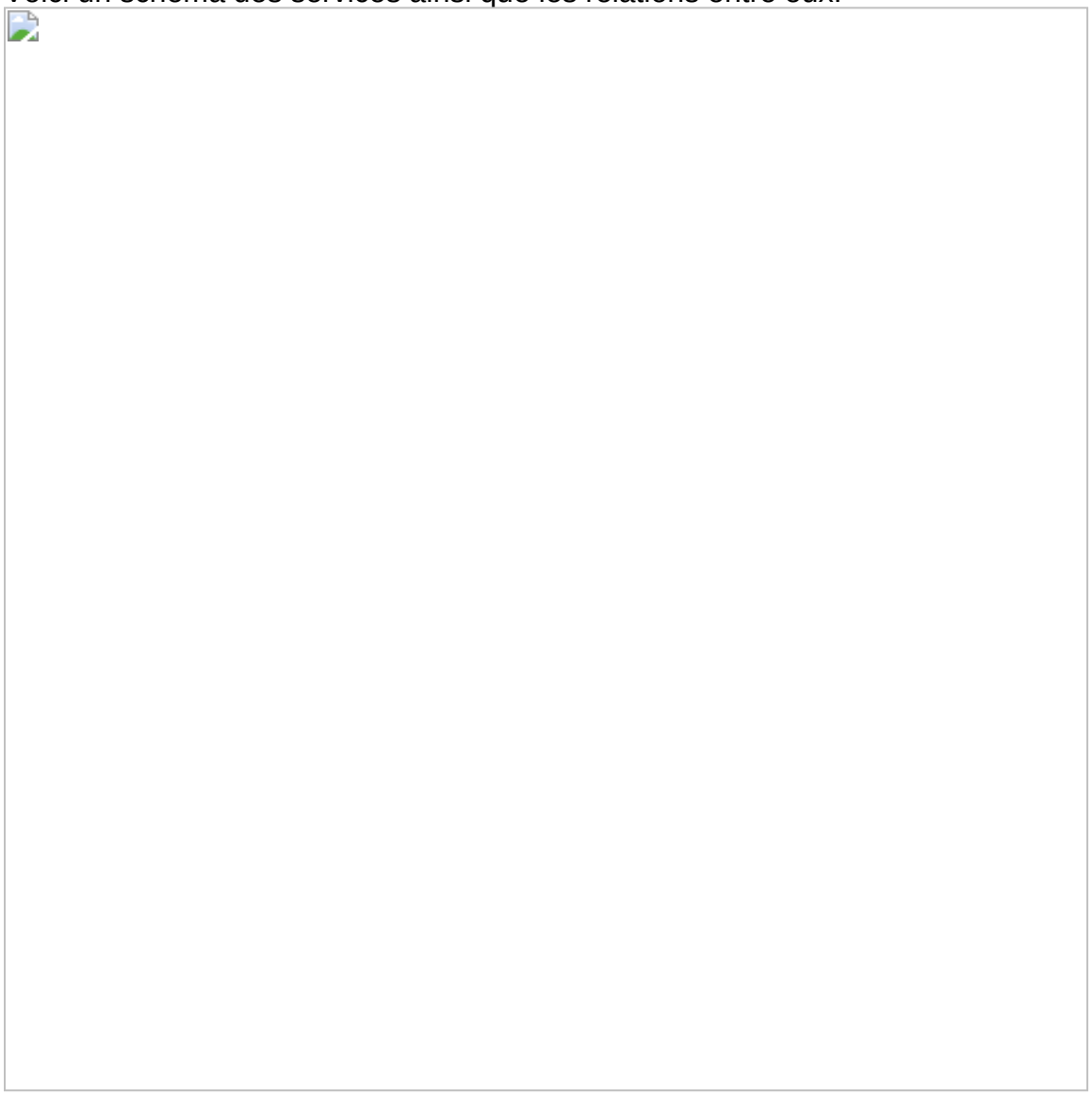
Objectifs

Notion	Description
Architecture	Création d'une architecture orientée micro service et la comparer avec des architectures déjà vues type MVC / Monolithique
Outils et langages	Analyse des besoins et choix des technologies en fonction des contraintes de fonctionnalités
Interconnexion	Faire communiquer un ou plusieurs services via des APIs
Déploiement / Configuration (Facultatif)	Mise en place des services via des outils de déploiement (Docker / Vagrant / Ansible / …)

Consignes

Dans ce projet vous devez mettre en place une application de sauvegarde de documents pour une entreprise.
Pour cela, vous allez développer 3 services orientés back-end qui vous donneront quelques fonctionnalités dont l'entreprise a besoin.
L'application aura un point d'entrée pour que des logiciels clients type CRM puissent se bind à lui.

Voici un schéma des services ainsi que les relations entre eux.



Service : Clients

Les clients qui vont utiliser notre application ne seront pas à développer.
Cependant, pour valider votre outil, vous pourrez utiliser les clients suivants :

- un pour votre service API : **Postman**
- l'autre pour votre service notification : **Amritb**

Service : Database

Vous pouvez utiliser la base de données de votre choix :

- mysql,
- postgresql,
- etc.

```
CREATE TABLE file (id INT AUTO_INCREMENT NOT NULL, user_id INT DEFAULT NULL, name VARCHAR(255) NOT NULL, path VARCHAR(
ALTER TABLE file ADD CONSTRAINT FK_8C9F3610A76ED395 FOREIGN KEY (user_id) REFERENCES user (id)
```

Service : API

Votre service API est le point central de votre application.
Vous avez deux grandes sections dans votre API :

- Gestion des utilisateurs
- Gestion des fichiers

Vous devez pouvoir créer / modifier un utilisateur et aussi vous connecter via un login / password.

La gestion des fichiers est plus complexe, vous devez créer / modifier / supprimer un fichier et une fois l'action exécutée, vous devrez envoyer une alerte vers votre service notification.

Service : Notifications

Le service notification permet d'alerter en temps réel de la modification d'un fichier.
Étant en temps réel, vous devrez mettre en place un socket qui lira différents événements :

- File_update / File_delete / File_create (évènement envoyé par l'API pour le transmettre au client)
- User_connect (événement envoyé par le client pour se connecter au service et recevoir les modifications des fichiers)
- Admin_connect (événement envoyé par le service backup pour se connecter au service et recevoir TOUTES les modifications des fichiers)

En plus de lire des événements, le service va émettre des événements :

- Notify_File_update / Notify_File_delete / Notify_File_create (événement envoyé aux personnes connectées pour les notifier du changement d'un fichier)

Service : Backup

Le service de backup est très simple, il devra se connecter et à chaque changement d'un fichier, il devra en faire une copie dans un répertoire de sauvegarde pour conserver un historique du fichier.
Le script doit donc tourner en continu.