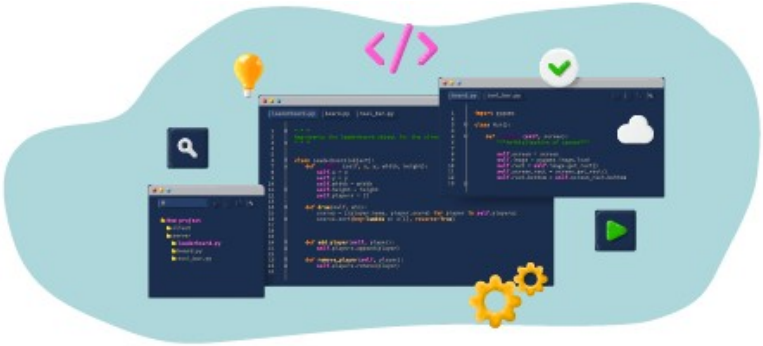
ORION



Justification des choix techniques

Projet MDD

Auteur : Frédéric Godard

Version 0.0.1

Aperçu / Synthèse....3
Choix techniques....4
Choix d’architecture Back....4
Choix de langage Back....5
La publication des services....5
Choix de framework sécurité....6
Le contrôle des données....6
La base de données....7
Le mapping des données (ORM)....7
Choix d’Architecture Front....8
Choix de bibliothèque de composant visuels....8
Gestion de version....9

Aperçu / Synthèse

L’application MDD sera développée en Java pour la partie Back et Angular pour la partie Front.

Le framework Spring permet de simplifier l’implémentation d’un application microservices et d’accélérer significativement les développements.

Les sources seront gérées avec git sur la plateforme GitHub.

Choix techniques

Différents choix s'imposent afin de répondre aux contraintes posées pour l'application MDD. Vous trouverez dans ce chapitre les motivation de ces choix.

Choix d'architecture Back

choix technique	lien vers le site / la documentation / une ressource	but du choix
Orientée services / MSA		*Définir le partage des données entre le back et le front*

Avantages : Fonctionne à 100 % sur internet avec des normes de sécurité standard et strictes. Peut donc servir facilement de nombreux clients.

Scalabilité
Les applications de type micro-services offre une meilleur scalabilté.
Cette architecture permet un meilleur respect des pratiques SOLID, en aidant à réaliser une bonne séparation des fonctionnalités.
On pourra facilement supprimer et ré-écrire un service si il est défaillant sans impact sur les autres fonctionnalités,

Inconvénient:
Légère dégradation des performances compensée par la possibilité de créer de multiples instances pour répondre aux besoins de la charge,

Choix de langage Back

choix technique	lien vers le site / la documentation / une ressource	but du choix
Java		

Pour la partie Back-End, l'équipe à porté son choix sur le langage Java. Java est l'un des premiers langages adapté aux serveurs d'application web. Souvent imité mais jamais égalé Comparé à .net Java à l'avantage d'être multi-plateformes ce qui permettra de déployer l'application sur des serveurs Linux, performants et moins onéreux que des machines Windows.

La publication des services

choix technique	lien vers le site / la documentation / une ressource	but du choix
spring-boot-starter-web	*https://spring.io/guides/gs/spring-boot*	*La publication des services web*

Globalement, la stack technique Spring à permis de simplifier le développement d'application Web Java.
L'implémentation de l'IOC de Spring à été plus rapidement au point que le framework natif EJB de Java et l'avènement de Spring Boot a permis de simplifier les problématiques DEVOPS par rapport aux anciens serveurs JEE.
Comme nous souhaitons développer une application de type MSA, avec un possible déploiement cloud pour sa tenue en charge, Spring Boot et son module de publication web est le choix adapté.

Choix de framework sécurité

choix technique	lien vers le site / la documentation / une ressource	but du choix
Spring-boot oauth2-resource-server	*https://docs.spring.io/spring-security/reference/servlet/oauth2/resource-server/index.html*	*la sécurisation*

Le module « resource server » de spring boot permet de mettre en œuvre rapidement un service d'authentification oAuth2.0 avec token JWT. Son utilisation permet de gagner beaucoup de temps de développement par rapport à une implémentation concurrente et fait partie de la stack spring boot choisie pour le développement de notre application.

Le contrôle des données

choix technique	lien vers le site / la documentation / une ressource	but du choix
spring-boot-starter-validation	*https://docs.spring.io/spring-framework/docs/4.1.x/spring-framework-reference/html/validation.html*	*Le contrôle de validité des données*

Parmi les tâches de développement back usuelle il y a le contrôle des données entrante permettant de vérifier les règles de gestion fonctionnelles ou encore le respect du format des données à injecter en base. Un framework de validation permet de gagner du temps sur des contrôles standards (champs obligatoires, tailles, types de saisie). Nous choisissons la bibliothèque validation de spring qui permet de rester homogène avec le reste de nos choix.

La base de données

choix technique	lien vers le site / la documentation / une ressource	but du choix
MySQL	*https://www.mysql.com/fr/*	*Le stockage des données*

MySQL à été choisi comme base de donnée.
Cette base de donnée, désormais gérée par Oracle, à fait ses preuves et est facile à mettre en œuvre.
Même si elle pourrait s'avérer limitée à terme pour sa gestion d'importantes volumétrie de données, elle est peu onéreuse et sera amplement suffisante dans le cadre du développement du MVP.
Notons que la gestion des accès et sauvegarde des données seront fait au travers d'un ORM (framework de mapping objet) ce qui permettra d'éviter une trop forte adhérence avec la base.
On pourra donc en changer si le succès de l'application le rends nécessaire.

Le mapping des données (ORM)

choix technique	lien vers le site / la documentation / une ressource	but du choix
Spring data jpa	*https://spring.io/guides/gs/accessing-data-jpa*	*L'accès aux donnés (ORM)*

L'implémentation Spring de JPA (Interface de persistance standard définie dans la JVM) est une évidence afin de respecter l'ecosystème Spring Boot mis en œuvre pour ce projet.

Choix d’Architecture Front

choix technique	lien vers le site / la documentation / une ressource	but du choix
Angular / SPA	*https://mobiskill.fr/blog/conseils-emploi-tech/quest-ce-quune-single-page-application/*	*L’interface utilisateur*

Le choix d’un framework SPA permet d’améliorer significativement l’expérience utilisateur en améliorant la fluidité de l’application.

Angular répond à ce critère et permet de développer rapidement des applications SPA qui sont « responsive ».

De plus Angular permet le développement d’application très structurée grâce à une approche orienté composant claire en se basant sur typescript au dessus de javascript et sur les styles scss. Ceci démarque nettement Angular de ses concurrents et contribue à une meilleure propreté de code.

Choix de bibliothèque de composant visuels

choix technique	lien vers le site / la documentation / une ressource	but du choix
Angular Material	*https://material.angular.io/*	*Accélérer le développement des pages.*

Un bibliothèque de composants visuels permet d’accélérer le développement des pages en offrant aux développeurs des composant enrichis comparés aux balises HTML Standards. Plusieurs bibliothèques existent et sont assez comparables.
Notre choix se porte sur Angular Material, car cette bibliothèque n’a pas grand-chose à envier à ses principaux concurrents comme Prime-NG, et ce choix permet d’éviter d’éventuel souci de compatibilité

Gestion de version

choix technique	lien vers le site / la documentation / une ressource	but du choix
Git / GitHub		*L'interface utilisateur*

Git a supplanté svn depuis déjà de nombreuses années concernant la gestion des version. Bien qu'un peu plus volumineux localement en terme d'espace disque, il offre des possibilités de duplication du projet avec l'ensemble de ses révisions facile d'utilisation. De plus il permet la création de branches de travail locales pour les développeurs et permet un meilleur travail collaboratif notamment grâce aux « merge request » ou « pull request ».

GitHub est un serveur git prêt à l'emploi disponible sur internet. Bien que pouvant poser des soucis de confidentialités GitHub donne la possibilité de repository privés ce qui semble une solution acceptable pour le développement d'un MVP.

L'installation d'un serveur local de type GitLab sera toujours possible dans un second temps grâce à la facilité de duplication de projet de git.