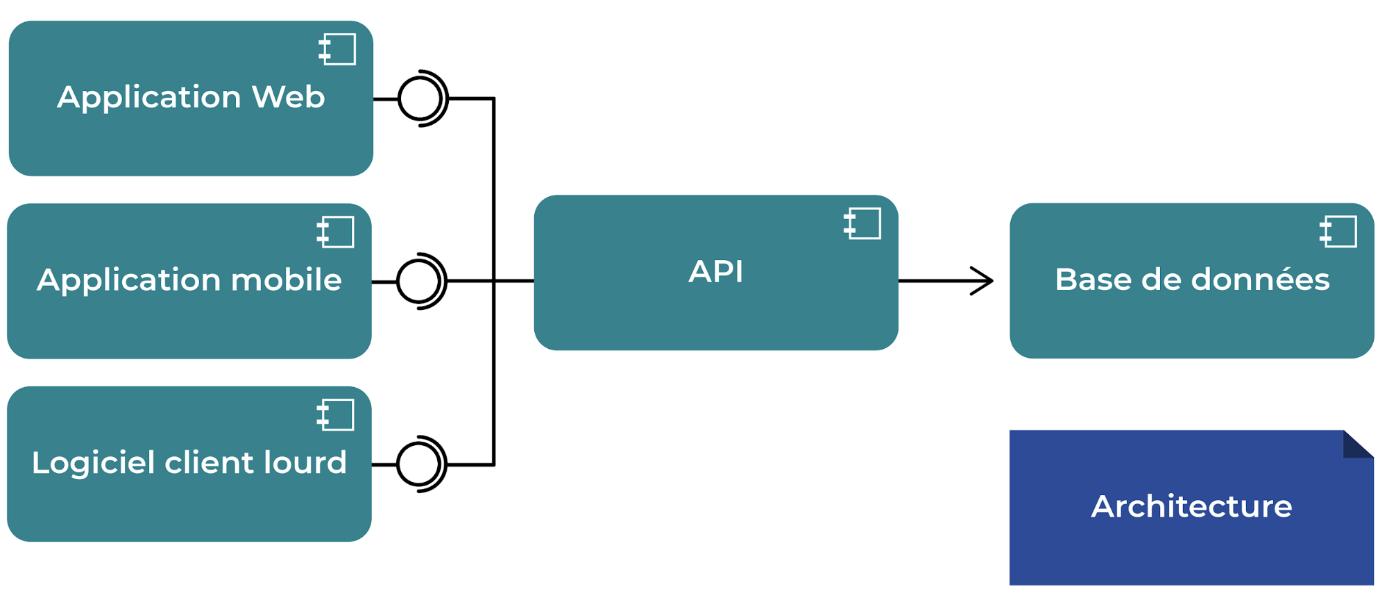
**Plongez-vous dans votre nouvelle mission**

Votre mission, si vous l’acceptez, est d’aider HR Association dans la création de sa suite d’outils de gestion des employés à destination des petites entreprises.

Voici l’architecture imaginée :

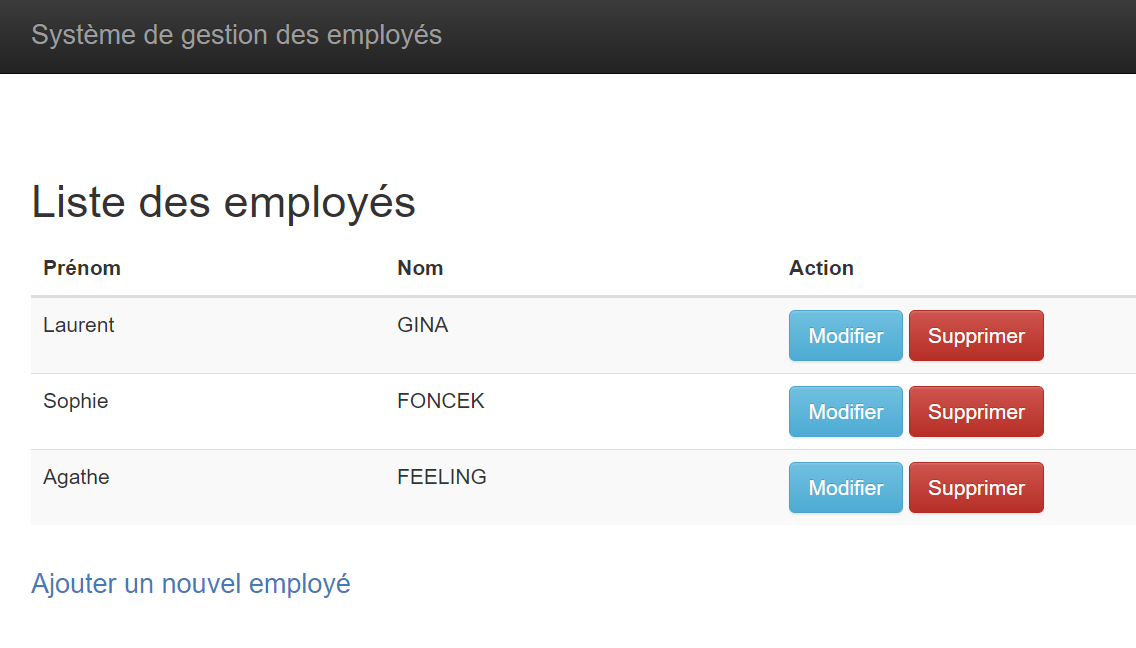
Architecture des composants logiciels de HR Association.

Lors de la partie 3 de ce cours, nous avons réalisé l’API qui sera exploitée par les différents clients.

Aujourd’hui, l’objectif est de développer une application web qui permettra de :

1. Visualiser les employés.
2. Ajouter un nouvel employé.
3. Modifier un employé.
4. Supprimer un employé.

Voici un visuel :

Page d’accueil du système de gestion des employés

Techniquement, nous allons évidemment utiliser **Spring Boot**, et je vous apprendrai comment renvoyer une page HTML et comment communiquer avec une API. Ce sera passionnant !

Allons-y ! 

**À vous de jouer !**



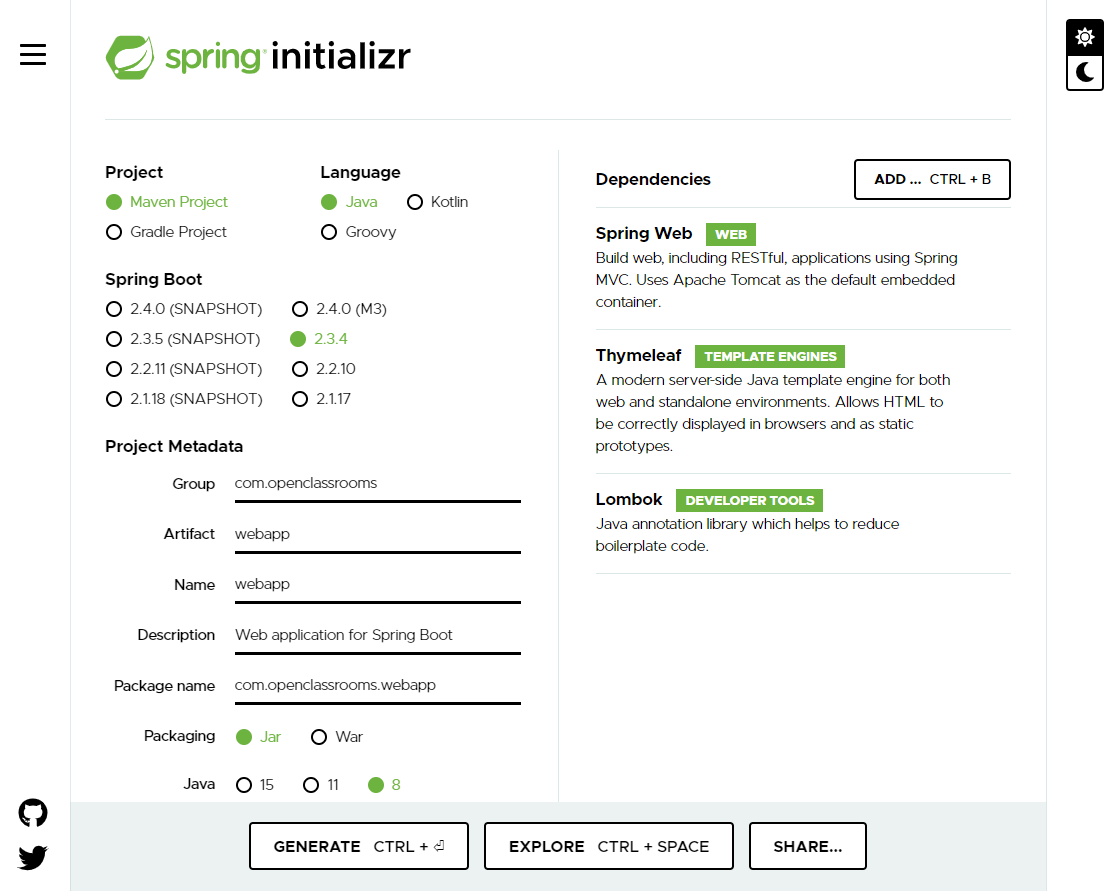
Je vous laisse réaliser la première étape, à savoir **créer la structure minimale** de ce projet. Utilisez **Spring Initializr** ou bien **STS**, le résultat sera le même !

L’important est que vous choisissiez bien vos **starters** en fonction de ce que vous souhaitez réaliser. Cela ne sera pas facile pour vous, mais en parcourant attentivement la liste des starters et en lisant leur description, vous devriez pouvoir vous approcher du bon résultat.

Dès que vous avez fini, venez voir la correction que je vous propose.

**Correction : identifiez les starters permettant de créer l’API**

Voici ma solution :

Formulaire du site Spring Initializr complété pour le projet

Pour les **“Project Metadata**”, je suis resté simple, vous avez toute latitude pour saisir les données de votre choix.

Pour les “**Dependencies**”, c’est ici que ça se complique !

Je pense que **Spring Web** a été une évidence pour vous ! Je vous félicite d’avoir mis le doigt dessus.  Sachez que cela ne nous permettra pas uniquement de fournir une page HTML à afficher. Ce starter contient également le code pour communiquer avec une API.

Lorsque vous avez parcouru les “**Template Engines**”, vous avez certainement noté les mentions du HTML, et cela vous a peut-être donné envie de choisir un de ces starters. Si c’est le cas, vous avez eu raison !

En plus de Spring Web, utilisons **Thymeleaf**qui est l’un des**moteurs de template** (template engine) les plus couramment utilisés.

En deux mots, un moteur de template HTML va nous**aider à formater la page HTML** que nous voulons renvoyer.

Mon dernier starter, Lombok, est une librairie qui permet d’optimiser certaines classes. Par exemple, grâce à Lombok, en ajoutant l’annotation @Data à une classe, je n’ai plus besoin d’écrire les getters et setters, magique !

**En résumé**

* Pour implémenter une application web qui communique avec une API, 2 éléments sont essentiels :
  + Le starter **Spring Web** qui permettra de renvoyer des pages HTML et communiquer avec l’API.
  + Un **moteur de template** comme Thymeleaf qui permet de formater le HTML.