



**Marius Le Douarin**  
IT - Information Systems  
2024

**Francois Taiani**  
Professor-researcher

**Sopra Steria**  
Carrer de Casanova, n°2, 08011 Barcelona  
[info\\_es@soprasteria.com](mailto:info_es@soprasteria.com)

**Amal Hafi**  
Senior Information Systems Analyst

# SI Formation - Rapport de stage

Year 2023/2024

## **Remerciements**

Je souhaite débuter ce rapport en adressant mes plus sincères remerciements à toutes les personnes qui m'ont accompagné durant ces six mois de stage.

Tout d'abord, je tiens à exprimer ma profonde gratitude à Amal Hafi, qui a été bien plus qu'une simple tutrice de stage. Tout au long de cette expérience, elle s'est révélée être une véritable mentore. Grâce à ses conseils pertinents et avisés, j'ai pu monter en compétences très rapidement et progresser de manière continue.

Je tiens également à remercier Romain Duclos, dont l'écoute attentive et la disponibilité ont été précieuses pour répondre à mes interrogations. Il m'a également offert l'opportunité d'explorer des tâches transverses au projet, ce qui a considérablement enrichi mon bagage de connaissances.

Enfin, je tiens à exprimer ma reconnaissance à l'ensemble de l'équipe. Les développeurs, avec qui les échanges ont toujours été enrichissants, et qui se sont montrés prêts à partager leurs connaissances lors de nos points ensemble. De même, l'équipe des business analysts a toujours été disponible pour expliquer en détail les aspects fonctionnels du projet. Cette collaboration et cette bonne entente globale ont grandement contribué à rendre ce stage non seulement formateur, mais également agréable, me permettant de travailler sur un sujet captivant dans des conditions optimales.

## **Abstract (en)**

The goal of this internship was to develop an innovative solution to address the digitization gap in training management within the French Customs Administration.

Given the crucial importance of continuous training for customs officers, it was imperative to replace outdated paper-based processes with a tailored digital platform. This project aims to create a website that centralizes all information related to customs officer training.

This ambitious project emphasizes security and accessibility for all users. An earlier initiative, known as OPHELIE, had already been partially implemented. However, this initial effort, which dates back to the pre-Covid era, experienced significant delays and was ultimately abandoned by the customs administration.

Over time, the need for such a solution resurfaced, and the requirements evolved, rendering the OPHELIE project obsolete. In this context of renewal, Sopra Steria proposed to revive and complete the project, now under the name SI Formation.

## **Résumé (fr)**

L'objectif de ce stage était de développer une solution innovante afin de combler le déficit de numérisation dans la gestion des formations au sein des douanes françaises.

En effet, en raison de l'importance cruciale de la formation continue des douaniers, il était impératif de remplacer les processus papier au profit d'une plateforme numérique adaptée. Ce projet vise à créer une application web qui centralise toute l'information relative aux formations des douaniers.

Ce projet ambitieux met l'accent sur la sécurité et l'accessibilité pour tous les utilisateurs. Une initiative précédente, connue sous le nom OPHELIE, avait déjà été partiellement mise en place. Toutefois, cette demande initiale, remontant à la période pré-Covid, a vu son avancement considérablement ralenti et finalement abandonné par la douane.

Avec le temps, la nécessité d'une telle solution est réapparue, et les besoins ont évolué, rendant le projet OPHELIE obsolète. C'est dans ce contexte de renouveau que Sopra Steria a proposé de reprendre et de mener à bien ce projet, désormais sous le nom de SI Formation.

# **Sommaire**

I Introduction .....	5
II Sopra Steria .....	6
III Analyse .....	7
III.1 Contexte .....	7
III.2 Etat de l'art .....	7
III.3 Méthodologie .....	13
IV L'Application .....	21
IV.1 Solution proposée .....	21
IV.2 Implémentation .....	24
IV.3 Résultats .....	26
V Evaluation et Conclusion .....	29
VI Bibliographie .....	30
Bibliography .....	30
VII Annexe .....	32
VII.1 Glossaire .....	32

## I Introduction

Mon stage s'est inscrit dans le contexte des douanes françaises. Pour rappel, la douane est une administration de régulation des échanges étatique, chargée de faciliter et de sécuriser les flux de marchandises.

La douane c'est une administration en capacité de veille, de surveillance et d'intervention sur terre mais également sur le territoire maritime, avec la protection contre les pollutions marines et le contrôle des activités de pêche.

La douane s'inscrit dans un contexte mondial dans lequel nous pouvons constater un développement des flux commerciaux dans le cadre de la mondialisation des circuits économiques et des échanges de biens liés aux déplacements des voyageurs. Cette mondialisation s'est accompagnée d'une ouverture économique, culturelle et politique des différentes zones géographiques.

Cette mondialisation, à bien des égards très positives, comporte néanmoins des risques, et c'est pourquoi elle se doit d'être régulée avec une vigilance accrue.

Les points de vigilance se portent prioritairement sur :

- La diversité des partenaires économiques, ces échanges ont lieu entre pays présentant différents niveaux de développement, de protection du consommateur, de préoccupation environnementale et de régimes fiscaux.

Le développement des échanges peut engendrer une recrudescence des fraudes, la criminalité organisée renouvelle perpétuellement ses modes opératoires, augmentant ainsi le volume de produits prohibés dans certains trafics, comme celui des stupéfiants, la contrebande ou la contrefaçon, le trafic de cigarettes, le blanchiment d'argent, les trafics d'armes et munitions...

L'émergence de nouvelles menaces, mettant en péril la pérennité et la protection de notre environnement, des espèces et espaces naturels ainsi que de notre patrimoine culturel. Ces menaces se retrouvent sous la forme de trafic illégal d'espèces animales et végétales menacées d'extinction (et représente par la même occasion la deuxième cause de disparition de celles-ci), de transferts de déchets illégaux (hospitaliers, chimiques, métaux lourds). Le patrimoine culturel, quant à lui, est soumis à des réglementations relatives à la protection d'objets d'art, de collection ou d'antiquités.

En clair, la douane fait face à de nombreuses menaces qui nécessitent une supervision rigoureuse pour stopper les activités illégales. Ces menaces, souvent difficiles à prévoir, doivent être anticipées au mieux. Pour ce faire, il est important que les douaniers soient correctement formés, préparés à n'importe quel scénario, informés sur les différents types de trafics, et prêts à appliquer des protocoles de vérification stricts et rigoureux.

## II Sopra Steria

Pour passer à la présentation du groupe au sein duquel j'ai pu évoluer, Sopra Steria est une entreprise de services du numérique, autrement dit une ESN, française et une société de conseil en transformation numérique des entreprises et des organisations. Elle est née de la fusion de la SOciété de PRogrammation et d'Analyses (Sopra) et la société Steria (Société d'étude et de réalisation en informatique et automatisme).

Sopra Steria est un leader européen de la transformation numérique et figure parmi les cinq principaux acteurs européens du secteur des Entreprises de Services Numériques (ESN), avec 56 000 collaborateurs répartis dans 30 pays, majoritairement en France (48%), au Royaume-Uni (19%), en Europe (31%) sinon dans le reste du monde (2%). L'entreprise intervient dans divers domaines, notamment les services financiers, le secteur public, les télécommunications, les médias, le divertissement, l'aéronautique et spatial, la défense, la sécurité, l'énergie, les services publics, les transports, la distribution, et bien d'autres. [1]

Ainsi Sopra Steria se distingue comme un acteur majeur sur la scène internationale, dont les entités peuvent être amenées à collaborer étroitement sur des projets de grande envergure. Cette dynamique collaborative s'est faite ressentir au sein de l'équipe où j'ai eu la chance d'évoluer. L'équipe était scindée en deux, géographiquement parlant : une partie était basée à Nantes, tandis que l'autre, à laquelle j'appartenais, se trouvait à Barcelone.

Chez Sopra Steria, l'organisation des équipes et des projets suit une structure en arborescence bien définie. À la base de cette structure, nous trouvons plusieurs pôles d'expertise, chacun spécialisé dans un domaine particulier. J'ai eu l'opportunité de travailler dans le pôle Douane, un domaine clé au sein de l'entreprise. Ce pôle est lui-même subdivisé en plusieurs sphères, chacune regroupant différents projets et applications spécifiques. C'est dans la sphère Fiscalité que le projet SI Formation, auquel j'ai participé, s'inscrit.

## III Analyse

### III.1 Contexte

Le projet s'intitule SI Formation. Il a pour objectif de fournir une alternative au format papier pour les agents de la douane française, leur permettant ainsi de mettre en place de nouvelles formations plus facilement. Comme mentionné dans l'introduction, le domaine des douanes est très complexe, nécessitant une connaissance approfondie de nombreux protocoles pour pouvoir réagir efficacement à toute situation.

Le besoin de passer au format numérique pour faciliter la gestion des équipes et l'organisation des formations a refait surface cette année, bien qu'il ait déjà été identifié il y a quelque temps. En effet, un premier projet, du nom de CLAF, était déjà existant mais sur des technologies très anciennes et trop difficile à maintenir. Une première tentative de succession, nommée OPHELIE, avait vu le jour avant 2020. Pour OPHELIE, des équipes de développement avaient déjà été mobilisées et le projet avait bel et bien débuté. Cependant, l'arrivée de la pandémie de Covid-19, qui a contraint les individus à rester confinés, a eu un impact sans précédent sur l'activité des entreprises françaises, provoquant un ralentissement global de la production. C'est pourquoi le projet OPHELIE a rapidement été suspendu jusqu'à la fin de cette période.

À la reprise des activités normales, le projet a pu redémarrer, mais une nouvelle contrainte est apparue : les besoins des clients avaient radicalement changé entre-temps. Il a donc été nécessaire de revoir entièrement le cahier des charges, ce qui a entraîné l'abandon du projet pour éviter des complications supplémentaires.

C'est de là qu'est né SI Formation, une version différente d'OPHELIE répondant aux nouveaux besoins exprimés.

### III.2 Etat de l'art

Maintenant que les bases du problème ont été posées, un premier travail de réflexion a été mené pour réussir la mission. Les technologies liées au développement Web (*“tout le processus de création de sites internet ou d'applications”* [2]) ne manquent pas de nos jours. C'est un domaine en constante évolution, et désormais, pour garantir rapidité et qualité, les développeurs n'ont d'autre choix que de se tourner vers les frameworks. Frame signifie cadre, et work travail ; les frameworks sont de véritables boîtes à outils et constituent une méthodologie rigoureuse pour les développeurs. En effet, ces outils sont généralement une bonne réponse aux différents points critiques d'une phase de développement : arborescence, normes, sécurité, et CRUD (l'acronyme pour Create, Read, Update, Delete, les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données [3]). Les frameworks sont des bibliothèques de développement logiciel qui offrent une structure préconstruite, avec des composants autonomes prêts à être réutilisés. Les frameworks modernes sont souvent open-source, ce qui signifie qu'ils sont maintenus par une communauté de développeurs passionnés et spécialisés, qui travaillent ensemble pour améliorer le code et ajouter continuellement de nouvelles fonctionnalités afin de s'adapter aux nouveaux besoins des développeurs. [4]

En définitive, les points à retenir pour souligner l'importance d'utiliser des frameworks dans des projets de grande envergure sont les suivants :

- Rapidité : une base de travail existe déjà, un bon développeur est un développeur qui ne s'ajoute pas du travail inutile et qui sait utiliser des outils existants à son avantage.
- Architecture : les frameworks sont optimisés pour garantir un code propre et fonctionnel, ce qui évite de ralentir le fonctionnement d'une application.
- Productivité : sur un projet impliquant une grande équipe, les frameworks permettent de disposer d'une base de développement commune, qui définit des stratégies de développement auxquelles chacun doit adhérer. Acquérir des compétences sur ce type de technologies permet de changer de projet plus facilement sans avoir à se former de nouveau. En théorie, il ne resterait plus que la logique métier à assimiler.
- Communauté : cet aspect, mentionné plus tôt, est essentiel. Travailler avec de tels outils permet de bénéficier du soutien d'une vaste communauté, qui pourra répondre aux questions via des supports et des forums, aidant ainsi à corriger des bugs ou à résoudre des problèmes de programmation plus ou moins connus. Cela constitue une nouvelle fois un gain de temps non négligeable. [4]

Une enquête de Stack Overflow, le forum d'échange le plus connu dans le monde de la programmation, a révélé que plus de 55% des développeurs utilisent un framework dans leur travail quotidien. Ce pourcentage peut s'expliquer par le fait que certains considèrent les frameworks comme un frein à la créativité. De plus, les entreprises ont tendance à développer leurs propres outils internes privés. [5]

Pour rappel, un projet web s'articule autour de deux sections distinctes mais interdépendantes : le frontend et le backend.

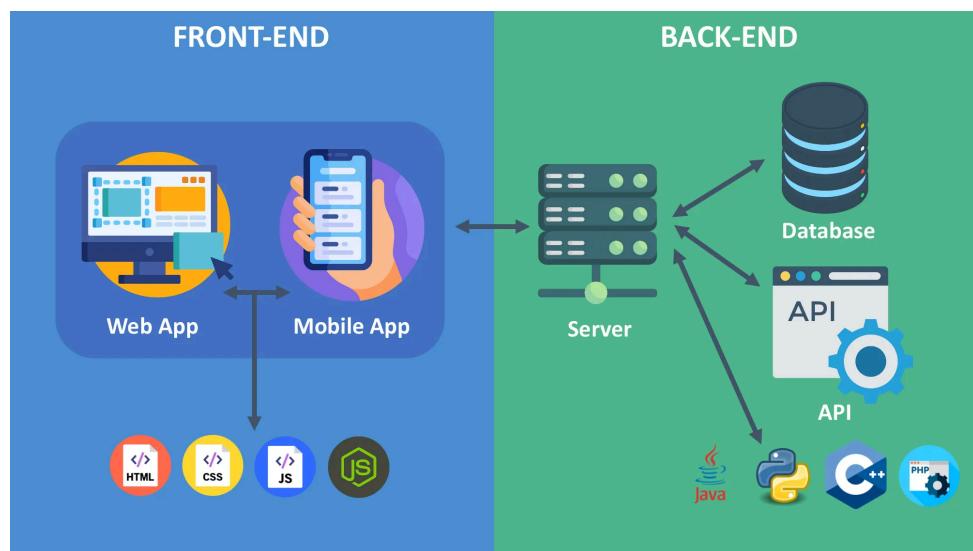


Figure 1: Architecture d'une application web [6].

Le terme frontend fait référence à la partie visible d'une application, celle que les utilisateurs manipulent directement, comme les menus, les boutons et les éléments visuels de la page.

De l'autre côté, nous avons la partie serveur. Le backend d'une application écoute les instructions et effectue différents types de traitement. Lorsque votre utilisateur interagit avec le

frontend, l'interaction envoie une demande au backend au format HTTP. Le backend traite la demande et renvoie une réponse. Ces éléments avec lesquels nous communiquons sont des :

- Serveurs de base de données pour récupérer ou modifier des données.
- Microservices qui exécutent un sous-ensemble des tâches spécifiques requêtées.
- API tierces pour exécuter des fonctions annexes.

Parmi les frameworks frontend les plus populaires [7], nous trouvons :

- React : Développé par Facebook, React est un framework open-source axé sur la création d'interfaces utilisateur dynamiques et réactives grâce à une architecture basée sur les composants. Son approche « déclarative » simplifie le développement, surtout pour les interfaces complexes. Il est aussi largement utilisé dans le développement mobile avec React Native, permettant ainsi une forte réutilisation de code entre les plateformes web et mobile.
- Vue.js : Vue est un framework JavaScript progressif, conçu pour être intégré progressivement dans un projet. Sa simplicité et sa réactivité en font un choix populaire, surtout pour les applications monopages. Vue se distingue par son (*“The Document Object Model (DOM) is a programming API for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated”* [8]) virtuel et sa syntaxe simple, facilitant la courbe d'apprentissage. Il est idéal pour les projets où la modularité et la performance sont essentielles.
- Angular : Angular, développé par Google, est un framework complet et robuste, principalement écrit en TypeScript. Il se distingue par sa structure organisée et son data binding bidirectionnel, qui synchronise automatiquement les données entre le modèle et la vue. Angular offre également des outils puissants comme Angular CLI pour le scaffolding et le testing, ainsi qu'une gestion avancée des dépendances via l'injection de dépendances. Ces caractéristiques en font un choix privilégié pour les applications d'entreprise nécessitant une grande échelle et une architecture maintenable.
- Node.js : Bien que Node.js soit généralement considéré comme un environnement d'exécution backend, il peut être utilisé côté front-end pour des tâches spécifiques. Sa capacité à gérer plusieurs requêtes simultanément grâce à une architecture orientée événements le rend adapté aux applications nécessitant des performances élevées et une forte scalabilité.

Pourquoi Angular pour ce projet ? Le choix d'Angular pour ce projet s'explique par plusieurs facteurs spécifiques aux besoins de l'application :

- Complexité et échelle : Le projet nécessite la gestion de nombreuses fonctionnalités complexes et une interaction intensive avec le backend. Angular, avec sa structure modulaire et ses outils intégrés, permet de développer une architecture robuste et maintenable, en garantissant une séparation claire des préoccupations.
- TypeScript et maintenabilité : Angular est conçu autour de TypeScript, un superset de JavaScript qui ajoute des fonctionnalités de typage statique. Cela améliore la maintenabilité du code, facilite la détection d'erreurs en amont et rend le développement plus structuré, ce qui est essentiel pour les grandes équipes ou les projets de longue durée.

- Data Binding Bidirectionnel : Le data binding bidirectionnel d'Angular simplifie la gestion des données côté frontend, particulièrement utile dans les applications où les données évoluent fréquemment et nécessitent une synchronisation en temps réel avec l'interface utilisateur.
- Outils de développement : Angular CLI accélère le processus de développement en offrant des commandes pour générer du code (module, pipe, resolver, service, guard, interface, component...), exécuter des tests et déployer l'application, ce qui a été particulièrement bénéfique pour le projet.
- Tests et fiabilité : Angular est réputé pour ses capacités de test intégrées, permettant de construire des applications fiables. Pour un projet où la fiabilité est critique comme le nôtre (et éviter des scénarios flaky), ces fonctionnalités sont essentielles.

Ainsi, bien que d'autres frameworks comme React ou Vue présentent des avantages certains, Angular s'est imposé comme le choix le plus adapté pour répondre aux exigences spécifiques de ce projet en termes de complexité, de performance et de maintenabilité.

En backend, le choix du framework est crucial pour déterminer la manière dont une application va traiter les requêtes, gérer les bases de données, et interagir avec les services tiers. Parmi les frameworks backend populaires, nous retrouvons :

- Django : Django est un framework web open-source écrit en Python, réputé pour sa simplicité et ses fonctionnalités intégrées. Il propose un ORM (Object-Relational Mapping) performant, une interface d'administration auto-générée, et suit l'architecture Model-Template-View (MTV). Ses capacités de sécurité et son système d'authentification robustes en font un choix privilégié pour des plateformes complexes comme Instagram.
- Ruby on Rails : Également connu sous le nom de Rails, ce framework open-source en Ruby suit l'architecture Model-View-Controller (MVC). Il est apprécié pour sa philosophie "Convention over Configuration", qui simplifie le développement en réduisant la nécessité de prendre des décisions sur la structure du code. Rails est utilisé par des géants comme GitHub et Airbnb, en raison de sa capacité à accélérer le développement grâce à des conventions préétablies et à son système de mappage objet-relationnel, Active Record.

De manière générale, il existe un large éventail de framework. En 2023 Statista, un site de statistiques, a dressé un histogramme des frameworks les plus utilisés dans le web.

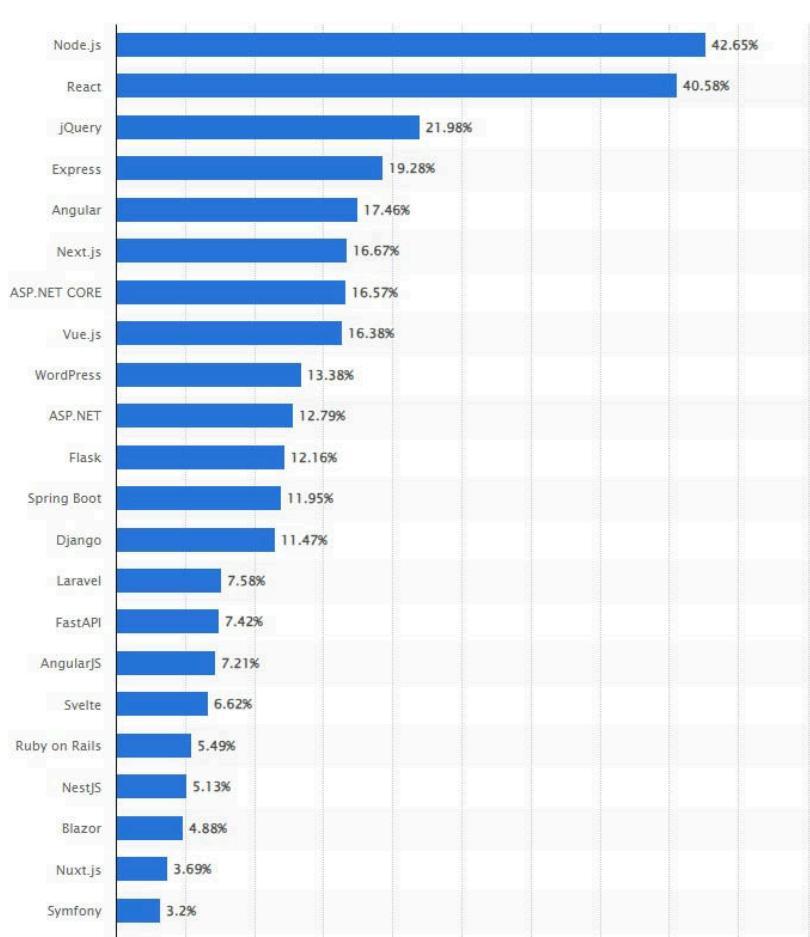


Figure 2: Most used web frameworks among developers worldwide, as of 2024 [9].

Il est bon de noter qu'un framework en bas dans ce graphique n'est pas nécessairement un mauvais framework, chaque framework a sa spécialité. Par exemple Symfony est toujours un framework très utilisé pour du PHP, ou bien Svelte en JavaScript qui fait abstraction du Document Object Model [8], une structure logique virtuelle sous forme d'arbre qui permet de définir la façon dont une page va être accédée et manipulée, permettant d'atteindre de hautes performances dans des cas d'usage. Un framework est fait pour répondre à un besoin, sa facilité d'apprentissage, d'intégration, ses fonctionnalités, sa communauté active, sa légereté, ses mises à jour sont les principaux critères qui influencent sa popularité. La réticence à changer de framework peut également provenir de la contrainte de devoir se former à nouveau.

### Spring VS Spring Boot [10]

De notre côté, le choix a été fait de s'orienter vers Spring Boot pour la gestion du projet côté backend. Avant de mentionner le nom de Spring Boot, nous devons faire un bon en arrière avec la naissance de Spring en 2002.

Spring c'est avant tout un framework open-source léger qui permet aux développeurs Java EE 7 de créer des applications d'entreprise simples, fiables et évolutives.

Java est un langage de programmation parmi les plus répandus. Multi-applicatifs, sa côte est due à ses nombreuses forces [11] :

- Langage multithread, permettant d'exécuter plusieurs tâches en parallèle pour un gain de temps de réponse, de meilleures performances et des coûts de maintenance réduits dans les applications.
- Gestion de la mémoire : bien que ce point soit controversé dans une partie de la communauté, la gestion de la mémoire reste une tâche complexe pour les développeurs ; elle est pourtant clé pour la performance. Java en version vanilla s'en occupe automatiquement. Lors des traitements, il est nécessaire de créer des objets, des entités virtuelles, qui donnent forme à nos données. Une fois qu'un objet devient obsolète, il est considéré comme un déchet que Java envoie sur un "tas" plus ou moins extensible, permettant de libérer de l'espace pour de futurs objets.
- Évolutivité : si les développeurs souhaitent faire évoluer une application verticalement ou horizontalement, Java le facilite grâce à sa communauté active depuis des dizaines d'années.
- Développement multiplateforme : les modifications nécessaires pour adapter une application d'un système d'exploitation à un autre sont mineures, ce qui réduit les efforts de développement.
- Son aspect sécurité est également largement mis en avant avec ses fonctionnalités telles que l'isolation de type sandbox, la cryptographie, la gestion des exceptions et le contrôle d'accès.

Pour revenir à Spring, il se distingue par sa capacité à gérer divers objets métiers, rendant le développement d'applications web plus facile par rapport aux frameworks et API Java classiques, comme JDBC, JSP et Java Servlet. Spring utilise des techniques modernes telles que la programmation orientée aspect (qui sépare le code technique du code métier), les POJO (Plain Old Java Object, les objets Java classiques sans surcouche par un framework) et l'injection de dépendances pour le développement d'applications d'entreprise. Pour être plus précis, Spring n'est pas un framework à part entière mais une collection de frameworks, comprenant : Spring AOP, Spring ORM, Spring Web Flow et Spring Web MVC. En tant que développeur, il est possible de choisir les modules que nous souhaitons employer, car ils sont indépendants les uns des autres.

Les présentations du petit frère faites, voici le grand frère : Spring Boot.

Spring Boot, basé sur le framework Spring classique, simplifie son utilisation tout en offrant toutes ses fonctionnalités. Ce framework orienté microservices permet de créer rapidement des applications prêtes pour la production grâce à une configuration automatique. Il suffit d'utiliser les bonnes configurations pour bénéficier de fonctionnalités spécifiques. Spring Boot est particulièrement efficace pour le développement d'API REST.

Les API REST sont des API, un contrat entre un utilisateur d'informations (émet une requête) et un fournisseur d'informations (répond à la requête), qui respect les principes REST comme [12] :

- Absence d'état : Les APIs ne stockent pas l'état de la session côté serveur.
- Mise en cache : Les réponses peuvent être mises en cache, éliminant certaines interactions client-serveur.

La question qui subsiste est : Pourquoi choisir Spring Boot plutôt que Spring (qui a longuement été utilisé et l'est toujours) ?

Bien que Spring résolve de nombreux problèmes, nous arrivons à un point où les applications tendent à évoluer vers des architectures de microservices, c'est-à-dire la séparation d'une application en plusieurs petits services web autonomes avec leurs propres fonctionnalités. Ces microservices nécessitent des outils spécifiques pour être développés rapidement. Les applications Spring traditionnelles demandent beaucoup de configurations, qu'elles soient en XML, en Java ou par annotations, ce qui ralentit le développement. Par exemple, pour utiliser un des modules de Spring, comme Spring MVC, il faut configurer plusieurs éléments tels que l'annotation ComponentScan, le servlet Dispatcher, le résolveur de vues et les web jars, tout ce qui est nécessaire à la bonne exécution du module. Spring Boot simplifie ce processus avec l'autoconfiguration, qui examine les frameworks disponibles et les configurations fournies par les développeurs ou déjà présentes.

Par exemple, s'il trouve Hibernate dans le classpath (un paramètre donné à la machine virtuelle Java pour indiquer où se trouvent les classes et packages afin de les exécuter), il va consulter la configuration fournie ou celle déjà en place dans l'application et configurer automatiquement la source de données et la base de données en mémoire, ainsi que le servlet Dispatcher (l'élément qui délègue les requêtes HTTP). Grâce à Spring Boot, un projet de démarrage est créé avec toutes les configurations XML et les dépendances par défaut, facilitant ainsi grandement le développement.

En conclusion, bien que d'autres frameworks comme Django ou Rails possèdent leurs propres avantages, Spring Boot s'est imposé comme le choix idéal pour notre projet en raison de sa compatibilité avec l'écosystème Java, de sa capacité à simplifier le développement de microservices, de ses performances, de sa gestion de la sécurité, et de sa robustesse éprouvée dans les environnements d'entreprise.

### III.3 Méthodologie

Pour mener à bien un tel projet en gérant une équipe divisée en deux, il est impératif de mettre en place des processus efficaces, et c'est ce à quoi Sopra Steria accorde une grande importance.

Ainsi, une semaine au sein de l'équipe SI Formation se déroule de la manière suivante : tous les jours, à l'exception du mercredi, une réunion d'équipe est prévue et appelée V0, ou plus communément les dailies. Le mercredi, cette réunion est remplacée par le V1, également connu sous le nom de weeklies.

Les réunions V0 ont un objectif très simple : faire le point sur les tâches effectuées la veille et celles prévues pour la journée à venir. En plus de permettre à chacun de prendre du recul et d'obtenir une vue d'ensemble sur l'avancement du projet, ces réunions sont également cruciales pour les responsables du suivi du projet. Elles permettent de détecter les tâches qui prennent plus de temps que prévu ou les points critiques nécessitant une attention particulière, qui seront alors traités en dehors des réunions V0.

Lors de la réalisation de leurs tâches, les développeurs doivent s'assigner des tickets sur la plateforme Jira. Cette dernière "fournit des outils de planification et de suivi permettant aux équipes de gérer les dépendances, les exigences fonctionnelles et les parties prenantes dès le démarrage du projet" [13]. Les responsables techniques (RT) et le chef de projet disposent de tableaux de bord pour suivre en permanence ces tickets, connaître leur attribution et leur avancement.

Les tickets sont généralement organisés avec une description comprenant un résumé de la tâche et les écrans à développer, avec une référence vers les spécifications. Les spécifications, rédigées par les business analysts (BAs), résultent des ateliers réalisés avec le client. Elles détaillent les besoins du client, les différents écrans à développer avec des maquettes, la description de ces maquettes, ainsi que des règles de gestion spécifiques à respecter. Ces règles peuvent concerner la gestion des accès des utilisateurs à certaines parties de l'application, l'accessibilité d'éléments de la page, le résultat des interactions avec ces éléments, etc.

En ce qui concerne les tickets, ils comprennent également des sous-tickets, plus spécifiques que le ticket principal. Ces sous-tickets permettent de déterminer l'ordre de réalisation des tâches ainsi que le temps alloué à chacune d'elles.

Ce temps est crucial et est discuté lors des réunions V0. Après avoir détaillé les tickets qui leur sont assignés, les développeurs doivent également mettre à jour leur RAE (Reste À Effectuer) afin qu'il soit reporté sur le fichier de suivi. Le RAE représente l'estimation du temps restant pour terminer une tâche. Lorsqu'une tâche commence, une estimation de temps est inscrite, basée sur le chiffrage effectué par les RT lors de la mise en place du devis.

Le temps estimé est indiqué sur tous les tickets, et le temps restant est affiché en dessous. Initialement, à la création de la tâche, le temps estimé est égal au temps restant. Ce temps est celui que le développeur doit suivre en permanence. Chaque jour, les développeurs doivent enregistrer huit heures sur leurs tâches en cours, puis comparer ces heures au temps restant. Si le développement est loin d'être terminé, le temps restant est réévalué à la hausse. À l'inverse, si la tâche prend moins de temps que prévu, ce temps est ajusté à la baisse.

En ce qui concerne les réunions V1, l'accent est mis sur l'avancement global du projet. Chaque partie (BA et développeur) fait le point sur ses propres progrès. Les BAs font souvent des retours sur divers sujets, tels que la livraison des spécifications pour la prochaine phase ou les retours des ateliers. Les développeurs, quant à eux, se concentrent sur la cohérence de l'avancement du développement par rapport aux délais. Ils mettent notamment en avant le burndown, une pratique courante dans les démarches agiles. Il s'agit d'un graphique montrant la quantité de travail effectuée et la quantité de travail restante au cours d'une période définie. Ce graphique est mis à jour quotidiennement pour estimer si le travail peut être réalisé dans les délais impartis. Un tableau indiquant la qualité du code, la couverture des tests du code source, et le taux de duplication du code (à maintenir aussi bas que possible) est également présenté.

À la fin des V1, une phase d'humeur de la semaine démarre. Un nouveau tour de table a lieu, où chacun énumère un point positif et un point négatif de la semaine écoulée. Les participants peuvent exprimer leurs inquiétudes ou partager de bonnes nouvelles. La réunion se conclut par un jeu ludique permettant de mieux connaître les différentes personnalités de l'équipe.

Les phases pré-livraison sont cruciales pour le projet, et les réunions V0 sont souvent privilégiées par rapport aux V1. Nous privilégions le suivi rapproché pour nous assurer que les objectifs de la phase sont réalisables. Parfois, le temps manque, auquel cas les objectifs à court terme sont réévalués, et la livraison se fait en deux étapes : une première livraison avec réserves, où nous détaillons au client ce qui n'a pas pu être réalisé, suivie d'une seconde livraison avec les éléments manquants.

Le V1 post-livraison est particulier car il concerne la préparation de la prochaine phase, avec l'exposé des futures stratégies.



Figure 3: V1 équipe de fin de phase 2 / début phase 3.

Durant ces réunions, le planning est affiché, avec la prise en compte des versions correctives de chaque phase :

## SI Formation – Plan de charges

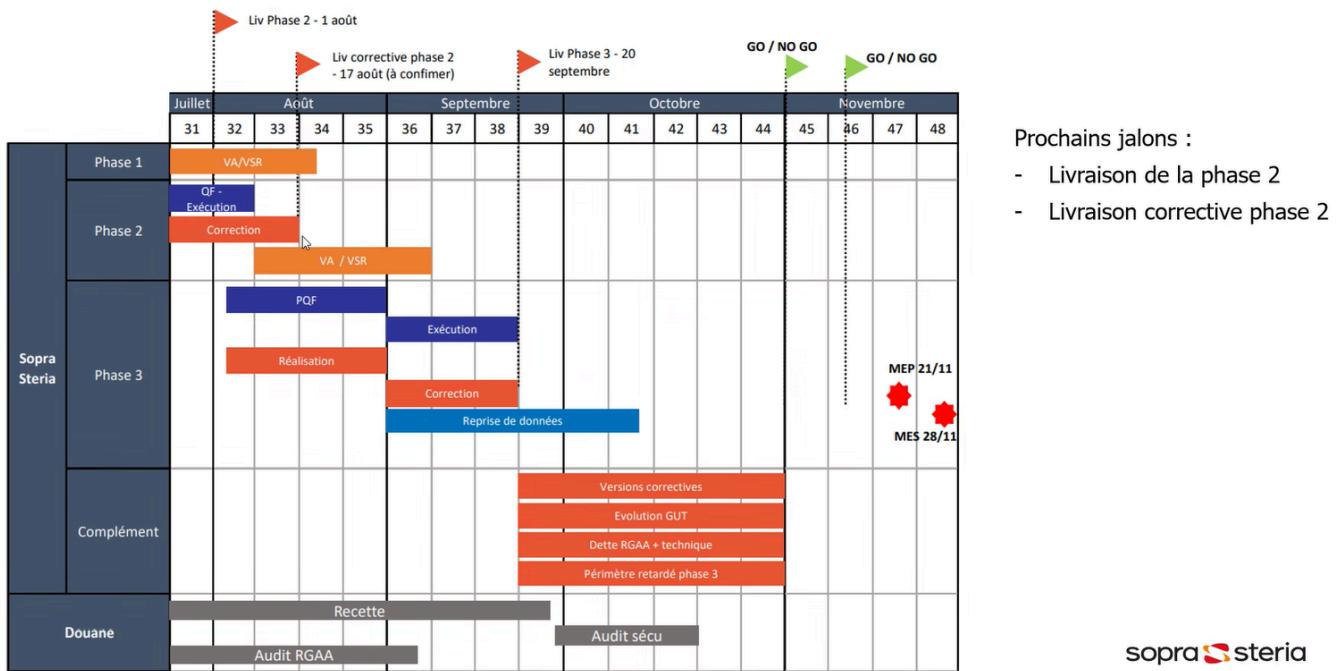


Figure 4: Plan de charges.

SI Formation est un projet relativement long mais a été planifié sur une période plutôt courte. Tout retard accumulé dans une phase se répercute sur les suivantes. C'est pourquoi, lorsque nous sommes censés débuter une nouvelle phase, l'équipe de développeurs est divisée : cer-

tains continuent de corriger la version de la phase précédente, tandis que d'autres commencent la phase actuelle dans les délais impartis. Ces délais exigeants ont notamment poussé l'équipe à faire plusieurs heures supplémentaires pour respecter les échéances de la phase 1. L'avantage de cette première phase est qu'elle nous a permis d'apprendre beaucoup sur le plan organisationnel, ce qui nous a aidés à éviter les mêmes erreurs dans les phases suivantes.

Ces ajustements ont eu un impact significatif sur les développeurs, en mettant l'accent sur le suivi des processus. L'objectif était de renforcer le temps consacré à la conception et à la validation de chaque tâche. Les processus suivent la trame suivante :

## Workflow tâches de dev



Figure 5: Workflow tâches de développeur - Partie 1

## Workflow tâches de dev - suite



Figure 6: Workflow tâches de développeur - Partie 2

Pour détailler les différentes étapes :

- Step 1 : Cette étape a été décrite plus tôt. Il s'agit pour le développeur de prendre connaissance de sa tâche à réaliser, en comprenant les tenants et les aboutissants.
- Step 2 : Cette étape cruciale, appelée la conception fonctionnelle, consiste à vérifier ce que nous avons réellement compris de notre tâche en consultant les spécifications. Si des erreurs de compréhension apparaissent, il appartient au BA de les corriger.
- Step 3 : Cette étape, appelée la conception technique, va de pair avec la conception fonctionnelle. L'objectif ici est de discuter avec un RT pour déterminer si la tâche est réalisable avec les ressources disponibles à ce moment-là. Les RTs ayant une vue d'ensemble sur les tâches en cours et celles déjà réalisées, leur consultation permet d'éviter le développement d'éléments en doublon, ce qui éviterait de perdre du temps inutilement.

Cette étape a malheureusement été souvent négligée pendant les phases 1 et 2 en raison du manque de temps. Cela a conduit à quelques problèmes techniques par la suite, ce qui nous a poussés à renforcer cette étape lors de la phase 3.

Step 4 : Il s'agit de la phase de réalisation de la tâche. À cette étape, il est crucial de procéder à une réévaluation continue du temps imparti pour la tâche.

- Step 5 : Utilisation de Wave. “*WAVE est un outil gratuit d'évaluation de l'accessibilité web, conçu pour identifier les moyens de rendre une page web plus accessible aux personnes handicapées*” [14].

WAVE met en évidence des informations importantes pour une évaluation de l'accessibilité avec des icônes intégrées :

- Les erreurs rouges indiquent des problèmes qui affecteront certains utilisateurs handicapés et représentent des non-conformités aux directives WCAG (Web Content Accessibility Guidelines). Cliquer sur une icône dans le panneau “Détails” met en évidence l'emplacement correspondant sur la page web et ouvre une info-bulle expliquant l'erreur.
- Erreurs de contraste, ces erreurs concernent le texte qui ne respecte pas les exigences de contraste des WCAG. Par exemple, du texte blanc sur un fond bleu avec un contraste insuffisant sera signalé.
- Les alertes jaunes indiquent des éléments de la page qui peuvent causer des problèmes d'accessibilité. L'évaluateur doit décider de l'impact potentiel de ces alertes.
- Les icônes vertes indiquent des fonctionnalités qui amélioreront l'accessibilité si elles sont correctement mises en œuvre, comme le texte alternatif descriptif pour les images.
- Les éléments structurels bleus montrent des titres, des listes non ordonnées, et des régions ou repères identifiés sur la page. Ils aident à organiser le contenu de manière logique et accessible.
- Les icônes violettes montrent l'utilisation des attributs ARIA (Accessible Rich Internet Applications) qui fournissent des informations d'accessibilité importantes en étiquetant les éléments, mais doivent être utilisés avec précaution pour ne pas nuire à l'accessibilité.

WAVE est facile à utiliser puisqu'il s'agit d'une extension installable sur n'importe quel navigateur et capable d'analyser toutes les pages. Au-delà des directives WCAG, mondialement reconnues, les sites gouvernementaux français se réfèrent à la norme RGAA (Référentiel Général d'Amélioration de l'Accessibilité).

Pourquoi ce point est-il autant mis en avant ? L'accessibilité numérique signifie que les sites web, technologies et outils sont conçus et développés pour être utilisables par les personnes

handicapées. Il est essentiel de suivre les grands principes de l'accessibilité : perceptible, utilisable, compréhensible, robuste, tant sur les plans suivants :

- Auditif, il faut pouvoir lire ce qui est dit dans une vidéo ou un son émis.
- Visuel, il faut pouvoir redimensionner du texte, faire attention au contraste, avoir une langue de lecture définie.
- Moteur, il faut avoir des fonctionnalités activables et désactivables au clavier et pas seulement à l'utilisation de la souris, un temps supplémentaire pour effectuer des actions, des liens en début de page pour accélérer la navigation via des accès rapides (et pour des cas plus poussés nous pouvons retrouver les commandes vocales, l'eye tracking...).
- Cognitif, le principe Facile À Lire et à Comprendre doit être respecté (FALC) les fonctionnalités ne doivent pas demander une interprétation, usage d'une police spécifique pour les personnes souffrant de dyslexie (par exemple la police Opendyslexique avec empattement et interligne augmenté).

Pour respecter les principes de l'accessibilité, nous utilisons en grande majorité le système de design de l'État (DSFR). Il est “*obligatoire pour tous les sites internet et applications mobiles de l'État depuis le mois de juillet 2023, il permet plus de rapidité dans la création et la mise à jour des sites et garantit la cohérence d'ensemble de l'écosystème pour le bénéfice de ses utilisateurs*” [15]. Ce système mis à disposition regroupe de nombreux composants, couleurs, émoticônes etc. Ces éléments respectent ou permettent de respecter la norme RGAA, mais c'est un devoir du développeur de s'assurer qu'aucune erreur Wave ne soit remontée, et c'est à un BA de s'assurer de la bonne réalisation de l'accessibilité ainsi que de la cohérence entre le DSFR et les maquettes validées avec le client.

- Step 6 : L'étape de validation de la tâche par un BA est une étape essentielle qui peut faire gagner beaucoup de temps comme nous le verrons par la suite. Cette tâche consiste tout simplement à vérifier que la fonctionnalité ou l'écran développé correspond aux specs. Pour ça, différents scénarios s'offrent à nous, un cas nominal et des scénarios alternatifs qu'il faut prendre en compte. Cette validation passe aussi par des tests de non-régression, nous nous assurons que la fonctionnalité ajoutée ne vient pas compromettre le comportement d'une autre.
- Step 7 : C'est l'étape finale où tout le développement s'est effectué sur une branche parallèle à la branche principale, celle de la version en cours. Il faut maintenant combiner ces branches. Pour cela, nous ouvrons une Merge Request (MR), qui permet de présenter les changements apportés par notre branche avant leur intégration dans la branche principale. Lorsque nous créons cette requête, une pipeline CI/CD se déclenche en arrière-plan. Elle commence par construire (build) notre application, puis lance les batteries de tests. Ces tests, qu'ils soient côté back-end ou front-end, sont ceux rédigés précédemment. L'ajout d'une nouvelle fonctionnalité pouvant impacter un développement antérieur, il est nécessaire d'adapter les tests existants (et non de les supprimer) pour refléter le nouveau comportement. De plus, de nouveaux tests doivent être rédigés par le développeur pour vérifier cette fonctionnalité.

En effet, un des steps de la pipeline consiste à lancer un Sonar. SonarQube est un “*outil d'assurance qualité du code qui collecte et analyse le code source pour fournir des rapports sur la qualité sur un projet*” [16]. Il analyse le code source sous différents aspects et le décompose couche par couche, du niveau du module jusqu'au niveau de la classe. À chaque niveau, il

produit des valeurs métriques et des statistiques qui révèlent les zones problématiques ou des erreurs de conception nécessitant des améliorations.

C'est donc un outil qui parcourt le nouveau code pour vérifier la couverture des tests. Une couverture de code inférieure à 80% est jugée insuffisante, et cela s'applique également à la couverture globale du projet. En parallèle, une vérification du code dupliqué est effectuée : un taux de duplication supérieur à 5% dans le nouveau code est considéré comme trop élevé. Il est donc nécessaire de revenir sur le développement réalisé pour mutualiser certaines parties du code. Sonar identifie également les "code smells" (mauvaises pratiques de conception) et exige leur correction. Après toutes ces vérifications, Sonar valide le code, et la pipeline est considérée comme réussie. Lorsque la validation est effectuée, le statut de la Merge Request (MR) passe de "En cours" à "À relire". Les Responsables Techniques (RTs) doivent alors effectuer une vérification manuelle supplémentaire en examinant toutes les lignes de code pour repérer d'éventuelles erreurs de développement.

S'il n'y a plus de correction à apporter à la MR, elle est alors merge à la version principale en cours (qui dépend de la phase actuelle) et ces nouvelles fonctionnalités sont alors accessibles aux autres membres de l'équipe de développement, pouvant débloquer le développement d'autres fonctionnalités.

Lorsqu'un lot de MR a été merge, et que de nouvelles fonctionnalités ou corrections de bugs (aussi appelés "defects") sont disponibles, une livraison est effectuée. Il ne s'agit pas d'une livraison au client, mais d'un déploiement sur l'environnement de qualification (qualif). Cet environnement permet aux BA de réaliser leurs séries de tests pour vérifier que tout est conforme aux spécifications.

Les tests des BA sont approfondis et couvrent l'ensemble de l'application, ils testent l'application comme si elle était utilisée par le client. En cas d'anomalies, ilsouvrent de nouveaux tickets sur Jira, annotés comme "Defect". En fonction de la priorité de ces anomalies (qui peuvent parfois être bloquantes), les développeurs s'empressent de corriger les problèmes. À chaque nouvelle version, des tests de non-régression sont également effectués pour s'assurer que les fonctionnalités précédemment déployées n'ont pas été altérées. Cette phase de test consiste généralement à relancer tous les tests ayant été passés avec succès précédemment.

En ce qui concerne les tests rédigés par les développeurs, nous avons principalement opté pour une stratégie de tests unitaires en termes de méthodologie. Les tests unitaires sont des tests dits "white box", c'est-à-dire des tests où nous examinons le comportement interne d'une unité de code individuelle, comme une méthode ou une fonction isolée. En revanche, les tests "black box" se concentrent uniquement sur les actions visibles et le comportement externe de l'application sans tenir compte de son fonctionnement interne. Bien que les tests "black box" puissent sembler essentiels car ils permettent de vérifier le fonctionnement global de l'application, les tests unitaires jouent un rôle crucial dans la vérification approfondie et précise des composants individuels du code. En pratique, les tests unitaires assurent une couverture détaillée des différentes parties du code, ce qui peut prévenir des problèmes qui pourraient ne pas être détectés par les tests "black box".

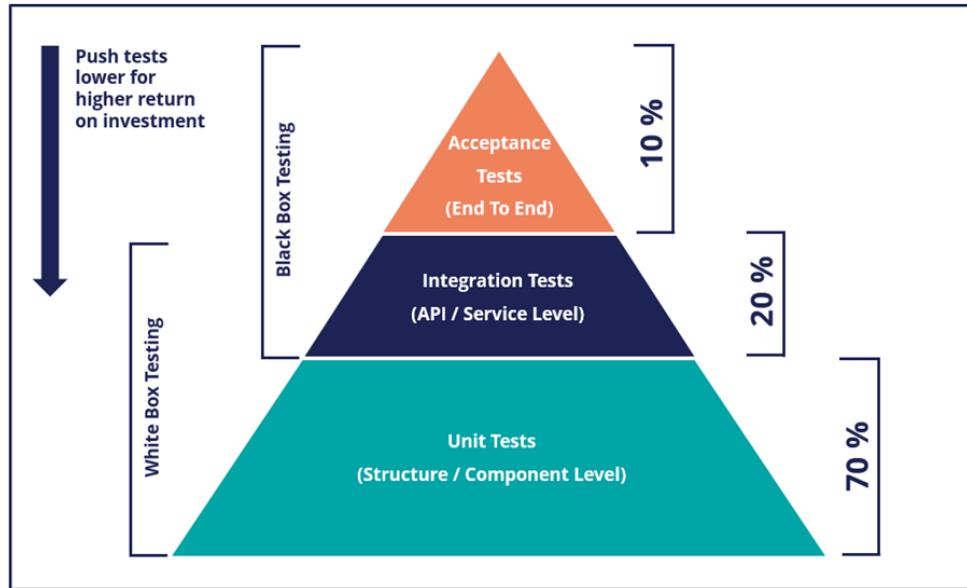


Figure 7: Pyramide de test [17].

La pyramide de tests est un modèle qui simplifie la complexité des tests logiciels en les organisant en une structure hiérarchique efficace.

Globalement les tests unitaires ont pour objectif d'identifier et corriger rapidement les bugs dans les petites parties du code. Les tests d'intégration quant à eux cherchent à garantir que les différents composants du système fonctionnent correctement ensemble. Enfin les tests End to End visent à valider le bon fonctionnement de l'application dans son ensemble, du début à la fin, dans des conditions réelles d'utilisation.

La question maintenant est : pourquoi nous sommes-nous contentés d'effectuer uniquement des tests unitaires dans notre développement ? La contrainte de temps pesait lourdement sur le projet. En mettant l'accent sur les tests unitaires, qui sont rapides à exécuter et faciles à maintenir, nous avons pu bénéficier de cycles de développement plus courts et plus agiles. Les tests unitaires fréquents et automatisés aident à maintenir un haut standard de qualité du code tout au long du cycle de vie du développement. En revanche, les tests de type "black box" sont plus complexes à mettre en place mais assurent une vérification complète du bon fonctionnement de l'application de bout en bout. Dans un contexte où le temps est limité, il est souvent préférable de privilégier une pyramide de tests avec une base solide, c'est-à-dire de nombreux tests unitaires, plutôt que de consacrer beaucoup de temps à des tests "black box" et avoir une base fragile. À l'avenir, l'objectif est de compléter cette pyramide de tests en ajoutant les tests nécessaires pour atteindre un niveau de couverture complet, jusqu'au pyramidion.

## IV L'Application

### IV.1 Solution proposée

SI Formation doit s'imprégner de toutes les notions métier liées à la douane et reproduire le workflow autour des formations douanières, et c'est ainsi que la solution proposée s'est développée naturellement.

L'application issue du projet doit permettre aux gestionnaires de créer des fiches de formation. Sur ces fiches, ils peuvent renseigner différentes informations, telles que le libellé de la formation, les nomenclatures associées (des étiquettes décrivant le type de formation), la direction responsable de cette formation, etc. À partir d'une fiche descriptive de formation, un gestionnaire peut créer des sessions. Les sessions représentent la mise en pratique des formations. Une session s'étend sur une période donnée, où chaque journée est découpée en vacations (généralement des demi-journées). Chaque vacation est attribuée à un formateur, rémunéré ou non, chargé de cette formation.

Les agents de la douane peuvent ensuite rechercher des formations pour trouver celles qui répondent à leurs besoins. À partir d'une fiche de formation, ils peuvent consulter les sessions publiées auxquelles ils peuvent postuler. Les agents effectuent une demande d'inscription via un formulaire où ils doivent fournir des informations telles que leur motivation et les documents obligatoires. C'est à ce moment que le workflow des candidatures commence.

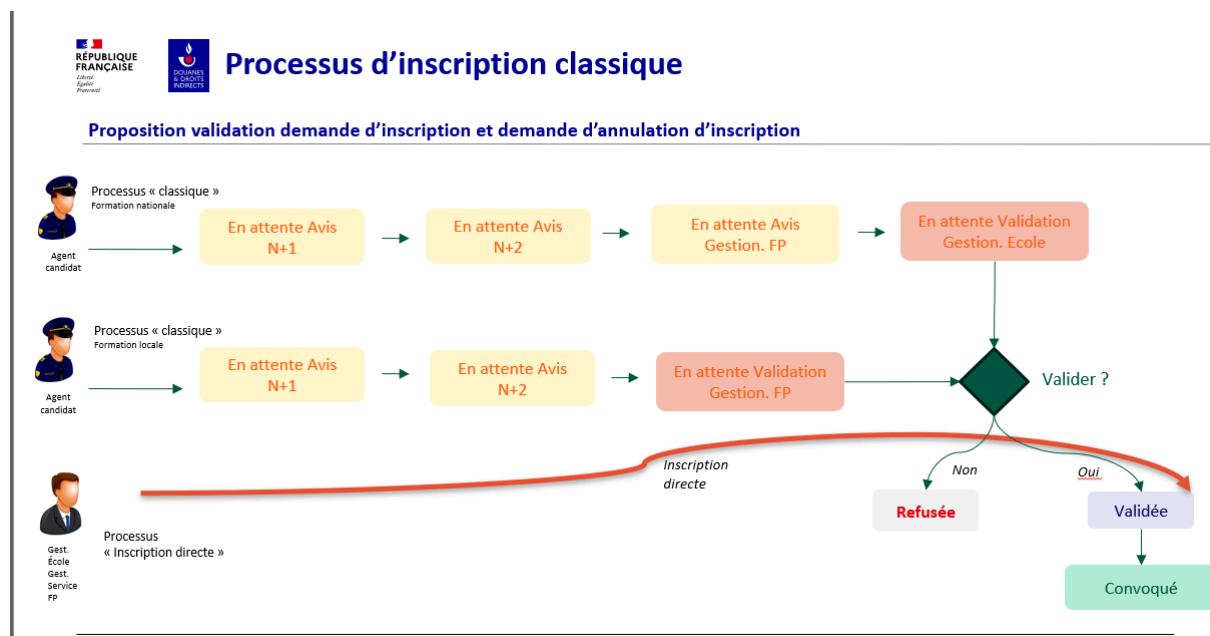


Figure 8: Workflow des inscriptions.

Lorsqu'une demande d'inscription est effectuée, un e-mail est envoyé au N+1 de l'agent pour l'informer qu'une nouvelle candidature est à traiter. Lorsque le N+1 se connecte à SI Formation, il peut visualiser ces candidatures sur la page d'accueil. Il peut alors émettre un avis sur chacune d'elles, qu'il soit favorable ou défavorable (avec justification), et valider ses choix. À ce moment-là, le N+2 recevra également un e-mail l'informant qu'une candidature est à traiter. Le N+2 aura généralement plus de candidatures à traiter et, sur sa page d'accueil, il aura une

vue d'ensemble des sessions avec des candidatures à examiner. Il pourra alors émettre son avis à son tour. Dans le cadre d'une formation nationale, une autre étape est impliquée : le gestionnaire FP doit donner son avis et dresser un classement en fonction des priorités. Étant donné que les formations ont un nombre limité de places, si de nombreuses candidatures sont reçues, tout le monde ne pourra pas y assister. Il est à noter qu'à chaque étape, l'acteur en charge peut consulter les avis et commentaires des étapes précédentes.

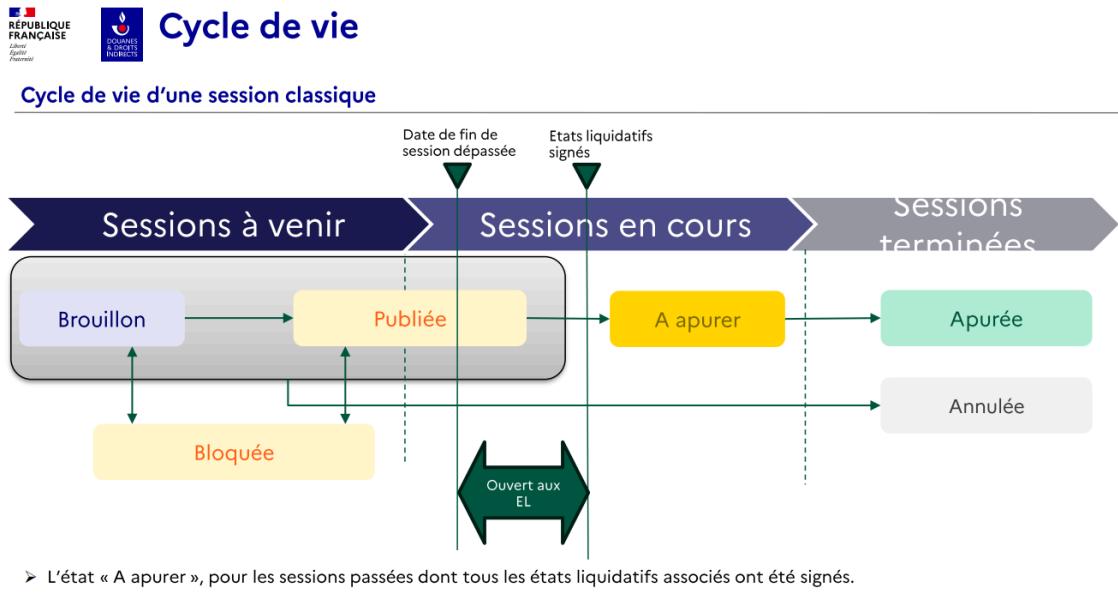
Quand tous les avis ont été donnés, les gestionnaires de session interviennent. Ils accèdent à leur outil dédié pour chaque session. Dans le cadre d'une formation nationale, il s'agit des gestionnaires d'école, tandis que pour une formation locale, ce sont les gestionnaires FP. Ils disposent alors d'un tableau de bord leur offrant une vue d'ensemble de l'avancement du workflow pour toutes les candidatures de la session. Bien qu'ils puissent consulter les détails des avis (qu'ils proviennent du N+2 ou du gestionnaire FP, selon le contexte), ils sont responsables de la validation ou du refus des demandes d'inscription.

Lorsqu'une demande est validée, ou à tout moment durant le cycle de vie d'une candidature, l'agent peut demander l'annulation de sa candidature en cas de contretemps. Si une demande d'annulation est effectuée, le workflow reprend à zéro pour cette demande spécifique. Une fois l'annulation traitée, le gestionnaire concerné peut également valider ou refuser cette demande.

Une fois la candidature validée, elle est considérée comme finalisée. Le gestionnaire envoie alors un e-mail et un document PDF confirmant de manière définitive la participation de l'agent à la formation.

À partir du moment où la date limite pour les inscriptions est dépassée, la session elle-même entre dans un nouveau cycle.

Cependant avant d'en parler, une autre notion à considérer est celle des séances Tir/TPCI. Ces séances sont très particulières et auraient presque nécessité une application distincte pour leur gestion, tant elles sont indépendantes du reste de l'application. Les douaniers ont régulièrement besoin de se former à des exercices de tir, que ce soit pour utiliser de nouveaux modèles d'armes ou pour améliorer leur dextérité dans le maniement des armes. Ces séances sont soumises à de nombreuses régulations, et la logique métier qui les sous-tend est très rigoureuse. Tout d'abord, des formations spécifiques existent pour ces séances, et elles sont confidentielles. Cela signifie que seuls les individus invités peuvent participer à ces formations, et il est impossible de rechercher ce type de séances via l'outil de recherche de SI Formation. Ces séances donnent naissance à la notion de "séance réalisée", ce qui permet de maintenir un historique pour réaliser des statistiques futures. Pour chaque séance réalisée, un certain nombre de métriques sont collectées, telles que les participants, les dates des séances, le nombre de balles tirées par agent, le formateur responsable de la supervision, etc.



Sous-Direction des Systèmes d'Information et télécommunication

16/04/2024

Figure 9: Workflow des sessions.

Deux notions importantes sont introduites dans ce nouveau workflow : les états liquidatifs et l'apurement. Tout d'abord, le schéma montre une période durant laquelle il est possible de signer les états liquidatifs. Ces états attestent de la possibilité de rémunérer un ou plusieurs formateurs présents lors d'une session ou d'une séance. Pour qu'un utilisateur puisse accéder à l'écran de gestion des états liquidatifs, deux prérequis doivent être remplis : la session doit être publiée ou bien la séance Tir/TPCI doit être terminée, et la date de fin doit être dépassée depuis les mois précédents. La signature des états liquidatifs est soumise à diverses contraintes, telles que s'assurer qu'aucun plafond annuel n'a été dépassé et que les barèmes de rémunération ont été respectés.

Une fois cette période écoulée, il est alors possible d'apurer la session. Une session ne peut être apurée que si la gestion des présences a été effectuée et les états liquidatifs signés. Les séances Tir/TPCI ne font pas l'objet d'apurement, car elles sont saisies a posteriori. Lorsque la session est à apurer, ni les formateurs, ni les agents inscrits, ni les périodes, ni les vacations ne sont modifiables. La session est alors définitivement figée et ne peut plus être modifiée. Une fois la séance terminée, le calcul de la rémunération est effectué et la session est considérée comme terminée.

Nous remarquons également sur le graphique l'ajout d'un nouveau statut pour les sessions, attribué à celles qui sont bloquées. Une session est bloquée lorsque le plafond de rémunération est dépassé pour au moins un formateur. Elle reste dans cet état tant que le problème n'est pas résolu, sinon elle est annulée si elle dépasse théoriquement le début de sa réalisation.

S'accompagne à ça les FIF (fiche individuelle de formation) que chaque agent détient. Il s'agit d'une fiche récapitulative permettant à un agent de consulter des informations telles que les :

- formations suivies (celles où il est présent sur au moins une vacation de la session).
- formations demandées non suivies (celles où il a une absence, une inscription refusée par le gestionnaire, une inscription annulée par l'agent lui-même, ou une session tout simplement annulée dans sa globalité).

- formations animées (celles pour lesquelles au moins une vacation a été animée).
- ses habilitations et qualifications.

Les statiques sont un point clé de l'application, aucune donnée ne sera jamais supprimée. Une session passée, une inscription annulée, les documents fournis lors d'une inscription, tout est imaginé de telle sorte à pouvoir retracer les différents cycles de vie que nous avons pu détailler (celui des candidatures ou encore celui des sessions).

## IV.2 Implémentation

Pour la réalisation et l'implémentation de l'application, il s'agissait principalement de mettre en œuvre la solution conçue, évaluée et budgétisée lors des ateliers avec le client.

Un point important à souligner est l'architecture du projet d'un point de vue technique. L'architecture choisie est l'architecture hexagonale, parfaitement adaptée à Spring Boot.

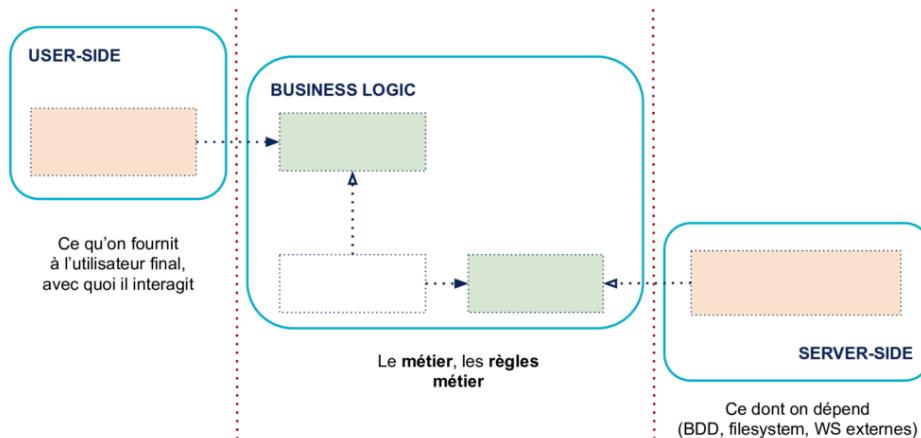


Figure 10: Architecture hexagonale [18].

L'architecture hexagonale, ou architecture “*ports and adapters*”, est une approche qui permet de structurer une application en séparant clairement la logique métier (Business Logic) des autres parties du système, telles que les interfaces utilisateur (User-Side) et les composants d'infrastructure comme les bases de données ou les services externes (Server-Side). Cette architecture est particulièrement bénéfique pour un projet Spring Boot, car elle facilite le développement, les tests, et la maintenance de l'application en isolant les préoccupations métier des détails d'implémentation technique [18].

En isolant la logique métier au cœur de l'application, nous, en tant que développeurs, pouvons tester automatiquement et de manière fiable son comportement indépendamment des autres composants. Cela permet d'obtenir un retour rapide et précis lors de l'exécution des tests. De plus, en utilisant des interfaces, appelées “*ports*”, pour interagir avec les systèmes externes et les interfaces utilisateur, l'architecture hexagonale garantit que les dépendances sont dirigées vers la logique métier, et non l'inverse. Ainsi, les problèmes rencontrés dans une partie du système, comme un changement de base de données ou une modification de l'interface utilisateur, n'ont aucun impact direct sur le cœur métier [18].

Cette structure modulaire rend l'application plus flexible et plus facile à maintenir, car les adaptations nécessaires, les “*adapters*”, peuvent être modifiées ou remplacées sans affecter le

reste du système. Dans le contexte d'un projet Spring Boot, où les besoins en scalabilité et en tests automatisés sont cruciaux, l'architecture hexagonale s'avère particulièrement efficace pour garantir que l'application reste robuste tout en permettant des itérations rapides et sûres.

Ensuite, pour aborder la gestion des données et la base de données utilisée, nous avons opté pour Hibernate. Un des défis majeurs avec les bases de données est la gestion des requêtes. Hibernate est un framework ORM (Object-Relational Mapping) très populaire dans la communauté Java. Il facilite la manipulation des données en permettant aux développeurs de mapper les objets de l'application aux tables d'une base de données relationnelle. Ainsi, il est possible de gérer les opérations CRUD (définies plus tôt) sur ces objets sans avoir à écrire de code SQL à la main, ce qui simplifie considérablement le processus de développement [19].

Pourquoi avons-nous spécifiquement choisi Hibernate dans notre cas ? Tout d'abord, parce que nous utilisons Java, et Hibernate repose sur la spécification JPA (Java Persistence API), qui définit un standard pour la persistance des objets Java. JPA permet de définir comment les objets Java doivent être stockés, récupérés et gérés dans une base de données relationnelle. Cependant, JPA n'a pas d'implémentation concrète, il ne s'agit que d'une spécification. C'est là qu'Hibernate intervient, en offrant une solution pour gérer la persistance des données grâce à son propre langage de requête spécialisé : Hibernate Query Language (HQL).

Il est également important de noter qu'Hibernate prend en charge le lazy loading, ce qui signifie que les données ne sont chargées que lorsqu'elles sont réellement nécessaires. Cette fonctionnalité est particulièrement utile dans les applications Spring Boot pour optimiser l'utilisation des ressources et améliorer les performances globales. Hibernate est facile à configurer et offre de nombreux autres avantages pour se concentrer sur la performance, ce qui est primordial dans une application à grande échelle, comme le caching et la gestion automatique des transactions. Les opérations de base sont déjà prises en charge par Hibernate, ce qui simplifie le développement [19].

Grâce à Hibernate, du côté base de données, il ne nous reste plus qu'à rédiger des scripts SQL pour créer nos tables et insérer des données de référence. Tout sera correctement interprété lorsque nous implémenterons les entités Java. Une Java Entity [20] est simplement un objet utilisé pour représenter une table dans une base de données relationnelle. Chaque instance d'une entité correspond à une ligne de cette table. C'est la liaison entre les données stockées en base et les données utilisables au sein de l'application.

Dans les tâches plus transverses ou industrielles, nous avons également dû nous pencher sur Harbor et Rancher.

Harbor est un outil essentiel pour le déploiement de notre application, car il garantit une couche de sécurité avec la détection de vulnérabilités, ainsi que la gestion des images de conteneurs [21]. Une image Docker y est stockée et décrit l'entièreté de la mise en place de notre application avec tous les services sur lesquels elle repose. Cette image se nomme %VERSION%-SNAPSHOT lorsqu'il s'agit d'un déploiement sur l'environnement de qualif, et SNAPSHOT est retiré lorsqu'il s'agit d'un déploiement en MOA (l'environnement de la douane) pour signaler qu'il s'agit d'une version stable de fin de phase.

À côté de ça, nous utilisons Rancher qui est une plateforme d'orchestration qui simplifie la gestion et le déploiement d'applications au sein de clusters de serveurs. Elle permet d'organ-

iser les conteneurs en services et en ensembles logiques appelés stacks, assurant ainsi une structuration claire des applications. [22]

En intégrant Harbor avec Rancher, nous améliorons la sécurité et l'efficacité des déploiements de notre application conteneurisée, en garantissant l'utilisation d'images fiables et conformes, tout en facilitant son intégration dans des processus de déploiement continu. Tout est automatisé au travers d'une pipeline CI/CD pour éliminer les opérations redondantes.

Les services externes appelés par notre application incluent Rush et InterRH, deux services de la douane permettant de collecter des informations indispensables, telles que le matricule d'un agent, le service auquel il appartient, et ses droits sur l'application (gestionnaire local ou national, administrateur, agent sans droit). En fonction de ces droits, des contrôles d'accès (guards) ont été mis en place pour limiter l'accès à certaines pages de gestion de l'application pour les agents sans droits particuliers. Nous récupérons également des informations personnelles telles que le nom, le prénom, l'adresse e-mail et les supérieurs hiérarchiques de l'agent (ces données étaient bouchonnées dans le cadre des tests sur nos environnements, ce qui signifie que nous avions créé des agents fictifs pour éviter d'accéder à des informations trop confidentielles).

### IV.3 Résultats

La fin de mon stage a coïncidé avec le milieu de la phase 3 du projet. Bien que la fin de la phase 2 ait été marquée par des difficultés, notamment une instabilité de l'application due à de nombreuses régressions causées par l'intégration trop précipitée de nouvelles fonctionnalités, la première livraison de la v2 avec réserves (report de certaines fonctionnalités) n'a pas été convaincante. Cependant, dès le début de la phase 3, nous avons rapidement redressé la situation en livrant une v2 bien plus fonctionnelle.

L'une des principales difficultés rencontrées sur le projet SI Formation concernait les échéances serrées, qui imposaient un rythme de développement rapide, rendant la qualité du code difficile à maintenir. À chaque début de nouvelle phase, nous devions encore nous concentrer sur les tâches de la phase précédente, accumulant ainsi du retard au fil du temps. C'est dans ce contexte que j'ai été chargé de travailler seul sur la notion des états liquidatifs pendant plusieurs semaines au cours de la phase 3. Ce sujet était particulièrement complexe en raison de problèmes de conception et de modélisation du modèle de données, qui n'avaient pas été identifiés suffisamment tôt. Le démarrage du développement sur cette partie a été difficile, car malgré des spécifications validées, celles-ci ont nécessité des révisions à la fin du processus via des points DEV/BA et des discussions directes avec le client.

Néanmoins, être responsable d'une partie aussi cruciale m'a poussé à adopter une approche méthodique dans ma stratégie de développement. Cela m'a également conduit à être particulièrement rigoureux dans ma compréhension du sujet, veillant à clarifier chaque point de doute, ce qui a permis d'identifier les problématiques rencontrées, pour finalement être efficace et rapide dans mon développement.

Un moment clé de la transition entre la phase 2 et la phase 3 a été le REX (retour d'expérience) de la phase 2. Un REX consiste à réunir l'équipe pour faire une rétrospective des semaines écoulées. Ce moment de partage s'articule autour de cinq axes :

- Start doing : les actions à entreprendre qui n'ont pas encore été mises en place.
- Stop doing : les actions à cesser, qu'elles aient été volontairement mises en place ou non.
- More of : les actions déjà en place mais qui pourraient être davantage exploitées ou améliorées.
- Less of : les actions en cours qu'il serait préférable de réduire.
- Keep doing : les bonnes pratiques à maintenir.

Chaque membre de l'équipe exprime son avis sur ces points pendant 15 minutes sous forme de tickets. Ensuite, lors d'un tour de table, les idées sont développées par les auteurs des différents tickets, puis une phase de vote permet de mettre en avant les tickets nécessitant le plus d'attention. Une fois les tickets prioritaires choisis, l'équipe réfléchit aux solutions pour résoudre les problèmes critiques, améliorer les points importants mais sous-exploités, et éliminer les mauvaises pratiques. Ce REX est essentiel car il permet à l'équipe de s'accorder sur les nouvelles directives à suivre afin de ne pas répéter les erreurs lors de la prochaine phase.

Globalement, le projet a été un franc succès. Nous avons reçu les félicitations directes de l'adjoint du directeur des opérations chez Sopra Steria, attestant de la qualité du travail réalisé. Cependant, il est important de réfléchir aux axes d'amélioration pour un projet de cette envergure avec des délais aussi courts. D'un point de vue technique, les technologies utilisées étaient parfaitement adaptées, la structure du projet était idéale, et les plateformes de monitoring étaient simples à utiliser et fiables, tant en termes de sécurité que de gestion de la charge. Cependant, la communication a été un point faible. Échanger pendant un sprint n'est pas simple, et les changements dans l'équipe n'ont pas facilité les choses. Les nouveaux développeurs n'ont pas toujours pu assimiler les pratiques discutées lors des points DEV, préférant parfois appliquer les méthodes de leurs précédents projets, en particulier lorsqu'ils sont externes à Sopra Steria. De plus, les points DEV, censés être hebdomadaires, n'ont pas toujours été maintenus comme prévu lors des phases de rush, bien qu'il s'agisse de points d'échange cruciaux.

La communication entre les BA (Business Analysts) et les développeurs est également essentielle. Cependant, le fait que de nombreux fils de discussion aient eu lieu en parallèle a conduit à des spécifications manquant de clarté ou comportant des zones d'ombre, alors que les développeurs auraient pu apporter une perspective technique précieuse. Bien que le point de conception fonctionnelle avec un BA au début de chaque tâche ait partiellement atténué ce problème, les spécifications étaient souvent rédigées par une seule personne, ce qui ne garantissait pas toujours une vision complète des différents workflows de l'application. Cette limitation a parfois conduit à des erreurs de compréhension, orientant le développement de manière incorrecte. Lors des phases de test, ces incohérences ont nécessité des révisions, engendrant parfois un rework. Revoir une fonctionnalité déjà développée est particulièrement problématique dans un contexte de rush, car cela implique de doubler le temps de travail nécessaire pour développer une même fonctionnalité.

Voici quelques exemples concrets :

- Une pop-in, développée conformément aux spécifications et validée lors des démonstrations avec les BA, s'est finalement révélée devoir être transformée en une page distincte. Cette transition a nécessité une nouvelle réflexion pour éviter les régressions.

- Une pop-in destinée à ajouter des nomenclatures à une fiche de formation lors de sa création manquait de règles de gestion claires, ce qui a entraîné diverses interprétations de son fonctionnement. Ces règles ont dû être révisées ultérieurement, nécessitant des ajustements supplémentaires.

Le manque de projection a également été problématique entre les phases. Par exemple, l'évolution de l'outil pour inscrire de nouveaux stagiaires à une session à la volée durant la phase 2 a nécessité un rework de l'outil développé lors de la phase 1, car les comportements étaient très différents, au lieu d'être une simple évolution.

En dehors des spécifications, des erreurs ont également été commises par les développeurs : mauvaise interprétation des spécifications, conception fonctionnelle et technique insuffisamment approfondie, manque de visibilité sur les autres développements (nous avons parfois travaillé sur la même fonctionnalité ou corrigé la même anomalie décrite dans deux tickets différents sans le savoir), et une stratégie de développement mal optimisée, entraînant des blocages fréquents. Les tâches étaient souvent trop dépendantes les unes des autres, empêchant de progresser pleinement. Cette situation a été partiellement corrigée à partir de la phase 2 grâce à une utilisation améliorée de draw.io, qui a permis de concevoir un diagramme des dépendances entre les différentes tâches et de mieux définir les priorités.

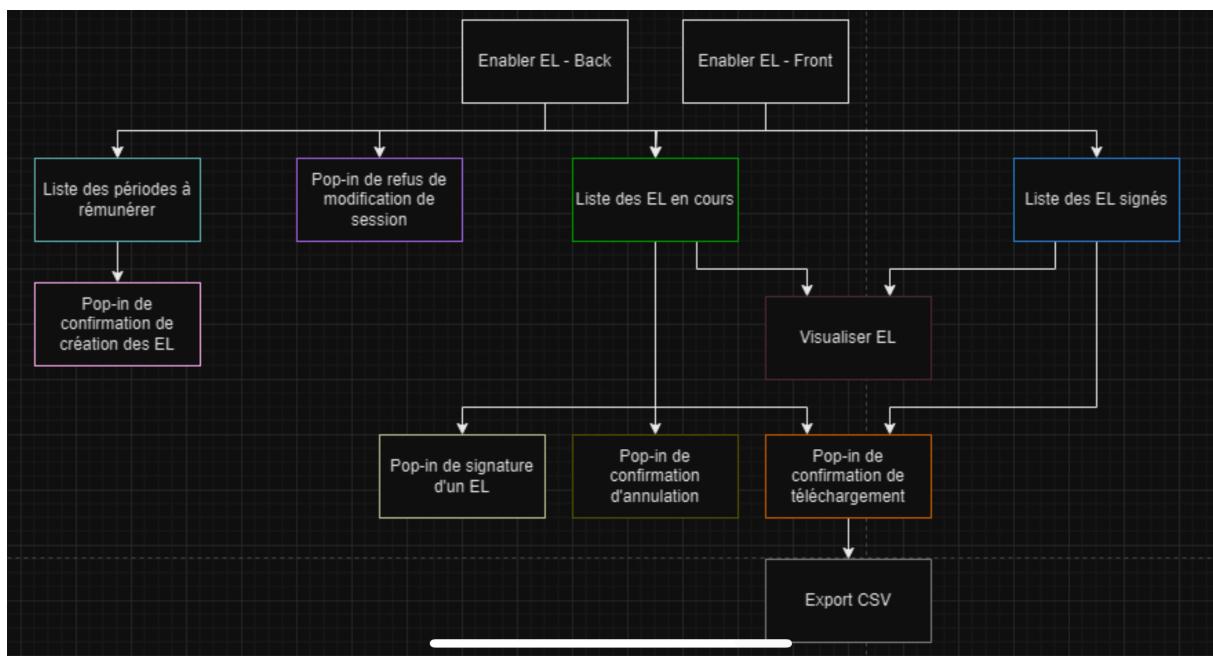


Figure 11: Diagramme de dépendances des états liquidatifs.

Cette expérience m'a montré à quel point une communication efficace est fondamentale pour le succès d'un projet.

## V Evaluation et Conclusion

En termes d'évaluation de mon stage, une fiche de satisfaction a été établie selon plusieurs critères, tels que les aptitudes générales, les méthodes de travail, le comportement et l'intégration.

Concernant le critère « savoir travailler au sein d'une équipe », qui inclut : interagir, partager les informations et coopérer avec chaque membre de l'équipe pour atteindre les objectifs collectifs ou individuels, il a été noté que j'ai me suis bien intégré à l'équipe, avec une participation active aux réunions V0 et V1 et un reporting efficace et clair.

Pour ce qui est de la « capacité à connaître et comprendre le fonctionnement de SI Formation », il a été mentionné que j'ai acquis rapidement des compétences sur le projet. Aujourd'hui, je suis décrit comme un profil autonome capable de réaliser les développements avec efficacité. Je suis suffisamment débrouillard pour gérer les sujets qui me sont affectés, avec un niveau de support efficace, une seule explication suffit pour comprendre le besoin. Un axe d'amélioration identifié est la nécessité d'avoir une vue d'ensemble des sujets en cours dans l'équipe, ce qui est considéré comme normal pour un junior dans un contexte professionnel, comme discuté lors de l'évaluation.

Concernant la « réalisation de développements simples et moyens dans le respect des SFD et des bonnes pratiques - complexité technique, la recherche de solutions, la conception, les nouvelles technologies et l'intégration (avec une autonomie progressive) » le responsable de l'évaluation a noté que mes résultats ont largement dépassé les attentes. Les commentaires soulignent que je réalise des développements de qualité, en vérifiant auprès de mes RTs et BA que les attentes sont bien respectées tout au long du processus. Mes RTs sont confiants dans ma capacité à gérer les tâches de développement, et j'ai réussi à prendre en charge des développements complexes, ce qui justifie une mention très favorable sur mon niveau technique et ma compréhension des besoins. Un axe d'amélioration suggéré est de traiter des sujets d'industrialisation tels que l'installation de l'application sur des environnements Kubernetes, ce que j'ai cherché à améliorer suite à cette évaluation.

Enfin, concernant « la capacité à évaluer mon RAE et à faire remonter de manière proactive les éventuelles difficultés rencontrées », il a été noté que mon niveau de reporting sur Jira est correct, que je sais évaluer mon RAE et que je remonte systématiquement les dérives auprès de mes RTs.

De mon point de vue, je suis plus que satisfait de cette expérience. J'en sors grandi avec des connaissances techniques renforcées et améliorées. J'ai constaté ma capacité à m'intégrer rapidement dans un projet dynamique dès ses débuts. J'ai eu l'occasion de faire de belles rencontres humaines dans un cadre professionnel, d'apprendre des connaissances partagées par mes RTs et autres développeurs expérimentés, et d'enrichir mon expérience professionnelle.

De plus, ayant mes efforts été remarqués, une opportunité de poursuivre mon parcours chez Sopra Steria à Nantes après mon stage m'a été offerte, une offre que j'ai acceptée avec enthousiasme, tant j'ai apprécié l'expérience.

## VI Bibliographie

### Bibliography

- [1] Sopra Steria, “À propos de Sopra Steria.” [Online]. Available: <https://www.soprasteria.com/fr/investisseurs/a-propos-de-sopra-steria/chiffres-cles>
- [2] Ecole DSP, “Tout savoir sur le développement web : métier, formations et salaire » Qu'est-ce que le développement web ?” [Online]. Available: <https://www.digitalschool.paris/guides/developpement-web/definition/#:~:text=Le%20d%C3%A9veloppement%20web%20englobe%20tout,et%20les%20mises%20%C3%A0%20jour>
- [3] Wikipedia, “CRUD.” 2024. [Online]. Available: <https://fr.wikipedia.org/wiki/CRUD>
- [4] Pure, “Définition de Framework.” [Online]. Available: <https://www.pure-illusion.com/lexique/definition-de-framework>
- [5] Charles Rémy, “L'essor des frameworks dans le développement web.” 2023. [Online]. Available: <https://agence-bb.ch/blog/lessor-des-frameworks-dans-le-developpement-web/>
- [6] Adityagaba, “Streamlining Backend-Frontend Integration: A Quick Guide.” 2023. [Online]. Available: <https://medium.com/@adityagaba1322/streamlining-backend-frontend-integration-a-quick-guide-145eca3cca05>
- [7] App Academy, “8 Most Popular Web Programming Frameworks (2024).” 2024. [Online]. Available: <https://www.appacademy.io/blog/8-most-popular-web-programming-frameworks>
- [8] Jonathan Robie, “What is the Document Object Model?” [Online]. Available: [https://www.w3.org/TR/WD-DOM/introduction.html#:~:text=The%20Document%20Object%20Model%20\(DOM,document%20is%20accessed%20and%20manipulated](https://www.w3.org/TR/WD-DOM/introduction.html#:~:text=The%20Document%20Object%20Model%20(DOM,document%20is%20accessed%20and%20manipulated)
- [9] Statista, “Most used web frameworks among developers worldwide, as of 2024.” 2024. [Online]. Available: <https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/>
- [10] GeeksforGeeks, “Difference between Spring and Spring Boot.” 2024. [Online]. Available: <https://www.geeksforgeeks.org/difference-between-spring-and-spring-boot/>
- [11] Ibrahim Cisse, “Quand et pourquoi Java est utilisé pour le développement d'applications ?” 2020. [Online]. Available: <https://ibracilinks.com/blog/quand-et-pourquoi-java-est-utilise-pour-le-developpement-d'applications#:~:text=Java%20permet%20d%C3%A9velopper%20des%20applications,comme%20Linux%2C%20avec%20quelques%20modifications>
- [12] DevUniversity - Team Redac, “Maîtriser les APIs REST : Principes et Avantages.” 2024. [Online]. Available: <https://www.devuniversity.com/blog/apis-rest-principes-et-avantages#:~:text=Les%20principes%20d'une%20API%20Rest%20sont%20simples,%C3%A9liminant%20certaines%20interactions%20client-serveur>

- [13] Atlassian, “Présentation de Jira - Jira pour les équipes.” [Online]. Available: <https://www.atlassian.com/fr/software/jira/guides/getting-started/who-uses-jira#for-agile-teams>
  - [14] Carolyn Quintero, “WAVE Web Accessibility Evaluation Tool.” [Online]. Available: <https://www.boisestate.edu/webguide/publishing/wave-web-accessibility-evaluation-tool/#:~:text=WAVE%20is%20a%20free%20web,identify%20many%20accessibility%20issues%20manually>
  - [15] Gouvernement français, “Service d'information du Gouvernement (SIG) - Système de design.” 2023. [Online]. Available: <https://www.info.gouv.fr/organisation/service-d-information-du-gouvernement-sig/systeme-de-design#:~:text=Cr%C3%A9%C3%A9%20et%20maintenu%20depuis%202020,sites%20internet%20et%20applications%20mobiles>
  - [16] Sean Malloy, “An Introduction on SonarQube.” [Online]. Available: <https://www.crestdata.ai/blogs/an-introduction-on-using-sonarqube#:~:text=SonarQube%20is%20a%20Code%20Quality,be%20measured%20continually%20over%20time>
  - [17] Pega, “Test pyramid.” [Online]. Available: <https://academy.pega.com/topic/test-pyramid/v1>
  - [18] Sébastien Roccaserra, “Architecture Hexagonale : trois principes et un exemple d'implémentation.” 2018. [Online]. Available: <https://blog.octo.com/architecture-hexagonale-trois-principes-et-un-exemple-dimplementation>
  - [19] Mohamed Elhamra, “Hibernate framework basics and architecture.” 2020. [Online]. Available: <https://medium.com/@mohamed.elhamra/hibernate-framework-basics-and-architecture-fe2bea5911ae>
  - [20] Oracle, “The Java EE 6 Tutorial - Entities.” [Online]. Available: <https://docs.oracle.com/javaee/6/tutorial/doc/bnbqa.html#:~:text=An%20entity%20is%20a%20lightweight,entities%20can%20use%20helper%20classes>
  - [21] Harbor, [Online]. Available: <https://goharbor.io/>
  - [22] Germain Lefebvre, “Présentation de Rancher.” 2018. [Online]. Available: [https://blog.ineat-group.com/2018/07/presentation-de-rancher/#:~:text=Rancher%20est%20un%20orchestrateur%20l%C3%A9ger,application%20\(ensemble%20de%20services\).](https://blog.ineat-group.com/2018/07/presentation-de-rancher/#:~:text=Rancher%20est%20un%20orchestrateur%20l%C3%A9ger,application%20(ensemble%20de%20services).)

## VII Annexe

### VII.1 Glossaire

Terme	Définition
<i>RT</i>	<ul style="list-style-type: none"> <li>• Responsable Technique : Encadre et supervise l'équipe de développeurs</li> </ul>
<i>BA</i>	<ul style="list-style-type: none"> <li>• Business Analyst : Etudie puis propose des solutions aux besoins commerciaux d'un client et traduit la solution retenue pour la partie tec</li> </ul>
<i>Specs</i>	<ul style="list-style-type: none"> <li>• Les spécifications sont le manuel de développement d'un développeur. Toutes les pages de l'application y sont décrites, ainsi que toutes les règles de gestion et d'enregistrement qui les accompagnent.</li> </ul>
<i>V0 / V1</i>	<ul style="list-style-type: none"> <li>• Successivement les réunions : quotidiennes et hebdomadaires</li> </ul>
<i>REX</i>	<ul style="list-style-type: none"> <li>• Retour d'expérience</li> </ul>
<i>Environnement de Qualif</i>	<ul style="list-style-type: none"> <li>• Environnement permettant à l'équipe BA de tester l'application</li> </ul>
<i>Environnement de MOA</i>	<ul style="list-style-type: none"> <li>• Environnement permettant à la douane de tester l'application</li> </ul>
<i>MR</i>	<ul style="list-style-type: none"> <li>• Merge Request, requête envoyée pour combiner une branche avec une nouvelle fonctionnalité ou une correction à la branche principale de l'application</li> </ul>
<i>Vocabulaire relatif au contexte métier</i>	
<i>Formation</i>	<ul style="list-style-type: none"> <li>• Représentée sous forme de fiche, il s'agit d'une thématique abordée dans le cadre de la montée en compétence des agents de la douane</li> </ul>
<i>Session</i>	<ul style="list-style-type: none"> <li>• Instance d'une formation, il s'agit d'une représentation de cette dernière dans la réalité</li> </ul>
<i>Période</i>	<ul style="list-style-type: none"> <li>• Intervalle de temps durant laquelle une session à lieu, il peut y avoir plusieurs reproductions de cette session donc plusieurs périodes</li> </ul>
<i>Vacation</i>	<ul style="list-style-type: none"> <li>• Demi-journée de formation durant une période</li> </ul>
<i>Séance</i>	<ul style="list-style-type: none"> <li>• Formation spécialisé dans les Tir/TPCI</li> </ul>

<i>Etat Liquidatif</i>	<ul style="list-style-type: none"> <li>Phase de contrôle après la cloture de la phase d'inscription à une session et que tous les agents y participant et formateurs placés sur des vacations sont convoqués. Un état liquidatif permet de valider la rémunération d'un formateur par le biais d'une signature</li> </ul>
<i>Apurement</i>	<ul style="list-style-type: none"> <li>Finalisation définitive d'une session</li> </ul>
<i>FIF</i>	<ul style="list-style-type: none"> <li>Fiche individuelle de formation, il s'agit d'une fiche propre à un agent pour qu'il puisse avoir un récapitulatif de sa participation à une séance ou session</li> </ul>
<i>Formateur</i>	<ul style="list-style-type: none"> <li>Personne qui encadre une session sur au moins une vacation</li> </ul>
<i>Gestionnaire</i>	<ul style="list-style-type: none"> <li>Personne qui valide les demandes d'inscription ou les demandes d'annulation pour la participation à une séance ou session</li> </ul>