



Marius Le Douarin IT - Information Systems 2024 ??? Professor-researcher	Sopra Steria Thakurova 9, 160 00 Praha 6 jakub.novak@fit.cvut.cz Amal Hafi Senior Information Systems Analyst
---	---

SI Formation - Rapport de stage

Year 2023/2024

Abstract (en)

The goal of this internship was to elaborate a solution of photo documentation of vehicles (cars particularly), based on a thesis previously achieved by Ing. Martin Vitek. The purpose of this solution was to provide to small companies of car reselling, a method of visualisation of the vehicles being sold, displayed in a website or an application. In effect, the traditionnal way of doing dynamic photographs of cars and other vehicles, is a costly method, that small companies can't afford : it needs a lot of ressources : rotating plate, high precision equipments, white background, lightning... The method used by resellers nowadays is much less precise, and not professionnal : pictures took on smartphone, with changing backgrounds, different lightnings, and different positions. Therefore, implementing this solution could help reseller selling their vehicles, by offering a better visualisation on websites, for instance flaws, color or possible hurts. This solution have three major components in its implementation : first, we need to acquire the images. These are took from a drone flying around a parked car, at the exact same place where other cars were shot. Because of the weather condition, and the background, the images needs to be normalized. This normalization operates in two steps : the light homogenization, and the segmentation. After this, we have normalized images ready to be merged into a visualisation of the car. Then, with the help of original images, we estimate the location of cameras. The final step is to render in an user interface the results of treatments applied on the images of the car.

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 un site 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.

Contents

I Introduction	5
II Sopra Steria	6
III Analysis	7
III.1 Context	7
III.2 State of the art	8
III.3 Methodology	13
IV The Application	21
IV.1 Proposed solution	21
IV.2 Implementation	24
IV.3 Results	26
V Evaluation	29
V.1 Results interpretation	29
V.2 Experiment return	30
V.3 Final state	31
VI Conclusion	33
VII Bibliography	35
Bibliography	35
VIII Annex	36
VIII.1 Glossary	36
VIII.2 Corporate Social Responsibility	36

I Introduction

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

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 à des zones géographiques nouvelles.

Cette mondialisation, à bien des égards très positives, comporte également 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 économies, 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 d'action, augmentant ainsi le volume de produits prohibés dans certain trafic, comme celui de 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 notre patrimoine culturel. Ces menaces se retrouvent sous la forme de trafic illégal des espèces animales et végétales menacées d'extinction (et représente par la même occasion la deuxième cause de disparition de celles-ci), des transferts de déchets illégaux (hospitaliers, chimiques, métaux lourds). Le patrimoine culturelle, quant à lui, est soumis à des réglementations relatives à la protection d'objets d'art, de collection ou d'antiquités.

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.

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 précis.

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).

<https://www.soprasteria.com/fr/investisseurs/a-propos-de-sopra-steria/chiffres-cles> 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%), autre part 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.

III Analysis

III.1 Context

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedit, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nolle me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini eruditio in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicitissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extreum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diurnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandi legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit. Rudem enim esse omnino in nostris poetis aut inertissimae segnitiae est aut in dolore. Omnis autem privatione doloris putat Epicurus terminari summam voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in voluptate aut a voluptate discedere. Nam cum ignorantia rerum bonarum et malarum maxime hominum vita vexetur, ob eumque errorem

et voluptatibus maximis saepe priventur et durissimis animi doloribus torqueantur, sapientia est adhibenda, quae et terroribus cupiditatibusque detractis et omnium falsarum opinionum temeritate derepta certissimam se nobis ducem praebeat ad voluptatem. Sapientia enim est una, quae maestitiam pellat ex animis, quae nos exhorrescere metu non sinat. Qua praeceptrice in tranquillitate vivi potest omnium cupiditatum ardore restincto. Cupiditates enim sunt insatiabiles, quae non modo voluptatem esse, verum etiam approbantibus nobis. Sic enim ab Epicuro reprehensa et correcta permulta. Nunc dicam de voluptate, nihil scilicet novi, ea tamen, quae te ipsum probaturum esse confidam. Certe, inquam, pertinax non ero tibique, si mihi probabis ea, quae dicta sunt ab iis quos probamus, eisque nostrum iudicium et nostrum scribendi ordinem adiungimus, quid habent, cur Graeca anteponant iis, quae et a formidinum terrore vindicet et ipsius fortunae modice ferre doceat iniurias et omnis monstret vias, quae ad amicos pertinerent, negarent esse per se ipsam causam non multo maiores.

III.2 State of the art

Maintenant que les bases du problèmes ont été posées, il y a eu un premier travail de réflexion pour mener à bien la mission. Les technologies autour du développement Web (tout le processus de création de sites internet ou d'applications) 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 pour cadre, et work pour travail, les frameworks sont de réels boîtes d'outils et une méthodologie rigoureuse de travail pour les développeurs. En effet ces outils sont généralement une bonne réponse aux différentes points critiques d'une phase de développement : arborescence, normes, sécurités, 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). Ces frameworks sont des bibliothèques de développement logiciel qui offrent une structure pré-construite, des composants autonomes qui ne demandent qu'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 de nouvelles fonctionnalités en continu pour s'adapter aux nouveaux besoins des développeurs qui les utilisent.

En définitif, les points à retenir pour déterminer l'importance d'utiliser des frameworks pour des projets de grande envergure sont :

- 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 avoir du code propre et fonctionnel qui ne ralentit pas le fonctionnement d'une application.
- Productivité : sur un projet avec une équipe importante, les framework permettent d'avoir une base de développement commune, qui fixe des process auxquels chacun doit se tenir. Acquérir des connaissances sur ce genre de technologies permet de changer de projet plus facilement sans avoir à monter en compétence à nouveau, en théorie il n'y aurait plus que la logique métier à apprendre.
- Communauté : cet aspect mentionné plus tôt est essentiel, travailler avec des tels outils permet de bénéficier d'un appui de toute une communauté qui pourra répondre à nos questions au travers de support et forum afin de corriger des bugs ou résoudre des problèmes de pro-

grammation plus ou moins connus. Il s'agit une nouvelle fois d'une source de gain de temps non négligeable.

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 les frameworks sont aussi vus comme un frein à la créativité pour certains. Les entreprises ont tendance à développer leurs propres outils privés internes.

Pour rappel, un projet web s'articule selon deux sections distinctes mais interdépendantes : le front end et le back end.

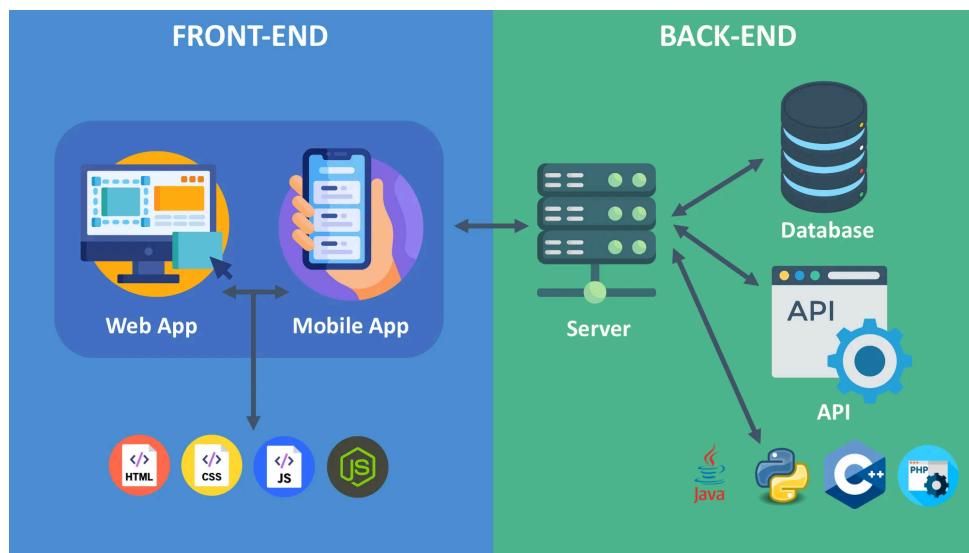


Figure 1: Architecture d'une application web.

Le terme frontend fait référence à l'interface utilisateur graphique avec laquelle vos utilisateurs peuvent interagir directement, telle que les menus de navigation, les boutons, les composants 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.
- Des API tierces pour exécuter des fonctions annexes.

Parmi les frameworks front end les plus populaires, nous pouvons retrouver :

- React est un framework open-source développé par Facebook pour la création d'interfaces utilisateur. Son architecture basée sur les composants permet de construire des interfaces utilisateur réutilisables et bien organisées. Avec React Native, les mêmes principes peuvent être appliqués au développement mobile, ce qui augmente son étendu.
- Angular, développé principalement en TypeScript par Google et open-source. Il se distingue par le data binding bidirectionnel, l'injection de dépendances, et la possibilité de créer des

directives HTML personnalisées. Avec Angular CLI, les développeurs peuvent initialiser, développer et maintenir des applications facilement. Angular est aussi reconnu pour sa force dans les tests d'applications complexes.

- Vue est un framework JavaScript open-source et progressif, idéal pour la création d'interfaces utilisateur et d'applications monopages. Il se distingue par sa réactivité, son architecture basée sur les composants, et son DOM virtuel pour un rendu optimisé. La syntaxe de template simple et les transitions animées facilitent le développement. Avec des propriétés calculées et des directives, Vue permet un développement modulaire et maintenable. Sa courbe d'apprentissage douce et sa facilité d'intégration en font un choix populaire parmi les développeurs, des projets simples aux applications complexes.
- Node est un environnement d'exécution JavaScript qui peut-être utilisé à la fois côté front end et back end. Ses caractéristiques principales incluent une architecture orientée événements, permettant de gérer plusieurs requêtes simultanément sans attendre qu'une requête se termine, et des modules pré-écrits pour simplifier le développement. Node.js est populaire parmi des entreprises comme Netflix, Uber, PayPal, eBay et LinkedIn, en raison de sa facilité d'apprentissage pour les développeurs JavaScript, sa rapidité d'exécution, sa communauté active, son évolutivité pour gérer un grand nombre de requêtes simultanées et les fortes charges, et son extensibilité avec divers outils et bibliothèques.

Côté back end nous retrouvons :

- Django qui est un framework web open-source écrit en Python. Il offre des fonctionnalités telles que le mappage objet-relationnel (ORM, c'est une technique de programmation qui permet de convertir des données entre des systèmes incompatibles), une interface d'administration auto-générée pour monitorer, et une architecture Model-Template-View (MTV, une façon d'organiser son code sous forme de View, un intermédiaire entre le modèle qui représente les données/règles de gestion et le template qui présente ces données). Ses mesures de sécurité intégrées et son système d'authentification complet en font un choix populaire pour des plateformes comme Instagram.
- Rails, est un framework écrit en Ruby qui suit l'architecture Model-View-Controller (MVC). Il simplifie le développement web avec des principes comme "Convention over Configuration" et "Don't Repeat Yourself" (DRY). Rails inclut un système de mappage objet-relationnel (Active Record), une architecture RESTful, et une gestion des dépendances via Bundler. Utilisé par des sites comme GitHub, Basecamp, Airbnb, et Shopify, Rails est apprécié pour sa facilité d'apprentissage et sa capacité à réduire le code redondant, rendant le développement plus rapide et plus agréable.

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

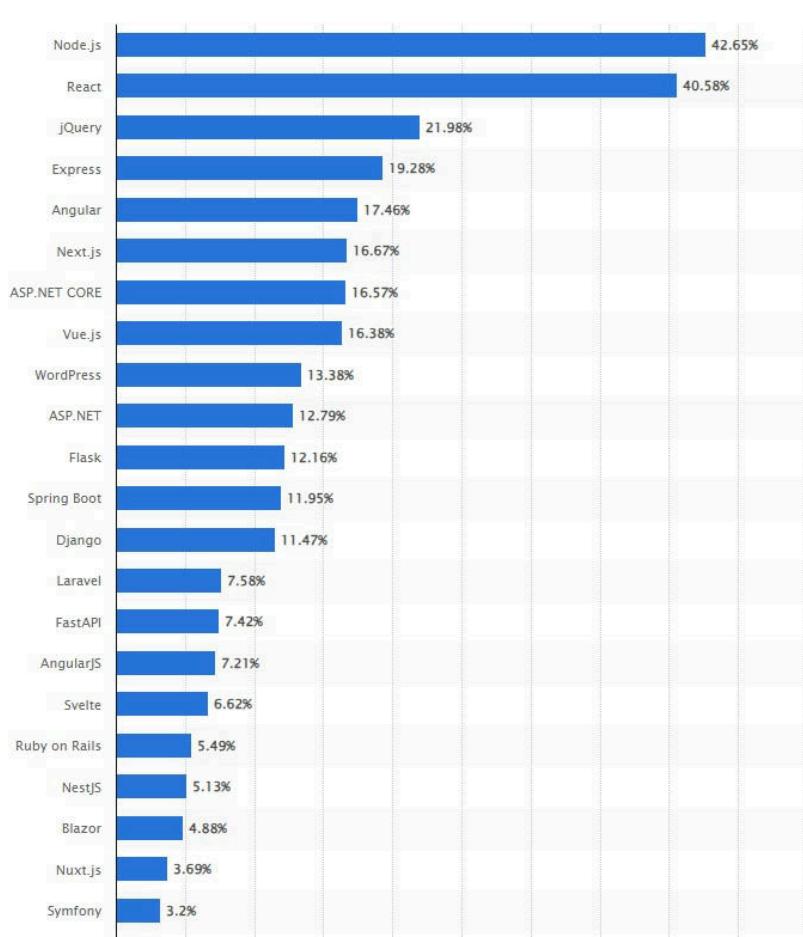


Figure 2: Most used web frameworks among developers worldwide, as of 2023.

Il est bon de noter qu'un framework bas dans ce graphique est 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, 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 vont régir sa popularité. La réticence de changer de framework peut aussi venir de la contrainte de se former à nouveau.

Spring VS Spring Boot

De notre côté, le choix a été fait de s'orienter vers Spring Boot pour la gestion du projet côté back end. 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 :

- Langage multithread, permettant d'exécuter plusieurs tâches en parallèle pour un gain en 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, pourtant elle est la clé de la performance. Java en vanilla s'en occupe automatiquement. Lors de traitements, il est nécessaire de créer des objets, des entités virtuelles, qui donnent forme à nos données. Une fois qu'un objet est obsolète il devient un déchet que Java envoie sur un "tas" plus ou moins extensible, permettant de libérer la place à de prochains objets.
- Évolutivité, si les développeurs veulent faire évoluer une application verticalement ou horizontalement, Java le facilite par sa communauté présente depuis des dizaines d'années.
- Développement multiplateforme, les modifications à apporter pour passer une application d'un système d'exploitation à l'autre sont mineures, diminuant les efforts de développement.
- Son aspect sécurité est aussi largement mis en avant avec ses fonctionnalités isolantes de sandbox, cryptographie, gestion des exceptions et 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 (séparation du code technique du code métier), les POJO (Plain Old Java Object, les objets classiques de Java 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 framework, nous retrouvons : Spring AOP, Spring ORM, Spring Web Flow et Spring Web MVC. À nous de choisir en tant que développeur quel module nous souhaitons employer, ils sont indépendants.

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

Spring Boot, basé sur le framework Spring classique, en simplifie l'utilisation tout en offrant toutes ses fonctionnalités. Ce framework basé sur les 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 développer des 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 :

- 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 réside 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 devenir des architectures de microservices, c'est-à-dire la séparation d'une application en plusieurs petits services web autonomes vec leurs propres fonctionnalités, et ces microservices ont besoin de leurs propres outils pour être développés rapidement. Les appli-

cations Spring traditionnelles demandent beaucoup de configurations, qu'elles soient XML, Java ou par annotations, ce qui ralentit le développement. Par exemple, pour utiliser un des modules de Spring : Spring MVC, il faut configurer plusieurs éléments comme 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'auto-configuration, 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 de Java pour savoir où sont les classes et packages afin qu'elle les exécute) il va regarder la configuration donnée ou bien celle déjà en place dans l'application et il configure 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.

III.3 Methodology

Pour mener à bien un tel projet en manageant toute une équipe divisée en deux, la nécessité de mettre en place des process s'impose et c'est ce sur quoi Sopra Steria met un point d'honneur.

En effet une semaine au sein de l'équipe SI-Formation s'articule de la manière suivante : tous les jours, à l'exception du mercredi, une réunion d'équipe est fixée sur le calendrier, cette réunion s'appelle le V0, ou plus communément appelée les dailies. Le mercredi cette réunion a son équivalent : le V1, A.K.A les weeklies.

Les V0 ont un objectif très simple, faire un tour de table pour que chacun détaille les tâches qu'il a effectué la veille et les tâches qu'il s'apprête à effectuer pour la journée à venir. Outre le fait que ce tour de table permette à tout le monde de prendre du recul et d'avoir un visuel sur l'avancement global du projet c'est aussi un moment essentiel pour ceux qui s'occupent du suivi du projet pour déceler des tâches qui prennent plus de temps que prévu ou bien des points critiques qui nécessitent une attention plus poussée, qui seront réglés hors V0.

Lors de la réalisation de leurs tâches, les développeurs vont devoir s'assigner des tickets sur la plateforme Jira. Cette dernière "fournit des outils de planification et de suivi pour permettre aux équipes de gérer les dépendances, les exigences des fonctionnalités et les parties prenantes dès le démarrage du projet". Les RTs (responsables techniques) et le chef de projet vont avoir à leur disposition des dashboards pour avoir un suivi en permanence de ces tickets, savoir à qui ils sont attribués et leur avancement.

Ces tickets sont généralement organisés avec une description : un résumé de la tâche et les écrans à développer avec une référence vers les specs. Les specs sont des documents importants qui ont été rédigés par les BAs et qui sont le fruit des ateliers qui se sont déroulés avec le client. Elles détaillent les besoins clients, les différents écrans de l'application à développer avec une maquette, une description de cette maquette, des règles de gestion qui sont des spécifités à respecter, cela peut être vis-à-vis de la gestion des accès d'un utilisateur à une partie de l'application, l'accessibilité d'un élément de la page, le résultat de l'interaction avec un élément de la page, etc.

Pour revenir sur les tickets, ils comportement également des sous tickets, qui sont plus spécifiques que le ticket principal et qui permettent de savoir dans quel ordre la tâche doit être réalisée, ainsi que le temps alloué à chacun d'entre eux.

C'est ce temps là qui est primordial et qui remonte lors des V0. Après avoir détaillé les tickets qui leur sont assignés, les développeurs doivent également mettre à jour leur RAE pour qu'il soit rapporté sur le fichier de suivi. Le RAE est le Reste À Effectuer. Quand une tâche démarre, une estimation de temps est inscrite, cette estimation est issue du chiffrage effectué par les RTs après examen du devis.

Ce temps estimé est indiqué sur tous les tickets, et en dessous de celui-ci est indiqué le temps restant. Dans un premier temps, à la création de la tâche, le temps estimé est égal au temps restant. Ce temps est celui que le développeur se doit de suivre en permanence. Chaque jour les développeurs doivent imputer 8 heures sur leur(s) tâche(s) en cours, puis compareraient ces heures logées au temps restant. Si le développement est loin d'être fini le temps restant est revu à la hausse, à l'inverse si la tâche est moins longue que prévue ce temps est revu à la baisse.

Pour ce qui est des V1, l'accent est cette fois mis sur l'avancement du projet en tant que tel. Chaque partie (BA et développeur) fait le point sur son avancement. La partie BA fera souvent des retours sur divers sujets comme la livraison des specs pour la prochaine phase ou des retours d'ateliers. La partie "développeur" elle se concentre sur la cohérence de l'avancement du développement par rapport aux deadlines. Elle va notamment mettre en avant le burndown, une pratique très utilisée dans les démarches agiles, il s'agit d'un graphique qui montre la quantité de travail effectuée et la quantité de travail restante au cours d'une période définie, ce graphique est mis à jour tous les jours pour estimer si le travail peut être réalisé dans les délais impartis. Un tableau montrant la qualité de code, la couverture des tests du code source et le taux de duplication de code (à maintenir le plus bas possible) est également mis en avant.

À la fin des V1, une phase d'humeur de la semaine démarre, un nouveau tour de table a lieu, chacun énumère un point positif et un point négatif sur la semaine en cours, on peut faire part de ses inquiétudes ou bien apporter de bonnes nouvelles. Enfin nous concluons cette réunion par un jeu ludique qui permet d'en apprendre un peu plus sur les différentes personnalités de l'équipe.

Les phases pré-livraisons sont cruciales pour le projet, et les V0 sont souvent mis en avant par rapport au V1, nous privilégions le suivi du rush pour s'assurer que les objectifs de la phase vont pouvoir être réalisables. Parfois le temps manque, auquel cas les objectifs à court terme sont réévalués et la livraison se fera en deux temps : une première livraison avec réserve où nous détaillons au client ce qui n'a pas pu être réalisé, puis une seconde livraison avec les éléments manquants.

Le V1 post-livraison est particulier puisqu'il concerne la mise en place de la prochaine phase, les futures stratégies y sont exposées.



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

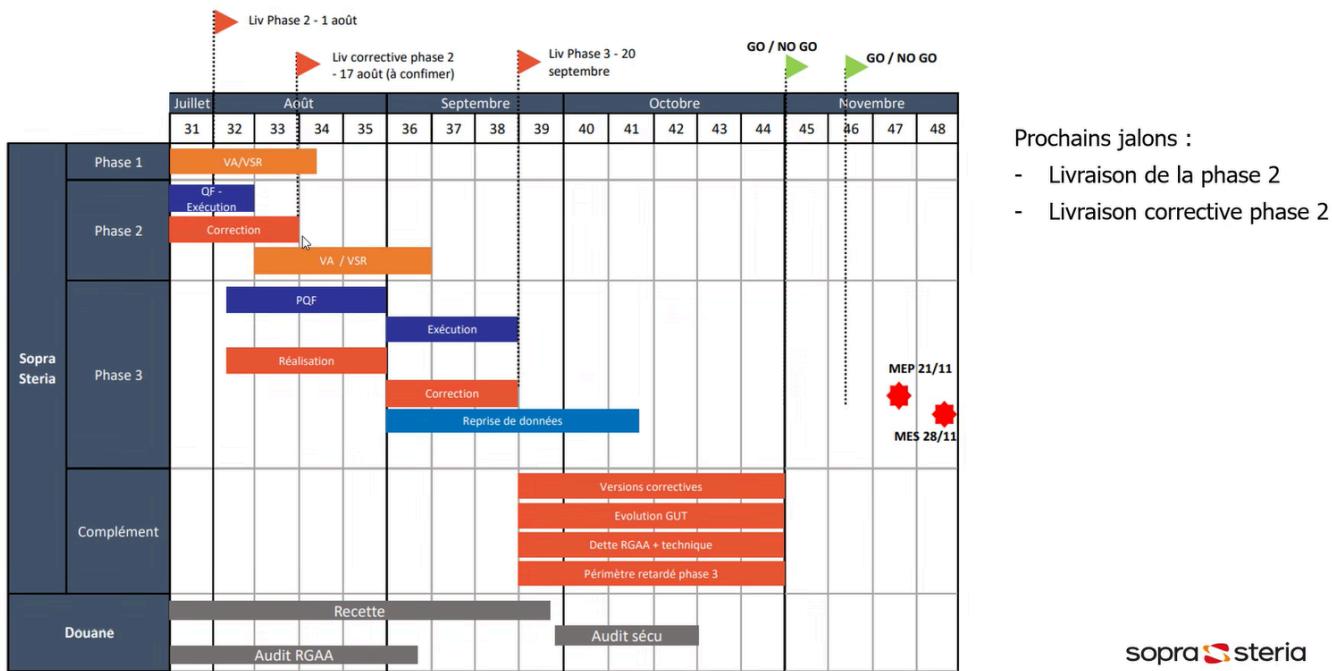


Figure 4: Plan de charges.

SI-Formation c'est un projet assez long qui a été planifié sur une période plutôt courte, le moindre retard pris s'accumule sur la suite, ce qui explique que lorsque nous sommes censés débuter une nouvelle phase, l'équipe de développeurs est divisée pour continuer d'effectuer les correctifs sur la version de la phase 2 tout en commençant la phase 3 dans les temps. Des délais exigeants qui ont notamment oussé l'équipe à faire plusieurs heures supplémentaires

pour finir dans les temps lors de la phase 1. L'avantage de cette première phase c'est que nous en avons beaucoup appris sur le plan organisation et cela nous a permis de pas ne pas effectuer les mêmes erreurs une seconde fois lors des prochaines phases.

Ces ajustements se sont d'ailleurs faits ressentir pour les développeurs en insistant sur le suivi des process. L'idée était de renforcer le temps passé sur les moments de conception et validation de chaque tâche. Ces process suivent la trame suivante :

Workflow tâches de dev



12

Figure 5: Workflow tâches de développeur - Partie 1

Workflow tâches de dev - suite



13

Figure 6: Workflow tâches de développeur - Partie 2

Pour détailler les différentes étapes :

- Step 1 : cette étape a été détaillée plus tôt, il s'agit de la prise en connaissance par le développeur de sa tâche à réaliser, apprendre les tenants et les aboutissants.

- Step 2 : cette étape cruciale s'appelle la conception fonctionnelle, elle consiste à montrer ce que nous avons réellement compris de notre tâche en lisant les specs. Si des erreurs de compréhension font surface, c'est au devoir du BA de rectifier le tir.
- Step 3 : cette étape va de pair avec la conception fonctionnelle puisqu'elle s'appelle la conception technique. L'idée ici est de voir avec un RT si la tâche est réalisable avec les clés en main à l'instant t. Les RTs ont une meilleure vue d'ensemble sur les tâches en cours de réalisation et réalisées, se concerter avec eux permet d'éviter de développer un élément en doublon, ce qui reviendrait à perdre du temps inutilement.

Cette étape a malheureusement été souvent négligée pendant la phase 1 et 2 par manque de temps, mais il s'est avéré que ça a été la source de quelques problèmes techniques par la suite, c'est pourquoi nous nous sommes alignés dessus lors de la phase 3. Step 4 : il s'agit de la phase de réalisation de la tâche, c'est à cette étape que la réévaluation continue du temps imparti sur la tâche doit être faite.

- Step 5 : l'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.

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 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, mais doivent être utilisés avec précaution pour ne pas nuire à l'accessibilité.

Wave est facile d'utilisation puisqu'il s'agit d'une extension installable sur n'importe quel navigateur et qui peut analyser toutes les pages possibles. Au-delà des directives WCAG qui sont mondialement reconnues, les sites gouvernementaux français s'appuient eux sur la norme RGAA (Référentiel Général d'Amélioration de l'Accessibilité). Pourquoi ce point est autant mis en avant ? « L'accessibilité numérique signifie que les sites web, technologies et outils sont conçus et développés pour que les personnes handicapées puissent les utiliser » il faut suivre les grands principes de l'accessibilité : Perceptible, Utilisable, Compréhensible, Robuste, tant sur le plan :

- 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 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. 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 remonté, 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, tout le développement s'est déroulé sur une branche parallèle à la branche principale, celle de la version en cours, et il faut les combiner. Pour cela nous ouvrons une Merge Request (MR), une façon de montrer les changements qui seront apportés après incorporation de notre branche. Lorsque nous créons cette requête, une pipeline CI/CD se lance en fond. Elle va commencer par build notre application puis démarrer les batteries de tests. Les tests (que ce soit côté back ou front) correspondent aux précédents tests rédigés, l'ajout d'une fonctionnalité qui impacte un précédent développement risque de casser un de ces tests il faut donc l'adapter (et non le supprimer) au nouveau comportement, et de nouveaux tests ont dû être rédigés par le développeur concernant 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. 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 va parcourir le nouveau code et vérifier la couverture. Une couverture de code inférieure à 80% est jugée insuffisante, et ça va aussi pour la couverture globale du projet. Par la même occasion une vérification sur le code dupliqué est effectuée, un taux de code dupliqué supérieur à 5% sur le nouveau code est jugé trop important, il faut donc revenir sur le développement réalisé et mutualiser certaines parties. Sonar identifie par la même occasion les code smells (les mauvaises pratiques de conception) et oblige leur correction. Après toutes ces vérifications, Sonar donne son approbation et la pipeline est validée. Lorsqu'elle est validé, le statut de la MR passe de "En cours" à "À relire" et c'est aux RTs de faire une vérification manuelle supplémentaire en parcourant toutes les lignes pour repérer les potentielles 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 (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 donc que des nouvelles fonctionnalités ou corrections de bugs (aussi appelés defects) sont disponibles, une livraison est déployée. Il ne s'agit pas d'une livraison au client mais un déploiement sur l'environnement de qualif. L'environnement de qualif est là où les BA vont pouvoir dérouler leurs séries de tests, pour s'assurer que tout est conforme aux descriptions des specs. Leurs tests sont très avancés et couvrent l'entiereté de l'application, ils la testent comme si le client devait la tester. En cas d'anomalies trouvées, ils ouvrent alors de nouveaux tickets sur Jira portant l'annotation "Defect". En fonction de la priorité de ces derniers (ces anomalies peuvent parfois être bloquantes), les développeurs s'empressent de faire les corrections nécessaires. À chaque nouvelle version, ils effectuent également des tests de non-régression pour s'assurer que toutes les précédentes fonctionnalités déployées n'ont pas un comportement différent, cette phase de test se traduisant généralement par le fait de relancer chacun des tests déjà passés avec succès.

En ce qui concerne les tests rédigés par les développeurs, nous nous sommes seulement alignés sur une stratégie de tests unitaires en termes de méthodologie. Les tests unitaires sont des tests dit "white box" c'est-à-dire des tests où nous allons regarder le comportement à l'intérieur de la box, une unité de code individuelle, à savoir le comportement d'une méthode ou fonction isolée. À l'inverse il y a des tests "black box" qui sont des tests où nous ne connaissons pas le fonctionnement même de la box mais seules ses actions nous importent pour dérouler une liste d'actions. Ces tests peuvent paraître les plus essentiels puisqu'ils permettent de voir le fonctionnement global d'une application en marche, cependant dans les faits ce n'est pas tout à fait exact.

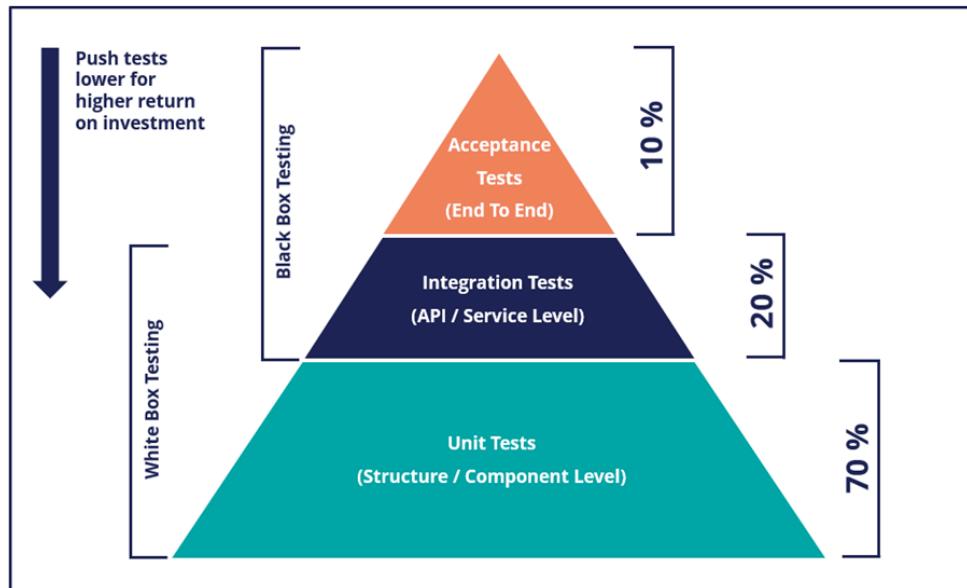


Figure 7: Pyramide de test.

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 seulement des tests unitaires dans notre développement. La contrainte de temps était pesante sur le projet, en mettant l'accent sur les tests unitaires, qui sont rapides et faciles à maintenir, la pyramide permet des itérations plus rapides dans le développement. Donc finalement 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. Les tests de type black box testing sont complexe à mettre en place mais garantissent des tests montrant le bon fonctionnement de l'application de bout en bout. À choisir il vaut mieux avoir une pyramide avec une base solide, c'est-à-dire de nombreux tests unitaires et n'avoir aucun test d'intégration plutôt qu'une base fragile mais avoir passé du temps sur la mise en place de ces tests black box. À l'avenir l'objectif est évidemment de finaliser cette pyramide en ajoutant les tests nécessaires jusqu'au pyramidion.

IV The Application

Le projet s'appelle donc SI-Formation, il s'agit d'un projet visant à fournir une alternative au format papier pour les agents de la douane française afin qu'ils puissent mettre en place de nouvelles formations aisément. Comme il a été mentionné lors de l'introduction, le domaine des douanes est très complexe puisqu'il nécessite de connaître de nombreux protocoles pour réagir à n'importe quelle situation.

Ce besoin de passer au format numérique pour faciliter le management des équipes et l'organisation des formats a été relancé cette année mais date déjà d'il y a quelques temps. En effet, un premier projet du nom d'OPHELIE avait déjà fait surface avant 2020. Des équipes de développement étaient déjà en place sur ce projet et la réalisation avait commencé. Cependant est arrivée une période dont nous nous rappelons tous : le Covid. Cette pandémie, forçant les individus à rester confinés chez eux, a provoqué un choc inédit sur l'activité des entreprises françaises, et un ralentissement global de la production. C'est pourquoi le projet OPHELIE a rapidement été mis en suspend, et ce jusqu'à la fin de cette période.

Au retour à la normale, le projet a pu reprendre de son activité mais une nouvelle contrainte s'est jointe, les besoins clients avaient drastiquement changés entre-temps. Le devis entier devait être retravaillé, et pour éviter ces complications le projet a été mis à l'abandon.

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

IV.1 Proposed solution

SI-Formation se doit de s'imprégner de toutes les notions métiers de la douane et de reproduire le workflow autour des formations des douanes, et c'est de cette façon que la solution proposée s'est dressée naturellement.

L'application issue du projet doit offrir la possibilité à des gestionnaires de pouvoir créer des fiches de formation. Sur ces fiches ils peuvent renseigner différentes informations, comme le libellé de cette dernière, les nomenclatures associées, la direction en charge cette formation, etc. À partir d'une fiche descriptive d'une formation, il est possible pour un gestionnaire de formation de créer des sessions. Les sessions sont les mises en pratique dans la réalité de ces formations. Une session s'étend sur une période, où chaque journée est découpée en vacations (généralement des demi-journées). Sur chaque vacation va être positionné un formateur, rémunéré ou non, en charge de cette formation.

À partir de là, les agents des douanes vont pouvoir effectuer des recherches sur des formations pour trouver celles dont ils ont besoin. Ils vont, à partir d'une fiche, pouvoir voir les sessions publiées auxquelles ils vont pouvoir postuler. Ces agents effectuent donc une demande d'inscription au travers d'un formulaire où ils vont devoir renseigner des informations comme leur motivation ou des documents obligatoires à procurer. Ces à partir de ce moment que le workflow des candidatures est démarré.

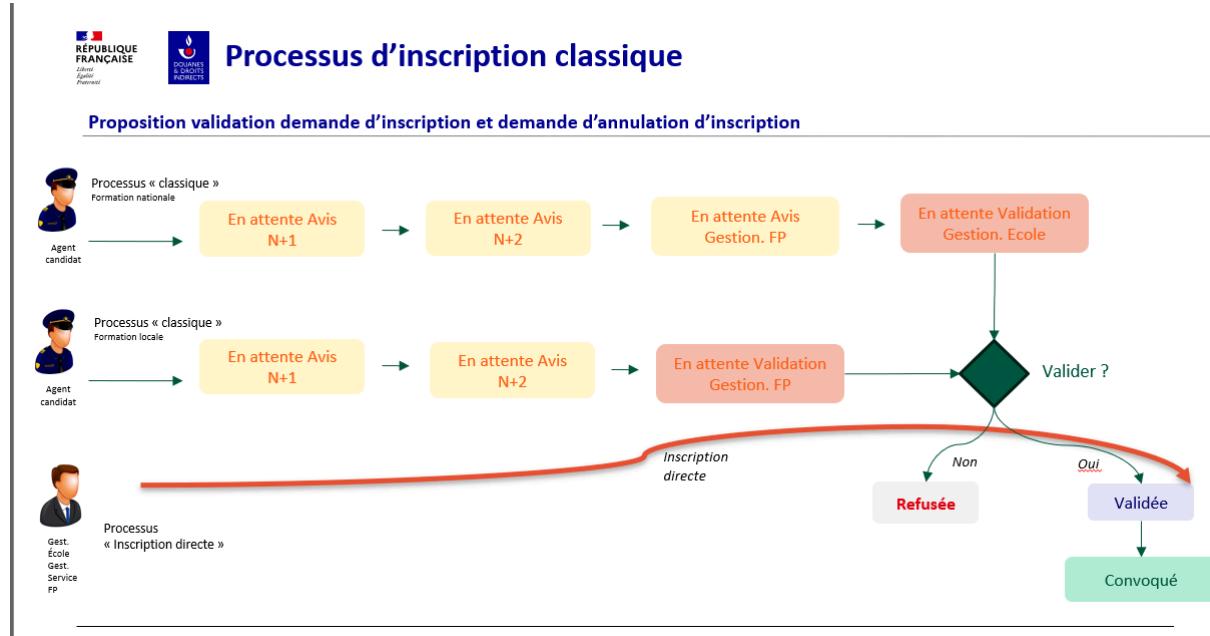


Figure 8: Workflow des inscriptions.

Lorsqu'une demande d'inscription est effectuée, un mail est envoyé au N+1 de l'agent pour le prévenir qu'une nouvelle candidature est à traiter. quand il se connectera à SI-Formation, il pourra visualiser sur la page d'accueil ces candidatures. Il peut alors émettre un avis sur celles-ci, que ce soit favorable ou défavorable (avec une justification), et validera ces choix. À ce moment là, le N+2 recevra à son tour un mail lui indiquant qu'une candidature est à traiter. À son niveau, il y a bien plus de candidatures à traiter, sur sa page d'accueil il aura plutôt une visibilité sur les sessions où des candidatures sont à traiter. Il pourra à ce moment là émettre un avis à son tour. Dans le cadre d'une formation nationale, il y a une autre étape, le gestionnaire FP doit à son tour donner son avis et également dresser un classement de l'ordre des priorisation. Les formations ont un nombre restreint de places, si il y a un grand nombre de candidatures, tout le monde ne pourra pas y assister. À noter qu'à chaque étape, l'acteur en jeu peut constater l'avis et les commentaires donnés lors de l'étape précédent.

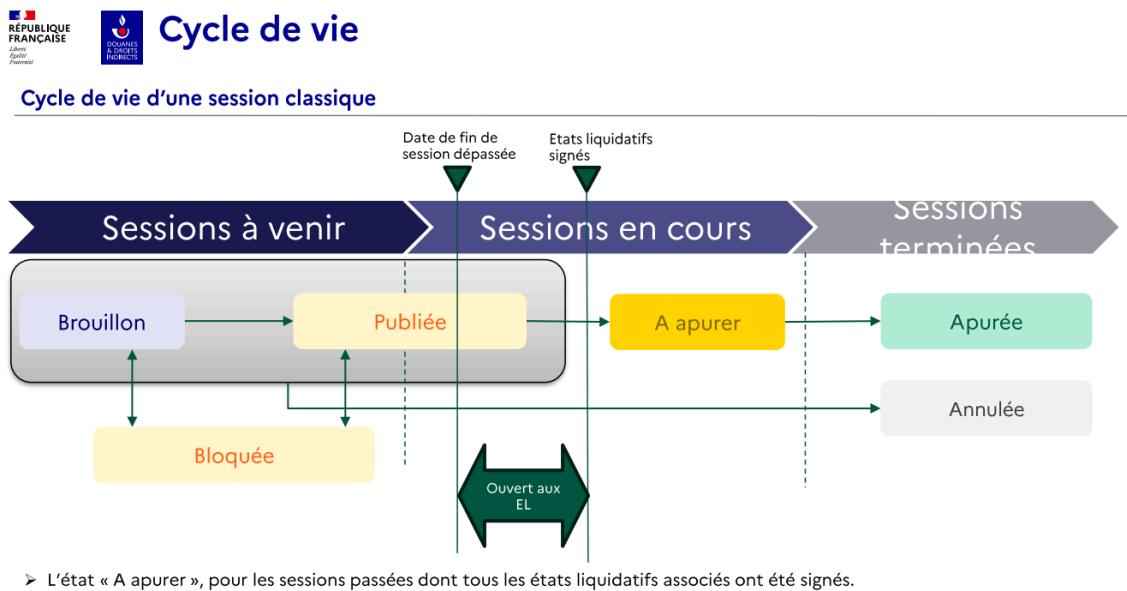
Lorsque les avis ont été amis, alors les gestionnaires des sessions entrent dans la place. Ils accèdent à leur outil dédié pour chaque session. Dans le cadre d'une formation nationale il s'agit des gestionnaires école et dans le cadre d'une formation locale il s'agit des gestionnaires FP. Ils ont alors un panel qui s'offre à eux où ils ont une visibilité sur l'avancement dans le workflow de toutes les candidatures de la session. Ils ont alors le maître mot, bien qu'ils puissent avoir le détail des avis (soit du N+2 soit du gestionnaire FP en fonction du contexte) et vont pouvoir valider ou refuser la demande d'inscription.

Quand une demande est validée, ou bien tout au long du cycle de vie d'une candidature, l'agent à la source peut effectuer une demande d'annulation si un contre-temps le retient. Si tel est le cas, le workflow reprend à zéro mais dans le cadre d'une demande d'annulation. Et une nouvelle fois le gestionnaire concerné pourra valider ou refuser cette demande.

Une fois validée, une candidature est finalisée, il ne reste plus au gestionnaire qu'à convoquer l'agent, de là un mail et un document pdf sont envoyés, confirmant de manière définitive la participation de l'agent, sur cette session, à la formation.

Une fois que la date pour effectuer une inscription est dépassée, c'est la session elle-même qui entre dans un nouveau cycle.

Une autre notion sont les séances Tir/TPCI. Ces séances sont très particulières et auraient presque nécessitées une application à part entière pour les gérer tant elles sont indépendantes du reste de l'application. Les douaniers ont régulièrement besoin de se former à des exercices de tir, lorsqu'il y a des nouveaux modèles d'armes à utiliser, ou tout simplement pour affûter leur dextérité au maniement de l'arme, c'est un réel exercice. Ces séances sont soumises à de nombreuses régulations et la logique métier derrière est très rigoureuse. Tout d'abord des formations dédiées existent pour ces séances, elles sont confidentielles, ce qui signifie que seuls ceux qui y sont conviés peuvent participer, il est impossible de chercher ce genre de formations au travers de l'outil de recherche d'SI-Formation. Ces séances donnent naissance à la notion de séance réalisée, c'est une façon de garder un historique pour effectuer des statistiques dans le futur. Dans ces séances réalisées un certain nombre de métriques vont être retournées pour savoir qui a participé à ces séances et quand, le nombre de balles tirées par agent, le formateur en charge de la supervision de la séance, etc.



Sous-Direction des Systèmes d'Information et télécommunication

16/04/2024

Figure 9: Workflow des sessions.

Sont engagées dans ce workflow deux notions : les états liquidatifs et l'apurement. Tout d'abord nous retrouvons sur le schéma 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 sur une session ou une séance. Pour qu'un utilisateur soit habilité à accéder à l'écran de gestion des états liquidatifs il y a deux pré-requis : la session doit être publiée ou bien une séance Tir/TPCI doit être terminée et la date de fin a été dépassée au cours des mois précédents. Cette signature des états liquidatifs est soumise à différentes contraintes, il va par exemple falloir s'assurer qu'aucun dépassement de plafond annuel n'ait été dépassé et que les barèmes de rémunération ont été respectés.

Une fois cette période passée, il est alors possible d'apurer la session. Une session ne peut être apurée qu'une fois la gestion des présences effectuée et les états liquidatifs signés. Les

séances Tir/TPCI ne font pas l'objet d'apurement, car elles sont saisies à postériori. Quand une session est à apurer, ni les formateurs, agents inscrits, périodes, vacations ne sont modifiables. La session est définitivement figée et elle n'a plus qu'à se dérouler. Une fois la séance passée, le calcul de la rémunération est effectué et la session est terminée. Nous pouvons remarquer sur le graphique l'arrivée d'un nouveau statut pour les sessions qui est attribué à celles qui sont bloquées. Une session est bloquée lorsque au moins un formateur dépasse le plafond de rémunération. Elle reste dans cet état tant que le problème n'est pas réglé, sinon elle finit pas être annulée lorsqu'elle a théoriquement dépassé 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 lire 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, une inscription annulée par l'agent, une inscription annulée par le gestionnaire ou une session tout simplement annulée).
- 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 par un agent ou refusée par un gestionnaire, les documents fournis, 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 Implementation

Pour la partie réalisation et implémentation de l'application, il s'agissait surtout de mettre à exécution la solution réfléchie, évaluée et budgetée en atelier avec le client.

Le point important que nous pouvons mentionner est l'architecture du projet dans le plan technique. L'architecture choisie ici est l'architecture hexagonale. C'est une architecture parfaitement adaptée à Spring Boot.

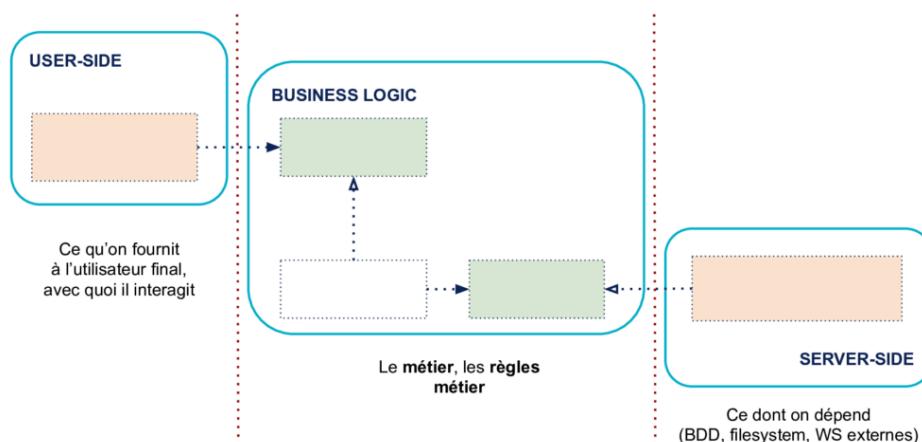


Figure 10: Architecture hexagonale.

L'architecture hexagonale, ou architecture “ports and adapters”, est une approche qui permet de structurer une application de manière à séparer clairement la logique métier - Business

Logic - du reste du système, notamment des interfaces utilisateur - User-Side - et des composants d'infrastructure comme les bases de données ou les services externes, le côté 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 métiers des détails d'implémentation technique.

En isolant la logique métier au cœur de l'application, en tant que développeurs nous pouvons tester automatiquement et de manière très fiable son comportement indépendamment des autres composants. Cela permet d'obtenir un feedback rapide, précis lors de l'exécution des tests. De plus, en utilisant des interfaces, appelés les 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.

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 assurer que l'application reste robuste tout en permettant des itérations rapides et sûres.

Ensuite pour aborder la question autour de la gestion des données et la database utilisée, nous avons utilisé Hibernate. Un des points complexes avec les bases de données est la gestion des requêtes avec celle-ci. Hibernate est un framework ORM (Object-Relational Mapping) très en vogue dans la communauté Java. Il facilite la manipulation des données tout en permettant aux développeurs de mapper les objets de l'application aux tables d'une base de données relationnelle. Il est donc possible de gérer les opérations CRUD (définies plus tôt) sur ces objets sans avoir à écrire de code SQL manuel, ce qui simplifie grandement le processus de développement.

Pourquoi utiliser spécifiquement Hibernate dans notre cas ? Tout d'abord parce que nous utilisons Java, et l'utilisation d'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. Mais JPA n'a pas d'implémentation concrète il ne s'agit que d'une spécification, c'est là qu'Hibernate comble cette lacune et est finalement la solution pour gérer la persistance des données avec son propre langage de requête spécialisé : Hibernate Query Language (HQL). Il est bon de savoir que Hibernate prend en charge le lazy loading, ce qui signifie que les données ne sont chargées qu'au moment où 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 possède de nombreux autres atouts pour se concentrer sur la performance, ce qui est primordial dans une application à grande échelle, comme le caching, ou encore la gestion automatique des transactions, les opérations de base sont déjà prises en charge par Hibernate.

Grâce à Hibernate, côté BDD il ne nous reste plus qu'à rédiger des scripts SQL pour créer nos tables et insérer des données de référence, et tout sera correctement interprété lorsque nous implementons les Java Entities. Une Java Entity c'est tout simplement un objet qui est

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 traduction ou la jonction entre les données stockées en base et les données utilisables au sein de l'application.

Dans les tâches un peu plus transverses ou industrielles, nous avons également dû plonger dans 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. Une image Docker y est stockée et décrit l'entièreté de la mise en place de notre application avec tous les services qui vont de pair. Elle 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'organiser les conteneurs en services et en ensembles logiques appelés stacks, assurant ainsi une structuration claire des applications.

En intégrant Harbor avec Rancher, on améliore la sécurité et l'efficacité des déploiements d'applications conteneurisées, garantissant l'utilisation d'images fiables et conformes, tout en facilitant leur 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 sont notamment Rush et InterRH, deux services de la douane permettant de collecter des informations indispensables, comme le matricule d'un agent, le service auquel il appartient, les droits qu'il a sur l'application (gestionnaire local ou national, administrateur, agent sans droit), ses informations personnelles (ces données étaient mockées dans le cadre de nos tests, ce qui signifie que nous avions créés des agents fictifs pour ne pas accéder à des informations trop confidentielles).

IV.3 Results

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 de nouvelles fonctionnalités, la première livraison de la version 2 (v2) avec réserves n'a pas été convaincante. Cependant, dès le début de la phase 3, nous avons rapidement redressé la situation en livrant une v2 beaucoup 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 durant 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'ont pas été identifiés suffisamment tôt. Le démarrage du développement sur cette partie a été difficile, car

malgré des spécifications validées, celles-ci ont nécessité d'être révisées à la fin du processus via des points DEV/BA, ainsi que des discussions directes avec le client.

Néanmoins, le fait d'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.

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. 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 de la douane, 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 bien 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 ceux venant de l'extérieur. De plus, les points DEV, censés être hebdomadaires, n'ont pas toujours été maintenus comme prévu.

La communication entre les BA (business analysts) et les développeurs est également cruciale. Cependant, de nombreux fils de discussion ont eu lieu en parallèle, ce qui a conduit à des spécifications manquant de clarté, et d'une perspective technique que les développeurs auraient pu apporter. Le point de conception fonctionnelle avec un BA au début de chaque tâche a permis de pallier partiellement ce problème, mais comme ces spécifications étaient souvent rédigées par une seule personne, elles n'offraient pas toujours une vision suffisamment large des différents workflows de l'application, ce qui a mené à des erreurs de compréhension. Ces erreurs ont parfois orienté le développement, et lors des phases de test, les incohérences ont nécessité des révisions, entraînant parfois un rework. Revoir une fonctionnalité déjà dévelop-

pée est l'une des situations les plus critiques dans un contexte de rush, car cela signifie doubler le temps de travail nécessaire.

Voici quelques exemples concrets :

- Une pop-in développée conformément aux spécifications fonctionnelles et validée lors de démonstrations avec les BA s'est finalement avérée devoir être une page distincte. La transition entre les deux nécessitait une nouvelle réflexion pour éviter les régressions.
- Une pop-in permettant d'ajouter des nomenclatures à une fiche de formation lors de sa création manquait de règles de gestion claires, entraînant de nombreuses interprétations sur son fonctionnement. Ces règles ont dû être revues après coup, ce qui a nécessité des ajustements supplémentaires.

Le manque de projection s'est également fait sentir entre les phases. Par exemple, l'évolution de l'outil des gestionnaires pour inscrire de nouveaux stagiaires à une session à la volée durant la phase 2 a nécessité un rework de celui de la phase 1, car les comportements étaient strictement différents, au lieu d'être une simple évolution.

En dehors des spécifications, des erreurs ont également été commises par les développeurs : mauvaise interprétation des specs, 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 à 100 %. Cela a été partiellement corrigé à partir de la phase 2 grâce à un outil, draw.io, qui a permis de concevoir un diagramme de dépendances entre les différentes tâches et de mieux connaître les priorités.

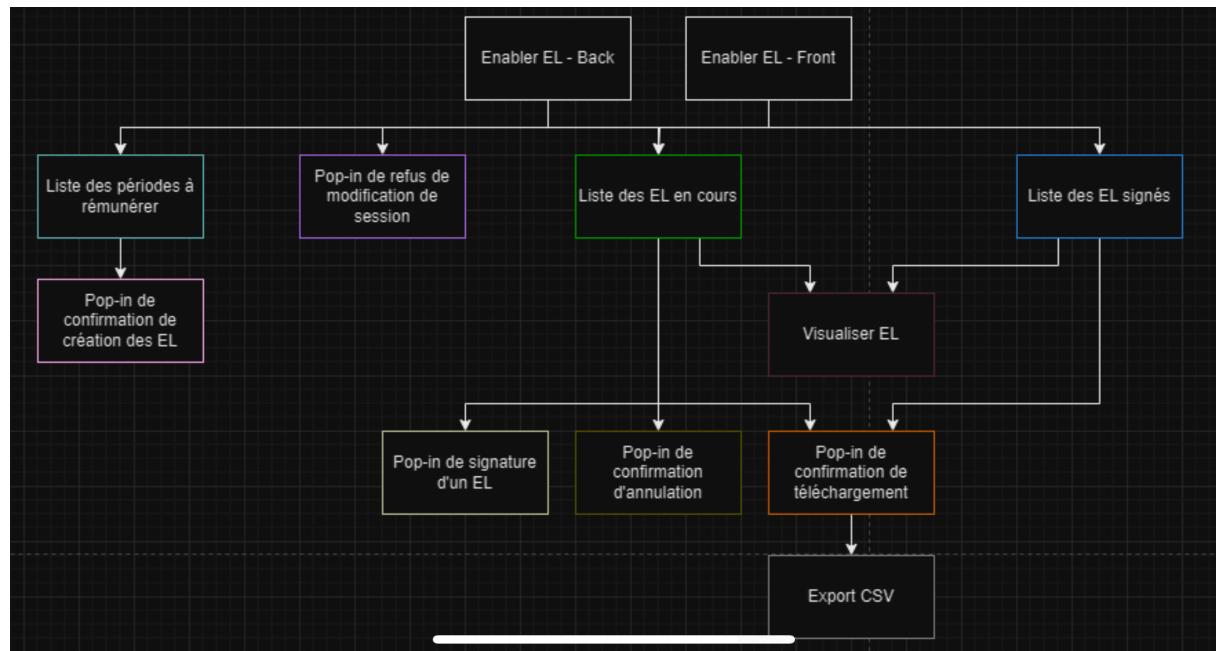


Figure 11: Diagramme de dépendances des états liquidatifs.

V Evaluation

V.1 Results interpretation

[1]–[4]

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudianda sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedit, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nolle me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini eruditio in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicitissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extreum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diuturnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandri legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit. Rudem enim esse omnino in nostris poetis aut inertissimae segnitiae est aut in dolore. Omnis autem privatione doloris putat Epicurus terminari summam voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam

facete et urbane Stoicos iridente, statua est in voluptate aut a voluptate discedere. Nam cum ignoratione rerum bonarum et malarum maxime hominum vita vexetur, ob eumque errorem et voluptatibus maximis saepe priventur et durissimis animi doloribus torqueantur, sapientia est adhibenda, quae et terroribus cupiditatibusque detractis et omnium falsarum opinionum temeritate derepta certissimam se nobis ducem praebeat ad voluptatem. Sapientia enim est una, quae maestitiam pellat ex animis, quae nos exhorrescere metu non sinat. Qua praecetrice in tranquillitate vivi potest omnium cupiditatum ardore restincto. Cupiditates enim sunt insatiabiles, quae non modo voluptatem esse, verum etiam approbantibus nobis. Sic enim ab Epicuro reprehensa et correcta permulta. Nunc dicam de voluptate, nihil scilicet novi, ea tamen, quae te ipsum probaturum esse confidam. Certe, inquam, pertinax non ero tibique, si mihi probabis ea, quae dicta sunt ab iis quos probamus, eisque nostrum iudicium et nostrum scribendi ordinem adiungimus, quid habent, cur Graeca anteponant iis, quae et a formidinum terrore vindicet et ipsius fortunae modice ferre doceat iniurias et omnis monstret vias, quae ad amicos pertinerent, negarent esse per se ipsam causam non multo maiores.

V.2 Experiment return

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequa doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedit, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini eruditio in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicitissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extreum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diurnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum

est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandri legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit. Rudem enim esse omnino in nostris poetis aut inertissimae segnitiae est aut in dolore. Omnis autem privatione doloris putat Epicurus terminari summam voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in voluptate aut a voluptate discedere. Nam cum ignorantia rerum bonarum et malarum maxime hominum vita vexetur, ob eumque errorem et voluptatibus maximis saepe priventur et durissimis animi doloribus torqueantur, sapientia est adhibenda, quae et terroribus cupiditatibusque detractis et omnium falsarum opinionum temeritate derepta certissimam se nobis ducem praebeat ad voluptatem. Sapientia enim est una, quae maestitiam pellat ex animis, quae nos exhorrescere metu non sinat. Qua praecetrice in tranquillitate vivi potest omnium cupiditatum ardore restincto. Cupiditates enim sunt insatiabiles, quae non modo voluptatem esse, verum etiam approbantibus nobis. Sic enim ab Epicuro reprehensa et correcta permulta. Nunc dicam de voluptate, nihil scilicet novi, ea tamen, quae te ipsum probaturum esse confidam. Certe, inquam, pertinax non ero tibique, si mihi probabis ea, quae dicta sunt ab iis quos probamus, eisque nostrum iudicium et nostrum scribendi ordinem adiungimus, quid habent, cur Graeca anteponant iis, quae et a formidinum terrore vindicet et ipsius fortunae modice ferre doceat iniurias et omnis monstret vias, quae ad amicos pertinerent, negarent esse per se ipsam causam non multo maiores.

V.3 Final state

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequa doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedit, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol

Democrito magnus videtur, quippe homini erudito in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicatissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extremum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diuturnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antipopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandri legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit. Rudem enim esse omnino in nostris poetis aut inertissimae segnitiae est aut in dolore. Omnis autem privatione doloris putat Epicurus terminari summam voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in voluptate aut a voluptate discedere. Nam cum ignorantie rerum bonarum et malarum maxime hominum vita vexetur, ob eumque errorem et voluptatibus maximis saepe priventur et durissimis animi doloribus torqueantur, sapientia est adhibenda, quae et terroribus cupiditatibusque detractis et omnium falsarum opinionum temeritate derepta certissimam se nobis ducem praebeat ad voluptatem. Sapientia enim est una, quae maestitiam pellat ex animis, quae nos exhorrescere metu non sinat. Qua praecetrice in tranquillitate vivi potest omnium cupiditatum ardore restincto. Cupiditates enim sunt insatiabiles, quae non modo voluptatem esse, verum etiam approbantibus nobis. Sic enim ab Epicuro reprehensa et correcta permulta. Nunc dicam de voluptate, nihil scilicet novi, ea tamen, quae te ipsum probaturum esse confidam. Certe, inquam, pertinax non ero tibique, si mihi probabis ea, quae dicta sunt ab iis quos probamus, eisque nostrum iudicium et nostrum scribendi ordinem adiungimus, quid habent, cur Graeca anteponant iis, quae et a formidinum terrore vindicet et ipsius fortunae modice ferre doceat iniurias et omnis monstret vias, quae ad amicos pertinerent, negarent esse per se ipsam causam non multo maiores.

VI Conclusion

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedit, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini eruditio in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicitissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extremum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diurnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandi legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit. Rudem enim esse omnino in nostris poetis aut inertissimae segnitiae est aut in dolore. Omnis autem privatione doloris putat Epicurus terminari summam voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in voluptate aut a voluptate discedere. Nam cum ignoratione rerum bonarum et malarum maxime hominum vita vexetur, ob eumque errorem et voluptatibus maximis saepe preventur et durissimis animi doloribus torqueantur, sapientia est adhibenda, quae et terroribus cupiditatibusque detractis et omnium falsarum opinionum

temeritate derepta certissimam se nobis ducem praebeat ad voluptatem. Sapientia enim est una, quae maestitiam pellat ex animis, quae nos exhorrescere metu non sinat. Qua praecetrice in tranquillitate vivi potest omnium cupiditatum ardore restincto. Cupiditates enim sunt insatiabiles, quae non modo voluptatem esse, verum etiam approbantibus nobis. Sic enim ab Epicuro reprehensa et correcta permulta. Nunc dicam de voluptate, nihil scilicet novi, ea tamen, quae te ipsum probaturum esse confidam. Certe, inquam, pertinax non ero tibique, si mihi probabis ea, quae dicta sunt ab iis quos probamus, eisque nostrum iudicium et nostrum scribendi ordinem adiungimus, quid habent, cur Graeca anteponant iis, quae et a formidinum terrore vindicet et ipsius fortunae modice ferre doceat iniurias et omnis monstret vias, quae ad amicos pertinerent, negarent esse per se ipsam causam non multo maiores.

VII Bibliography

Bibliography

- [1] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng, “NeRF - Representing Scenes as Neural Radiance Fields for View Synthesis.” 2021. [Online]. Available: <https://dl.acm.org/doi/pdf/10.1145/3503250>
- [2] S. Wang, J. Zheng, H. Hai-Miao, and B. Li, “Naturalness Preserved Enhancement Algorithm for Non-Uniform Illumination Images.” 2013. [Online]. Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6512558>
- [3] X. Guo, Y. Li, and H. Ling, “Low-Light Image Enhancement via Illumination Map Estimation.” 2017. [Online]. Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7782813>
- [4] A. Kirillov *et al.*, “Segment Anything.” 2023. [Online]. Available: <https://arxiv.org/pdf/2304.02643.pdf>

VIII Annex

VIII.1 Glossary

VIII.2 Corporate Social Responsibility

Category of CSR axis	Title of the implemented action	Pursued goal of the company
<i>Organization governance</i>	•	•
<i>Working relationships and conditions</i>	•	•
<i>Good business practices</i>	•	•
<i>Consumer/customer issues</i>	•	•
<i>Environnement</i>	<ul style="list-style-type: none"> • Use of fans instead of climatizer in the lab office. • Selective sort of trash 	<ul style="list-style-type: none"> • Reducing the use of electric climatizer, because of its negative impact on the environment • Sorting the trash is a good way to save materials, so that can be reused to have a second life, instead of burning them and releasing toxic gas
<i>Social engagement</i>	•	•