La Plateforme

Rapport d'alternance

Johan Bouguermouh

Promo : Master of Science IT & Business

Période de formation : 01/09/2023 au 31/08/2025

Entreprise : L’ATelier de la Plateforme

Tuteur : Mr ATHOMAS Julien

Étudiant : Mr Johan Bouguermouh

Remerciements

Je tiens à remercier chaleureusement **l’ensemble de l’équipe de La Plateforme**, et plus particulièrement les membres de **l’Atelier**, pour leur encadrement, leur confiance et les responsabilités qui m’ont été accordées tout au long de mon alternance. Cette expérience a été bien plus qu’une simple mise en pratique de mes apprentissages : elle m’a permis de m’impliquer dans des projets concrets à fort impact, d’exercer un rôle à la croisée du développement, de la gestion de projet et de la conception métier, et surtout, de prendre la mesure des exigences d’un environnement professionnel.

Merci à **mes encadrants techniques et pédagogiques**. Notamment :

- Monsieur Julien Athomas, CTO de L'Atelier de la Plateforme, pour la transmission de sa passion, l'éveille et sa pertinence sur la compréhension métier et les enjeux d'un rapport pragmatique entre Développement et concrétisation métier.
- Madame Ana Stanko, Manager et Chef de Projet, pour son rapport humain, son écoute et la transmission de son savoir sur la communication adapté envers les client et les mécanismes liées à la conduite de projet sans jamais brider mon autonomie dont j'avais besoin pour expérimentée et continuer d'apprendre.
- Monsieur Jérôme Savajol, co-créateur de l'Atelier, pour son mentora et sa bienveillance. Sa connaissance approfondie des fonctionnement d'entreprise de la Tech et le regard avertis qu'il a pu me transmettre à travers des discussions passionnantes.
- Je tiens a remercier aussi l'ensemble de l'équipe, dont le recrutement semble avoir fait l'objet singulier d'un attachement particulier events leurs sympathies et leurs bienveillances. Notamment aux encadrants seniors sur les projets avec qui j'ai eu l'honneur de travailler : Madame Fatima Elmouhine et Monsieur Morad Labrid. Merci pour leur disponibilité, leur professionnalisme et leur sympathie tant appréciable.

Je suis également reconnaissant envers mes camarades de promotion, avec qui les échanges, les questionnements et les soutiens mutuels ont constitué un véritable moteur collectif.

Enfin, je souhaite exprimer ma gratitude à **La Plateforme** pour avoir construit un modèle de formation en alternance si proche des réalités du terrain, où la montée en compétence ne se fait pas uniquement par accumulation de savoirs, mais par confrontation constante à des problématiques réelles, dans un cadre bienveillant, stimulant et profondément humain.

Sommaire

Rapport d’alternance..... 0
Remerciements..... 2
Sommaire..... 3
Introduction..... 4
L’entreprise..... 5
Présentation de l'entreprise L'Atelier de la Plateforme..... 5
Recrutement..... 5
Fonction au sain de l'Atelier..... 6
Présentation Projet Technicert..... 7
Choix du projet Technicert..... 7
Context & présentation de l’entreprise..... 8
Qui est Technicert..... 8
Qu'elle est son rôle ?..... 8
Contexte et problématique..... 8
Organisation interne..... 9
Contribution technique au projet Technicert..... 10
Contexte..... 10
Réflexion et solutions envisagées..... 11
Choix de l’environnement backend..... 11
Base de données..... 12
Choix de l'environnement Frontend..... 12
Retour d'expérience concernant le choix de la stack technique..... 13
Architecture & Infrastructure de Déploiement..... 15
Décision liée à l'architecture..... 15
Avantage de l’approche..... 15
Retour d'expérience et limites rencontrées..... 15
Déploiement, maintenance et scalabilité..... 16
Modélisation de la base de données et cartographie des flux de données..... 17
Conception de la base de données..... 17
Méthodologie de modélisation..... 18
Cartographie des flux de données..... 18
Intégration à l'analyse d'impact (AIPD)..... 19
Identification des données sensibles et mesures de protection..... 20
Quelles sont les données personnelles traitées ?..... 21
Principe d’accès grâce à une gestion des rôle..... 21
Mesures de sécurisation mises en place..... 21
Archivage et intégrité des certificats..... 22
Autres garanties..... 22
Planification et coordination du Projet..... 22
Projet SRIAS..... 25
Présentation des enjeux de l'avant-projet..... 25

Présentation de l'entreprise SRIAS..... 25
Qu'est-ce que la SRIAS ?..... 25
Quelle était sa problématique ?..... 25
Phase d'avant-projet..... 26
Appel à projet et phase de développement..... 26
Rôle de Product Owner..... 26
Adaptation du livrable..... 27
Organisation de gestion de projet..... 27
Contribution à la phase de conception et déclaration technique..... 28
Conduite de projet et organisation autour de cérémonies Scrum et rédaction d’artefacts..... 28
Burn Chart du projet SRIAS..... 29
Conclusion de l’expérience..... 30

Introduction

Le rapport suivant constitue le témoignage de mon expérience dans le cadre de mon alternance, réalisée au sein de **l'Atelier de La Plateforme**, agence interne de l'école La Plateforme à Marseille.

J'ai eu la chance d'évoluer dans un cadre professionnel particulièrement riche, formateur et **singulièrement adapté aux objectifs du Master Business IT** que je poursuis. Cette alternance m'a en effet permis d'expérimenter, de manière concrète et progressive, **l'ensemble des domaines de compétences que couvre ce cursus** : de la **chefferie de projet** à la **conception produit**, de la **rédaction technique** au **développement web**, en passant par des responsabilités de **Product Owner** ou de pilotage d'équipes sur des cycles courts.

Ce rapport a pour objectif de restituer cette expérience dans toute sa diversité, en mettant en lumière les projets sur lesquels j'ai travaillé, les rôles que j'ai occupés, les outils et méthodes que j'ai utilisés, ainsi que les enseignements que j'ai pu tirer de cette immersion professionnelle.

À travers ce document, je souhaite retracer les différentes missions qui m'ont été confiées, les responsabilités que j'ai exercées, et les compétences que j'ai pu mobiliser ou acquérir au fil de cette immersion professionnelle. Cette alternance s'est inscrite dans un contexte stimulant, où les enjeux techniques, organisationnels et humains se croisent au quotidien.

Dans un souci de cohérence, je vais tâcher de conserver une ligne directrice centrée sur une mission principale. Certaines dimensions de l'expérience acquise seront cependant enrichies par l'observation de missions annexes, lorsque leur pertinence éclaire ou complète le sujet abordé.

L’entreprise

Présentation de l'entreprise L'Atelier de la Plateforme

La Plateforme est une école du numérique basée à Marseille, qui propose une pédagogie axée sur la pratique, l’inclusion et l’accompagnement vers l’emploi. Elle forme chaque année plusieurs centaines d’apprenants aux métiers du développement web, de la cybersécurité, de la data, de l'IA ou encore du numérique appliqué au management via le **Master Business IT**.

Dans ce cadre, **L’Atelier** occupe une place stratégique : il s’agit de **l’agence interne de La Plateforme**, où les étudiants en alternance, encadrés par des professionnels, conçoivent et développent des solutions numériques réelles pour des partenaires publics, privés ou associatifs. À la croisée de l’agence web, du pôle R&D et de l’incubateur pédagogique, L’Atelier permet de prolonger les apprentissages théoriques par des expériences concrètes, porteuses de sens et d’impact.

L’un des engagements forts de L’Atelier réside dans **l’accompagnement individualisé des alternants**, avec un positionnement volontairement hybride : chaque étudiant y est à la fois **acteur d’un projet de production réel** et **observateur privilégié des dynamiques d’équipe**, des choix stratégiques et des interactions client.
L’Atelier structure son organisation autour du **mentorat** et d’une **politique de compagnonnage**. Chaque alternant est accompagné par des développeurs et chefs de projet expérimentés, dans une logique d’apprentissage par la pratique, de transmission des savoir-faire, et d’exigence professionnelle. Grâce à l’apport de **clients concrets et de problématiques réelles**, les étudiants disposent d’un **terrain d’expérimentation solide** pour construire un savoir opérationnel.

Recrutement

Ma collaboration avec **L’Atelier de La Plateforme** débute en mars 2022, dans le cadre du **projet Liberty**, une initiative expérimentale portée par l’école visant à réimaginé les réseaux sociaux. À cette occasion, certains étudiants de La

Plateforme sont invités à contribuer au développement de l’application, sous la supervision directe de l’Atelier.

À la rentrée 2023, dans le cadre de ma recherche d’alternance pour le **Master Business IT**, je choisis de candidater auprès de L’Atelier. Mon profil est retenu, et une proposition de poste me place sur des fonctions de **chef de projet alternant**, avec la responsabilité de suivre plusieurs missions transverses.

Dès le début des échanges, j’exprime néanmoins le souhait de conserver un lien opérationnel avec le développement, domaine auquel je suis également attaché. Un accord est alors établi : **mon alternance s’organisera autour d’un équilibre entre pilotage de projet (environ 70 % du temps) et développement web (environ 30 %)**.

Cette volonté m'a permis de renforcer la posture en tant que chef de projet tout en consolidant ma compréhension technique.

Fonction au sain de l'Atelier

Au sein de L’Atelier, mon poste principal est celui de **chef de projet digital alternant**. Mon rôle s'articule autour de trois axes majeurs :

1. **La coordination de projet**, depuis la définition du besoin jusqu’au suivi de production ;
2. **L’interface client**, avec une attention particulière portée à la communication, à la reformulation des attentes, et à la présentation des livrables ;
3. **Le cadrage technique et fonctionnel**, en lien direct avec les développeurs, pour garantir la cohérence des solutions et faciliter les arbitrages.
4. La conduite de projet à travers les cérémonies et la rédaction d'artefact

Une dimension notable de mes missions résidait également dans ma **participation en tant qu'intervenant aux phases d’avant-projet**, notamment lors de la réception de nouvelles demandes ou d’opportunités externes. J’ai ainsi contribué à l’analyse initiale des besoins, à la formulation d’avant-propositions, à l’évaluation des charges potentielles et à la définition d’un premier périmètre fonctionnel. Ces interventions ont renforcé ma capacité à structurer des projets dès leur genèse et à anticiper les conditions de leur faisabilité.

L'ensemble de ces missions s'inscrivent dans une organisation en mode agile, inspiré du Framework Scrum, toutefois adapté à l'envergure des projets et la sensibilité du groupe.

Dans le cadre de cette alternance, j’ai souhaité conserver une **part active dans le développement web**, afin de maintenir une compréhension directe des contraintes

techniques et de renforcer un dialogue concret auprès des équipes et des leads développeurs.

Ces 30 % ont été majoritairement consacrés à un projet spécifique de développement, que je présenterai dans la suite de ce rapport. Ce projet m'a permis d'exercer mes compétences techniques en parallèle de mes responsabilités de pilotage.

Présentation Projet Technicert

Choix du projet Technicert

Parmi les différents projets auxquels j'ai pu contribuer durant mon alternance, **Technicert** s'est imposé comme un des projets pertinent à présenter dans ce rapport. Ce choix repose sur plusieurs facteurs complémentaires :

- Il s'agit du projet sur lequel j'ai été **le plus engagé techniquement**, en tant que **développeur**. J'ai ainsi pu intervenir sur des problématiques concrètes de **conception logicielle**, de **structuration de base de données**, et de **mise en œuvre d'API**.
- L'organisation concrète entre **l'équipe de L'Atelier** et les **parties prenantes** a offert une approche professionnelle intéressante à souligner avec une perspective différente de mes missions de pilotage habituelles.
- La nature même du projet, orientée autour de l'univers de la **certification professionnelle**, soulevait des exigences spécifiques, notamment en matière de **gestion documentaire** et de **conformité au RGPD**.
- Le projet répondait à un ensemble de **règles de développement** et d'**infrastructure** intéressantes, avec des services offrant une **cohérence d'usage**.

En résumé, ce projet couvre un **large spectre des compétences visées** dans le cadre du passage de titre, et permet de proposer une **ligne narrative presque complète** à travers cette expérience.

Certaines missions annexes, notamment mes **interventions en amont de projets** ou mon **rôle de chef de projet**, viendront également **soutenir cette analyse** en apportant un regard plus large sur mes **compétences professionnelles**.

Context & présentation de l’entreprise

Qui est Technicert

Technicert est un organisme certificateur spécialisé dans la validation des compétences des diagnostiqueurs immobiliers en France. Basée dans les Yvelines, cette société exerce une mission essentielle dans le secteur du bâtiment et de l’immobilier en délivrant les certifications obligatoires pour les professionnels intervenant dans les diagnostics réglementaires tels que l’amiante, le plomb, les termites, le gaz, l’électricité, le diagnostic de performance énergétique (DPE) ou encore l’audit énergétique.

Accréditée par le **Cofrac** (Comité français d’accréditation) pour la certification de personnes, Technicert garantit la conformité des compétences certifiées aux exigences légales et normatives en vigueur. Son activité consiste à organiser des examens de certification, de transfert et de renouvellement à un rythme soutenu, avec une session programmée chaque semaine, en présentiel ou en distanciel.

Quel est son rôle ?

L’entreprise organise des sessions d’examen (certification initiale, renouvellement, transfert) en présentiel ou en distanciel. Ces examens visent à vérifier que les diagnostiqueurs maîtrisent les exigences techniques et réglementaires définies par l'État. Les résultats sont délivrés sous quelques jours, et les certifications sont valables pour une durée fixée par la réglementation.

Le métier de certificateur implique le suivi administratif des dossiers, la planification des examens, l’évaluation des candidats, le respect strict de normes (notamment celles imposées par le Cofrac), et le maintien de l’impartialité. L’enjeu est de tracer, sécuriser et historiser les parcours de certification de manière fiable.

Contexte et problématique

Technicert s’appuie sur un ensemble d’outils numériques hétérogènes et partiellement déconnectés. La gestion des activités s’effectue à travers une combinaison de solutions dispersées : tableaux de suivi (type Excel), logiciels de notation indépendants, gestion des rendez-vous par échanges de mails, et circulation de documents papier. Cette fragmentation freine l'efficacité opérationnelle, augmente le risque d’erreur humaine, et complexifie la traçabilité des parcours candidats.

Le site internet actuel, conçu comme une vitrine institutionnelle, ne propose que quelques fonctionnalités d'information. Il n’offre pas de services interactifs ou de portail dédié aux utilisateurs (candidats, formateurs, certificateurs), limitant ainsi les possibilités de suivi autonome et de gestion dématérialisée.

Dans un souci d’optimisation de ses processus et d’amélioration de l’expérience utilisateur, Technicert souhaite se doter d’un **espace numérique unifié**. Cet espace aurait une double vocation :

1. **Un espace candidat**, permettant :
    - la création de compte et le dépôt de dossier,
    - le suivi du cycle de vie d’une certification (inscription, convocation, résultats, renouvellement),
    - la consultation et le téléchargement de certificats.
2. **Un espace administratif**, permettant :
    - la gestion centralisée des candidats, sessions, convocations et résultats,
    - le suivi des échéances et la génération automatisée des documents,
    - l'historisation complète des parcours de certification,
    - une meilleure coordination entre les équipes.

La mission nécessite de concevoir un système intégré, sécurisé et interopérable, capable de remplacer les outils dispersés existants. Les enjeux techniques incluent :

- La **gestion de workflows métier** complexes (parcours de certification),
- L’**automatisation des notifications** et des échéances,
- La **gestion d’identités et des accès différenciés** (rôle candidat, évaluateur, gestionnaire),
- La gestion des paiement en ligne
- La gestion des passage d'examen
- L'entrée de notation des résultat aux examens
- La **traçabilité et archivage des données**, dans le respect des normes de certification.

Organisation interne

Comme évoqué l'or de l'introduction, l’Atelier s’appuie sur une organisation inspirée des méthodes Agile. L’un des cofondateurs, **M. Jérôme Savajol**, est d’ailleurs **formé à l’approche Agile**, et a largement contribué à structurer les méthodes de travail internes. Cette structuration repose sur une **interprétation souple du framework Scrum**, adaptée aux dynamiques d’équipe, à leurs tailles, et selon la nature des projets.

L’approche retenue privilégie le travail en **sprints courts** (de trois à cinq semaines), permettant une **itération rapide** et une **flexibilité** importante face aux problématiques rencontrées, que ce soit côté développement ou côté parties prenantes.

Il est cependant important de noter que l’Atelier intervient sur des **projets très variés**, notamment dans le cadre de **contrats publics**. Or, ces derniers laissent peu de place à l’adaptabilité offerte par les méthodes Agiles. Dans d’autres cas, les modalités contractuelles sont pensées pour laisser **une ouverture suffisante à une démarche Agile**.

Le projet **Technicert** s’est inscrit dans un format **hybride** : un devis avec un chiffrage précis définissait le périmètre et les attendus, ce qui imposait certaines **contraintes temporelles**, tout en laissant assez de souplesse pour une **organisation par sprints**.

Enfin, il est utile de préciser que, dans le cadre habituel des projets de l’Atelier, les **développeurs alternatifs participent activement aux cérémonies Agile**, y compris les revues de sprint. Cependant, dans le cas de **Technicert**, et **en raison du profil particulier du client**, la décision a été prise de **limiter la représentation côté Atelier** à Mme Ana Stanko (cheffe de projet) et M. Julien Athomas (CTO), afin de **préserver les développeurs** d’une relation client initialement perçue comme potentiellement **conflictuelle ou instable**.

À noter que, sur d’autres projets que j’ai moi-même pilotés (et que je présenterai plus loin), j’ai veillé à intégrer pleinement les développeurs aux échanges, afin d'accroître la **communication** et observer une **compréhension fine des enjeux métier.**

Contribution technique au projet Technicert

Dans ce projet, j’ai principalement été chargé de la structuration du backend, de la conception de la base de données et du développement des API. Ce rôle m’a confronté à plusieurs enjeux concrets. Cette section revient sur ces points, en s’appuyant sur les décisions prises et les solutions mises en œuvre.

Contexte

Le lancement du projet Technicert a coïncidé avec une phase de réflexion vis-à-vis de la prochaine transition au sein de l’Atelier de La Plateforme. En effet, l'ensemble des alternants en charge des développements web arrivant en fin de cycle d'ici août

2025, le projet devait s’adapter à l’arrivée d’une nouvelle promotion, encore peu expérimentée.

Dans ce contexte, il a été nécessaire de repenser l’organisation technique et les outils utilisés. L’objectif était double : stabiliser un socle technique compréhensible rapidement par les nouveaux arrivants, tout en maintenant un cadre de travail structuré et propice à l’évolution du projet dans la durée.

Il était essentiel d’aborder cette problématique dès le lancement du projet, compte tenu d’une échéance de livraison fixée à septembre 2025. Le projet devait en effet s’adapter à un rythme d’exécution progressif, marqué par une phase d’acclimatation de plusieurs mois, le temps que les nouveaux profils montent en compétence.

La solution technique devait donc répondre de manière pragmatique à cette contrainte de calendrier, tout en tenant compte des spécificités de l’équipe en place. La question centrale était :

Comment concevoir une architecture technique à la fois :

- suffisamment simple pour permettre une prise en main rapide par des profils en formation,
- suffisamment solide pour supporter des fonctionnalités métier complexes,
- et suffisamment cohérente pour garantir la maintenabilité et la continuité du projet dans un contexte de forte rotation des contributeurs.

Réflexion et solutions envisagées

Notre premier choix a porté sur le maintien d’un environnement JavaScript/TypeScript, côté backend comme frontend. Ce choix n’était pas imposé. D'autres environnements auraient pu être considérés (PHP, Python/Django, Go, etc.), mais la priorité a été donnée à la continuité de l’existant, à l'expertise des encadrants, ainsi qu'à la possibilité pour les développeurs débutants de s’appuyer sur un écosystème déjà exploré lors de leur formation.

Choix de l’environnement backend

Nous avions précédemment utilisé NestJS sur plusieurs projets. Ce framework offre une architecture solide et bien outillée, mais nous avons constaté que sa courbe d’apprentissage pouvait ralentir les premières livraisons, en particulier pour des développeurs juniors. Sa complexité structurelle (modules, injection de dépendances, décorateurs) introduit une charge cognitive non négligeable.

Nous avons donc opté pour une solution plus légère et pragmatique : Hono, un microframework HTTP écrit en TypeScript. Inspiré par la simplicité d’Express.js, Hono permet une structuration claire des routes et des middlewares, tout en offrant une compatibilité native avec TypeScript, sans nécessiter de configuration complexe.

Ce choix s'est révélé pertinent à plusieurs niveaux. D'une part, il a facilité l'onboarding grâce à une architecture facilement lisible et accessible, sans sacrifier pour autant les bonnes pratiques en matière de typage, de validation des données ou de modularité. D'autre part, il a permis de réduire la surface d'exposition aux dépendances tierces, en privilégiant un écosystème réduit, plus maîtrisable, et composé de paquets généralement mieux maintenus.

Base de données

Le choix de PostgreSQL s'est imposé naturellement. Cette base relationnelle, bien maîtrisée dans l'écosystème, offre à la fois des performances fiables, une compatibilité SQL standard, et des possibilités avancées de modélisation. Pour interagir avec la base, nous avons retenu Prisma comme ORM. Au fil des dernières années, Prisma a connu des évolutions qui le rendent aujourd'hui particulièrement adapté aux environnements TypeScript. En particulier, sa gestion des relations a été considérablement enrichie -- ce qui offre aujourd'hui une approche très intuitive des requêtes -- et les classes générées permettent une prise en main intuitive, même pour les cas d'usage complexes.

Choix de l'environnement Frontend

L'Atelier utilise historiquement Next.js comme framework principal côté frontend. S'il s'agit d'une solution puissante et bien intégrée à l'écosystème React, Next.js a connu, ces deux dernières années, une série d'évolutions majeures qui ont profondément transformé son fonctionnement :

- Mai 2023 : introduction du nouvel App Router basé sur les React Server Components, remettant en question l'usage du dossier pages/ et bouleversant l'organisation des routes et des layouts.
- Fin 2023 : changement du système de gestion des métadonnées (viewport, themeColor, etc.), avec la dépréciation des API historiques.
- 2024 / Next.js 15 : passage à un bundler Rust (Turbopack) et nécessité d'utiliser Node.js 18+, avec de nouveaux comportements asynchrones sur les fonctions cookies(), headers(), params, etc.

Ainsi, dans un projet en production continue, avec une équipe active et disponible, ces transitions n'auraient représenté que quelques jours de travail pour monter en version et appliquer les ajustements nécessaires.
Mais le contexte typique des clients de l'Atelier est tout autre : dans de nombreux cas, nous intervenons pour produire un MVP ou un MLP, servant à démontrer la valeur d'un concept, lever des fonds, ou amorcer le recrutement d'une équipe technique. Ces projets sont souvent mis en pause pendant plusieurs mois, avant d'être relancés.

Hors, lors de la reprise du développement, l'instabilité du framework rend la reprise difficile pour les clients en potentielle difficulté financière: les dépendances deviennent obsolètes, les API changent de comportement, et le code devient

rapidement incompatible avec la version actuelle de Next.js. Chaque relance demande donc un temps important de migration, souvent imprévu, pour simplement revenir à un état de fonctionnement stable. Cela induit des coûts supplémentaires et retarde la reprise du développement.

Dans la mesure où les applications développées sont destinées à un portail fermé, sans besoin de référencement public, nous n'étions pas contraints d'adopter une approche SSR, bien que celle-ci offre d'autres avantages, notamment en matière de performance perçue ou de pré-rendu des données. Ce contexte nous a permis d'explorer des solutions plus simples à maintenir dans la durée.

Compte tenu des problématiques évoquées avec Next.js, nous avons donc fait le choix de revenir à un socle plus classique, centré sur React. Cette approche offre une grande flexibilité, une large documentation communautaire, et surtout une courbe d'apprentissage plus progressive pour des développeurs encore en formation ou en début de parcours professionnel.

Retour d'expérience concernant le choix de la stack technique

J'ai pu voir que les choix opérés ne reposent pas uniquement sur des préférences techniques. Ils s'inscrivent dans une logique de pédagogie, de continuité et de pragmatisme propre à l'environnement de l'entreprise et au type de contrat dans lequel s'inscrit les service de l'Atelier. Plutôt que de privilégier des architectures dites "idéales", nous avons cherché à définir un socle réaliste, réutilisable, et adapté à un environnement de production partiellement tournant (comme c'est le cas avec une équipe majoritairement composée d'alternants). J'ai également mis l'accent sur la lisibilité du code, la rigueur typée, et la centralisation de la logique métier. Ces principes de base permettent de réduire la dette technique dès les premières semaines du projet, tout en favorisant la transmission entre développeurs successifs.

La mise en place d'un boilerplate avec des exemples de bonnes pratiques a permis de fournir un point de départ clair pour les nouveaux arrivants. Ils peuvent ainsi reproduire les schémas d'implémentation existants et s'approprier l'architecture progressivement.

Cependant, j'emet quelques réserves concernant le choix de React. Bien que cette bibliothèque soit facile à prendre en main et particulièrement puissante pour créer des interactions complexes via les hooks, elle ne constitue pas un framework structurant à proprement parler. Cette liberté peut s'avérer problématique dans certains contextes.

Plus précisément, React offre une modularité immédiate qui peut donner l'illusion d'une montée en compétence rapide. Mais cette simplicité apparente masque une complexité croissante dès lors que l'application devient mature. L'utilisation abusive

ou mal encadrée des hooks peut entraîner une multiplication des rendus, une sur-sollicitation du DOM, voire une augmentation involontaire de la charge serveur en cas de mauvais placement des appels API.

Il existe donc un écart important entre la phase d'exploration initiale – souvent maîtrisée rapidement par des profils juniors – et les exigences de performance et de robustesse d'une application en production, manipulant des volumes de données croissants sur plusieurs années.

À cela s'ajoute un manque de structuration claire de l'architecture des dossiers, qui peut rapidement devenir un frein pour les développeurs les moins expérimentés. Une arborescence peu rigoureuse, ou laissée à l'appréciation de chacun, rend l'exploration du code difficile et favorise des décisions d'implémentation isolées, parfois incohérentes. Cela peut entraîner un effet "code spaghetti", où les dépendances se croisent de manière désordonnée, et où il devient compliqué de comprendre la logique d'ensemble ou de localiser une fonctionnalité.

Dans ce contexte, je préconise que l'Atelier se dote d'un pattern architectural clair, partagé et documenté, afin de limiter ces dérives. Cela permettrait à chaque nouveau contributeur de s'intégrer plus efficacement au projet, en réduisant la charge mentale liée à l'orientation dans le code.

Je recommande donc, sur ce dernier point, la mise en place d'un boilerplate plus structuré, intégrant des conventions explicites sur l'organisation des composants, des hooks, des services et de la logique métier. Cela constituerait une base commune à tous les projets, facilitant leur cohérence, leur maintenabilité, et leur évolutivité — en particulier dans un contexte d'équipe tournante.

Concernant la gestion des performances et les risques liés aux re-rendus excessifs, il convient de prendre en compte les évolutions récentes de React :

- À partir de React 19, la gestion du cache et l'optimisation des re-rendus sont grandement facilitées grâce à l'introduction du React Compiler (RC). Celui-ci permet une optimisation automatique, réduisant considérablement la nécessité d'utiliser manuellement des hooks comme useMemo ou useCallback. Dans la mesure du possible, je recommande donc de migrer vers cette version ou une version ultérieure.
- Pour les projets utilisant une version antérieure à React 19 et ne pouvant pas faire l'objet d'une montée de version, je recommande l'installation systématique de l'outil why-did-you-render, proposé par welldone-software. Cet outil permet de détecter les re-rendus inutiles et d'en comprendre les causes. C'est également un excellent levier pédagogique pour sensibiliser les étudiants ou développeurs en montée en compétence aux problématiques de performance frontend, tout en intégrant progressivement de bonnes pratiques.

Architecture & Infrastructure de Déploiement

Décision liée à l'architecture

Toujours dans la continuité des principes précédemment évoqués, nous avons opté pour une architecture en monorepo, organisée autour de l'outil `pnpm`, et structurée en trois packages principaux :

- `client/` — *l'application frontend en React,*
- `server/` — *l'API HTTP développée avec Hono,*
- `shared/` — *un espace mutualisé contenant la logique métier commune, les DTOs, les schémas de validation, et les utilitaires.*

Ce découpage répond à une volonté claire : éviter les duplications de logique entre frontend et backend, tout en garantissant une cohérence forte autour du modèle métier. L'approche type-safe induite par ce partage de code a permis à l'ensemble des services d'utiliser une base commune pour les types, la validation, et les règles métiers.

Avantage de l'approche

Sans aller jusqu'à une stratégie formelle de Contract-Driven Development, notre dossier shared/ a néanmoins rempli un rôle de contrat implicite, facilitant considérablement la communication entre services et jouant un rôle d'auto-documentation très utile, notamment côté frontend pour anticiper la structure des données.

Nous avons pu observer des bénéfices dans la pratique d'une telle approche. Ainsi cela nous a permis un alignement rapide entre frontend et backend, accroissant notre vélocité de développement, une réduction significative des erreurs de typage ou de format d'échange grâce à l'alignement automatique via les types partagés et une unification des pratiques de codage autour d'un seul dépôt source.

Retour d'expérience et limites rencontrées

Cependant, cette architecture a également révélé certaines limites. Notre organisation par entité dans shared/ a mené à la cohabitation de types fortement interdépendants, notamment lorsque deux entités sont mutuellement liées *(example:* `Candidate` *peut avoir l'attributs* `certifications[]`*. Et* `Certification` *peut avoir l'attribut* `candidate`*)* Cette co-dépendance a parfois entraîné des risques de dépendances circulaires et une surcharge cognitive liée à la complexité croissante des fichiers centralisant toute la logique liée à une entité. Les fichiers de définition devenaient trop volumineux, rendant la lecture et la navigation difficiles (certains

fichiers atteignant plus de 800 lignes), ainsi que les modifications des DTOs fastidieuses en cas de nouvelles migrations.

Aujourd'hui je recommanderais de procédé ainsi afin d'améliorer l'usage de ce type d'architecture :

1. Externaliser la génération des schémas Zod depuis Prisma avec la librairie zod-prisma-types vers le package shared/. Cela permet de disposer d'un fichier central synchronisé avec le schéma de données, et qui assure la cohérence continue entre migrations Prisma et logique de validation côté services.
2. Centralisation de toutes les co-dépendances dans ce fichier généré unique, qui joue alors le rôle de module pivot entre les entités fortement couplées, réduisant les risques de cycles.
3. Enfin, pour améliorer la lisibilité et faciliter la navigation dans le code partagé, en envisageant de découper les schémas et types par entité et par type de requête. L'arborescence pourrait ressembler à :

```
shared/
└── dto/
    └── [entity]/
        └── [verb]/        # GET, POST, PUT, DELETE
            └── [controller_or_use_case].ts
```

Ce découplage permettrait de segmenter les responsabilités et d'accélérer l'accès à une information spécifique sans avoir à parcourir de longs fichiers monolithiques.

Déploiement, maintenance et scalabilité

Le projet repose sur une architecture monolithique optimisée, intégrée dans un environnement de développement reproductible grâce à Flox. Ce dernier permet une gestion fine des dépendances système (Node.js, PostgreSQL, Vault, etc.) ainsi qu'un bootstrap automatisé de l'environnement local (via pnpm, initialisation conditionnelle de la base PostgreSQL, etc.).
Nous avons mis en place une pipeline CI/CD via GitHub Actions, comprenant :

- Lancement automatique des tests serveur, avec analyse de couverture et génération de rapports de logs lisibles en cas d'échec ;
- Versionnement par tag Git, déclenchant des déploiements conditionnels ;
- Déploiement automatisé vers les environnements de développement et de production à la suite de la validation du Pull Request.

- Mise en production sur un cluster Kubernetes maintenu par l'équipe de La Plateforme ;
- Surveillance des erreurs et agrégation de logs, pour un suivi post-déploiement plus réactif.

L'ensemble favorise une boucle de feedback rapide, tout en assurant la stabilité des déploiements et la reproductibilité des environnements entre développeurs.

Modélisation de la base de données et cartographie des flux de données

Conception de la base de données

La modélisation de la base de données de l'application Technicert a constitué une étape centrale dans la conception de l'outil et dans la résolution de la compréhension métier. Elle nous a permis de traduire les besoins métiers de la certification professionnelle en structures relationnelles robustes et adaptées à un environnement évolutif. Ce travail s'est aussi révélé être un levier concret de réflexion collective, en matérialisant les concepts métiers dans une structure manipulable et partageable. Il a permis d'ouvrir des discussions structurantes avec l'ensemble des parties prenantes.

Dans une démarche agile, la base a été conçue pour être scalable, en anticipant les évolutions à venir sans nécessiter de remaniement profond. Certaines décisions ont parfois privilégié une approche pragmatique à une modélisation strictement académique, afin de répondre efficacement aux besoins métiers exprimés — même de manière partielle ou implicite. C'est le cas, par exemple, du lien entre les dossiers de certification (`CertificationFile`) et les entités employeurs (`Companie`). Bien qu'aucune exigence claire n'ait été formulée, nous avons opté pour une relation One-To-Many entre `Companie` et `CertificationFile`, reflétant le cas fréquent où plusieurs candidatures sont portées ou financées par un même employeur. Les informations relatives au manager (nom, fonction, email, téléphone) ont été intégrées directement dans le modèle, plutôt que modélisées dans une entité dédiée, car elles ne sont pas toujours présentes, ni nécessaires à la logique fonctionnelle.

Un autre exemple de révision structurelle a concerné la gestion des sessions d'examen. Initialement, le client avait décrit une organisation en deux temps systématiques — théorique puis pratique. Mais l'analyse plus fine de documents officiels a révélé que certains domaines ne comportaient qu'un seul type d'épreuve. Cela nous a conduit à modéliser les `Session` en lien avec des `Module`, pouvant être au nombre de un ou plusieurs. Le candidat ne souscrit donc plus directement à une session, mais à chaque module via une entité `Participation`. Cette granularité permet une plus grande souplesse, tout en restant conforme aux évolutions potentielles imposées par la COFRAC.

Méthodologie de modélisation

Le modèle de données repose sur une approche relationnelle classique, implémentée via le framework Prisma. Il est accompagné d'un schéma DBML pour la documentation technique. Les entités ont été identifiées à partir de l'analyse des processus métiers et des exigences de la COFRAC. Parmi les tables principales, on retrouve :

- `Candidate` : représente une personne préparant une certification ;
- `CertificationFile` : dossier administratif contenant les pièces justificatives ;
- `CertificationRequest` : demande officielle de certification ;
- `Certification` : résultat de la procédure de certification ;
- `Participation`, `Session`, `Module` : gestion des épreuves ;
- `PaymentIntent`, `CertificationPayment` : données liées aux paiements ;
- `Companie`, `Examinator` : informations contextuelles et encadrantes.

Le modèle inclut les types de relations, les champs sensibles, les contraintes d'unicité, et les dates de création/modification/suppression. Cette structuration vise à favoriser la scalabilité et à anticiper les besoins en archivage et en contrôle.

Cartographie des flux de données

En parallèle de la modélisation, une analyse des flux de données a été réalisée pour identifier les moments où les données personnelles sont créées, modifiées, stockées, transmises ou supprimées. Voici un résumé synthétique :

Étape métier	Données concernées	Origine → Destination	Finalité
Création du compte	Nom, prénom, email, mot de passe	Frontend → Backend → BDD	Authentification et accès utilisateur
Dossier de candidature	Adresse, CNI, diplômes, justificatifs	Utilisateur → API → Bucket S3 + BDD	Constitution du dossier à valeur probante

Paiement	Montant, identifiants Stripe, statut	Frontend → Stripe → Webhook → Backend	Traitement comptable et suivi des paiements
Inscriptions aux sessions	Domaine, session ID	Utilisateur → Backend → BDD	Planification du passage de l'épreuve
Délivrance du certificat	Résultat, identifiant, métadonnées	Backend → BDD → Générateur PDF	Archivage et contrôle de conformité

Chaque flux est analysé en fonction de sa finalité, des types de données impliquées et des mesures de sécurité en place :

- Chiffrement des communications (HTTPS/TLS)
- Chiffrement des fichiers sensibles dans les buckets S3 (AES)
- Accès restreint par rôle (RBAC)
- Journalisation des actions utilisateur

Intégration à l'analyse d'impact (AIPD)

Dans le cadre du projet **Technicert**, une **AIPD complète** a été rédigée afin d'anticiper les risques pour les droits et libertés des personnes concernées. Ce document central s'appuie sur une **analyse fonctionnelle du parcours utilisateur**, une **description détaillée des traitements** et une **cartographie technique des flux de données**, actuellement enrichie d'un schéma visuel.

L'AIPD identifie précisément :

- Les **données collectées** à chaque étape du parcours (création de compte, candidature, évaluation, paiement, archivage) ;
- Leur **base légale** (exécution contractuelle, obligation légale, mission d'intérêt public) ;
- Leur **durée de conservation**, conforme aux exigences **COFRAC** et **du Code du travail / commerce** ;
- Les **destinataires internes et externes**, y compris les **sous-traitants** comme **Stripe** (encadré par des clauses contractuelles types RGPD) ;

- Les **mesures techniques** (chiffrement, stockage sécurisé, séparation des environnements) et **organisationnelles** (contrôle d'accès, journalisation) mises en place.

L'ensemble du traitement a été pensé de manière **privacy by design** dès les premières itérations du projet. En particulier :

- Un **registre de traitement** a été produit selon le modèle CNIL PME, précisant les finalités, bases légales et transferts éventuels ;

- Une **analyse de risques** a été conduite, identifiant les menaces critiques (accès non autorisé, perte de données, flou informationnel), et les **mesures correctrices** associées (audit des accès, backups, documentation RGPD, formation des agents) ;

- Les **flux de données sensibles** ont été décrits et documentés, à la fois dans l'AIPD et dans un **fichier visuel** (schéma d'architecture logique en cours de validation avec l'équipe technique).

Ces livrables permettent de démontrer la **conformité progressive** de l'outil avec le RGPD, tout en facilitant le dialogue avec la **COFRAC** et la préparation des **audits de certification**. La transparence vis-à-vis des personnes concernées a également été anticipée via la préparation d'un **formulaire d'exercice des droits**, intégré au tableau de bord utilisateur, et d'une **politique de confidentialité** en cours de finalisation.

En complément, un travail de **cartographie des données sensibles** a été initié à partir du **modèle de base de données**. Les entités contenant des informations à caractère personnel (par exemple `Candidate`, `CertificationRequest`, `PaymentIntent`) ont été annotées avec leur sensibilité, les rôles pouvant y accéder (via **RBAC**), et les mécanismes de protection mis en place (chiffrement, accès restreint, anonymisation à venir pour les exports statistiques).

Identification des données sensibles et mesures de protection

Dès les premières phases de conception de l'application **Technicert**, l'analyse métier ainsi que le diagramme de base de données relationnel ont permis de cibler les données à caractère personnel ainsi que les données sensibles. Permettant ainsi d'adopter une posture préventive en termes d' exigences de conformité RGPD. Ces exigences devaient notamment intégrer les contraintes normatives liées aux audits de la COFRAC, autorité de référence en matière d'accréditation. Cela nous a permis de cartographier les données, classé leur niveau de sensibilité et établi des règles claires d'accès, de stockage et d'usage synthétisées dans un document d'Analyse d'Impact relative à la Protection des Données.

Quelles sont les données personnelles traitées ?

Les données à caractère personnel identifiées concernent majoritairement les candidats aux certifications. On y trouve :

- des **données d'identité** (nom, prénom, date de naissance),
- des **données de contact** (email, téléphone),
- des **données administratives** (documents justificatifs, pièces d'identité, attestations d'expérience),
- et, dans certains cas, des **informations professionnelles** (poste, structure d'appartenance, habilitations).

La présence de pièces justificatives et de documents d'identité introduit un **niveau de sensibilité élevé**, particulièrement en cas de faille de sécurité. Ces documents sont nécessaires à la validation des candidatures et à la constitution des dossiers techniques soumis à la COFRAC, mais leur gestion exige rigueur et précaution.

Principe d'accès grâce à une gestion des rôle

L'architecture applicative repose sur un principe de séparation stricte des privilèges. Chaque utilisateur se voit attribuer un rôle (`USER`, `ADMIN`, `SUPER`), encadrant les données accessibles en lecture ou modification. Ainsi :

- Les candidats peuvent consulter uniquement leurs propres données ;
- Les agents Technicert n'ont accès qu'aux informations nécessaires à l'évaluation des dossiers ;
- Les administrateurs disposent d'un accès complet, mais leurs actions sont systématiquement journalisées.

L'authentification est assurée via des **tokens JWT**, garantissant à la fois une sécurité robuste et une intégration simple côté client. Des rôles sont utilisés pour limiter les permissions au strict nécessaire.

Mesures de sécurisation mises en place

Les échanges entre les clients (navigateurs) et les serveurs sont sécurisés via le protocole HTTPS (TLS), garantissant la confidentialité des données en transit. Les mots de passe sont stockés de manière sécurisée grâce au hachage bcrypt.

Par ailleurs, les documents sensibles (pièces d'identité, attestations, justificatifs de diplôme) sont stockés dans un bucket S3 privé hébergé par Scaleway. Ce stockage est sécurisé par chiffrement côté serveur, et l'accès à ces documents est contrôlé par des URLs temporaires signées, empêchant tout accès direct ou non autorisé.

Archivage et intégrité des certificats

Les certificats délivrés sont archivés de manière dématérialisée et générés dynamiquement à chaque consultation. Cela garantit la cohérence entre les données de la base et les documents délivrés, tout en évitant les risques de falsification. Leur durée de conservation est alignée avec les recommandations du COFRAC : dix ans à compter de la date de délivrance.

Autres garanties

Les formulaires de saisie sont conçus pour indiquer clairement les champs obligatoires et optionnels, conformément au principe de minimisation. Les données collectées sont limitées à ce qui est strictement nécessaire à la finalité du traitement, et aucun profilage ou prise de décision automatisée n'est effectué sur les candidats.

L'ensemble de ces mesures s'inscrit dans une logique de "privacy by design" et constitue une réponse proactive aux exigences du RGPD comme aux attentes opérationnelles du COFRAC en matière d'auditabilité, de traçabilité et de sécurisation des certifications professionnelles.

Planification et coordination du Projet

Le projet a été entièrement versionné via GitHub. Chaque fonctionnalité faisait l'objet d'une branche dédiée, nommée selon une convention précise (`feature/`, `fix/`, etc.). Les pull requests étaient soumises à relecture par le CTO (Julien Athomas), et les commits étaient accompagnés de messages explicites, favorisant la traçabilité des développements et la revue technique.

Le projet Technicert a été mené dans un contexte où le chef de projet n'était pas disponible à plein temps. Ce déficit d'encadrement a conduit l'équipe de développement à assumer une partie de la conduite de projet. Sur le papier, le projet Technicert prévoyait une gestion structurée via GitHub Project et une méthodologie agile, reposant sur des itérations de 4 à 5 semaines.

La structure de notre kanban suivait un flux clair, accompagné de définitions of done, réparties comme suit :

- **Backlog** : tickets correspondant aux besoins métier, décomposés en objectifs spécifiques ;
- **En cours** : ticket assigné à un développeur et actuellement en développement ;

- **À tester** : ticket terminé, testé par le développeur, soumis à une pull request en attente de relecture ;
- **À tester client** : fonctionnalité intégrée à la recette et soumise à évaluation par le client ;
- **Done/Go** : fonctionnalité validée comme conforme aux attentes.

Chaque ticket s'inscrivait dans une matrice Lean, basée sur trois critères :

- **Importance** (Must Have / Should Have / Could Have / Won't Have) ;
- **Estimation horaire** par l'équipe technique ;
- **Priorité**, définie comme la synthèse des deux précédents.

Les tickets étaient également typés :

- **Feature** : ajout de nouvelles fonctionnalités ;
- **Bug** : correction de dysfonctionnements imprévus ;
- **Task** : actions techniques non fonctionnelles (tests, documentation, etc.).

L'absence de véritables Sprint Planning et de priorités partagées a engendré un déséquilibre dans le traitement du backlog. Certaines fonctionnalités mineures, notamment visuelles, ont été survalorisées pour rassurer le client, au détriment d'une construction cohérente du MVP. Les retours clients, souvent imprécis, passaient par une interprétation technique faute d'un lien direct avec les développeurs. Ce glissement a conduit l'équipe à endosser des responsabilités de pilotage, induisant surcharge cognitive, désynchronisation fonctionnelle, et ralentissements.

Pour y remédier, plusieurs pratiques ont été mises en place :

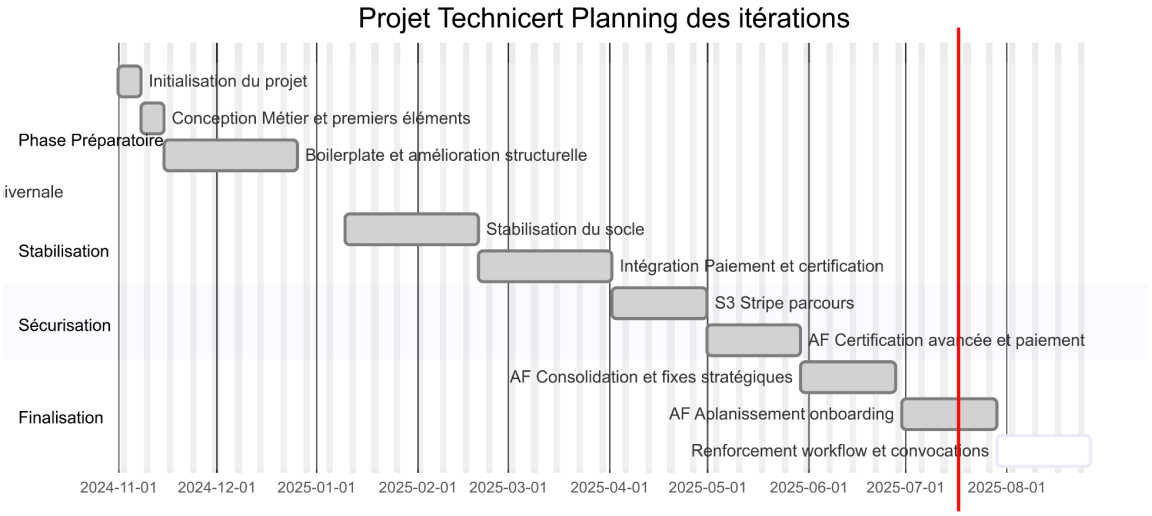
- des pull requests plus fréquentes, réduites à des unités fonctionnelles simples pour faciliter la relecture collective ;
- un récapitulatif en en-tête de chaque pull request, permettant une meilleure compréhension des enjeux techniques par le CTO ;
- une documentation interne des blocages, consignée au fil du développement, transmise à la cheffe de projet pour clarification ;
- un effort accru sur la **communication osmotique** au sein de l'équipe, rendant visibles sans demande explicite les problèmes techniques ou métiers rencontrés.

Ce fonctionnement adaptatif témoigne de l'application concrète des principes agiles : responsabilisation, auto-organisation, et ajustement progressif. Il met aussi en évidence une leçon fondamentale : sans transparence ni artefacts partagés, l'adaptation devient une improvisation. Ce que Scrum appelle empirisme — observer, ajuster, documenter — n'est possible que si l'environnement permet aux connaissances d'émerger, de circuler et d'être comprises collectivement.

En l'absence de certains rituels et repères, l'équipe a ainsi tenté de reconstituer, par la pratique, les conditions minimales de circulation d'information, de compréhension partagée et de validation des choix techniques. Le projet a ainsi montré que l'intelligence collective peut pallier, un temps, l'absence d'un cadre formel, mais que cela nécessite un effort constant de structuration interne pour ne pas céder à la dérive ou à la saturation.

En définitive, nous avons choisi de renforcer la documentation métier via le wiki du projet. Toutes les questions liées aux problématiques fonctionnelles ont été archivées dans une page dédiée, afin de conserver l'historique des réflexions. Le workflow de certification — cœur logique de l'application — a ainsi été formalisé et mis à jour avec rigueur. De plus, la base de données, les documentations Swagger, les routes, les README, ainsi que la description des flux de données, constituent désormais un socle facilitant l'appropriation du projet par de nouveaux développeurs. Cette structuration progressive vise à consolider une compréhension partagée des enjeux métier, essentielle à la continuité et à l'évolutivité du projet.

Dans ce contexte, **le diagramme de Gantt a volontairement été écarté**. Ce type de représentation, fondé sur une planification rigide et séquentielle, est en contradiction avec l'approche Agile pratiquée dans le projet. Il tend à masquer la réalité mouvante du développement logiciel, en particulier lorsqu'il s'agit d'un MVP encore en maturation. Pour autant, la planification n'a pas été négligée : elle a été assurée via un **backlog produit découpé en milestones**, représentant des jalons fonctionnels structurants (stabilisation du socle, intégration des paiements, sécurisation, etc.).



Ces itérations, définies collectivement et priorisées par valeur métier, ont permis à l'équipe de conserver un cap, tout en s'adaptant aux retours progressifs du commanditaire. Le backlog ainsi organisé constituait le véritable **artefact de pilotage**, substitut pragmatique à tout outil de planification rigide. En fin de projet, un diagramme de Gantt rétrospectif a été produit à des fins de restitution, mais sans vocation prescriptive.

Projet SRIAS

Présentation des enjeux de l'avant-projet

Le projet SRIAS a débuté par une phase d'avant-projet, dans le cadre d'un accompagnement à l'émergence d'une solution numérique. Cette phase visait à clarifier les besoins de la SRIAS et à poser les fondations fonctionnelles et techniques du futur développement. En tant que consultant, j'ai été mandaté pour identifier les problématiques internes de la SRIAS, formaliser les fonctionnalités clés attendues de l'application, rédiger la documentation de cadrage et concevoir des wireframes illustrant les futurs parcours utilisateurs.

L'avant-projet s'est déroulé en quatre demi-journées réparties entre les membres de l'équipe (Ana, Julien et moi-même). Les deux premières séances furent dédiées à la stabilisation des attentes et à la présentation des premiers wireframes. La dernière séance permit la remise de l'ensemble des documents de cadrage, validés par la SRIAS.

Présentation de l'entreprise SRIAS

Qu'est-ce que la SRIAS ?

La SRIAS (Section Régionale Interministérielle d'Action Sociale) est une structure régionale rattachée au ministère de la Fonction publique. Elle agit en faveur du bien-être des agents de l'État en région, en mettant en œuvre des actions sociales : logement, garde d'enfants, aide aux vacances, etc.

Quelle était sa problématique ?

La SRIAS PACA faisait face à une complexité croissante dans la gestion des prestations sociales destinées aux agents publics. Le traitement des demandes de subvention, la vérification des droits, la gestion des contrats avec les partenaires, et la traçabilité des prestations reposaient sur une organisation artisanale, dispersée entre tableurs, emails, et échanges manuels. Ce mode de fonctionnement entraînait plusieurs difficultés majeures :

- **Manque d'efficacité et de fluidité** dans le traitement des déclarations des partenaires.
- **Absence de centralisation des données**, rendant difficile le suivi des subventions accordées.
- **Manque de transparence pour les agents** sur leurs droits et sur les offres disponibles.
- **Charge administrative importante** pour la SRIAS et les partenaires, avec des risques d'erreur élevés.
- **Difficulté à fiabiliser les conditions d'éligibilité** (revenu fiscal, nombre de parts, code ministère, etc.) sans système intégré.

Cette situation rendait difficile l'évaluation globale de l'efficacité des actions sociales menées et générait des frustrations tant pour les agents bénéficiaires que pour les partenaires. La nécessité d'un portail numérique complet, intuitif et sécurisé s'imposait pour fluidifier les échanges, fiabiliser les données, et simplifier les processus pour l'ensemble des parties prenantes.

Phase d'avant-projet

Dans le cadre de la phase d'avant-projet du portail SRIAS, j'ai été chargé de conduire l'analyse initiale des besoins aux côtés de la responsable de la SRIAS PACA. Cette analyse, structurée autour d'ateliers de cadrage et d'entretiens qualitatifs, a abouti à la rédaction d'un document d'avant-projet détaille. Ce document constituait une pièce justificative supplémentaire et essentielle en vue d'un dépôt de subvention auprès de la préfecture, condition préalable à tout lancement du projet via appel d'offre.

Le besoin principal identifié concerne la fluidification du traitement des demandes de subventions et de prestations destinées aux agents publics, avec une attention particulière portée à la traçabilité, la simplification du parcours usager et la conformité légale. Nous avons mis en évidence trois profils utilisateurs : les agents, les administrateurs SRIAS, et les partenaires prestataires. Chacun de ces profils impliquait des droits spécifiques et des parcours distincts liés à la politique de la SRIAS en matière de droit des agents.

L'analyse a également mis en lumière des contraintes externes fortes : nécessité d'un accès via FranceConnect, gestion des éligibilités sur la base du quotient familial, conservation des justificatifs, exigence de transparence et de reporting. Ces éléments ont guidé la proposition de structuration fonctionnelle du futur portail, notamment à travers des maquettes, des flux décrits, et une proposition d'architecture modulaire, souple et interopérable.

Ce projet a également permis de sensibiliser les parties prenantes aux enjeux liés aux données à caractère personnel, et à la nécessité d'engager une étude préalable, condition sine qua non à l'usage d'API nationales restreintes. Ce travail a permis à la SRIAS PACA de préparer les éléments nécessaires à la mise en conformité de la solution et au bon déroulement de ses missions. Le cadrage mené a ainsi permis, à travers ses discussions avec l'ensemble des parties prenantes, de matérialiser une problématique concrète, tout en posant les bases solides d'un futur appel à projet respectueux des exigences du secteur public.

Appel à projet et phase de développement

Rôle de Product Owner

Après la validation de l'appel à projet « SRIAS Connect », l'Atelier a été retenu pour développer la solution. Durant cette nouvelle phase, j'ai assuré un rôle de Product Owner sur l'ensemble du cycle de vie du projet. Bien que la cheffe de projet (Ana) ait conservé une supervision générale, c'est moi qui ai pris en charge la gestion opérationnelle du backlog, l'animation des cérémonies Scrum, et la coordination technique avec les développeurs.

Adaptation du livrable

Dans un contexte de commande publique, le projet SRIAS s'est inscrit dans une phase préparatoire préalable à un appel à projet formalisé. Le travail réalisé fut donc une **formalisation structurée des besoins**. J'ai pu donc traduire ces besoins en **backlog fonctionnel** découpé à partir du document d'avant-projet transmis que j'avais préalablement édité et qui fut validé par la SRIAS PACA. Ce backlog constitue l'artefact central ayant permis d'aligner les équipes sur les objectifs à atteindre.

Plutôt qu'un cahier des charges rigide, cette approche a permis de maintenir une **dynamique agile**, tout en répondant aux contraintes spécifiques du secteur public et aux exigences institutionnelles. Les critères d'acceptation ont été intégrés dans la définition des éléments du backlog (Definition of Done), et les indicateurs de performance ont été reportés dans des **milestones** jalonnant l'évolution du projet.

Ce travail démontre ma capacité à **formaliser les besoins métier dans un format exploitable**, à structurer un projet en tenant compte de ses contraintes sectorielles, et à produire des livrables **juridiquement recevables et techniquement opérationnels**, sans sacrifier la logique itérative propre à une démarche agile.

Organisation de gestion de projet

Le développement s'est inscrit dans une approche Agile stricte, avec des itérations mensuelles. Le projet a suivi un cadre méthodologique basé sur les rituels Scrum :

- **Sprint planning** avec les développeurs
- **Daily meetings** réguliers
- **Séances de grooming** avant chaque sprint review
- **Sprint reviews** en présence de toutes les parties prenantes (client, devs, PO)

Chaque sprint se clôturait par la prise de note formelle des ajustements demandés, une réorganisation du backlog, puis le lancement du cycle suivant.

La gestion du projet était centralisée dans un GitHub Project structuré autour des colonnes suivantes :

- **Backlog** : ensemble des issues identifiées à partir du besoin client
- **Ready** : issues sélectionnées pour le sprint en cours
- **En cours** : développement en cours
- **Fini à tester** : ticket terminé, avec pull request soumise et en cours de validation
- **À tester client** : intégré à la recette, prêt à être testé côté client
- **GO** : fonctionnalités validées par le client

Les tickets comportaient systématiquement une estimation horaire (par les devs uniquement) et une priorité définie initialement par le PO, puis réajustée collectivement selon les enjeux métier.

Contribution à la phase de conception et déclaration technique

En tant que PO, j'ai également contribué à la formalisation d'une déclaration technique issue de l'avant-projet. J'ai participé à l'architecture fonctionnelle, à la validation des choix techniques avec les développeurs, et à la clarification des flux de données, notamment à travers un lexique métier et une structuration de la base de données. Le modèle relationnel, les entités, et les interactions prévues ont été précisés dès le lancement du développement.

Particulièrement, j'ai documenté l'approche technique en rédigeant des tickets détaillés sur les composants transversaux, comme le tableau principal de gestion des données. Cette documentation, enrichie de schémas et captures d'écrans, permettait de mutualiser la conception de composants réutilisables, facilitant ainsi le travail des développeurs. Chaque fonctionnalité intégrée était soigneusement référencée dans le backlog avec un lien vers ses itérations précédentes, assurant une continuité technique et une montée en compétence rapide pour les nouveaux contributeurs.

Conduite de projet et organisation autour de cérémonies Scrum et rédaction d'artefacts

J'ai personnellement animé l'ensemble des cérémonies, rédigé tous les tickets du backlog, et produit les comptes rendus de chaque sprint review. Lors de la phase de test client, je veillais à ce que toutes les issues soient effectivement fonctionnelles avant la présentation. En cas de dysfonctionnement, le ticket était renvoyé dans « En cours » avec un commentaire daté, listant les points de friction et les actions attendues.

N'ayant pas pu être présent durant les deux premiers sprints pour des raisons de calendrier académique, j'ai redoublé de rigueur dans la structuration du backlog initial. J'ai veillé à anticiper les enjeux techniques et fonctionnels, afin de fournir aux développeurs une vision claire et opérationnelle des objectifs. Cette anticipation s'est notamment traduite par :

- Une documentation exhaustive dans les tickets clés
- Une structuration logique des milestones par fonctionnalité
- Une formalisation du lexique métier pour lever toute ambiguïté

Cette rigueur a permis d'assurer la traçabilité des évolutions et d'ancrer un véritable processus d'amélioration continue. Chaque étape était documentée, chaque problème traité, et chaque retour client intégré avec méthode.

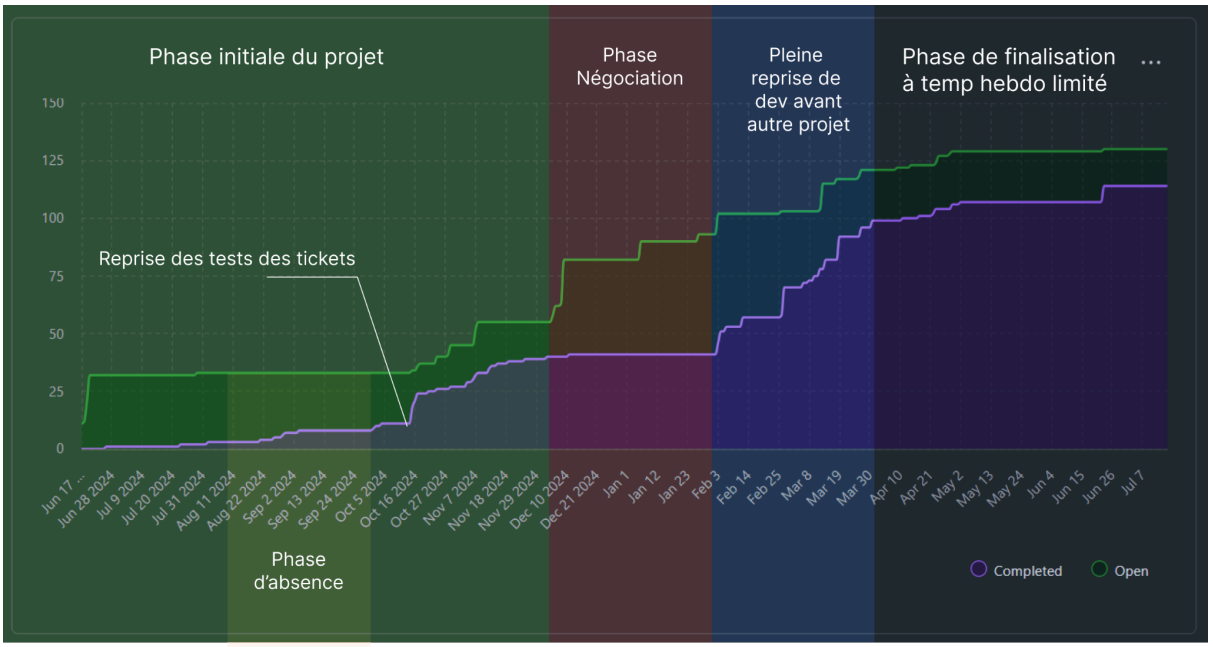
Un point central de cette démarche fut l'attention portée à la production régulière de comptes rendus exhaustifs. Ces derniers ont permis de nourrir la confiance du client et de renforcer la transparence tout au long du projet. La sprint review est ainsi devenue un véritable artefact d'inspection et d'adaptation, capable de refléter la progression, d'identifier les points de friction, et de guider les décisions métier dans une logique d'amélioration continue.

À la date de début décembre, la burn chart était conforme aux prévisions : seules deux fonctionnalités restaient à intégrer, l'une dans les temps estimés, l'autre dépendante d'une

autorisation de France Connect. C’est alors qu’est intervenu un tournant organisationnel : une structure partenaire de la SRIAS, à vocation juridique, a exprimé de nouvelles exigences non prévues initialement. Cette structure – dont les responsabilités portaient sur l’inspection des partenaires – n’avait jusqu’ici pas été directement impliquée dans les décisions fonctionnelles. Un désaccord structurel est apparu : la SRIAS n’ayant juridiquement pas compétence pour certaines vérifications, alors même qu'elles étaient incluses dans l’application.

Face à cette nouvelle donne, un rendez-vous exceptionnel a été organisé avec l’équipe de développement pour évaluer les nouveaux tickets. Une estimation des risques techniques a été réalisée collectivement, permettant à Ana de statuer sur les ajustements à intégrer, en arbitrant entre besoin client et faisabilité technique. Cela a abouti à un compromis : une révision de certaines fonctionnalités a permis de répondre partiellement aux nouvelles exigences sans remettre en cause l’équilibre du projet ni la charge de travail allouée. Cette expérience a renforcé notre capacité collective à adapter la trajectoire du projet tout en préservant sa viabilité.

Burn Chart du projet SRIAS



La burn chart ci-jointe permet d’observer les dynamiques concrètes du projet SRIAS au fil du temps. On distingue clairement plusieurs phases :

- Une **phase initiale** d'ouverture et de planification des tickets, avec une montée progressive des issues ouvertes (courbe verte), mais peu de tickets finalisés (courbe violette), en partie à cause de l’absence du PO sur les deux premiers sprints.
- Durant cette absence, la validation des tickets en attente est restée en suspens, ce qui crée un plateau artificiel sur les "completed". À la reprise, une validation rapide en escalier correspond à des correctifs sur des tickets techniques déjà développés mais non validés.
- S’ensuit une **phase de développement actif** : les livraisons reprennent de manière régulière, la courbe "completed" gagne en linéarité, et reflète un rythme d’équipe retrouvé.

- Lors de la **période de négociation**, un désaccord de périmètre avec une entité juridique extérieure à la SRIAS (impliquée tardivement) vient bousculer les objectifs initiaux. Cela génère une augmentation subite du backlog (+25 %), traduisant un risque de débordement projet. Une analyse des risques est alors menée avec les devs, permettant de réévaluer chaque nouvelle issue.

- Enfin, la **phase de finalisation hebdomadaire** montre un ralentissement assumé, correspondant à une mobilisation réduite de l'équipe et à une clôture progressive des fonctionnalités.

Malgré une trajectoire légèrement biaisée par un manque de suivi initial des flux de tickets, cette burn chart met en évidence une capacité d'adaptation, une dynamique de rattrapage rapide, et un pilotage itératif conforme à la méthodologie agile.

Conclusion de l’expérience

Cette expérience fut particulièrement enrichissante, car elle a ancré ma participation dans un projet d’envergure. Depuis plusieurs années, peu de décisions ministérielles avaient permis le développement d’une telle application régionale. Ce travail a été rendu possible grâce à l’engagement de la SRIAS PACA, déterminée à moderniser des pratiques administratives devenues inadaptées face aux enjeux de transformation numérique.

Bien que portée initialement à l’échelle régionale, l’ambition du projet dépasse ce cadre : une extension nationale est déjà envisagée. L’avant-projet a ainsi permis de démontrer qu’une solution pragmatique, pensée pour un budget contraint, pouvait répondre efficacement aux besoins métier.

La conduite de projet que j’ai assurée n’était pas ma première, mais elle se distinguait par son cadre contractuel exigeant, inhérent à tout projet public. Elle m’a obligé à faire preuve d’une rigueur accrue, tout en restant attentif aux contraintes économiques de l’entreprise. Mon rôle de Product Owner s’est révélé déterminant : il m’a permis d’observer de manière active l’impact stratégique de cette fonction sur la réussite globale du projet.

Par ailleurs, mes compétences techniques m’ont permis de pallier mon absence initiale. En anticipant les besoins à travers un backlog détaillé et une documentation technique claire, j’ai facilité la progression autonome des développeurs. Cette préparation a également permis aux devs de s’approprier progressivement la logique métier, jusqu’à devenir eux-mêmes porteurs d’intentions fonctionnelles.

Ce retour progressif de la maîtrise métier vers l’équipe technique témoigne d’une montée en compétence collective, portée par une dynamique empirique et un engagement partagé envers la qualité du produit final.