Objectifs

- Comprendre la différence entre une architecture monolithique et microservices (voir cours d'introduction aux microservices).
- Faire communiquer des microservices via des APIs.
- Utilisation d'outils comme Docker ou Vagrant pour déployer des applications et des infrastructures.

Consignes

Lors du projet myAPI, vous avez mis en place une API pour la gestion des utilisateurs et des vidéos.

Un investisseur vous propose de développer un site similaire à YouTube contre la somme de $200 000.
Pour répondre à sa demande, vous devez développer des outils autour de votre API déjà existante.

Vous devez fournir 5 microservices décrits ci-après (l'API *myAPI* étant comptée dans les microservices).
Chaque microservice doit être dans un langage ou framework particulier, et aura un rôle spécifique.

Afin d'obtenir une solution finale fonctionnelle, vous devez connecter chacun de ces services une fois mis en place.
De plus, vos microservices doivent être scalables et leur fonctionnement doivent pouvoir s'adapter à leur niveau de charge.

Le schéma suivant présente les interactions entre les différents microservices.
Ce schéma est donné à titre d'exemple. Votre choix d'architecture est libre.

schéma_microservices

En complément des micro-services, vous devez également développer une application mobile native répondant aux mêmes attentes que le site principal.
MyAPI ne sera pas davantage développée dans ce sujet.

Site de diffusion

Le développement de ce service doit se faire avec Node.js en utilisant VueJS et Nuxt.

Vous devez mettre en place une application basée sur votre API (myAPI).

Le site devra avoir les fonctionnalités suivantes en utilisant votre API :

- la création de comptes,
- la connexion,
- la modification de comptes,
- l'upload d'une vidéo,
- le listage des vidéos,
- la recherche de vidéos,
- la consultation des détails d'une vidéo.

Pour ceux qui auraient pris de l'avance, vous pouvez implémenter ces fonctionnalités facultatives :

- l'ajout de commentaires,
- le listage des commentaires,
- la suppression d'une vidéo par l'utilisateur.

Conseils

Il n'est pas nécessaire de s'attarder sur l'UI (design).
L'utilisation d'outils comme *material-ui* ou autre est fortement recommandée pour rester concentré sur l'essentiel.

L'encodage

Vous devez développer une API dans un langage de votre choix.

Lorsqu'un utilisateur met une vidéo en ligne, elle doit être convertie dans un format et une définition spécifique.
Afin que l'encodage ne soit pas bloquant durant la navigation de l'utilisateur, nous souhaitons qu'il s'exécute en tâche de fond grâce à un microservice.

Pour encoder la vidéo, vous devez suivre les instructions suivantes :

- Le format utilisé doit être *mp4*.
- une copie de la vidéo doit également être faite dans toutes les définitions inférieures à la sienne, parmi 1080p, 720p, 480p, 360p et 240p.
  Par exemple, si votre vidéo d'origine est en 480p, vous ferez une copie en 360p et 240p.

Pour ce développement, vous avez plusieurs possibilités :

- un appel API à votre microservice quand l'utilisateur upload une vidéo, renvoyant instantanément une réponse pour ne pas bloquer l'encodage.
- un appel API à intervalle régulier pour récupérer la liste des vidéos et appliquer l'encodage adéquat.
- un watcher sur le répertoire où sont stockées les vidéos originales.

Une fois l'encodage terminé, un appel à l'API doit être fait pour mettre à jour les données de la vidéo.

Aide

Mot-clé : **FFMPEG**

Mailer

Certains événements du site nécessitent un envoi de mail à l'utilisateur : un changement de mot de passe, à la fin de l'encodage...

Vous devrez donc mettre en place un microservice sous la forme d'une API qui permettra d'envoyer des mails.

Le microservice prend pour seuls paramètres l'adresse email et le type de mail à envoyer.
Le mail doit être envoyé par `Postfix` (l'utilisation d'outils de mailing n'est pas autorisée).

Aide

Pour tester l'envoi du mail, vous pouvez utiliser **Maildev**.

Recherche

Vous devez mettre en place un moteur de recherche avec une recherche syntaxique.

Votre moteur de recherche doit se mettre à jour à chaque ajout, modification ou suppression de vidéo.

Lorsque l’utilisateur fait une recherche de vidéos, la requête passe par le moteur de recherche.

Différents moteurs de recherche existent, vous n'avez pas besoin d'en créer un.

Voici une liste non exhaustive de moteurs de recherche :

- Solr.
- Elasticsearch.
- Algolia (version gratuite).
- Sphinx search.

Application mobile

Vous devez mettre en place une application mobile native (iOS, Android ou cross-platform) dans un(e) des frameworks/technologies suivantes :

- **Android:** Java ok Kotlin
- **iOS:** Swift
- **Cross-platform:** ReactNative ou Flutter

L'application mobile devra fonctionner comme le site de diffusion et devra disposer des mêmes fonctionnalités :

- la création de comptes,
- la connexion,
- la modification de comptes,
- l'upload d'une vidéo,
- le listage des vidéos,
- la recherche de vidéos,
- la consultation des détails d'une vidéo.
- **En option:** l'ajout de commentaires,
- **En option:** le listage des commentaires,
- **En option:** la suppression d'une vidéo par l'utilisateur.