

Dossier Technique

Projet SIVAC

HENNAOUI Riyad

CDA8 session 2021/2022.



Tuteur

CATALA Remy

AVERTISSEMENT

La société Inetum demeure unique propriétaire des documents et données liés aux réalisations sur l'application SIVAC. Aucun transfert des composants de l'outils tels que le code, les données ou la documentation technique ne peut être réalisé sans accord préalable écrit. Il a été validé auprès du tuteur, CATALA Remy, qu'aucune donnée à caractère confidentiel n'a été transmise au titre de ce rapport technique.

Remerciements

Avant tout, je tiens à remercier toutes les personnes qui ont contribué au succès de ce stage et qui ont aidé à ce que cette reconversion se réalise.

Je tiens tout d'abord à remercier Monsieur **Remy CATALA**, Directeur de Projet de la société INETUM, de m'avoir accueilli comme stagiaire au sein de son équipe, et qui s'est assuré que tout se passe pour le mieux.

Marie Stéphanie LEROY, Chef de Projet, pour son immense aide à la compréhension du projet, sa disponibilité, son investissement et ses encouragements.

Marie CHABOUT-COMBAZ Analyste fonctionnel, pour son accueil, son humour et son soutien.

Yann ALBAC, responsable technique, pour son expertise, sa disponibilité, ses précieux conseils et sa bonne humeur.

Séverine SALAHUN, développeuse Java, pour sa disponibilité, ses précieux conseils, sa bonne humeur et ses encouragements.

Laurent CREVOLA, développeur full stack, pour ses conseils, sa disponibilité et ses encouragements.

Emeric COURTADE, développeur full stack, pour tout le temps qu'il m'a consacré, son aide à la compréhension du code côté front et back.

Patrice HERVE, développeur full stack, pour sa disponibilité, son professionnalisme, sa bonne humeur et ses précieux conseils.

David BEUGNET, développeur Java, pour son aide précieuse à l'installation de tous les outils nécessaires, ses conseils, sa bonne humeur et son humour.

Farhan BARREH-BOUNI, développeur Java et ancien CDA à l'AFPA, pour ces précieux conseils, sa disponibilité, sa bonne humeur et sa relecture.

Mes camarades de promotion, grâce à notre entre aide, notre bonne humeur et notre soutien mutuel qui nous a permis de réussir nos reconversions.

Je remercie également mes formateurs **Pascal DANGU**, **Philippe VIGUIER** et **Svetlana POPLAVSKAYA** pour leur transmission de savoir, leurs connaissances et conseils avisés.

Ma famille qui, grâce à leur soutien et attention, m'ont permis d'être ce que je suis actuellement.

Enfin je tiens particulièrement à remercier **ma compagne** pour son soutien indéfectible, sa patience pendant cette année, ses encouragements constants. Elle est pour moi, une vraie source d'inspiration et a été toujours à mes côtés même durant les moments difficiles.

Abstract

Contents

AVERTISSEMENT	2
Remerciements	3
Abstract	4
1. Introduction	8
2. Présentation de l'entreprise	9
2.1. Présentation du groupe	9
2.2. Implantations en France	10
2.3. Présentation de l'agence	11
2.4. Organisation des services	12
2.5. Organigramme de l'équipe SIVAC:	12
3. Presentation du projet	13
3.1. Problématique :	13
3.2. Contexte interministériel d'aide programme SIVAC	14
3.3. Présentation du programme SIVAC:	14
3.4. Présentation du projet	14
A. Glossaire	14
B. La plateforme PJS	15
C. Les applications partenaires:	16
3.5. Planification du projet	16
3.6. Les principes de la méthode agile:	17
3.7. Réunions	18
3.8. Environnement et contraintes techniques	21
A. Environment Technique	21
B. Contraintes du Projet	26

4. Analyse des besoins	26
4.1. Cartographie fonctionnelle	27
4.2. Vue d'ensemble du système	27
4.3. Principales fonctionnalités de l'application	28
A. Événement:	28
B. Individus:	29
4.4. Liens événements / individus	29
4.5. Dossier d'accompagnement	31
4.6. Diagrammes de cas d'utilisation :	31
A. Utilisateur	32
B. Événement	32
C. Individu	33
4.7. Diagrammes d'activité	34
A. Consulté l'écran d'accueil	34
B. Rechercher un événement	35
C. Rechercher un individu	36
5. Conception	36
5.1. Module MAI (Front End)	36
A. Composant squelette (sivac-layout)	37
B. Module (NgModule)	38
C. Store	40
D. Cache	41
E. Joindre le serveur	43
F. Formulaires	43
5.2. Module MAS (Back end)	45
A. Modèle de données	45
B. Présentation des packages	45
C. Configuration de l'application	46
D. Tests	46
E. Utilisation de Swagger / OpenApi	46

5.3. Sécurité applicative	46
A. Injection XML (A1)	47
B. Failles d'injection Javascript XSS(A3)	47
6. Implémentation	48
6.1. Liste des Listes à partagée	48
A. Base de données	48
B. Back end	49
C. Front end	51
6.2. Maintenance corrective	53
7. Tests et validation	55
7.1. Tests	55
A. End point	55
B. IHM	55
C. Maintenance corrective	56
7.2. Validation	57
A. Jenkins	57
B. SonarQube	57
C. Validation fonctionnel	58
8. Bilan de projet	58
9. Bilan et perspectives	58
10. Annexes	60

1. Introduction

Après avoir choisi la voie de la reconversion dans les métiers du développement informatique, j'ai entamé une formation de concepteur développeur d'application dispensée par l'AFPA de Balma de septembre 2021 à Août 2022.

Pour mettre en pratique les acquis durant la formation, il est nécessaire de réaliser une période d'application en entreprise (PAE) de 12 semaines. C'est dans ce cadre que j'ai effectué un stage, du 16 Mai au 10 Août 2022, au sein de la division AS SUD-OUEST de la société Inetum.

Plus largement, cette période en entreprise a été l'opportunité pour moi d'intégrer une équipe d'experts qui m'ont permis, non seulement de mettre en pratique mais également d'obtenir de nouvelles connaissances, autour d'une application qui tourne en production.

Tout au long de cette période, l'objectif général était l'élaboration de nouvelles fonctionnalités commandées par le ministère de la Justice. Pour ce faire j'ai dû mobiliser les connaissances acquises durant la formation afin de déterminer un certain nombre d'objectifs spécifiques pour atteindre ce dernier, à savoir :

- ✓ Découvrir le cœur de métier de l'application et intégrer les différents environnements de développement.
- ✓ Monter en compétence sur la partie fonctionnelle.
- ✓ Planifier et réaliser les tâches qui me sont affectées.
- ✓ Réaliser la maintenance évolutive et corrective demandée par le client.
- ✓ Préparer et exécuter les tests.
- ✓ Préparer et exécuter le déploiement sur les serveurs d'intégration.

2. Présentation de l'entreprise

2.1. Présentation du groupe

L'entreprise Inetum, anciennement appelée « Groupe Informatique Français » (Gfi) est une **entreprise de service numérique** (ESN) inscrite dans une dimension internationale.

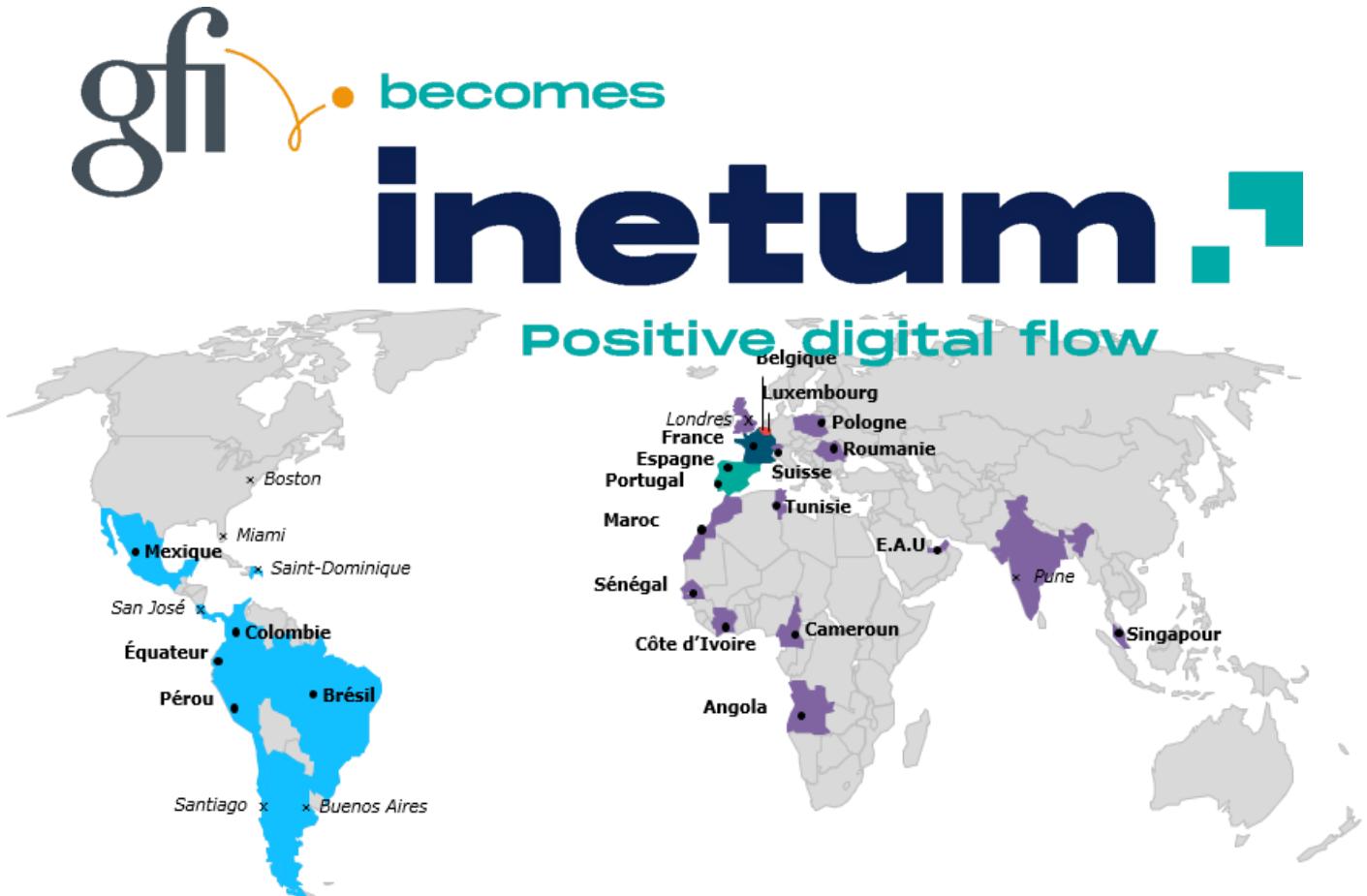
La volonté d'Inetum est explicitée dans son slogan « Positive digital flow », soutenir la transformation digitale des entreprises. Sa particularité est d'accompagner sur la durée de façon évolutive et personnalisée, grâce à une maîtrise agile qui permet de s'adapter aux besoins de leurs clients.



2.2. Implantations en France

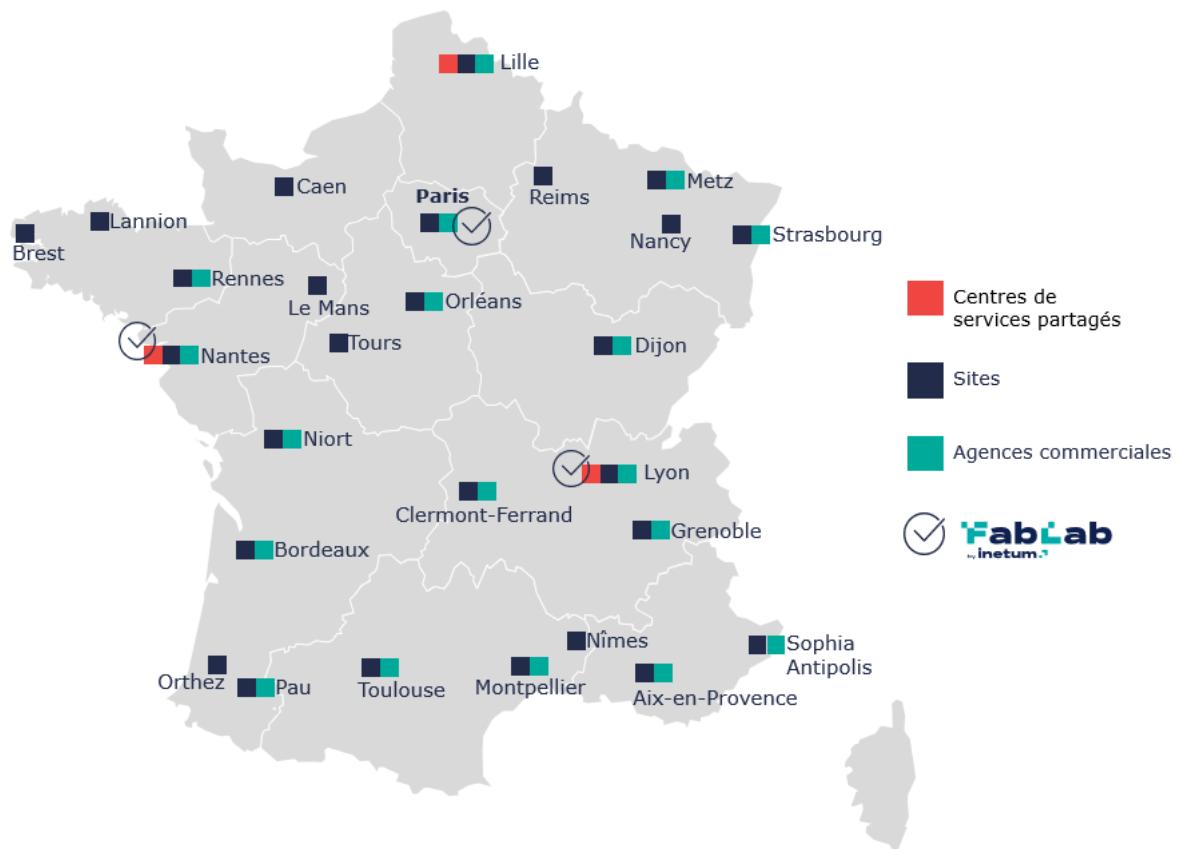
La France génère **39% du chiffre d'affaires** avec un effectif de plus de **11000 consultants**.

Avec une forte implantation sur le territoire par son historique puisqu'elle a été créée en 1970 sous le nom de Gfi « Groupe Français d'Informatique ».



En 2020 la société a choisi de se rebaptiser avec un nom qui puise ses racines latines dans le mot « incrementum » signifiant croissance, une nouvelle image pour incarner son **ADN**, un nom qui revendique **l'esprit entrepreneurial**, une **ambition** et un **esprit de conquête**.

45 sites y compris
18 agences commerciales



2.3. Présentation de l'agence

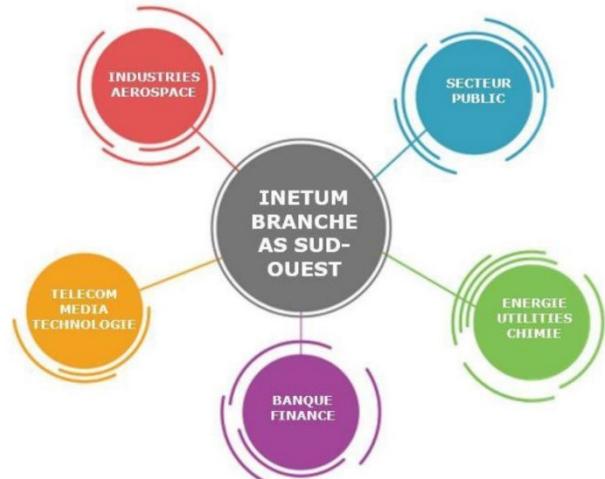
S'inscrivant dans cette dynamique de « Positive digital flow », la branche Sud-Ouest constitue un pôle important sur le territoire. Avec ses 940 consultants, son chiffre d'affaires de 108 millions d'euros et ses 5 centres de compétences (ALM/PLM, SIG, MOBILITE, SOLUTIONS TELECOMES et PERFORMANCE ET TRANSFORMATION), cette dernière quadrille le sud-ouest avec 4 implantations : Toulouse, Bordeaux, Pau et Orthez.

L'agence de Toulouse dans laquelle j'ai fait mon stage est la plus importante de la Branche Sud Ouest.



2.4. Organisation des services

Inetum AS Sud-Ouest (AS pour Applications Services) est organisé en plusieurs pôles représentant des secteurs d'activités diverses. Ces pôles entretiennent des partenariats avec différentes entreprises comme : le Ministère de la Justice français, Tigo, Orange, Rolls Royce en Europe, Auchan et pleins d'autres : Ville de Madrid, EDF, Gouvernement du Mexique, etc

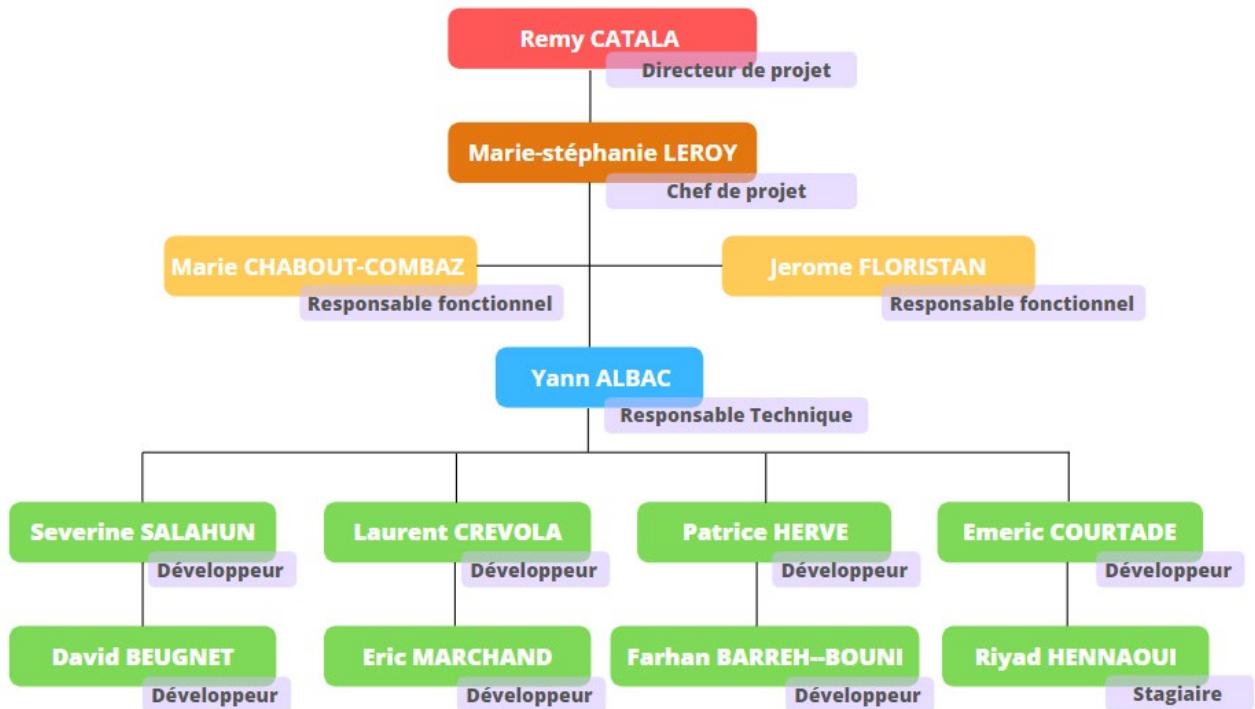


2.5. Organigramme de l'équipe SIVAC:

L'équipe SIVAC est composée d'une quinzaine de personnes.

Elle est divisée en deux parties :

- Les business analystes s'assurent de comprendre le besoin client, documentent correctement et transmettent aux développeurs dans un langage adapté.
- Les développeurs qui s'occupent de l'implémentation du front end et du back end en suivant les spécifications fonctionnelles détaillées rédigées par les business analystes.



3. Presentation du projet

3.1. Problématique :

A la suite d'un événement dramatique, les victimes ont des besoins très concrets, comme :

- Un accès privilégié à toute information relative à l'événement.
- Un accompagnement administratif dans l'ensemble de leurs formalités de santé, d'emploi, de logement et d'indemnisation.
- Un besoin de reconnaissance.

Et dans ces cas les Etats devraient :

- Mettre en place des points de contact appropriés pour l'information des victimes, concernant notamment leurs droits.
- De créer des organismes et des structures de soutien, d'assistance, de conseils pratiques et juridiques notamment en termes de réparation des préjudices subis.
- Veiller à leur fournir des informations appropriées, entre autres sur le suivi de l'enquête.
- Prendre la décision finale concernant les poursuites.
- Préciser la date et le lieu des audiences.
- Définir les conditions dans lesquelles il est possible de prendre connaissance des décisions rendues.

Les attentats commis ces dernières années ont montré les difficultés rencontrées par les différents acteurs publics pour gérer efficacement les informations concernant les victimes lors d'évènement exceptionnels.

3.2. Contexte interministériel d'aide programme SIVAC

Les attentats commis en 2015 et 2016, de même que l'ouragan Irma, ont montré les difficultés rencontrées par les différents acteurs publics pour gérer efficacement les informations concernant les victimes lors d'évènements exceptionnels.

La réunion interministérielle du 13 avril 2017 a donc décidé la création d'un Système d'Information Interministériel des Victimes d'Attentats et de Catastrophes (**SIVAC**) permettant de mutualiser l'information entre les acteurs concernés et de disposer d'une vision à 360° des victimes lors de la crise et dans la durée pour l'accompagnement des victimes et de leurs proches. Cette mission est confiée à la Délégation Interministérielle à l'Aide aux Victimes (DIAV).

Ce système doit couvrir les fonctions de dénombrement, d'aide à l'identification des victimes et à l'information des proches, d'établissement et de diffusion des listes de victimes, d'accompagnement et de suivi des victimes.

3.3. Présentation du programme SIVAC:

Deux enjeux principaux motivent ce projet :

1) Parvenir à gérer efficacement les événements produisant de nombreuses victimes : (fournir des données pertinentes et fiables pour les prises de décision et la communication, savoir accompagner le plus rapidement possible toutes les victimes et leurs proches).

2) Assurer à toutes les victimes et à leurs proches, à travers une démarche aussi simple que possible, une prise en charge et des aides rapides et personnalisées dans le respect de leurs droits et vie privée.

« Le projet sur lequel nous avons travaillé ces six derniers mois s'intitule « plateforme justice SIVAC ». C'est la partie du programme qui englobe uniquement le traitement de dossier qui concernent le terrorisme. Les données sur les autres types d'événements sont présentes dans l'outils mais sans aucune fonctionnalité spécifique à ceux-ci. Ces fonctionnalités seront développées pour les versions ultérieures. »

3.4. Présentation du projet

Pour que sur le plan judiciaire, l'Etat tienne dûment compte du statut, du rôle et des droits des victimes du terrorisme, y compris de leurs proches, il a été confié au Ministère de la Justice la mise en œuvre d'un nouveau système d'information nommé « Plateforme Justice SIVAC ».

A. Glossaire

Abréviations	Description
PJS	Plateforme Justice SIVAC
SIVAC	Système d'Information Interministériel des Victimes d'Attentats et de Catastrophes
Rôles	Il y a 3 types de rôles (Personne présente, Personne recherchée, Témoin)

Victime	Individu avec un rôle Personne présente ou un rôle Personne recherchée.
Les Personnes Présentes	Individu avec un rôle Personne Présente dans un événement = Victime = Victime Directe (VD)
Proche	Individu avec un lien de proximité = Victime indirecte (VI) = Rôle Relation entre un individu (victime) et un événement
Lien de proximité	Relation entre un individu (proche) et un rôle (c'est-à-dire un autre individu (victime) pour un événement
PNAT GESTION	Le Parquet National Anti-Terroriste
SDAT GESTION	La Sous-Direction Anti-Terroriste
ADMIN LOCAL	L'Administrateur local
BAVPA TERRORISME	Bureau de l'Aide aux Victimes et de la Politique Associative
FGTI	Fonds de Garantie des victimes
DIAV	La Délégation Interministérielle à l'Aide des Victimes
MJ	Ministère de la Justice
UIVC	Unité nationale d'Identification des Victimes de Catastrophes
UPIVC	Unité Police d'Identification des Victimes de Catastrophes
UGIVC	Unité Gendarmerie d'Identification des Victimes de Catastrophe
CNAF	La Caisse nationale des Allocations Familiales
GESTION DGFIP	La Direction Générale des Finances Publiques
UIVC	Unité nationale d'Identification des Victimes de Catastrophes
UPIVC	Unité Police d'Identification des Victimes de Catastrophes

B. La plateforme PJS

En tant que composante du programme SIVAC, la plateforme PJS a pour objectifs :

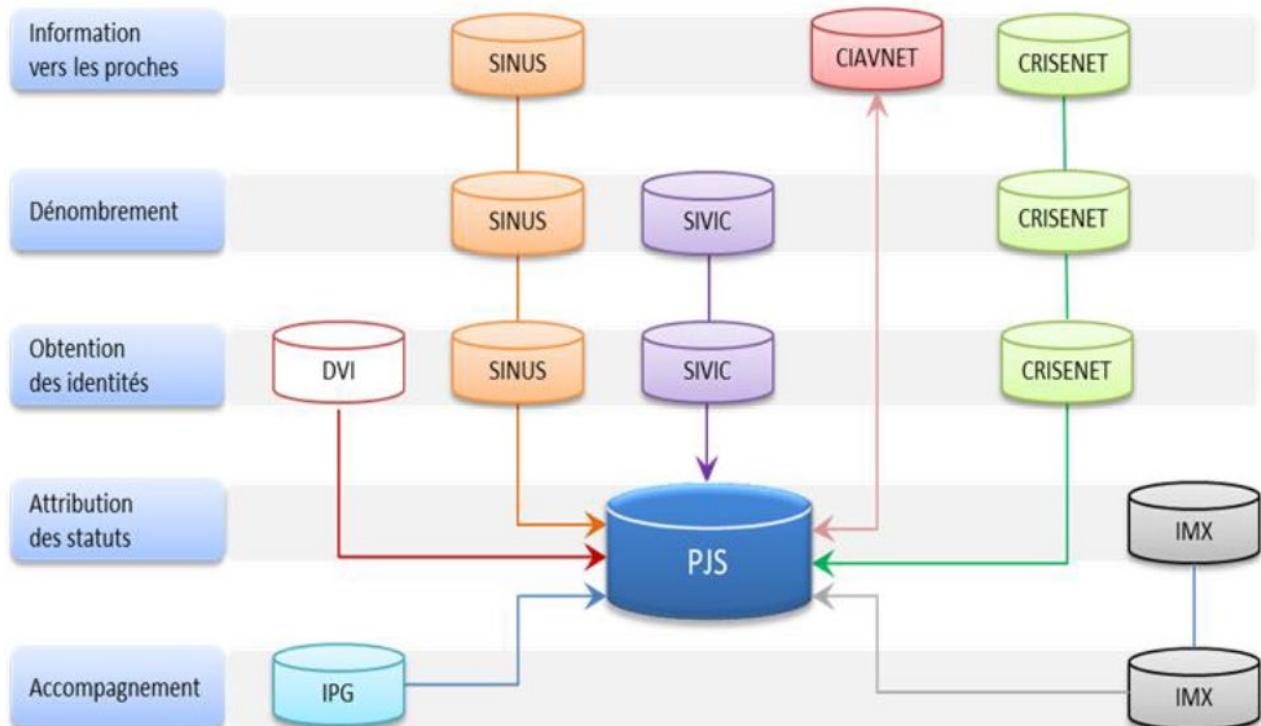
- De mettre en place des systèmes d'échanges avec toutes les applications partenaires participant au programme SIVAC pour collecter, rapprocher, mettre à jour, diffuser et centraliser les données liées aux victimes et à leurs proches.
- De mettre à disposition de chaque acteur, une interface de consultation et de gestion des données dont il a la charge, ainsi qu'une vision globale et partagée des événements et des individus gérés au sein du programme SIVAC.
- De permettre aux victimes enregistrées dans PJS, de consulter leurs données et d'être informées sur l'avancement de leurs démarches.

- La mise en place de PJS est envisagée de manière incrémentale. Le premier cycle de déploiement vise déjà à couvrir la prise en charge par les acteurs français des victimes et le suivi d'attentats terroristes en France et à l'étranger. Les versions ultérieures seront notamment destinées à gérer l'élargissement du périmètre aux autres types d'événements et aux fonctionnalités supplémentaires nécessaires.

C. Les applications partenaires:

Plusieurs applications partenaires sont déjà mises en place, et sont utilisées par les différents acteurs, impliqués dans le programme SIVAC.

Le schéma suivant présente les flux d'échanges qui pourraient être mis en place entre PJS et ces applications positionnées en fonction des processus métier pris en compte.



3.5. Planification du projet

La gestion de projet s'impose dans les structures de toute taille comme un mode d'organisation particulièrement efficace.

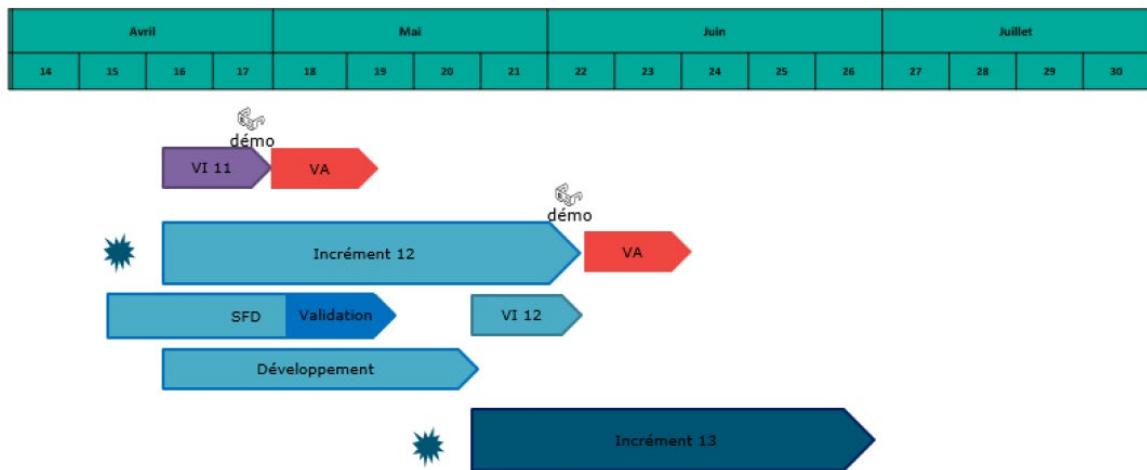
Chez Inetum nous basculons petit à petit sur les méthodes agiles plus précisément sur la méthode SCRUM.

Au cœur de la méthode agile réside une plus grande implication du client et une meilleure réactivité des équipes.

Notre cheffe de projet dirige le projet, anime l'équipe et assure le bon déroulement des tâches afin de satisfaire les exigences des clients.

La planification du projet est en mode hybride : côté client le mode est plutôt cycle en V, alors que côté équipe nous utilisons la méthode agile (en partie car le product owner n'est pas présent).

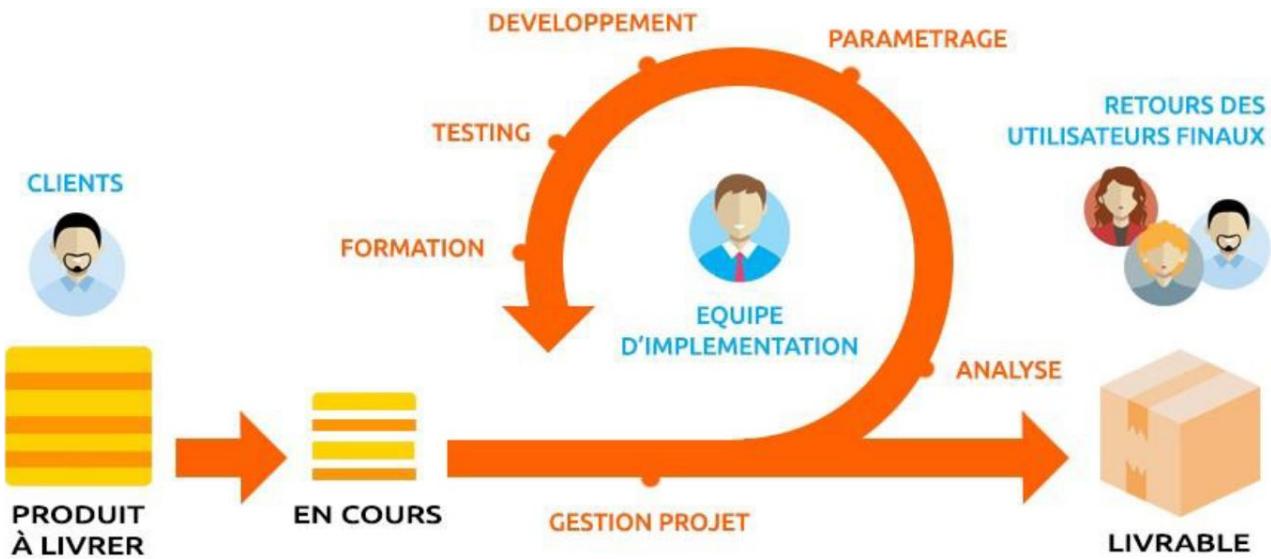
Roadmap



La Roadmap du projet est une séquence d'incrémentations (sprints), dans lesquels sont définis un nombre de fonctionnalités à livrer au client avec l'écriture des spécifications fonctionnelles, du développement et la validation de celles-ci. Chaque incrément dure 6 semaines et se termine par une démonstration aux utilisateurs de l'application.

3.6. Les principes de la méthode agile:

- **La collaboration** : c'est à dire la communication avec le client plutôt que la contractualisation des relations.
- **L'équipe** : le travail en équipe avec une cohésion.
- **L'application** : favoriser les fonctionnalités opérationnelles plutôt que la documentation exhaustive.
- **L'acceptation** : accepter les modifications et être souple.



3.7. Réunions

- **Sprint planning (planification du sprint)**

Le sprint planning (ou kick-off) est une réunion qui se déroule le premier jour du sprint. Il dure 6 semaines et c'est pendant le sprint planning que les développeurs vont concevoir et tester de nouvelles fonctionnalités.

Cette réunion est faite avec le client durant laquelle sont abordées les anomalies rencontrées et les corrections à réaliser.

- **Daily Scrum (mêlée quotidienne)**

Le Daily est une réunion très rapide qui a lieu chaque jour en fin de matinée pour ne pas couper la journée. Chaque participant prend la parole pour répondre brièvement à trois questions :

- ✓ Qu'est-ce que j'ai fait hier ?
- ✓ Qu'est-ce que je vais faire aujourd'hui ?
- ✓ Est-ce que je rencontre actuellement des problèmes ?

C'est également l'occasion pour l'équipe d'avoir une vue globale des tâches en cours ainsi que des difficultés rencontrées.

Le tableau de bord Jira montre l'état des tickets pour le sprint 14. Les colonnes sont : À FAIRE, BLOQUÉ, CONCEPTION, EN DEV/UTU, MERGE REQUEST, A DEPLOYER, EN TEST, FAIT. Des filtres rapides sont disponibles pour les tickets uniques et les dernièrement mis à jour.

A FAIRE	BLOQUÉ	CONCEPTION	EN DEV/UTU	MERGE REQUEST	A DEPLOYER	EN TEST	FAIT
MIPIS-206 Import 0208866, MANTIS	MIPIS-1508 Internationalisation Objectifs secondaires	MIPIS-1949 PoC sécurisation des flux 2	MIPIS-1508 Internationalisation	MIPIS-2020 [Test partagé] Modification de l'export (tableau VO/VI)	MIPIS-2011 [Conf parquet] Remplacer la liste déroulante par un toggle	MIPIS-2019 [Conf parquet] Remplacer la liste déroulante par un toggle	MIPIS-1938 [Pochette individuel] Performances dégradées pour les profils BAVPA, FGFI
MIPIS-2029 test de perf / test de charge	MIPIS-1533 Internationalisation - Choix de la langue	MIPIS-1534 Internationalisation, Service de gestion et sauvegarde du choix de la langue	MIPIS-1534 Internationalisation, Service de gestion et sauvegarde du choix de la langue	MIPIS-2021 [LP/LAF] Ajout de "-" pour les numéros tel dans les exports LAP et LP	MIPIS-2051 0209737: LP partagée mise en attente de publication, mais aucun	MIPIS-2053 [Conf parquet] Modification affichage	MIPIS-1939 [Pochette individuel] Performances dégradées pour les profils BAVPA, FGFI
MIPIS-2030 qualité / sécurité de code	MIPIS-2060 Internationalisation - mise à jour du CTD	MIPIS-1535 Internationalisation - Modification des services d'export	MIPIS-1535 Internationalisation - Modification des services d'export	MIPIS-2032 [export zed] mise en place	MIPIS-2054 [Conf parquet] Script de reprise de données	MIPIS-2055 [Conf parquet] Mise à jour du mondiale de dimension et	MIPIS-1939 [Pochette individuel] Performances dégradées pour les profils BAVPA, FGFI
MIPIS-2024 Traitement de la CVE-2022-26520 + postgresql	MIPIS-2025 Sécurisation des flux partie 2	MIPIS-1536 Internationalisation - Mise à jour du script création utilisateur / Reprise de	MIPIS-1536 Internationalisation - Mise à jour du script création utilisateur / Reprise de	MIPIS-2036 [export zed] industrialisation mtt-zed			MIPIS-1939 [Pochette individuel] Performances dégradées pour les profils BAVPA, FGFI

- **Weekly (hebdomadaire)**

Le Weekly est une réunion qui a lieu chaque vendredi pour faire un point sur les tâches en cours, les problèmes éventuels et l'humeur de la semaine. Le but est de favoriser la progression et l'implication totale de l'équipe. Il s'agit donc d'un moment de partage où chacun peut s'exprimer.

Les sujets à échanger dans le weekly :

- ✓ **1 - Daily**
- ✓ **2 - Weekly – commun à toute équipe**
 - Résumé de la météo hebdomadaire
 - Actualités importantes
 - Difficultés rencontrées
 - Congés/absence
- ✓ **3 - Informations / sujets communs**
- ✓ **4 - Échéances et objectifs passés et à venir**

L'équipe utilise l'outil « Miro », une application qui crée un tableau blanc collaboratif qui permet de répondre à différents besoins : réunions, atelier, etc ...

Dans notre cas cela nous sert, entre autre, pour la météo de la semaine, la présence au bureau la semaine suivante, les idées, les irritants, etc .

The Miro board displays two main sections:

- Présence bureau INETUM:** A grid showing the presence of team members (Yann, Patrice, Marie-Stéph, David, Emeric, Julien, Séverine, Rémy, Laurent, Marie, Farhan, Jérôme, Eric, Riyad) from Monday to Friday. Icons represent different working statuses: house (present), office (INETUM office), palm tree (remote work), and cloud (absent). A callout box provides instructions: "On copie la croix sur le jour de présence souhaité (2 jours/semaine) Inetum : présentiel bureau inetum Palmier : congé/absent Maison : télétravail".
- Météo de l'équipe:** A table showing the weather forecast for each team member from Monday to Friday. The table includes columns for Member, Monday, Tuesday, Wednesday, Thursday, Friday, and "Votre fait marquant" (notable action). The "Votre fait marquant" column contains various notes such as "Livraison ...", "Fin de sprint", "arrivée de Jérôme / livraison incr13 / démo", "Démô", "Jerôme / Audit RGAA", "RAS", "Rétro", "liste evt vi vd", "Zen (mais la chaleur revient, ça va changer!)", "RAS", "Content d'être opérationnel rapidement", "RAS", and "Rétro Patrice".

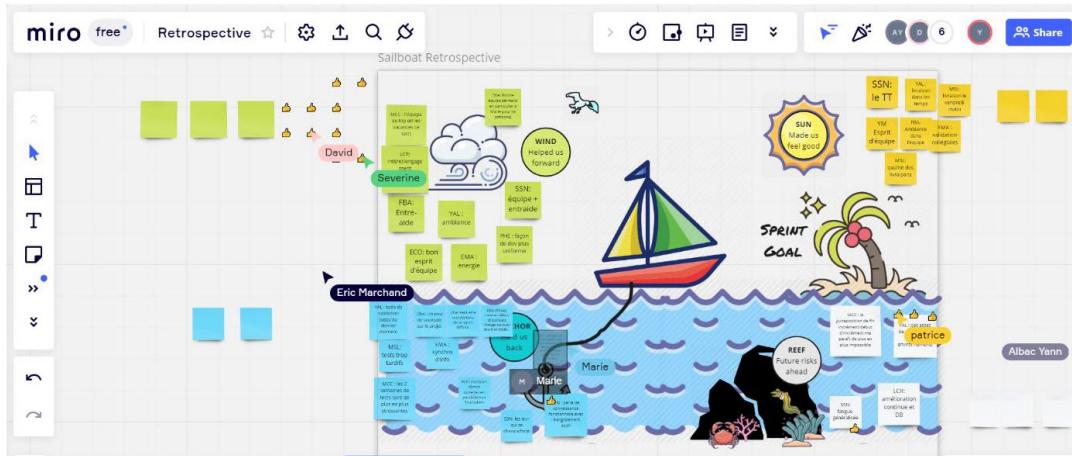
- **Sprint review (fin de sprint)**

Le sprint review est une réunion où sont présentés aux parties prenantes les différents livrables terminés. Plus qu'une simple présentation, c'est l'occasion de faire une démonstration en condition réelle afin de s'assurer que le produit réponde aux besoins exprimés par le client.

- **Rétrospective (fin de sprint)**

Cette rétrospective est une réunion qui vient clôturer un sprint. Elle a lieu après la revue de sprint et avant la réunion de planification du sprint suivant.

Elle permet de faire le point sur le sprint qui vient de s'achever, afin d'en tirer des enseignements et devenir encore plus efficace lors du prochain. L'idée est de suivre une démarche d'amélioration continue.



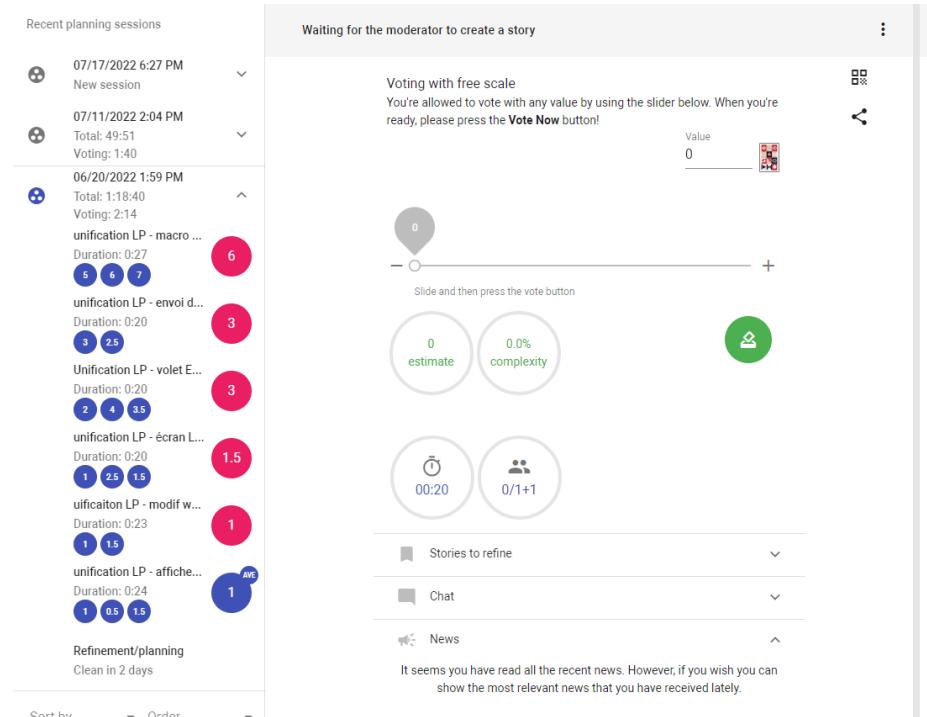
- **REX (à la demande)**

Réunion sur le retour d'expérience utilisateur, planifiée après chaque livraison, elle permet d'avoir les retours clients afin de s'assurer que le produit livré correspond bien à ses besoins.

- **Session de chiffrage (hebdomadaire)**

Cette réunion permet d'estimer le nombre de jours nécessaires au développement d'une ou plusieurs fonctionnalité(s). Après description de la fonctionnalité par la cheffe de projet, les développeurs débattent sur le meilleur découpage et s'accordent pour définir un nombre de jours pour la réalisation de chaque tâche, en utilisant le poker planning.

Le principe est que chaque développeur a 20 secondes pour estimer le temps nécessaire à la réalisation de la tâche. A la fin du temps imparti, l'équipe découvre les estimations de chaque membre et débat si l'écart entre elles est important.



- **Synchronisation technique (hebdomadaire)**

Le but de cette réunion est de partager entre développeurs des éléments techniques mis en place sur le projet. Ce temps peut être consacré si l'un d'entre eux souhaite monter en compétence sur un aspect particulier.

3.8. Environnement et contraintes techniques

A. Environment Technique

1. Eclipse :



L'IDE (environnement de développement intégré) Eclipse est un projet décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation du même nom, visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent. Il est d'abord conçu pour le langage Java mais ses nombreux plugins en font un environnement de développement pour de nombreux autres langages de programmation tel que : C/C++, Python, PHP, Ruby etc ..

2. IntelliJ IDEA community :



Également appelé « IntelliJ », « IDEA » ou « IDJ » est un environnement de développement intégré (en anglais Integrated Development Environment - IDE) destiné au développement de logiciels informatiques reposant sur la technologie Java. Il est développé par JetBrains (anciennement « IntelliJ ») et disponible en deux versions, l'une communautaire (open source) sous licence Apache 2, et l'autre propriétaire, protégée par une licence commerciale. Tous deux supportent les langages de programmation Java, Kotlin, Groovy et Scala.

3. Visual Studio Code :



Est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS. Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré. Les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires.

Le code source de Visual Studio Code provient du projet logiciel libre et open source VSCode de Microsoft publié sous la licence MIT permissive. Mais les binaires compilés constituent un freeware, c'est-à-dire un logiciel gratuit pour toute utilisation mais privatif.

Dans un sondage auprès de développeurs réalisé par Stack Overflow en 2021, Visual Studio Code a été classé comme l'outil d'environnement de développement le plus populaire, avec 71,06% des 82 277 répondants déclarant l'utiliser.

4. Java :



Java est un langage de programmation orienté objet qui reprend en grande partie la syntaxe du langage C++. Néanmoins, Java a été épuré des concepts les plus subtils du C++ et à la fois les plus déroutants, tels que les pointeurs et références, ou l'héritage multiple contourné par l'implémentation des interfaces. De plus, depuis la version 8, l'arrivée des interfaces fonctionnelles introduit l'héritage multiple (sans la gestion des attributs) avec ses avantages et inconvénients tels que l'héritage en diamant. Les concepteurs ont privilégié l'approche orientée objet de sorte qu'en Java, tout est objet à l'exception des types primitifs (nombres entiers, nombres à virgule flottante, etc.) qui ont cependant leurs variantes qui héritent de l'objet Object (Integer, Float...).

5. Maven :

Maven Apache Maven (couramment appelé Maven) est un outil de gestion et d'automatisation de production des projets logiciels Java en général et Java EE en particulier. Il est utilisé pour automatiser l'intégration continue lors d'un développement de logiciel. Maven est géré par l'organisation Apache Software Foundation. L'outil était précédemment une branche de l'organisation Jakarta Project.

L'objectif recherché est de produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication.

Maven utilise un paradigme connu sous le nom de Project Object Model (POM) afin de décrire un projet logiciel, ses dépendances avec des modules externes et l'ordre à suivre pour sa production. Il est livré avec un grand nombre de tâches prédéfinies, comme la compilation de code Java ou encore sa modularisation.

6. Hibernate :



Hibernate est un framework open source gérant la persistance des objets en base de données relationnelle.

Hibernate est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un développement client lourd, que dans un environnement web léger de type Apache Tomcat ou dans un environnement Java EE complet : WebSphere, JBoss Application Server et Oracle WebLogic Server.

Hibernate apporte une solution aux problèmes d'adaptation entre le paradigme objet et les SGBD en remplaçant les accès à la base de données par des appels à des méthodes objet de haut niveau.

7. Spring & Spring Boot :



Spring est un framework open source pour construire et définir l'infrastructure d'une application Java, dont il facilite le développement et les tests.

Spring s'appuie principalement sur l'intégration de trois concepts clés :

- L'inversion de contrôle est assurée de deux façons différentes : la recherche de dépendances et l'injection de dépendances,
- La programmation orientée aspect,
- Une couche d'abstraction.

La couche d'abstraction permet d'intégrer d'autres frameworks et bibliothèques avec une plus grande facilité. Cela se fait par l'apport ou non de couches d'abstraction spécifiques à des frameworks particuliers. Il est ainsi possible d'intégrer un module d'envoi de mails plus facilement.

Spring Boot facilite la création d'applications Spring autonomes de qualité production afin de les exécuter simplement.

Il permet de simplifier le développement en Spring (configuration maven, programme principal, etc.). Les éléments nécessaires pour une application stand-alone en SpringBoot sont :

- un fichier pom.xml récupérant les éléments SpringBoot (pom parent)
- un fichier Main.java qui initialise le conteneur Spring
- Des annotations dans les classes

8. Postgres :



PostgreSQL est un système de gestion de base de données relationnelle orienté objet puissant et open source qui est capable de prendre en charge en toute sécurité les charges de travail de données les plus complexes. Alors que MySQL donne la priorité à l'évolutivité et aux performances, Postgres donne la priorité à la conformité et à l'extensibilité SQL.

Les entreprises qui souhaitent maintenir un haut niveau d'intégrité et de personnalisation de leurs données choisissent généralement Postgres en raison de sa fiabilité, l'intégrité de ses données, la robustesse de ses fonctionnalités, et parce qu'il fournit des solutions toujours performantes et

innovantes. PostgreSQL fonctionne sur tous les principaux systèmes d'exploitation et est conforme à ACID depuis 2001.

Postgres peut être téléchargé gratuitement et déployé sur du matériel standard, ou peut être exécuté dans le Cloud par le biais d'une variété de fournisseurs. Bien que Postgres soit riche en fonctionnalités et adapté aux charges de travail OLAP, les performances de Postgres ont tendance à atteindre une limite lorsque les volumes de données dépassent plusieurs téraoctets.

Contrairement aux autres bases de données transactionnelles, PostgreSQL est implémenté sur un seul serveur et n'est généralement pas conçu pour distribuer ses fonctions de stockage ou de calcul sur plusieurs nœuds. Bien qu'il soit possible d'utiliser des techniques de regroupement, de réPLICATION et de mutualisation pour faire évoluer votre cluster Postgres en termes de performances et de capacité supplémentaire, ces solutions sont complexes et peu courantes. De nombreuses entreprises choisissent d'augmenter la taille de Postgres en adaptant verticalement la base de données, ce qui implique l'achat d'un serveur plus grand et plus puissant.

9. DBeaver :



DBeaver est un logiciel permettant l'administration et le requêtage de base de données. Pour les bases de données relationnelles, il utilise un driver JDBC. Pour les autres bases de données (NoSQL), il utilise des pilotes de base de données propriétaire. Il fournit un éditeur qui prend en charge la complétion de code et coloration syntaxique. Ce logiciel est écrit en Java et est basé sur la plate-forme Eclipse.

10. Gitlab :



GitLab est un logiciel libre de forge basé sur git proposant les fonctionnalités de wiki, un système de suivi des bugs, l'intégration continue et la livraison continue.

```

SIVAC-PIS > pjs-core > Merge requests > !715
Open Created 5 days ago by riyad.hennaoui Developer
Edit Mark as draft ▾
Jira 2051 ne pas mettre en attente de publication la liste partagée si la confidentialité parquet est déjà prise en compte sur l'événement unitaire après création de l'événement global
Overview 3 Commits 20 Changes 4
2 unresolved threads ▾ ▾ ▾
4 files +340 -10 Viewed ▾
pj-s-core
Project information
Repository
Issues 0
Merge requests 5
CI/CD
Security & Compliance
Deployments
Monitor
Infrastructure
Analytics
Wiki
Snippets
pj-s-core
pj-s-mas-bo/src/main/java/fr/gouv/justice/penal/sivac/api/evenement/RegroupementEvenementsCommonService.java
RegroupementEvenementsCommonService.java +78 -6
pj-s-mas-bo/src/main/java/fr/gouv/justice/penal/sivac/api/evenement/DataSetBuilder.java +174 -0
pj-s-mas-bo/src/main/java/fr/gouv/justice/penal/sivac/api/evenement/listes/partagee/ListePartageeCommonServiceTest.java +74 -4
pj-s-mas-bo/src/main/java/fr/gouv/justice/penal/sivac/api/evenement/listes/partagee/RegroupementEvenementsCommonServiceTest.java +14 -0
Search files (Ctrl+P)
pj-s-mas-bo/src/main/java/fr/gouv/justice/penal/sivac/api/evenement/RegroupementEvenementsCommonService.java
1 package fr.gouv.justice.penal.sivac.api.evenement;
2 import java.text.MessageFormat;
3 import java.util.ArrayList;
4 import java.util.Date;
5 import java.util.List;
6 import java.util.Set;
7 import java.util.stream.Collectors;
8 import java.util.stream.Collectors;
9
10 import java.util.Objects;
11 import java.util.Collections;
12
13 import fr.gouv.justice.penal.sivac.api.evenement.listes.IndividuPartage;
14 import fr.gouv.justice.penal.sivac.api.evenement.listes.partagee.ListePartagee;
15 import fr.gouv.justice.penal.sivac.api.evenement.listes.partagee.ListePartageeRepository;
16 import fr.gouv.justice.penal.sivac.api.individu.Individu;
17 import fr.gouv.justice.penal.sivac.api.referentiel.ReferentielConstantes;
18 import org.apache.commons.lang3.tuple.Pair;
19 import org.slf4j.Logger;
20 import org.slf4j.LoggerFactory;

```

Nous utilisons seulement la partie versionning de cet outil, plusieurs règles sont appliquées à l'équipe, chaque branche doit être nommée en suivant le format suivant v<V>l<L>i<I>_MJPJS-XXX[_<description>] (ex: v11i3_MJPJS-582_rapprochement_manuel) :

- V correspond au n° de version en cours de développement.
- L correspond au n° de lot.
- I correspond au n° de l'itération.
- Description correspond au label facultatif de description de la branche.
- XXX correspond au numéro de la tâche Jira associée.

Une procédure particulière est appliquée pour les merge requests :

Pré-requis : respecter les critères de la Definition of Done (DOD) qui sont :

- Conception effectuée, présentée et rédigée dans le CTD (conception technique détaillé)
- Tests Unitaires / Intégrations développés et valides
- Les exigences sont présentes dans les commentaires du code
- La revue du code est faite via MR
- Pas de nouveaux bugs / vulnérabilités présente sur Sonar
- Tests automatisés (pour plus tard une fois mis en place)
- Jira à jour

L'étape suivante est de faire un pull rebase afin de récupérer les modifications postérieures à la création de la branche sur la branche develop et de résoudre les conflits si besoin (PI, après avoir récupérer les modifications de develop, les commits seront ajoutés en dernier). Dans le cas contraire, le merge ne sera pas possible et sera bloqué par Gitlab.

Le code proposé devra être exécuté sur un pipeline Jenkins spécifique pour s'assurer que tout se passe bien. Enfin une vérification de la qualité du code est faite par sonarQube.

Chaque MR est soumise à relecture par au minimum deux développeurs, si ceux-ci ne soulèvent pas de commentaire, la branche peut être mergée sur la branche principale « develop ». Dans le cas contraire l'auteur doit effectuer les modifications ou exposer ses choix techniques, jusqu'à fermeture des commentaires.

11. Angular :



Angular est un framework côté client, open source, basé sur TypeScript, et codirigé par l'équipe du projet « Angular » à Google et par une communauté de particuliers et de sociétés.

Il permet la création d'applications Web et plus particulièrement d'applications web monopage : des applications web accessibles via une page web unique qui permet de fluidifier l'expérience utilisateur et d'éviter les chargements de pages à chaque nouvelle action. Le framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d'architecture qui a largement fait ses preuves et qui permet une forte maintenabilité et une amélioration du travail collaboratif.

12. Cypress :



Cypress.io, outil open source permettant de mettre facilement en place des tests d'applications utilisant des frameworks JavaScript modernes. Il permettra de tester tout ce qui fonctionne dans le navigateur.

Pour le projet cette solution est utilisée pour effectuer des tests RGAA (Référentiel général d'amélioration de l'accessibilité)

Les services publics numériques et certains services privés ont l'obligation d'être accessibles de façon équivalente à tout citoyen, qu'il soit ou non en situation de handicap (visuel, auditif, moteur, trouble dys...). Un service numérique accessible est plus facile à utiliser pour les personnes porteuses d'un handicap et de meilleure qualité pour tous.

Pour faciliter la mise en œuvre de l'accessibilité numérique, la DINUM édite depuis 2009 le référentiel général d'amélioration de l'accessibilité – RGAA, créé pour mettre en œuvre l'article 47 de la loi handicap de 2005 et son décret d'application actualisé en 2019. Il fait régulièrement l'objet de nouvelles versions et mises à jour afin de s'adapter aux évolutions du Web mais aussi aux changements de normes et réglementations.

13. Jenkins :

 **Jenkins** Jenkins est un outil open source de serveur d'automatisation. Il aide à automatiser les parties du développement logiciel liées au build, aux tests et au déploiement, et facilite l'intégration continue et la livraison continue. Écrit en Java, Jenkins fonctionne dans un conteneur de servlets tel qu'Apache Tomcat, ou en mode autonome avec son propre serveur Web embarqué.

Il s'interface avec des systèmes de gestion de versions tels que CVS, Git et Subversion. Il exécute des projets basés sur Apache Ant et Apache Maven aussi bien que des scripts arbitraires en shell Unix ou batch Windows.

Les générations de projets peuvent être amorcées par différents moyens, tels que des mécanismes de planification similaires au cron, des systèmes de dépendances entre générations, ou par des requêtes sur certaines URL spécifiques.

14. SonarQube :

 SonarQube (précédemment Sonar) est un logiciel libre de qualimétrie en continu de code. Il aide à la détection, la classification et la résolution de défaut dans le code source. Il permet également d'identifier les duplications de code, de mesurer le niveau de documentation et connaître la couverture de test déployée.

SonarQube permet une surveillance continue de la qualité du code grâce à son interface web permettant de voir les défauts de l'ensemble du code et ceux ajoutés par la nouvelle version. Le logiciel peut être interfacé avec un système d'automatisation comme Jenkins pour inclure l'analyse comme une extension du développement.

15. Docker :

 Docker est une plateforme permettant de lancer certaines applications dans des conteneurs logiciels.

Selon la firme de recherche sur l'industrie 451 Research, « Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur ». Il ne s'agit pas de virtualisation, mais de conteneurisation, une forme plus légère qui s'appuie sur certaines parties de la machine hôte pour son fonctionnement. Cette approche permet d'accroître la flexibilité et la portabilité d'exécution d'une application, laquelle va pouvoir tourner de façon fiable et prévisible sur une grande variété de machines hôtes, que ce soit sur la machine locale, un cloud privé ou public, une machine nue, etc.

Techniquement, Docker étend le format de conteneur Linux standard, LXC, avec une API de haut niveau fournissant une solution pratique de virtualisation qui exécute les processus de façon isolée. Pour ce faire, Docker utilise entre autres LXC, cgroups et le noyau Linux lui-même. Contrairement aux machines virtuelles traditionnelles, un conteneur Docker n'inclut pas de système d'exploitation, mais s'appuie au contraire sur les fonctionnalités du système d'exploitation fournies par la machine hôte.

La technologie de conteneur de Docker peut être utilisée pour étendre des systèmes distribués de façon à ce qu'ils s'exécutent de manière autonome depuis une seule machine physique ou une seule instance par nœud. Cela permet aux nœuds d'être déployés au fur et à mesure que les ressources sont disponibles, offrant un déploiement transparent et similaire aux PaaS pour des systèmes comme Apache Cassandra, Riak, ou d'autres systèmes distribués.

16. Microsoft Teams :

 Microsoft Teams est une application de communication collaborative, officiellement lancée par son propriétaire Microsoft en novembre 2016. L'utilisation de cette application est faite par l'équipe pour l'ensemble des échanges et des réunions.

17. Jira :

 L'outil Jira est destiné aux sociétés désireuses de mettre en place un fonctionnement en méthode agile et faciliter le travail des utilisateurs concernés dans leur organisation. Il permet notamment la création et la planification de tâches via un système de rédaction et de gestion des récits utilisateurs.

Jira Software est à destination des équipes de développeurs pour la mise en place de méthode Kanban et d'organisation de type scrum. Il permet le découpage des projets en récits utilisateurs (tâches, composants) et leur affectation aux développeurs.

B. Contraintes du Projet

Les contraintes de l'application sont multiples. Elles s'expliquent généralement par la rigidité du service de sécurité informatique du ministère de la justice : ce service impose des versions non récentes voir plus maintenue. Une migration vers des versions plus récentes a été demandée par le ministère récemment. Cependant, les besoins et les chiffages n'étaient pas encore effectués pendant ma période en entreprise. Le choix de cette migration est dû essentiellement aux risques de failles trop importantes.

Les versions problématiques :

- Angular 8.2.14
- Angular Material 7.3.7
- Spring Boot
- Navigateur cible Firefox 52.

Une autre contrainte est le mode fonctionnement hybride entre le ministère de la justice et l'équipe. En effet, étant donné que le ministère n'est pas intégré dans la méthode agile, cela engendre des contraintes concernant l'écriture des Spécifications fonctionnelles détaillées. Parfois elles ne sont pas complètes au début du développement des fonctionnalités. Se rajoute à cela le chevauchement des incrémentations car chaque nouvel incrément débute deux semaines avant la fin de l'incrément courant.

4. Analyse des besoins

Au début de ma période en entreprise, la cheffe de projet m'a accompagné afin de monter en compétences. Cela s'est traduit par la présentation des fonctionnalités de l'application, et à l'issue de chaque présentation, j'ai effectué des travaux pratiques pour qu'elle valide la compréhension de cette fonctionnalité.

Au total, j'ai réalisé six travaux pratiques afin de me familiariser avec la partie fonctionnelle de l'application.

La réflexion autour des besoins du client est le point de départ de tout processus de développement. Cette étape consiste à analyser la situation pour tenir compte des contraintes et des risques. Les questions principales auxquelles on cherche à répondre sont :

- À quoi et à qui va servir le système qu'on cherche à développer ?
- Comment ce système va-t-il être utilisé ?
- Pourquoi faire ?

Le système doit assurer le suivi des victimes à partir de la création d'événements jusqu'à gérer la liste partagée des individus.

Pour cela, deux fois par semaine des réunions sont organisées et animées par l'équipe fonctionnelle PJS avec les différents intervenants côté clients afin d'analyser leurs besoins.

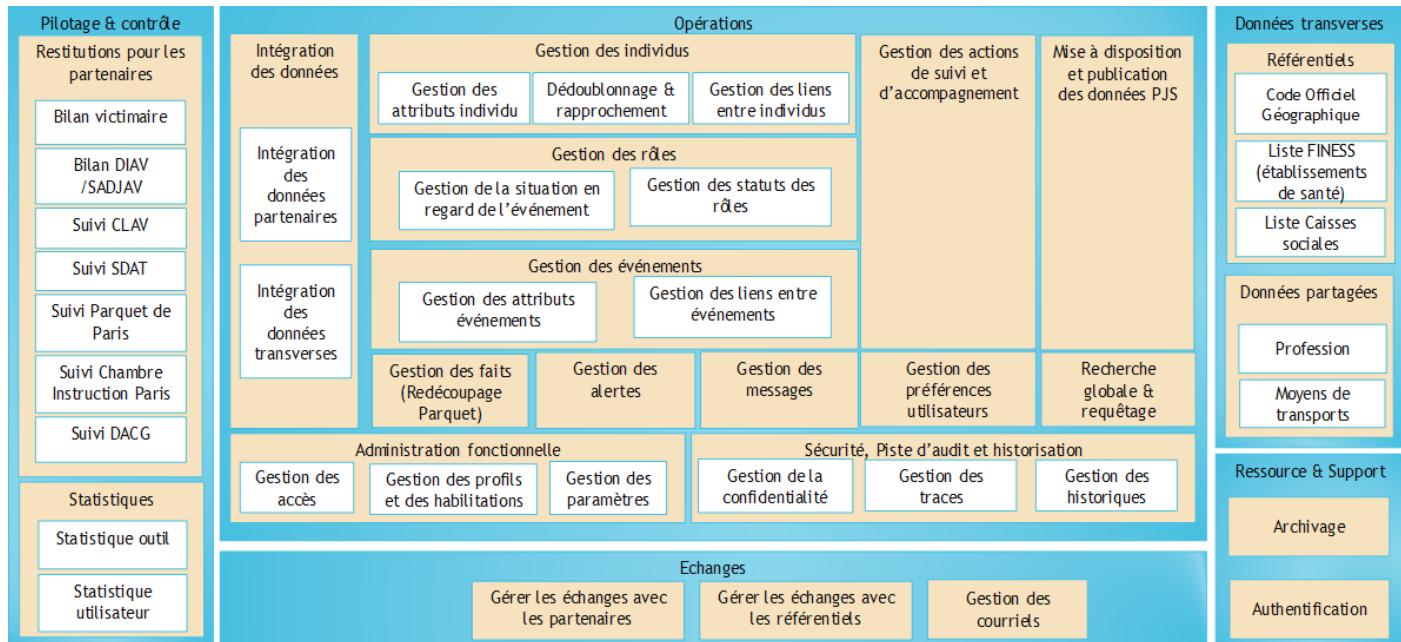
Ces ateliers visent à comprendre et lister les exigences et les éléments attendus par le client.

Les ateliers se divisent en plusieurs étapes :

- Définition du périmètre.

- Suivi du cycle de vie et identification des cas d'utilisation.
- Listage des fonctions attendues et les caractériser.

4.1. Cartographie fonctionnelle

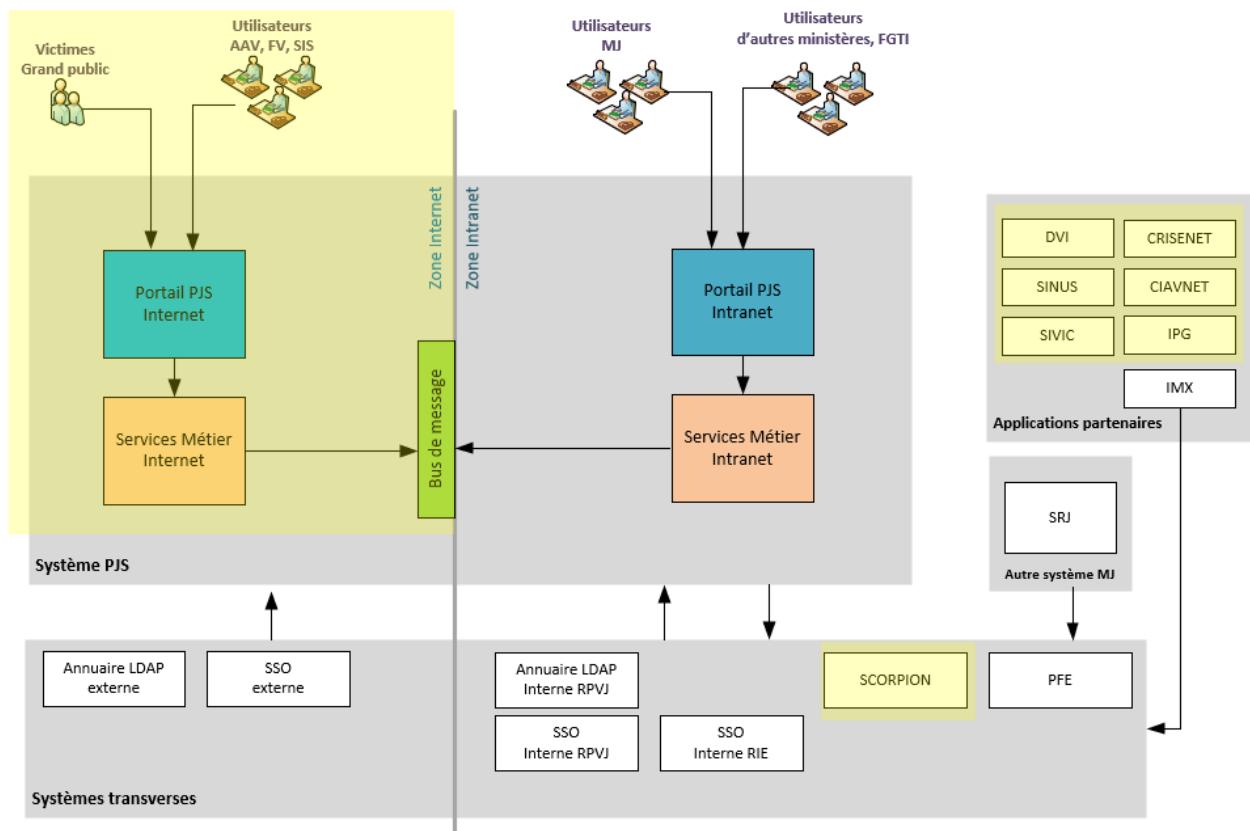


4.2. Vue d'ensemble du système

L'application PJS se présente sous la forme de deux portails web permettant l'accès aux utilisateurs en fonction de leur provenance :

- ✓ Portail intranet pour les utilisateurs du ministère de la Justice et pour les utilisateurs d'autres ministères et ceux du FGCI .
- ✓ Portail intranet pour le grand public (victimes) et pour les utilisateurs des associations/partenaires (AAV, FV, SIS).

Sous-système	Portail	Acteurs	Couverture fonctionnelle
PJS-BO	Portail PJS intranet	Utilisateurs du ministère, d'autres ministères et FGCI	<ul style="list-style-type: none"> ➢ Gestion des données Victimes & Evénement ➢ Importation de données des applications partenaires
PJS-FO	Portail PJS internet	AAV, FV, SIS Victimes & grand public	Consultation des données Victimes



Vue d'ensemble du système PJS

4.3. Principales fonctionnalités de l'application

Diagrammes de classe en annexe.

A. Événements :

L'objet événement dans PJS décrit le lieu, la date et les caractéristiques des faits en cours ou survenus.

- ✓ Créer manuellement un événement depuis l'application PJS
- ✓ Consulter la fiche événement
- ✓ Consulter la liste des individus d'un événement (Personne présente)
- ✓ Modifier la fiche événement via l'application PJS
- ✓ Ajouter une personne présente
- ✓ Rechercher un événement
- ✓ Créer un événement via import (FGTI)
- ✓Modifier la fiche événement via import (FGTI)
- ✓ Ajouter une personne présente via import (FGTI)
- ✓ Consulter l'arborescence d'un événement
- ✓ Ajouter un lien de regroupement (cas lier un événement)
- ✓ Ajouter un lien de regroupement (cas ajouter à l'événement global)

- ✓ Supprimer un lien de regroupement
- ✓ Créer, modifier, supprimer, afficher un événement global

B. Individus:

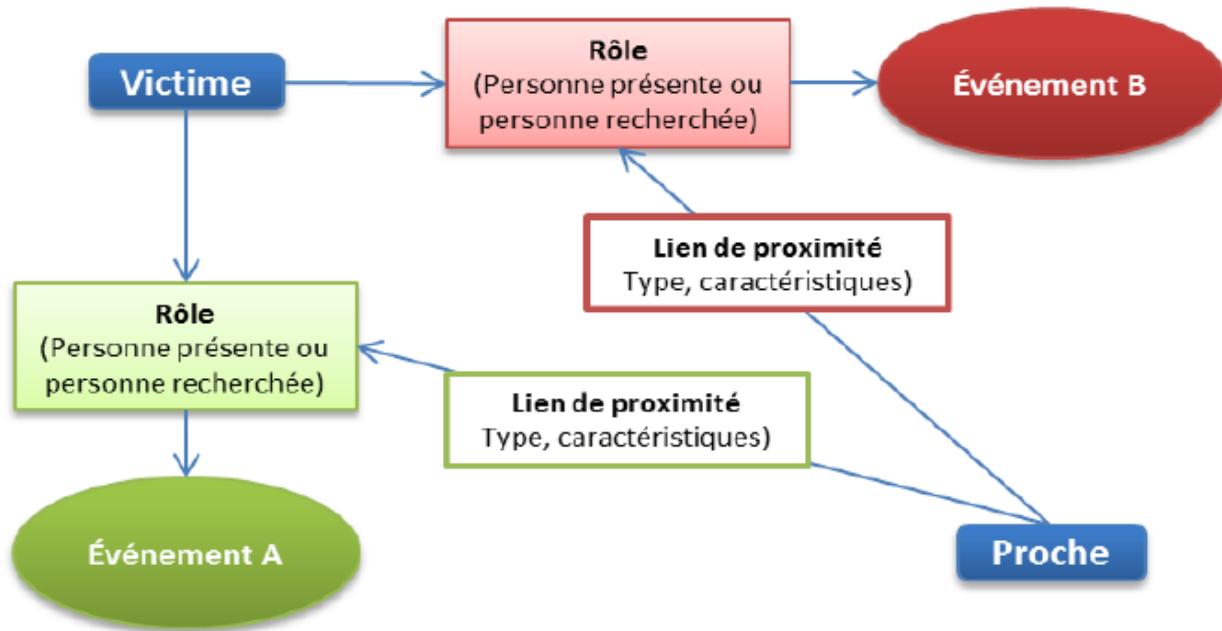
Un individu est une personne physique gérée dans le cadre d'un événement.

- ✓ Créer un individu avec rôle (PP) via PJS
- ✓ Ajouter un rôle (PP) à un individu existant via PJS
- ✓ Créer un proche via PJS
- ✓ Ajouter un lien de proximité à un individu existant via PJS
- ✓ Rechercher en cours de saisie
- ✓ Modifier la fiche individu via PJS
- ✓ Modifier le détail du rôle d'un individu via PJS
- ✓ Modifier le détail du lien de proximité d'un individu via PJS
- ✓ Consulter la fiche individu
- ✓ Consulter la liste des rôles d'un individu
- ✓ Consulter le détail d'un rôle
- ✓ Consulter la liste des proches
- ✓ Consulter la liste des liens de proximité
- ✓ Consulter le détail d'un lien de proximité
- ✓ Rechercher un individu
- ✓ Créer un individu avec rôle (PP) via import FGTI
- ✓ Ajouter un rôle (PP) à un individu existant via import FGTI
- ✓ Créer un proche via import FGTI
- ✓ Ajouter un lien de proximité à un individu existant via import FGTI
- ✓ Modifier la fiche individu via import FGTI
- ✓ Modifier le détail du rôle d'un individu via import FGTI
- ✓ Modifier le détail du lien de proximité d'un individu via import FGTI

4.4. Liens événement / individu

Avant d'aborder les fonctionnalités de l'application, il est important de comprendre le lien entre un événement et un individu.

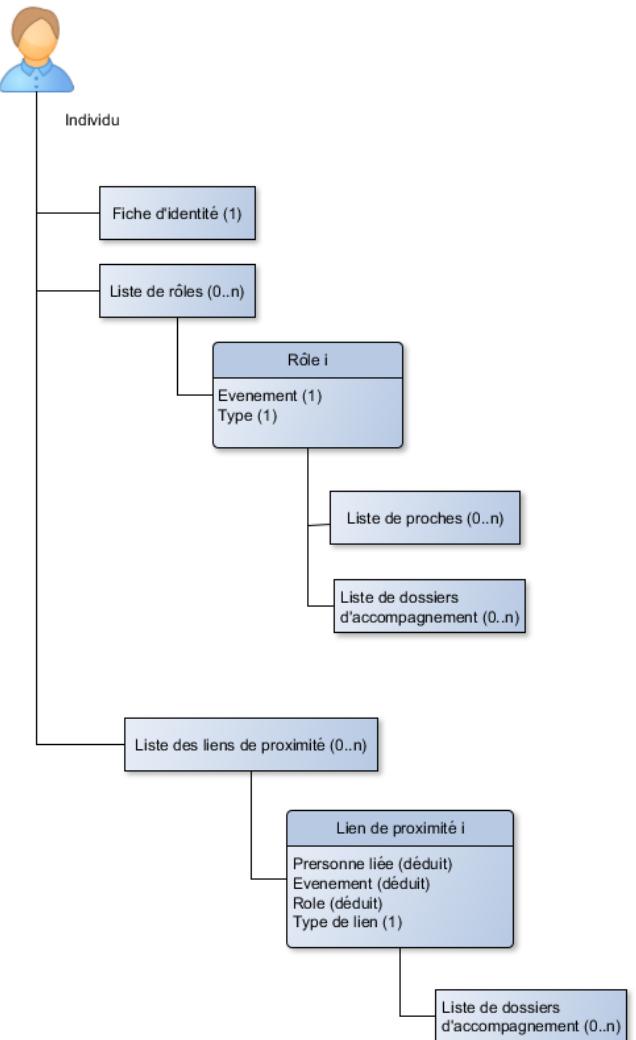
Un individu est une personne physique gérée dans le cadre d'un événement. Il est lié directement à un événement via un rôle.



Trois rôles sont retenus dans PJS :

- Personne présente (sur le lieu de l'événement).
- Personne recherchée (dans le cadre de l'événement).
- Témoin.

- ✓ Un individu est identifié par sa fiche d'identité (ensemble de données personnelles) et est soit impliqué dans un événement (c'est-à-dire a un rôle), soit identifié comme un proche.
- ✓ Un individu n'est créé dans PJS que dans le cadre d'un événement (il est rattaché à un rôle ou, en tant que proche, rattaché par un lien de proximité à une victime dans un événement).
- ✓ Un rôle apporte des données sur la situation de l'individu par rapport à l'événement et sur son éventuelle prise en charge sanitaire.
- ✓ Le type de lien (fratrie, parent, enfant, etc...)



4.5. Dossier d'accompagnement

Un dossier d'accompagnement est attribué à chaque individu. Il est rattaché au rôle de la victime (personne présente) ou, pour un proche, au lien de proximité qui le lie à une victime.

Un individu peut posséder plusieurs dossiers d'accompagnement du même acteur (en tant que victime de plusieurs événements, ou en tant que victime et proche dans le même événement).

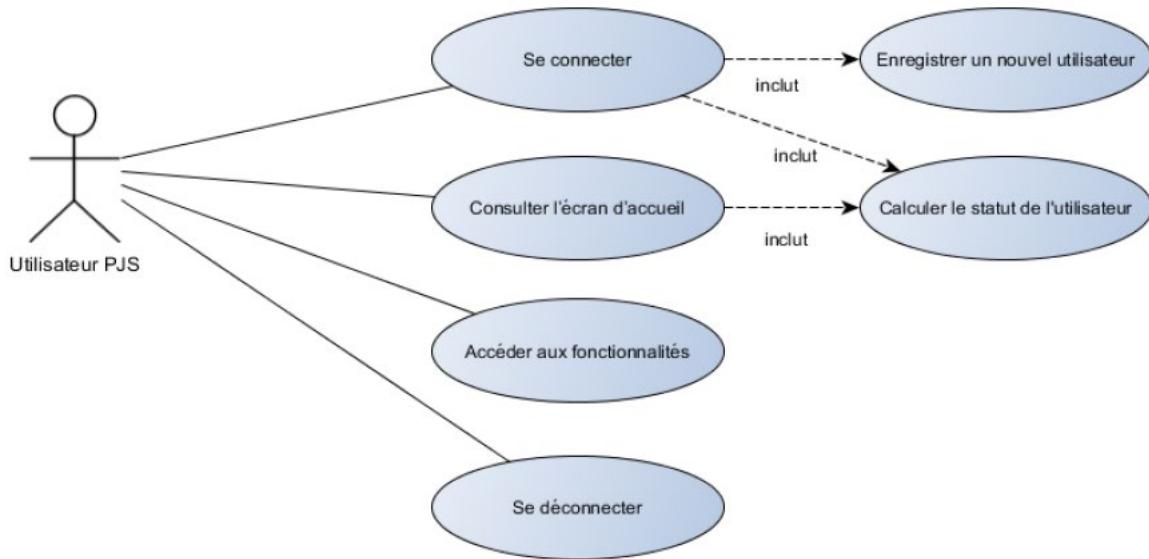
Il est transmis à PJS (importation par le flux FGTI) ou saisi directement par l'acteur via l'interface PJS.

4.6. Diagrammes de cas d'utilisation:

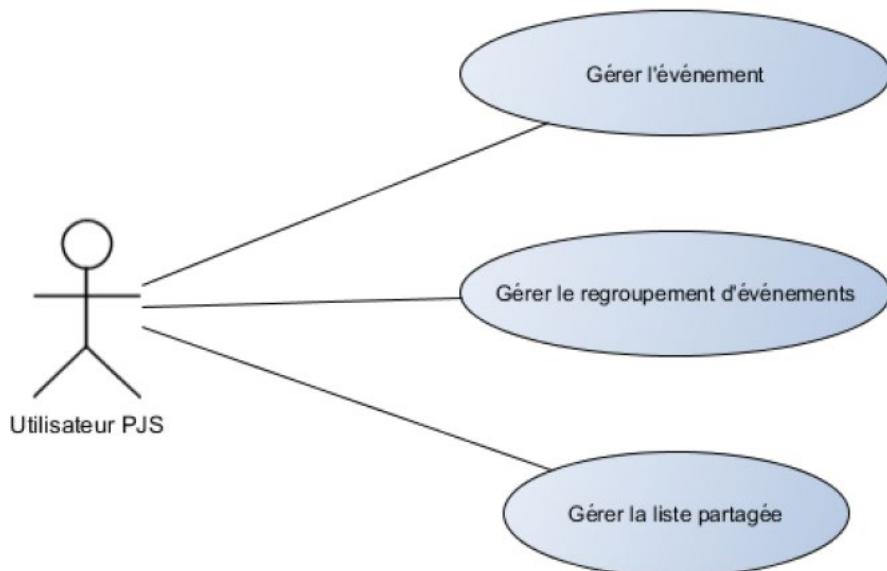
La spécification fonctionnelle décrit dans le détail la façon dont les exigences seront prises en compte. Pour les vues dynamiques, on montre le fonctionnement et le comportement du système résultant de l'analyse faite ci-dessous. Pour cela, la création de diagrammes de cas d'utilisation est nécessaire.

A. Utilisateur

Un utilisateur va se connecter à l'application et accéder aux différentes fonctionnalités selon son profil.



B. Événement



Ce cas d'utilisation décrit comment l'application permet à l'utilisateur de regrouper plusieurs événements unitaires au sein d'un événement global.

Suivant le choix de l'utilisateur deux cas peuvent se présenter :

- ✓ Ajouter un événement unitaire à un événement global existant.
- ✓ Lier 2 événements unitaires ensemble, ce qui créera un événement global.

Il n'est pas possible de lier 2 événements globaux entre eux.

- ✓ L'utilisateur clique sur le bouton « Ajouter un événement » dans l'arborescence d'un événement global ou d'un événement unitaire regroupé.
- ✓ L'utilisateur clique sur le bouton « Lier à un événement » dans l'arborescence d'un événement unitaire non regroupé.

Ce cas d'utilisation décrit comment sont gérés les regroupements d'événements, et en particulier la consultation de l'arborescence d'un événement, l'ajout d'un lien de regroupement, la suppression d'un lien de regroupement et le rapprochement rapide des individus.

C. Individu



Les données unifiées d'un individu, sont automatiquement constituées par PJS dans une version unifiée de l'individu, à partir des données fournies par les différents acteurs et partenaires. L'objectif est d'obtenir une vision de convergence la plus complète possible et servant de référence pour l'ensemble des acteurs.

Ces données concernent :

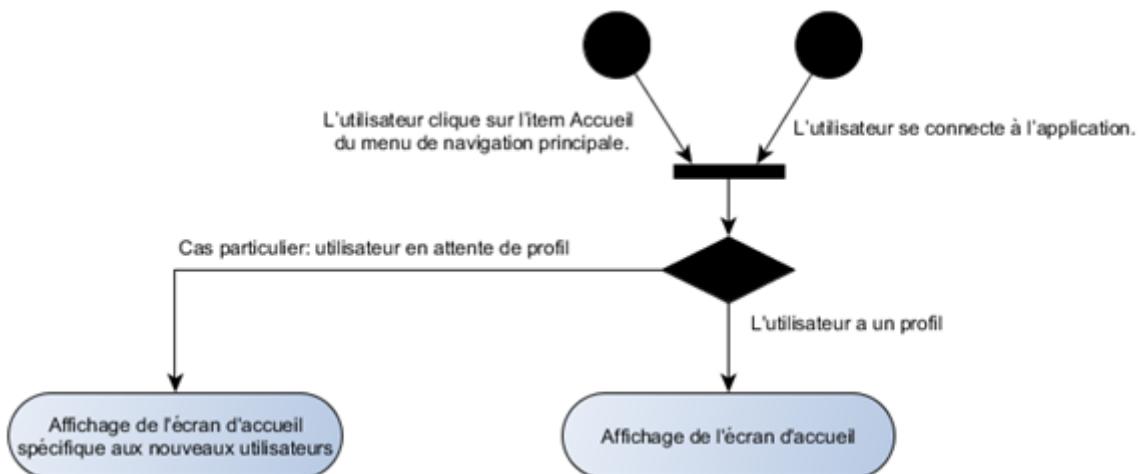
- ✓ **Fiche de l'individu** (identité, description, coordonnées).
- ✓ **Rôles de l'individu** (ses liens dans les différents évènements). Les liens de proximité de l'individu (ses liens avec d'autres individus).
- ✓ **Fiche unitaire** : Fiche individu créée par un utilisateur de PJS ou par import de flux partenaire.
- ✓ **Fiche unifiée** : Fiche unifiée générée par PJS après rapprochement de plusieurs fiches unitaires.
- ✓ **Rapprochement de fiches unitaires** : Quand un utilisateur fait un rapprochement.
- ✓ **Rôle unifié** : Rôle généré par PJS après création ou mise à jour d'un individu unifié ou d'un événement global.
- ✓ **Lien de proximité unifié** : Lien de proximité généré par PJS après création ou la mise à jour d'un individu unifié ou d'un événement global.

4.7. Diagrammes d'activité

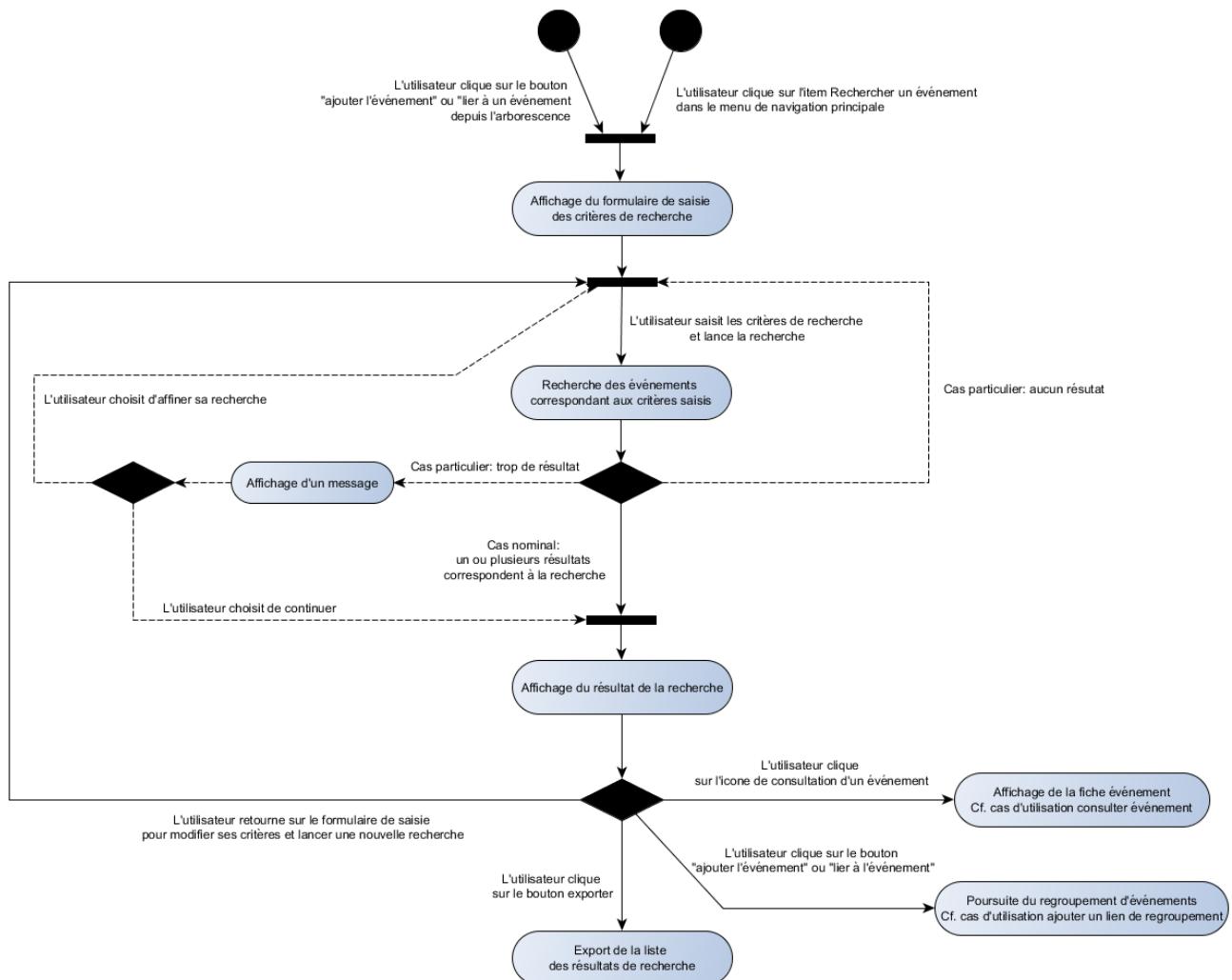
Les diagrammes d'activité représentent le système physique et l'interaction avec le système.

En langage UML, le diagramme d'activité fournit une vue du comportement d'un système en décrivant la séquence d'action d'un processus.

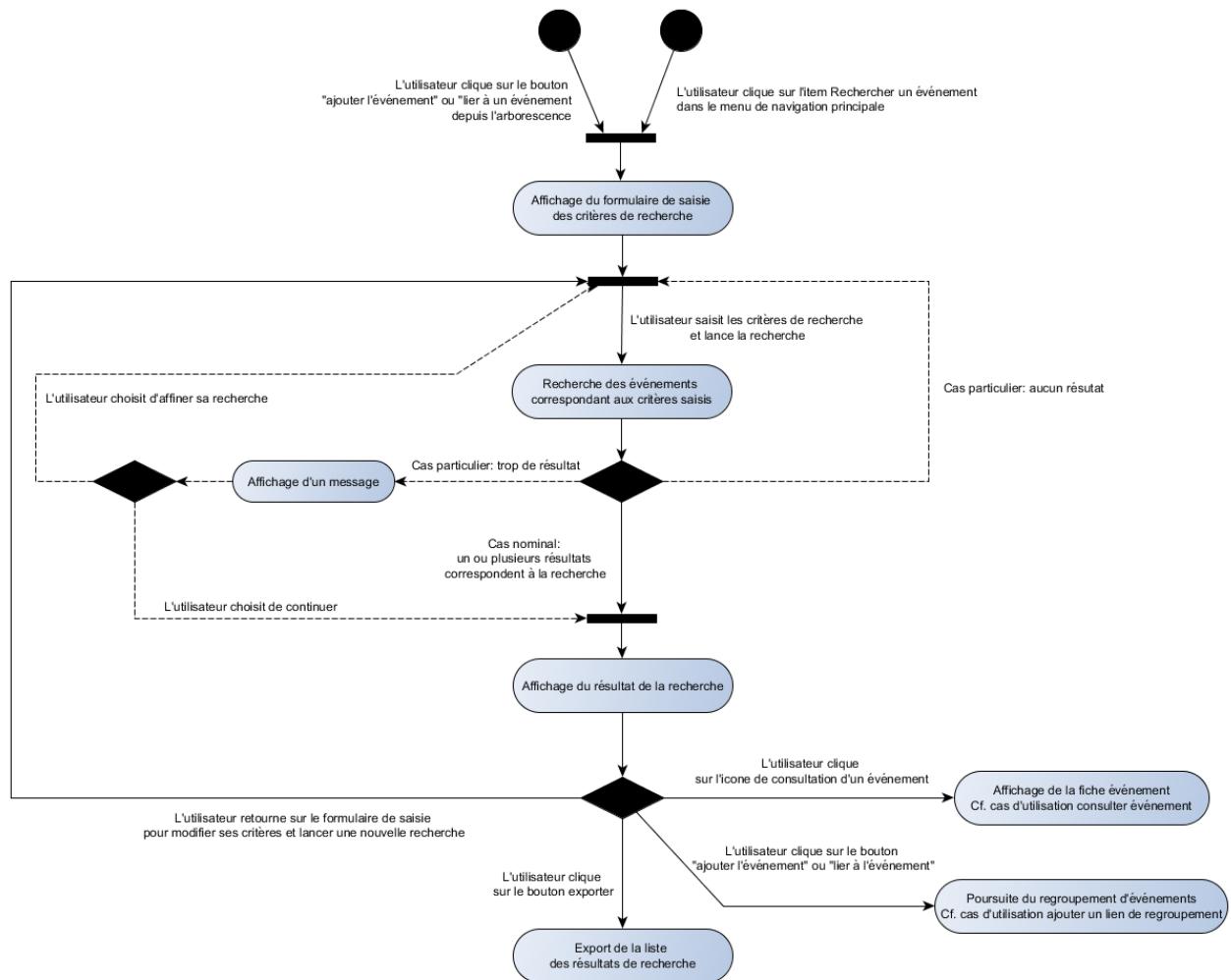
A. Consulter l'écran d'accueil



B. Rechercher un événement



C. Rechercher un individu



5. Conception

5.1. Module MAI (Front End)

Le Module Applicatif Interactif Homme Machine (MAI), est une Encapsulation des IHM précisée dans les spécifications fonctionnelles détaillées.

Elle pilote les enchaînements des services pour réaliser un objectif métier, déployé dans le navigateur du poste de travail utilisateur, elle porte le contexte utilisateur.

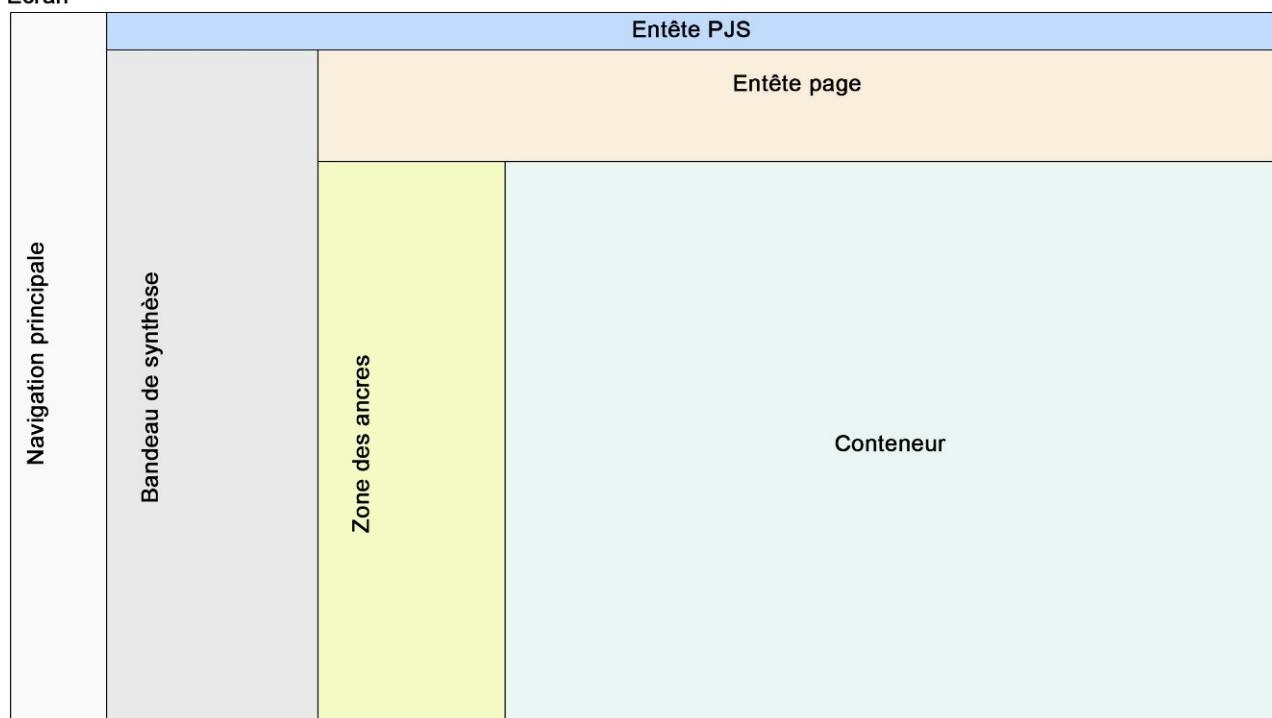
The screenshot shows the SIVAC application's event detail page for the "Le Bataclan" incident. The left sidebar has sections for MINISTÈRE DE LA JUSTICE, ACCUEIL, ÉVÉNEMENTS, INDIVIDUS, and ADMINISTRATION. The main content area has tabs for "Fiche événement" and "Liste individus / Liste partagée". The "Fiche événement" tab is active, displaying event details like date (13/11/2015), location (France, Non renseignée), and type (Terrorisme). A right sidebar contains a "Données confidentielles" section with placeholder text and buttons for "SUPPRIMER" and "MODIFIER". The footer includes copyright information and a link to the accessibility statement.

A. Composant squelette (sivac-layout)

Le composant Angular sivac-layout est utilisé dans la totalité des écrans présents dans l'application et permet de structurer le contenu de la page en définissant les différentes zones à afficher et leurs contenus.

La structure générale d'une page de l'application est composée de plusieurs zones dont certaines sont obsolètes. Le modèle le plus complet est celui comportant les zones optionnelles.

Ecran



Ce composant s'appuie sur les directives structurelles Angular 'ng-template' et 'ng-component' permettant de définir dans le composant générique des contenus (les différentes zones du schéma ci-dessus).

L'utilisation d'un composant générique sur tous les écrans permet, en plus de rendre le développement plus rapide, une homogénéité dans l'ensemble de l'application.

Ce composant est constitué de 5 parties correspondant aux zones contenant les informations qui varient entre les pages :

- ✓ La zone dans l'entête de la page où se trouvent les boutons d'actions
- ✓ La partie haute de la zone de synthèse comportant un résumé de l'élément consulté
- ✓ La partie basse de la zone de synthèse contenant un menu de navigation
- ✓ La zone d'ancre du contenu de la page
- ✓ La zone réservée au contenu principal de la page

De plus, l'affichage de ces zones est conditionné par deux paramètres du composant sivac-layout :

- ✓ hasLeftPart correspondant à la zone de synthèse
- ✓ hasAnchorsZone correspondant à la zone des ancrés

B. Module (NgModule)

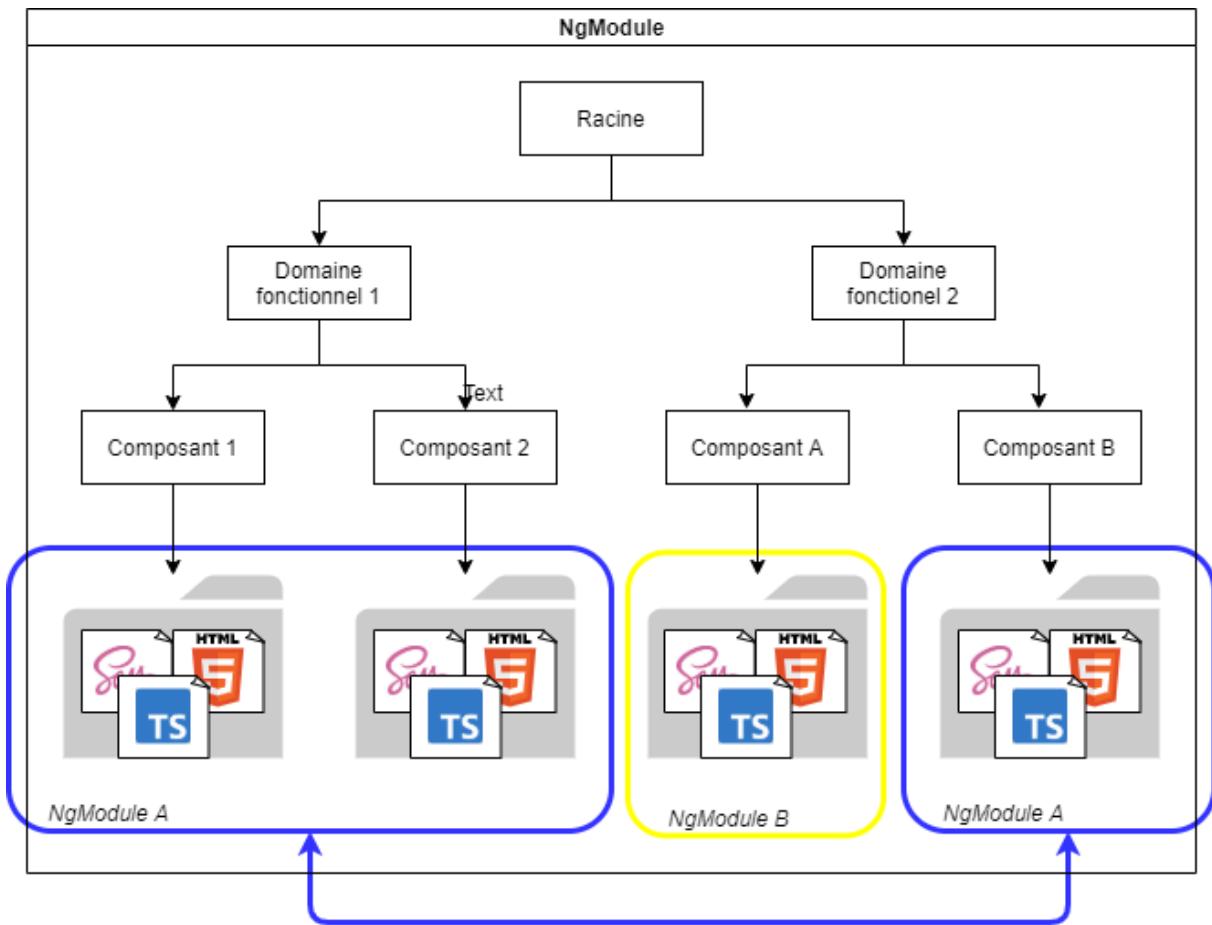
Angular englobe naturellement la notion de module telle qu'elle est définie dans le langage de programmation Typescript. Cet outil permet le cloisonnement étanche entre différents fichiers du logiciel. Il existe une seconde notion de module, propre à Angular, que l'on appelle NgModule pour les différencier des modules Typescript.

Les modules Angular définissent le contenu de l'application pour plusieurs raisons (principalement techniques). Les définitions incluses dans ces fichiers permettent à Angular :

- ✓ De ne pas à avoir à découvrir dynamiquement le contenu de l'application
- ✓ De réaliser des tâches de contrôles de cohérences et d'optimisation
- ✓ De grouper des éléments pour retarder leur chargement et diminuer la consommation réseau et mémoire.

Les NgModules sont définis dans des fichiers qui incluent, entre autres, les composants du module et aussi les modules dont il dépend. Angular peut de cette façon découvrir l'ensemble des modules/composants de l'application afin de procéder à la construction de celle-ci. Le schéma suivant illustre le fait que ces modules regroupent les composants indépendamment de leurs emplacements.

Dans l'exemple ci-dessous l'application est découpée en deux NgModules A & B et 4 modules Typescript.



Les NgModule dans PJS

L'application PJS suit les bonnes pratiques Angular qui définissent 4 types de modules :

- ✓ Le module de démarrage (*bootstrap module*)
- ✓ Le module de composant commun (*sharedModule*)
- ✓ Le module de services commun (*core module*)
- ✓ Un ou plusieurs modules fonctionnels (*feature module*).

La liste des NgModules dans PJS :

- ✓ **AppModule** (*bootstrap module*) :
Ce module est le premier module lu par Angular. Il permet la découverte des autres modules ainsi que le lancement de l'application.
- ✓ **SharedModule** : Composants partagés.
- ✓ **CodeModule** : Services partagés.
- ✓ **AuthModule** (*feature module*) : Composants et services pour l'authentification.
- ✓ **DashboardModule** (*feature module*) : Composants et services pour les tableaux de bord.

- ✓ **EvenementModule** (feature module) : Tous les composants/pages pour la gestion des événements dans PJS.
- ✓ **IndividusModule** (feature module) : Tous les composants/pages pour la gestion des individus dans PJS.
- ✓ **ReferentielModule** (feature module) : Tous les composants/pages pour la gestion des référentiels dans PJS.

C. Store

Angular est un outil modulaire fonctionnant par l'assemblage de composants qui sont les unités simples de l'application. Il faut néanmoins mettre en place un système de communication entre les composants pour permettre le fonctionnement de l'application. Il existe un système de communication simple pour le dialogue entre parents et enfants directs à travers le passage de paramètres. Ce mécanisme ne fonctionne pas pour un dialogue entre composants « frères », éloignés (sans parent commun) ou distants dans l'arbre de descendance.

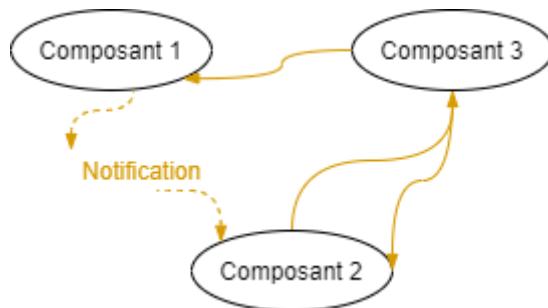
Les bonnes pratiques d'Angular résolvent ce problème par les patrons de code « observable » et « service à état commun ». PJS utilise de nombreuses fois les observables pour des communications entre composants proches.

Pour une communication plus globale, PJS utilise un gestionnaire d'état centralisé qui réalise deux fonctions principales :

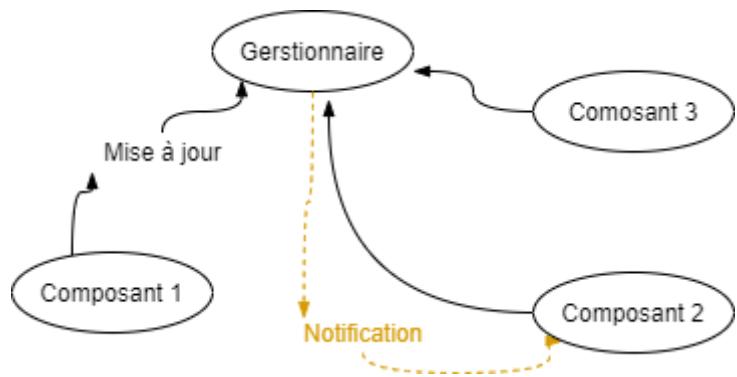
1. **Être référent pour certaines données.** Les composants demandent des changements de valeurs auprès du gestionnaire pour une mise à jour de valeur. Cette façon de faire élimine les problèmes de modifications concurrentes.
2. **Notifier des changements les composants intéressés** (patron observable) afin qu'ils puissent se mettre à jour.

Ce gestionnaire évite une communication en toile d'araignée entre composants et assure l'unicité des valeurs pour toute l'application.

Le schéma suivant illustre une communication entre composants sans gestionnaire. Cette forme de dialogue est difficile à suivre lorsqu'on veut connaître les échanges.



A l'opposé, le schéma suivant montre la communication avec un gestionnaire. Les échanges entre composants n'existent plus et tous partagent la même valeur puisque celle-ci est gardée par le gestionnaire. Celui-ci informe certains composants lors des changements.



D. Cache

Les formulaires de PJS proposent très souvent à l'utilisateur de choisir parmi une sélection de possibilités.

L'exemple suivant montre ce processus pour indiquer le type d'une voie :

Type de voie

r

- Aérodrome
- Agglomération
- Aire
- Ancienne Route
- Arcade

Ces listes de choix sont stockées dans la base de données du serveur et doivent, par conséquent, être chargées pour être affichées. Ce chargement se fait par un transfert aller-retour de données avec le serveur. Ce mécanisme provoque une utilisation de la bande passante et des différentes ressources matérielles (CPU...) alors que ces données sont constantes.

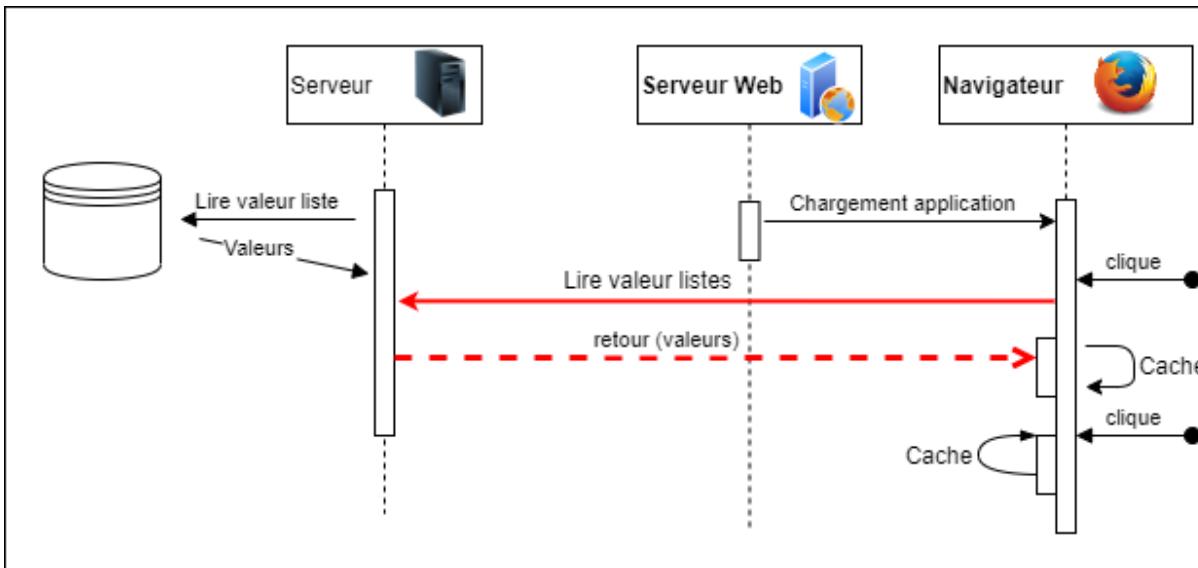
Ces transferts peuvent être évités en utilisant un stockage local au navigateur. HTML5 définit différents types de stockages aux caractéristiques différentes :

Quantité données	Faible volume	Gros volume
Access	Synchrone	Asynchrone
Index	Non	Oui

Stockage objet	Non	Oui
Firefox Private sessions	Oui	Non

PJS a fait le choix d'utiliser la technologie **IndexedDB** pour sa gestion de gros volume et son fonctionnement asynchrone qui évite de figer le navigateur.

Le diagramme suivant montre la chronologie des actions avec notamment, l'utilisation du cache lors du second clique qui évite un couteux appel http.



Durée de vie

Une date de mise en cache des données est adjointe à celles-ci lors de l'enregistrement. Cette date permet à l'application de connaître la fraîcheur des données. L'application peut ainsi décider de les charger dans le cas où les données seraient considérées obsolètes. La longueur de validité peut être définie par type de donnée.

Il est possible de visualiser ces données à partir de l'outil Devtool en sélectionnant le local storage de l'application.

#	Key	Value
0	"civilité"	<pre> >ttl: '2022-06-02T13:55:16.855Z', value: Array(12) >ttl: '2022-06-02T13:55:16.855Z' >value: Array(12) >0: {actif: true, id: 0, libelleCount: '0', libelleLong: 'Non renseigné', idFonctionnel: 0} >1: {actif: true, id: 1, libelleCount: '1', libelleLong: 'FEMME', idFonctionnel: 5} >2: {actif: true, id: 2, libelleCount: 'M', libelleLong: 'MÉSIEUR', idFonctionnel: 2} >3: {actif: true, id: 3, libelleCount: 'M', libelleLong: 'MÂTRE', idFonctionnel: 6} >4: {actif: true, id: 4, libelleCount: 'M', libelleLong: 'MÉSIEURNE', idFonctionnel: 8} >5: {actif: true, id: 5, libelleCount: 'M', libelleLong: 'MÉSIEURE', idFonctionnel: 3} >6: {actif: true, id: 6, libelleCount: 'M', libelleLong: 'MÉSIEURE', idFonctionnel: 23} >7: {actif: true, id: 7, libelleCount: 'M', libelleLong: 'MADAME', idFonctionnel: 1} >8: {actif: true, id: 8, libelleCount: 'M', libelleLong: 'MESDAMES', idFonctionnel: 21} >9: {actif: true, id: 9, libelleCount: 'M', libelleLong: 'MADAME ET MÔSIEUR', idFonctionnel: 22} >10: {actif: true, id: 10, libelleCount: 'P', libelleLong: 'PROFESSEUR', idFonctionnel: 7} >11: {actif: true, id: 11, libelleCount: 'V', libelleLong: 'VEUVE', idFonctionnel: 4} length: 12 </pre>
1	"etatInitial"	>ttl: '2022-06-02T13:55:16.868Z', value: Array(8)
2	"naturelien"	>ttl: '2022-06-02T13:55:16.230Z', value: Array(25)
3	"niveauxValidationsIdentites"	>ttl: '2022-06-02T13:55:16.231Z', value: Array(3)
4	"payInformation"	>ttl: '2022-07-15T12:17:14.695Z', value: Array(217)
5	"precisions-coordonnees-type-REL"	>ttl: '2022-06-02T13:55:16.864Z', value: Array(3)
6	"precisions-coordonnees-type-TEL"	>ttl: '2022-06-02T13:55:16.860Z', value: Array(4)
7	"structuresAppartenance"	>ttl: '2022-06-02T13:55:16.872Z', value: Array(5)
8	"sexes"	>ttl: '2022-06-02T13:55:16.880Z', value: Array(4)
9	"statusPersonnesPrésentes"	>ttl: '2022-06-02T13:55:16.880Z', value: Array(3)
10	"structureAppartenance"	>ttl: '2022-07-15T12:17:14.375Z', value: Array(19)
11	"typeEvenement"	>ttl: '2022-07-15T12:17:14.583Z', value: Array(17)

E. Joindre le serveur

Angular fournit une liste d'outils intégrés (built-in features) comme la navigation, l'internationalisation... La communication avec le serveur fait partie de cette boîte à outils. Les appels http se font, comme d'habitude avec Angular, avec un service unique disponible dans toute l'application. Ce service propose l'ensemble des « verbes » http : **DELETE, PUT, GET, POST, OPTIONS**.

Voici un exemple d'usage de ce service dans la méthode qui permet d'interroger le serveur sur un code postal :

```
getCodesPostauxByCodePostalAutoCompletion(codePostal: string): Observable<string[]> {
  const getUrl = `${this.url}/auto-completion/${codePostal}`; 1
  return this.http.get<string[]>(getUrl); 2
}
```

1 Construction de l'URL pour joindre le serveur.

2 Utilisation du service Angular HttpClient pour initier le GET http.

F. Formulaires

Ce chapitre explique la création technique d'un FormControl sur le formulaire SIVAC en prenant en exemple un formulaire de modification d'un lien de proximité.

Pour élaborer ce formulaire de consultation, il est nécessaire au préalable de fournir le html du ou des composants permettant la création des différents champs de renseignements du chapitre. (Nature du lien, Proche informant...).

```
<mat-expansion-panel class="mat-elevation-z" [expanded]="true" id="Lien de proximité" (afterCollapse)="getOffsets()" (afterExpand)="getOffsets()">
  <mat-expansion-panel-header>
    <mat-panel-title>Lien de proximité</mat-panel-title>
  </mat-expansion-panel-header>
  <!-- Edition -->
  <div *ngIf="detailMode != 'cons'">
    <app-lien-proximite-proximite [form]="lienForm"></app-lien-proximite-proximite>
  </div>
  <!-- Consultation -->
  <div *ngIf="detailMode === 'cons'">
    <app-consult-table [dataTable]="dataConsultation.lienProximite"></app-consult-table>
  </div>
</mat-expansion-panel>
```

Pour notre exemple, le code html est rangé dans le composant app-lien-proximite-proximite, lui-même définissant les champs de renseignements un par un de la manière suivante :

```
<mat-form-field appearance="outline" class="form-element">
  <mat-label>Nature du lien </mat-label>
  <mat-select formControlName="natureLien" [compareWith]="comparer">
    <mat-option *ngFor="let nature of naturesLien" [value]="nature">{{nature.lien }}</mat-option>
  </mat-select>
  <mat-hint>{{ 'forms.hint_obligatoire' | translate }}</mat-hint>
  <mat-error *ngIf="natureLien.errors?.required && natureLien.touched">
    {{'forms.obligatoire' | translate }}
  </mat-error>
</mat-form-field>
```

Puis de lui appliquer un contrôle de formulaire. Les différents FormControls ont été définis dans un seul service : CréerFormService

```
this.lienForm = this.formService.creerLienProximiteForm(data);
```

Ce FormControl en particulier, va définir l'obligation de renseigner Nature du lien (1), ainsi que sa possibilité d'édition (2), puis de la possibilité d'édition des champs Proche informant et Proche recherchant (2) et enfin la consultation pure (3) sans possibilité d'édition des autres champs :

```
creerLienProximiteForm(lienProximite: LienProximite) {
  let lienForm = this.formBuilder.group({
    // lien de proximite
    'personnelle': [{ value: lienProximite.prenomVictime, disabled: true }],
    'natureLien': [{ value: lienProximite.natureLien, disabled: false, Validators.required}], 1
    'informant': [{ value: lienProximite.informant, disabled: false }], 2
    'recherchant': [{ value: lienProximite.recherchant, disabled: false }], 2
    'contactSivic': [{ value: lienProximite.contactSivic, disabled: true }],
    'contactCdcs': [{ value: lienProximite.contactCdcs, disabled: true }],
    // confidentialité de proximité
    'confidentialiteEmployeur': [{ value: 'NR', disabled: true }], 3
    'confidentialiteParquet': [{ value: 'NR', disabled: true }], 3
    // statuts
    'statutJustice': [{ value: 'NR', disabled: true }],
    'dateStatJust': [{ value: 'NR', disabled: true }],
  });
  return lienForm;
}
```

Les trois champs modifiables ont une propriété disabled à false, et le champ natureLien a l'obligation d'être rempli avec la propriété Validators.required.

Il faut enfin prendre soin de bien reporter chaque clé du groupe FormBuilder (par exemple « natureLien ») dans la propriété spécifique formControlName du composant, pour que la règle de contrôle s'applique au composant.

```
<mat-select formControlName="natureLien" [compareWith]="comparer">
| <mat-option *ngFor="let nature of naturesLien" [value]="nature">{{nature.lien }}</mat-option>
</mat-select>
```

5.2. Module MAS (Back end)

A. Modèle de données

Le modèle de données se trouve en annexe.

B. Presentation des packages

L'organisation du code du module mas-bo est la suivante :

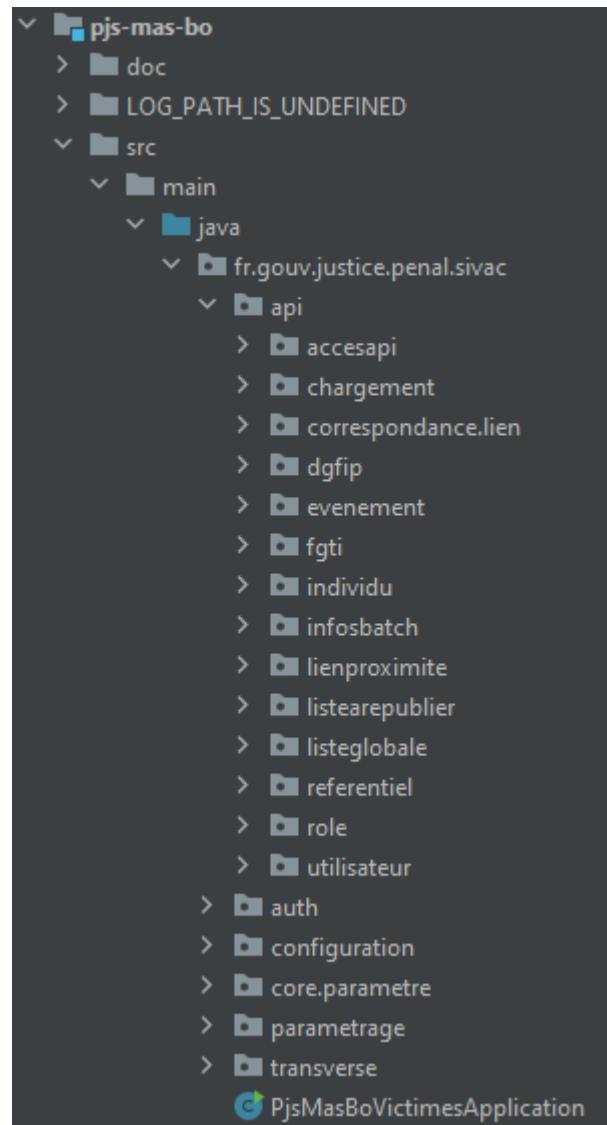
Le package api contient le découpage au sens fonctionnel / thématique des cas d'utilisation.

Nous y trouverons un package pour chaque notion comme : Individu, Evenement, FGTI ... où nous allons retrouver les entités, contrôleur, services dans chacun d'entre eux.

Dans le package « auth », les classes pour la gestion de l'authentification, pour générer les jetons JWT par exemple.

Dans le package « configuration », les classes de configuration applicatives comme SpringSecurityConfiguration.

Dans le package « transverse », les classes communes pour l'ensemble du projet, comme par exemple les classes contenant les constantes, des classes utilitaires etc...



C. Configuration de l'application

Afin de configurer les différents éléments de l'application, nous utilisons le fichier application.yml situé sous src/main/ressources.

Nous y retrouverons plus spécifiquement les configurations du port du serveur, la connexion de la base de données et JPA, flyway, CROS, Actuator et les profils.

D. Tests

L'organisation des packages pour les tests est similaire à celle du code applicatif.

Pour la réalisation des tests, nous utilisons une base de données mémoire de type H2 pour les tests unitaires. Celle-ci est configurée dans le fichier application-test.yml situé dans /src/test/ressources.

Afin de pouvoir bouchonner les appels de certains services / repository nous utilisons l'API Mockito. Pour chaque élément bouchonné il faut utiliser l'annotation @Mock lors de sa déclaration.

Le but des tests unitaires est de valider un comportement, un traitement réalisé au sein de l'application. Il est inutile de tester si une sauvegarde / modification / suppression d'une donnée en base se déroule bien.

Les tests d'intégration nécessitent la création d'une vraie base de données PostgreSQL via une image docker. L'annotation @DockerApplicationTest est une classe de test permettant l'autoconfiguration et le démarrage d'une base Postgres à l'aide de docker. Ce mode permet la vérification des scripts flyway et le comportement de l'application en mode nominal.

E. Utilisation de Swagger / OpenApi

Les interactions avec le module MAI et MAS se font via des requêtes. Un fichier nommé sivac-api.yml est défini pour l'ensemble des endpoints et des DTO.

L'avantage de cette solution pour la définition des endpoints et DTO est qu'elle permet de fixer un contrat d'interface avec le module MAS (back) et les autres modules comme MAI (front).

Il est également possible avec l'utilisation de maven de générer de façon automatique les interfaces pour la partie contrôleur et les DTO. Du côté du MAI, il est possible d'utiliser également ce fichier afin de générer automatiquement en Typescript les DTO. Cela permet d'avoir l'assurance que les objets transités entre le MAS et MAI aient la même structure.

5.3. Sécurité applicative

L'audit du code des modules applicatifs révèle souvent un niveau de risque de sécurité notable avec la présence de quelques failles de sécurité usuellement constatées dans ce type d'application (application web internet, intranet...). Ces failles sont classées par OWASP en 2013 parmi les dix failles les plus couramment rencontrées

Les failles de sécurité détectées concernent les injections (A1) et le XSS (A3).

Ces failles peuvent permettre à un attaquant, disposant d'une connaissance technique suffisante :

- D'altérer les traitements en provoquant des erreurs ou un déni de service à travers des injection XML
- De récupérer les informations personnelles de l'utilisateur (cookies) par des attaques XSS

L'utilisation de token de session permet de réduire les risques d'attaques de type CSRF (A8). Cependant, l'ajout de contre-mesures supplémentaires par le contrôle de quelques paramètres de l'en-tête HTTP (REFERER, HTTP-HOST, USER-AGENT) diminuera considérablement ces risques.

Les actions urgentes sont la mise en place des contre-mesures pour parer à des éventuelles attaques.

2013 : Le Top 10 des risques de sécurité des applications Web

A1 Injection	<i>Injection (de code)</i>
A2 Broken Authentication and Session Management	<i>Violation de Gestion d'authentification et de Session</i>
A3 Cross-Site Scripting (XSS)	<i>XSS</i>
A4 Insecure Direct Object References	<i>Références directes non sécurisées à un objet (mis dans 2017-A4)</i>
A5 Security Misconfiguration	<i>Mauvaise configuration sécurité</i>
A6 Sensitive Data Exposure	<i>Exposition de données sensibles</i>
A7 Missing Function Level Access Control	<i>Manque de contrôle d'accès au niveau fonctionnel (mis dans 2017-A4)</i>
A8 Cross-Site Request Forgery (CSRF)	<i>CSRF</i>
A9 Using Components with Known Vulnerabilities	<i>Utilisation de composants avec des vulnérabilités connues</i>
A10 Unvalidated Redirects and Forwards	<i>Redirection et Renvois non validés</i>

A. Injection XML (A1)

Les failles d'injection XML concernent les opérations de parsing de données XML non sécurisées. Les données XML peuvent contenir des séquences de caractères spécifiques constituant du code malveillant pour altérer les traitements (attaque par déni de service par exemple), afin accéder à des données en base.

B. Failles d'injection Javascript XSS(A3)

Il faut échapper les caractères sur composant.innerHTML en réalisant une fonction d'échappement en javascript, équivalente à la fonction en Java escapeHtml() :

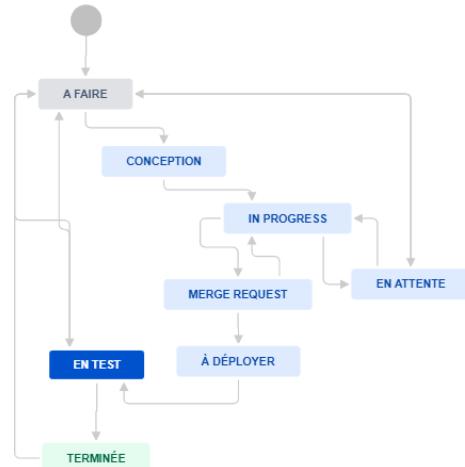
```
function escapeHtml(unsafe) {
    return unsafe
        .replace(/&/g, "&amp;")
        .replace(/</g, "&lt;")
        .replace(/>/g, "&gt;")
        .replace(/"/g, "&quot;")
        .replace(/'/g, "&#039;");
}
```

Si pour diverses raisons le composant.innerHTML doit contenir des tags html – et donc sans échapper les caractères – il faut vérifier si le contenu ne contient pas <script>. Il suffira alors de rechercher la présence de <script dans le contenu et de le remplacer par <script par exemple.

6. Implémentation

Lors de ma période de stage en entreprise, j'ai réalisé plusieurs tâches à la suite de l'expression de besoin du client, traduit par des spécifications fonctionnelles détaillées. Ces tâches ont d'abord été chiffrées par l'équipe lors d'une session de chiffrage et m'ont été attribuées par la cheffe de projet.

Ces tâches ont été réalisées en suivant les étapes suivantes :



6.1. Liste des Listes à partager

Un utilisateur a besoin de connaître facilement quelles sont les listes partagées à republier sans avoir à consulter chaque événement.

Pour cela, l'application va déterminer quelles sont les listes partagées qui nécessitent une republication à l'instant T et construire une liste de listes partagées à republier. Cette première étape va être lancée via un batch qui pourra être déclenché toutes les nuits, tous les mois ... L'utilisateur pourra alors consulter cette liste via l'application et effectuer les republications nécessaires.

A. Base de données

La création d'une nouvelle table était nécessaire pour cette fonctionnalité, puisque ces données devaient être persistées afin d'être appelées par le backend.

```

1  -- Création de la table
2  CREATE TABLE liste_partagee_a_republier (
3      id BIGINT NOT NULL,
4      evenement_id BIGINT NOT NULL UNIQUE,
5      nom_evenement VARCHAR (256) NOT NULL,
6      date_evenement TIMESTAMP NOT NULL,
7      version_liste_partagee INT NOT NULL,
8      date_publication_liste_partagee TIMESTAMP NOT NULL,
9      republiee BOOLEAN NOT NULL DEFAULT FALSE,
10     applicable BOOLEAN NOT NULL DEFAULT TRUE,
11     nombre_individus_ajoutes INT NOT NULL DEFAULT 0
12 );
13
14  -- Primary Key
15  ALTER table ONLY liste_partagee_a_republier ADD CONSTRAINT liste_partagee_a_republier_pk PRIMARY KEY (id);
16
17  --Création de la sequence liste_partagee_a_republier_id_seq pour la clé primaire
18  CREATE SEQUENCE liste_partagee_a_republier_id_seq
19      START WITH 1
20      INCREMENT BY 1
21      NO MINVALUE
22      NO MAXVALUE
23      CACHE 1;
24
25  --Utilisation de la sequence
26  ALTER SEQUENCE liste_partagee_a_republier_id_seq OWNED BY liste_partagee_a_republier.id;
27  ALTER TABLE ONLY liste_partagee_a_republier ALTER COLUMN id SET DEFAULT nextval('liste_partagee_a_republier_id_seq');
28
29 COMMIT;
  
```

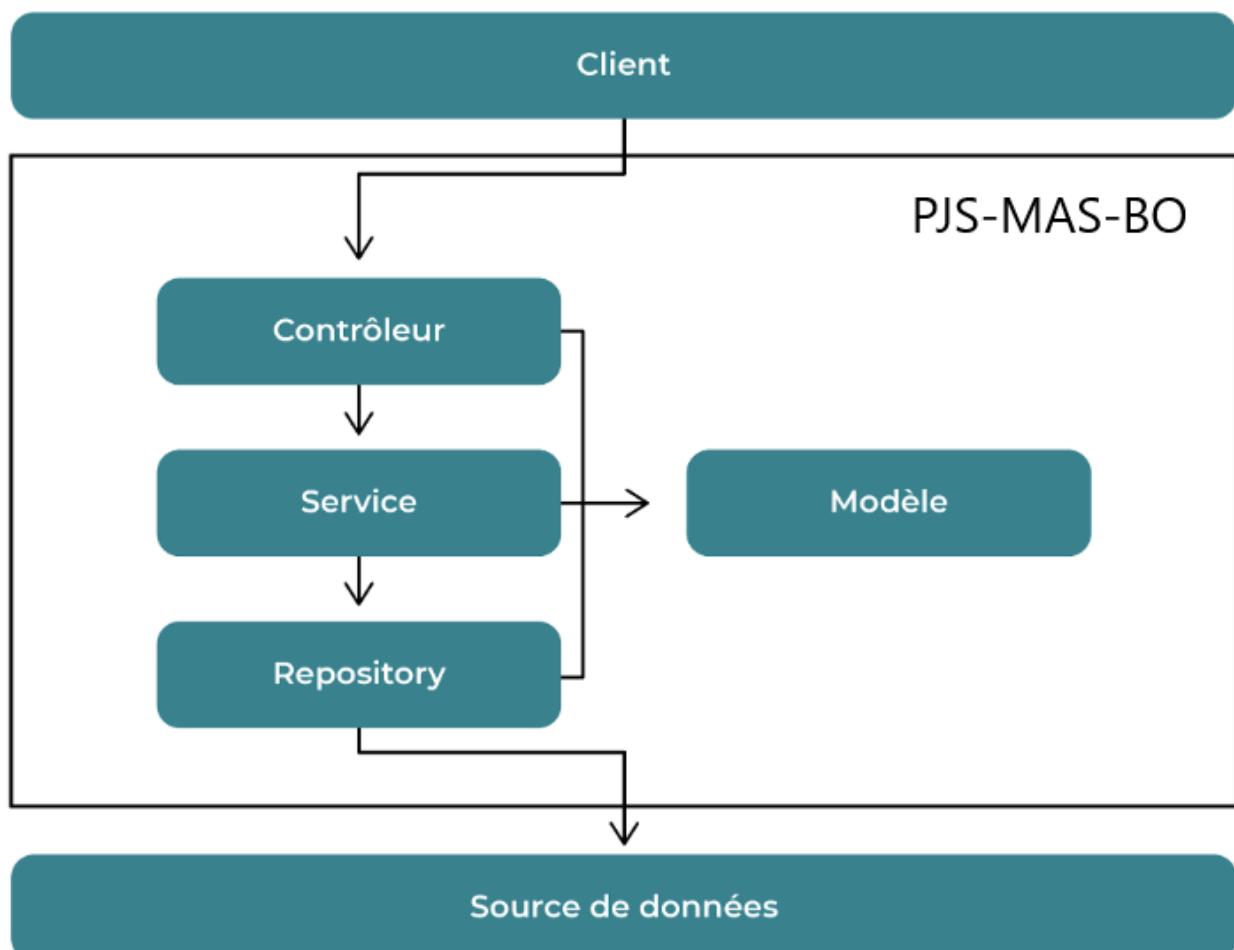
B. Back end

Pour cette partie je devais réaliser le endpoint que devait appeler le client (front end), pour récupérer les données triées et paginées, en base. En suivant les règles de conception de l'application.

Pour ce faire j'ai créé le package correspondant à la fonctionnalité dans l'arborescence du projet.

Le développement suit une architecture particulière. Il y a d'abord la création de l'entité JPA, qui n'est autre qu'une classe avec des propriétés qui sont associées aux valeurs en base ; le repository qu'utilise JPA pour construire les requêtes faites en base de données puis une classe Common service qui permet d'appeler une méthode de l'interface repository afin d'y injecter les paramètres de la requête (l'ordre de tri par exemple). Celui-ci est appelé par l'API service, qui joue un rôle de passe plat entre les différents services et le contrôleur.

▼	📁	listearepublier
	⌚	ListePartageeeARepublier
	⌚	ListePartageeeARepublier ApiService
	⌚	ListePartageeeARepublier Batch
	⌚	ListePartageeeARepublier CommonService
	⌚	ListePartageeeARepublier Controller
	🕒	ListePartageeeARepublier Mapper
	🕒	ListePartageeeARepublier Repository

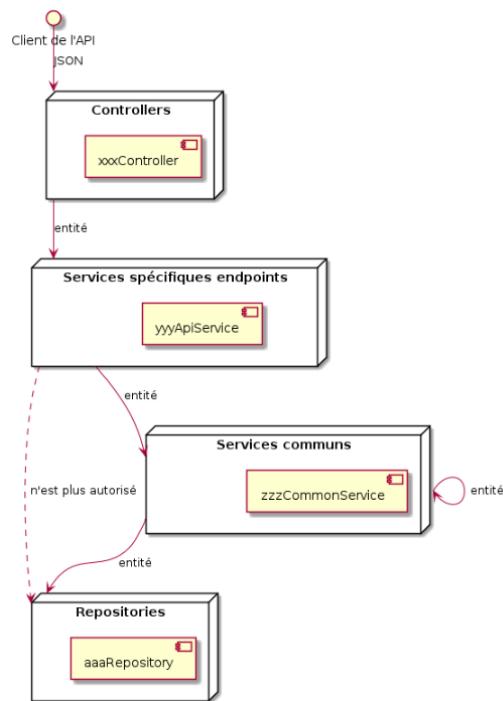


Sur ce diagramme, le découpage est légèrement différent d'un découpage classique avec des règles de conception qui définissent le rôle de chaque couche. Les contrôleurs ont uniquement comme rôle de gérer l'interfaçage REST et réalisent les conversions DTO ↔ Entité.

La couche service est découpée particulièrement, les end points peuvent appeler seulement les Api Services (service spécifiques) qui implémentent la logique métier et gèrent les transactions et la sécurité via l'annotation @Transactional.

Les Commons Service peuvent s'appeler entre eux, ne doivent pas faire appel à des services spécifiques et peuvent faire appel aux repositories pour gérer la persistance des données.

Le repository, qui étend l'interface JpaRepository qui contient un ensemble de méthodes des requêtes CRUD pouvant être complexes ou la création de Query en JPQL pour des requêtes plus spécifiques.



```

53     @Override
54     @PreAuthorize("hasAuthority(T(DroitsConstants).GESTION_LISTE_PARTAGEE)")
55     public ResponseEntity<PageListeListePartageeDTO> getListePartageeARepublierSortedPaged(
56         @NotNull @Valid OrderEnumDTO order, @NotNull @Valid String page, @NotNull @Valid String size,
57         @NotNull @Valid ColonneTableauListesPartageeARepublierDTO sort) {
58
59         // <R_SFD_LIS_CLR_FCT_003-1> L'application récupère les données de la liste des listes partagées à republier
60         // préalablement construite.
61         Page<DetailsListePartageeARepublierDTO> pageListePartageeARepublier = this.listePartageeARepublierApiService
62             .getListePartageeARepublier(Integer.parseInt(page), Integer.parseInt(size), order, sort.name())
63             .map(listePartageeARepublierMapper::asDTO);
64
65         // <R_SFD_LIS_CLR_FCT_002-1> L'application affiche l'écran de la liste des listes partagées à republier.
66         PageListeListePartageeDTO pageDto = new PageListeListePartageeDTO();
67         pageDto.setContent(pageListePartageeARepublier.getContent());
68         pageDto.setNumberOfElements(pageListePartageeARepublier.getNumberElements());
69         pageDto.setTotalElements(pageListePartageeARepublier.getTotalElements());
70         pageDto.setTotalPages(pageListePartageeARepublier.getTotalPages());
71         pageDto.setDateBatch(
72             this.dateMapper.convertOffsetDateTimeToDate(listePartageeARepublierApiService.getDateBatch()));
73
74         // <R_SFD_LIS_CLR_FCT_005-1> Lors de l'affichage de l'écran, l'application génère une trace utilisateur
75         // indiquant que l'utilisateur a consulté les listes partagées à republier
76         actionsUtilisateurService
77             .traceActionUtilisateur(LoggerTraceConstants.TRACE_CONSULTATION_LISTEPARTAGEEAREPUBLIER);
78
79         return ResponseEntity.ok(pageDto);
80     }
  
```

La méthode ci-dessus est extraite du contrôleur. Elle a pour fonction la création de l'end point et respecte les contraintes de conception. Les particularités de cette méthode sont les annotations utilisées :

@PreAuthorize (Ligne 54) : est une annotation qui dépend de Spring Security. Cela permet de vérifier les droits de l'utilisateur, envoyés par le **Json Web Token** dans le header de la requête.

@NotNull (ligne 56) : est une annotation qui permet de vérifier que le paramètre n'est pas nul.

@Valid (ligne 56) : est une annotation qui permet de vérifier les valeurs du paramètre.

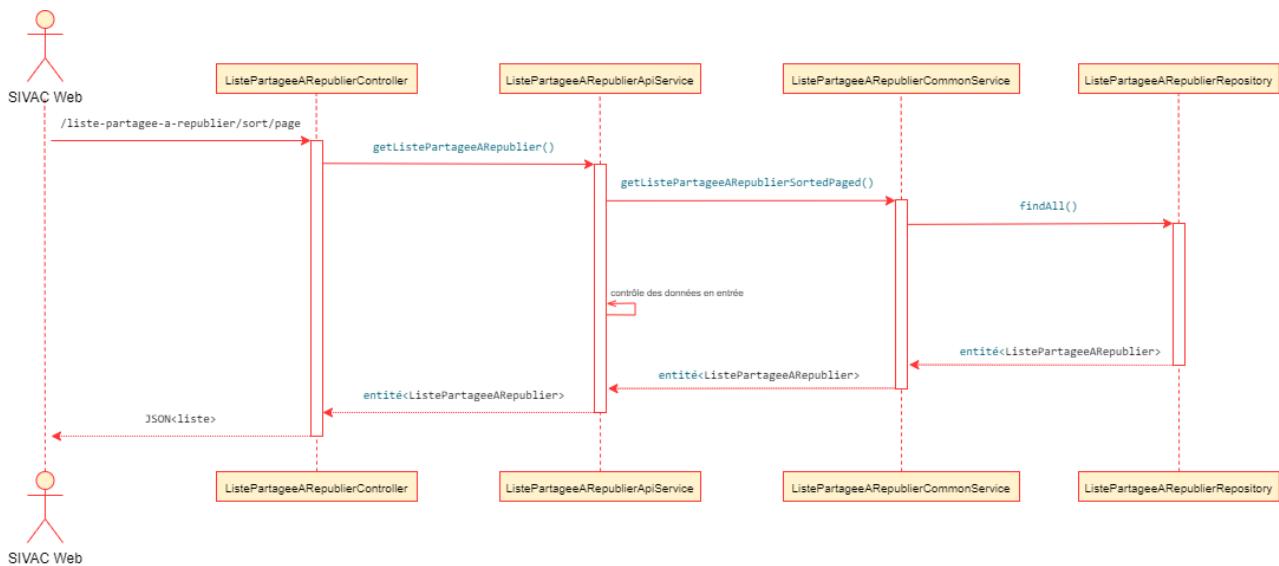


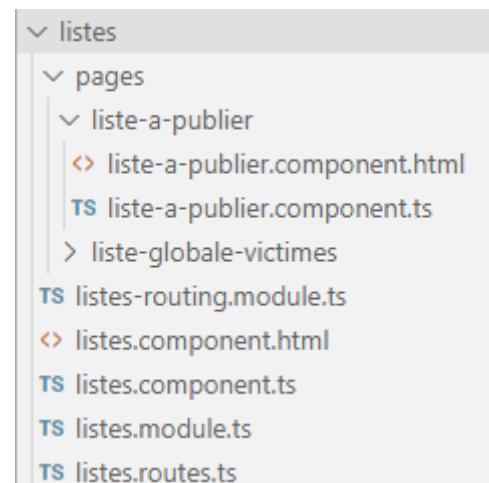
Diagramme de séquence de la fonctionnalité

C. Front end

Pour cette partie je devais réaliser l'écran permettant à l'utilisateur, avec les droits suffisants, de consulter la liste des listes à republier. Cette page est composée d'un tableau avec un ensemble de données paginées et triées, le résultat final est disponible en annexe.

L'architecture des packages de la fonctionnalité doit correspondre aux modules de l'application. Elle est créée dans le module listes qui comprend la liste globale des victimes et la liste des listes à republier. Ce composant comprend une classe Typescript avec des attributs, méthodes et constructeur, qui sert d'unité de découpage visuel et un Template (html), déclaré en tant que templateUrl dans la classe.

```
@Component({
  selector: 'app-liste-a-publier',
  templateUrl: './liste-a-publier.component.html'
})
```



Avant la construction de la page il a fallu déterminer les routes afin que le composant soit connecté au menu et au plan du site.

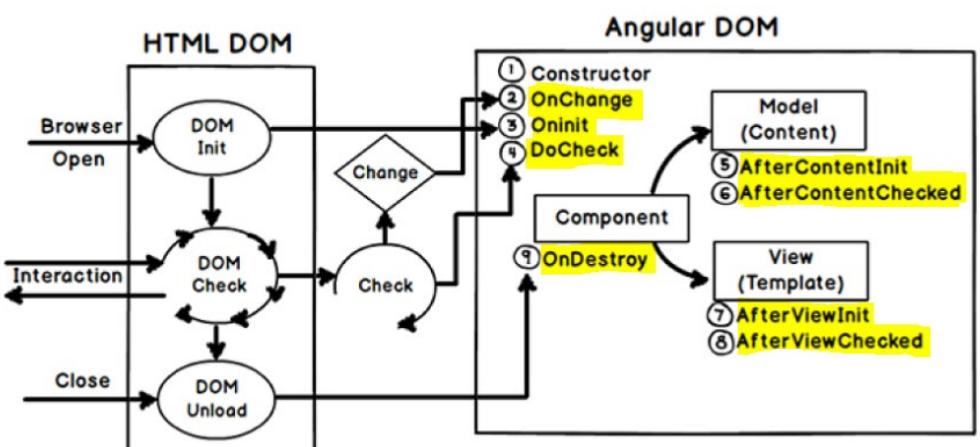
Pour ce faire, j'ai ajouté un chemin comme dans l'exemple ci-dessous qui est extrait de la classe « listes.routes.ts »

```

28
29     {
30         path: 'listes-partagee-a-republier',
31         component: ListeAPublierComponent,
32         canActivate: [PermissionGuard],
33         data: {
34             breadcrumb: breadcrumbConfig['liste_a_publier'],
35             droits: ['GESTION_LISTE_PARTAGEEE'],
36             title: 'LISTES.LISTE_A_PUBLIER.TITRE'
37         }

```

Avant de continuer il semble important d'aborder le cycle de vie d'un composant. Il s'agit de la définition de toutes les étapes de la naissance du Composant à sa mort (comme le montre le schéma). Dans une application Angular, il est utile de pouvoir se greffer à ces différentes étapes. Cela est réalisable grâce à ce que l'on appelle des **hooks**.



Dans l'extrait de code suivant, la méthode qui est appelée pour récupérer les données reçues du back end, je m'assure que cette récupération s'effectue après l'initialisation des vues, conformément au cycle de vie du composant, je l'appelle donc dans le ngAfterViewInit().

```

119    ngAfterViewInit() {
120        this.loadData().subscribe((data) => {
121            this.isLoadingResults = false;
122            if(data) {
123                const count = data.totalElements;
124                this.resultsLengthString = this.i18nSivacService.instant('LISTES.LISTE_A_PUBLIER.NOMBRE_RESULTAT', { count });
125                this.resultsLength = count;
126                const date = Optional.ofNullable(data.dateBatch)
127                    .map((d) => moment(d))
128                    .filter((m) => m.isValid())
129                    .map((m) => m.toDate())
130                    .map((d) => DateHeureUtils.dateVersString(d))
131                    .map((dh) => DateHeureUtils.extractDate(dh))
132                    .orElse('-');
133                const heure = Optional.ofNullable(data.dateBatch)
134                    .map((d) => DateHeureUtils.dateStrVersString(d, DateHeureUtils.FORMAT_HEURE_MINUTES_SECONDES))
135                    .orElse('-');
136                this.versionText = this.i18nSivacService.instant('LISTES.LISTE_A_PUBLIER.GENEREE_LE', { date, heure });
137                const content = data.content || [];
138                this.data = content.map((resultat) => {
139                    return this.resultToViewModel(resultat);
140                });
141            } else {
142                this.data = [];
143            }
144        });
145    }

```

La méthode « loadData » retourne un Observable, ce patron de conception permet de mettre en place un mécanisme de souscription pour envoyer des notifications à ces souscripteurs. Cet observable produit un « stream » de valeur. Dès réception du stream il faut effectuer plusieurs traitements afin de « préparer » les données à l'affichage côté template.

Le dernier extrait de code correspond à l'affichage de la colonne « nom événement » du tableau de liste des listes à republier dans le Template du composant.

```

18 <div table>
19   <!-- debut tableau -->
20   <mat-table [dataSource]="data" matSort class="sivac-table">
21
22     <!-- Nom evenement -->
23     <ng-container matColumnDef="nomEvenement">
24       <mat-header-cell class="wider" *matHeaderCellDef mat-sort-header>{{ 'LISTES.LISTE_A_PUBLIER.COLS.NOM_EVENEMENT' | i18nSivac }}</mat-header-cell>
25       <mat-cell class="wider" *matCellDef="let row">
26         <span>
27           {{ row.nomEvenement || "-" }}
28         </span>
29       </mat-cell>
30     </ng-container>

```

A la ligne 20 à l'intérieur de la balise <mat-table> la variable « data » correspond à la variable data de la classe Typescript (ligne 138 de l'extrait précédent). Afin de pouvoir afficher le nom de la colonne, je dois prendre en compte l'internationalisation de l'application. Pour ce faire je stipule la clé pour récupérer la valeur dans le fichier Json de langue correspondant à la langue par défaut de l'utilisateur, gérée directement par la classe i18nSivac. Ensuite chaque cellule créée par la balise <mat-cell> affichera la valeur de la variable nomEvenement.

Les balises du tableau proviennent de la librairie Angular Material.

6.2. Maintenance corrective

Le Client a relevé une anomalie liée à la gestion de la confidentialité parquet d'un individu, cette anomalie est sous garantie.

Cas testé :

1) Événement PNAT avec :

- VD Pierre Duval confidentiel, qui a un proche Monique Duval,
- VD Jean Marchand.

2) Publication LP : Seul Jean Marchand est dans la liste.

3) Événement FGTI avec :

- VD Pierre Duval qui a un proche Monique Duval ;
- VD Monique Duval qui a un proche Pierre Duval ;
- VD Jean Marchand.

4) Création EG :

- VD unifié : Monique Duval qui a un rôle FGTI, pas de lien de proximité, et qui a un poche unifié Pierre Duval dans l'événement FGTI.
- VD unifié : Jean Marchand.
- VD unifié mais confidentiel : Pierre Duval.

5) Liste à partager :

- Jean Marchand PNAT_FGTI ajouté : Ok
- Monique Duval (FGTI) ajouté : Ok

A cette étape du traitement, la liste partagée est en attente de publication.

Lorsque la confidentialité parquet d'un rôle est mise à « Oui », si une liste partagée existe sur l'événement et que l'individu y est présent :

- L'individu et ses proches sont retirés de la liste partagée de l'événement,
- la liste partagée est notée comme étant à publier.

Dans le cas testé, Pierre Duval n'a jamais été dans la liste partagée puisqu'il est confidentiel avant la publication de cette liste. La liste partagée ne doit pas être mise en attente de publication selon cette règle.

Après analyse du code, lors de l'unification de l'individu (Pierre Duval dans l'exemple), à la création de l'événement global, il est seulement vérifié que la confidentialité parquet de l'individu unifié (lors de l'unification des rôles) existe sans vérifier si la liste partagée de l'événement PNAT contient cet individu. Sans cette vérification la méthode considère que si une personne de l'événement a cette confidentialité la liste partagée devient provisoire.

La phase corrective nécessite l'ajout d'une vérification supplémentaire qui se trouve dans la méthode `unifierRole()` de la classe `RegroupementEvenementsCommonService`.

La première étape est la récupération de l'événement unitaire avec une structure d'appartenance PNAT

```

1 usage ANNUAIRE\riyad.hennaoui +1
329 @
private Optional<Evenement> getEvenementPnatDansEvenementGlobal(Evenement evenementGlobal) {
330     return evenementGlobal.getEvenementsRegroupes().stream()
331         .filter(e -> Objects.equals(e.getStructureAppartenance().getIdFonctionnel(),
332             ReferentielConstantes.STRUCTURE_APPARTENANCE_PARQUET_ID_FONCTIONNEL))
333         .findFirst();
334 }

```

L'extrait de code ci-dessus est une méthode qui récupère la liste des événements unitaires que je stream afin de filtrer par ID fonctionnel qui retourne le premier élément de type Optional.

La deuxième étape consiste à récupérer la liste des ID des individus originaires d'un événement PNAT, pour ce faire

```

1 usage ANNUAIRE\riyad.hennaoui +1
342 @
private List<Long> getListePartageePnatIndividusId(Evenement evenement) {
343     return getEvenementPnatDansEvenementGlobal(evenement).map(
344         evenementPnat -> listePartageeRepository.findFirstByEvenementOrderByListePartageeIdDesc(evenementPnat) Optional<ListePartagee>
345             .stream().flatMap(listePartagee -> listePartagee.getIndividusPartages().stream()) Stream<IndividuPartage>
346                 .map(IndividuPartage::getId).collect(Collectors.toList()))
347             .orElseGet(Collections::emptyList);
348 }

```

L'extrait de code ci-dessus est une méthode récupérant l'événement PNAT retourné par la première méthode. Grâce au type Optional, il est possible d'utiliser un stream afin de récupérer la liste des individus de la liste partagée pour créer une liste d'identifiants de chaque individu, ou une liste vide si la liste partagée n'existe pas.

La dernière étape est de vérifier la correspondance entre les id des individus unitaires et les id des individus de la liste partagée de l'événement unitaire.

```

1 usage ANNUAIRE\riyad.hennaoui
357 @
private boolean isRolesLiesEqualsIndividusListePartagee(RoleIndividuEvenement roleLie,
358     List<Long> individusIdListePartagee) {
359     return roleLie.getVictime().getIndividusUnitaires().stream().map(Individu::getIndividuId)
360         .anyMatch(individusIdListePartagee::contains);
361 }

```

Cette méthode est appelée afin de compléter la vérification de la confidentialité parquet de la méthode unifierRole().

```

310     if (roleLie.isConfidentialiteParquet())
311         && isRolesLiesEqualsIndividusListePartagee(roleLie, listePartageeIndividusId)) {
312             isNouvelleConfidentialiteParquet = true;
313         }

```

7. Tests et validation

7.1. Tests

Les tests se font pour la majorité du code de l'application. Nous y trouverons des tests unitaires, tests d'intégration et tests RGAA côté IHM pour l'accessibilité des personnes ayant des déficiences visuelles.

A. End point

Dans le cadre du développement de la fonctionnalité de consultation de la liste des listes à partager, j'ai réalisé un test d'intégration.

Pour ce faire j'ai dû créer un jeu de données que j'injecte dans la base de données.

```

255     riyad.hennaoui +1
256     @Test
257     @Sql({scripts = {"/clean_data.sql", "classpath:listearerepublier/script_test_listes-a-recalculer-existantes.sql"}, executionPhase = ExecutionPhase.BEFORE_TEST_METHOD})
258     @WithMockCustomUser(authorities = {DroitsConstants.GESTION_LISTE_PARTAGEE}, structureAppartenance = 3L)
259     public void testGetListeARepublier_checkListeARepublier() throws Exception {
260         Utilisateur utilisateur = utilisateurUtils
261             .creerEtInjectInSecurityContextUtilisateur(SecurityUtil.getCurrentUser().getUserId());

```

Sur l'extrait ci-dessus, pour pouvoir tester je dois en plus de lancer le script sql du jeu de données, mocker un utilisateur avec le droit et le bon profil pour utiliser le end point.

```

ResultActions result = mockMvc
    .perform(MockMvcRequestBuilders.get(urlTemplate: ENDPOINT_URL + "/sort/page")
        .param(ApiParamsConstants.PAGE_NAME, ...values: "1").param(ApiParamsConstants.SIZE_NAME, ...values: "10")
        .param(ApiParamsConstants.DIRECTION_NAME, ApiParamsConstants.DIRECTION_ASC.toUpperCase())
        .param(ApiParamsConstants.SORT_NAME, ...values: "NOM_EVENEMENT")
        .contentType(MediaType.APPLICATION_JSON_VALUE).content(json))
    .andExpect(jsonPath(expression: "$.content.[0].evenementId").value(expectedValue: "16000"))
    .andExpect(jsonPath(expression: "$.content.[0].nomEvenement").value(expectedValue: "PN-EVT-L"))
    .andExpect(jsonPath(expression: "$.content.[0].versionListePartagee").value(expectedValue: "1"))

```

Le test vérifie les données retournées par le fichier JSON de l'end point, comme indiqué dans le code ci-dessus.

B. IHM

Les tests RGAA d'accessibilité sont réalisés avec Cypress et Cucumber, le principe étant de décrire ce que doit faire un utilisateur comme dans l'extrait de code ci -dessous.

```

1  @browser
2  @rgaa
3  @rgaa-listes
4  @rgaa-listes-listings
5
6  Feature: Structuration de listes - Listings
7
8  Vérification des listes affichées
9
10 Background:
11   Given je fixe ma résolution à 1920 par 1080 pixels
12
13 @rgaa-listes-listings-1
14 Scenario: Vérification des listes affichées (bavpa)
15   Given je me connecte à l'application en tant que "bavpa"
16
17   When je navigue vers l'écran "PJS_ECRAN_053_LIS_LISTES_PARTAGEES_A_REPUBLIER"
18   Then les listes suivantes sont affichées :
19     | fil d'ariane | liste ordonnée | |
20
21   Then je me déconnecte de l'application

```

Afin de lancer ces tests, il est nécessaire de déployer l'application sur les serveurs AWS que j'ai réalisé sur une machine distante dédiée à la réalisation des tests.

C. Maintenance corrective

Pour la correction de l'anomalie soulevée par le client, j'ai créé un test d'intégration. Enfin pour le jeu de test j'ai réalisé un dataset en java afin de reproduire le cas du client.

```

464  ANNUAIREVriyad.hennaoui +1
465  @Test
466  @WithMockCustomUser(authorities = { DroitsConstants.GESTION_LISTE_PARTAGEE, DroitsConstants.GESTION_ARBO })
467  public void testAjouterLienRegroupementAvecRapprochementAutomatiqueIndividus_confParquetAvantCreationEvenementGlobal()
468      throws ParseException {
469      TestDataSet dataSet = dataSetBuilder.initDataSetAvec2EvtsUnitaires3VictUnitaires();
470      Evenement evenementPnat;
471      Evenement evenementFgti;
472
473      evenementPnat = dataSet.getEvenements().stream()
474          .filter(evenement -> evenement.getStructureAppartenance().getIdFonctionnel()
475              .equals(structureAppartenanceService.getStructureAppartenanceParquet().getIdFonctionnel()))
476          .findFirst().orElseThrow();
477
478      evenementFgti = dataSet.getEvenements().stream()
479          .filter(evenement -> evenement.getStructureAppartenance().getIdFonctionnel()
480              .equals(structureAppartenanceService.getStructureAppartenanceFgti().getIdFonctionnel()))
481          .findFirst().orElseThrow();

```

Dans l'extrait de code ci-dessus, je commence par récupérer le dataset créé et en extrais les événements PNAT et FGTI. Cela servira à vérifier que lors de la création de la liste partagée l'individu avec une confidentialité parquet n'y apparaît pas.

```

assertEquals( expected: 1, listeIndividus.size());
assertEquals( expected: "Jean",
    listeIndividus.get(0).getPersonneLiee().getFicheIdentite().getPrenoms().get(0).getLibellePrenom());

```

J'ai également modifié plusieurs tests unitaires existants, appelant la méthode unifierRole(), en mockant une liste partagée vide, comme dans l'exemple ci-dessous.

```
when(listePartageeRepository.findFirstByEvenementOrderByListePartageeIdDesc(any()))
    .thenReturn( value: Optional.empty());
```

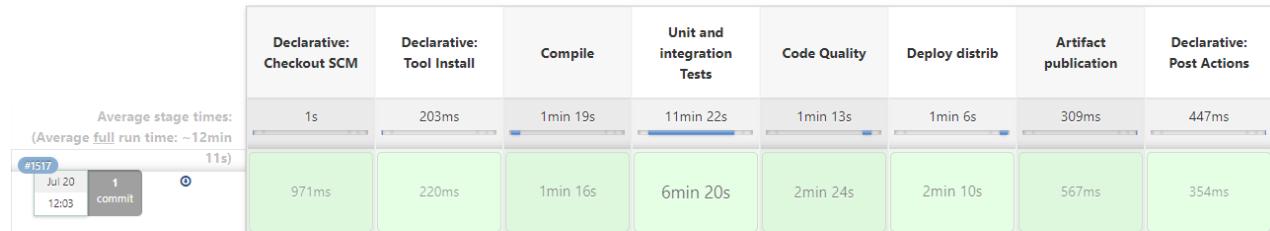
7.2. Validation

Les étapes de validation de chaque tâche avant l'intégration de celle-ci aux fonctionnalités livrées au client sont le passage par l'outil Jenkins puis Sonar. Enfin, la dernière étape est la validation par l'équipe fonctionnelle.

A. Jenkins

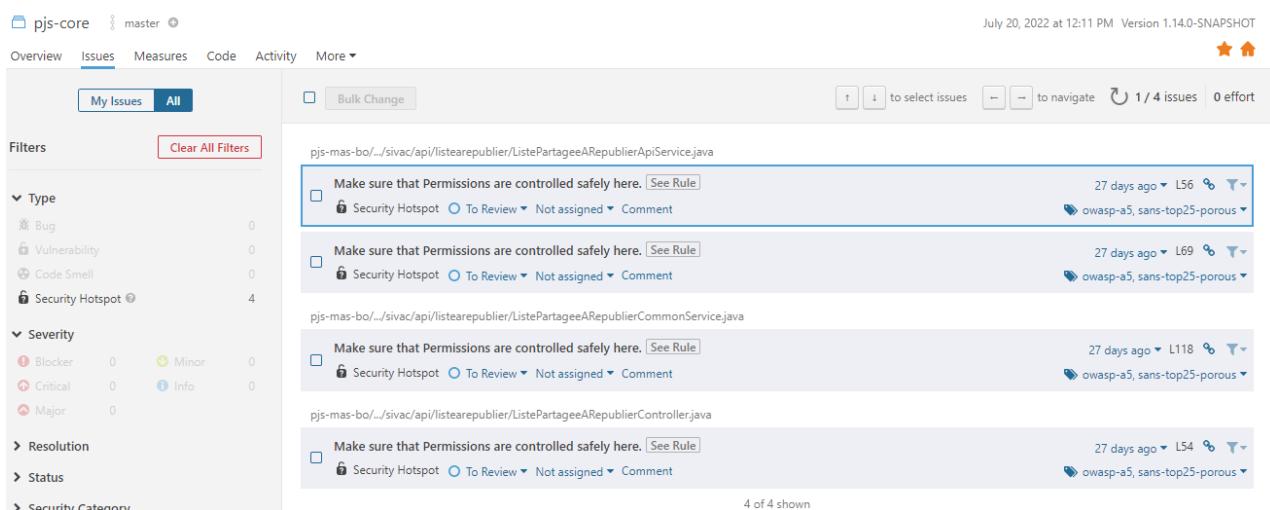
Avant la mise en Merge Request le code produit doit être lancé sur le pipeline de test de Jenkins qui s'occupe de vérifier l'exécution du code, dans une démarche d'intégration continue.

Stage View



B. SonarQube

Après le passage du code par Jenkins, il faut vérifier la qualité du code avec SonarQube qui est capable d'identifier, classifier et propose la résolution des défauts. Il permet également d'identifier les duplications de code et de mesurer la couverture de test déployé.

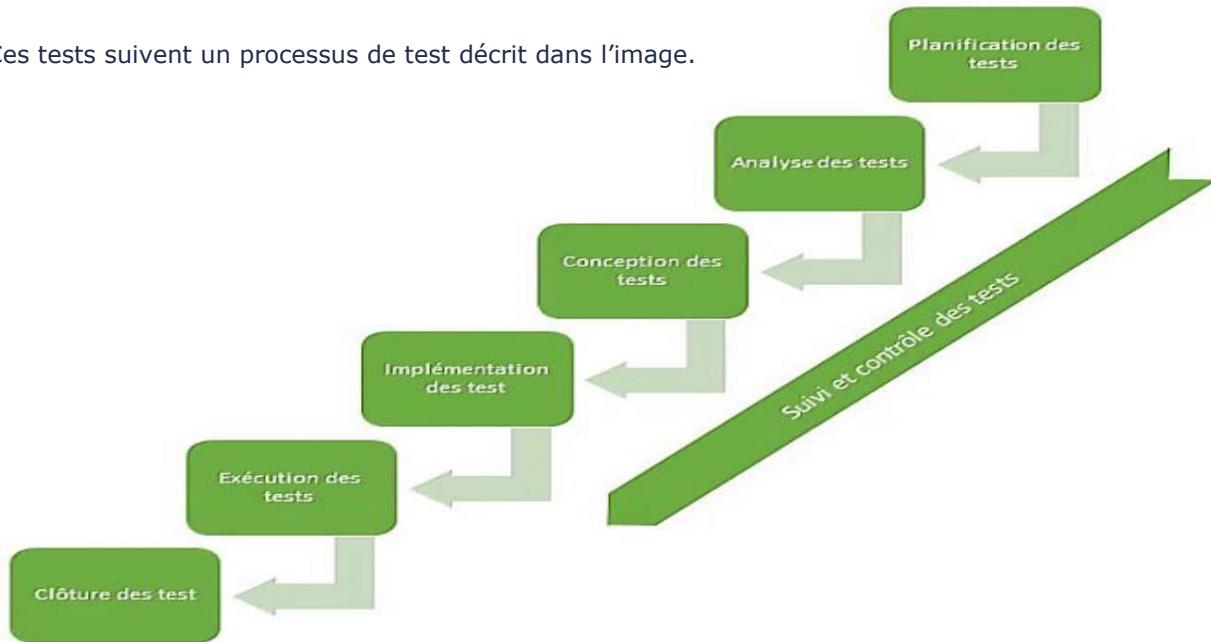


C. Validation fonctionnel

Cette dernière étape, est effectuée par l'équipe fonctionnelle, qui s'assure que la fonctionnalité développée respecte les spécifications et n'entraîne pas de régression au niveau de l'application.

Pour ce faire il est nécessaire de déployer la nouvelle version de l'application embarquant les fonctionnalités développées et mergées.

Ces tests suivent un processus de test décrit dans l'image.



8. Bilan de projet

L'ensemble des objectifs fixes ont été atteints, j'ai participé pleinement à deux sprints dans l'équipe de développement. Pour chacun d'entre eux, j'ai été épaulé par l'ensemble de l'équipe, tant côté développement que côté fonctionnel. Un autre point positif est le processus d'intégration au projet me permettant d'appréhender le projet sereinement, les différents travaux pratiques m'ont permis de valider la théorie présentée par la cheffe de projet.

Au total j'ai eu la responsabilité de la réalisation de 5 tâches, 3 fonctionnalités et deux retours validation. J'ai écrit plus de 1000 lignes de code, pour le développement des fonctionnalités. Cela m'a pris en moyenne 5 jours pour chacune et une demi-journée pour les retours validation.

Mes réalisations ont toutes été livrées au client.

Au terme de ce stage, j'ai réussi à prendre en main les différentes technologies du périmètre du projet.

9. Bilan et perspectives

Les 12 semaines que j'ai passées au sein de l'équipe SIVAC se sont écoulées rapidement. J'ai particulièrement apprécié la collaboration et l'esprit de groupe qui y régnait. Je suis très satisfait d'avoir mené à bien les tâches qui m'ont été confiées et très fier qu'elles aient été livrées au client.

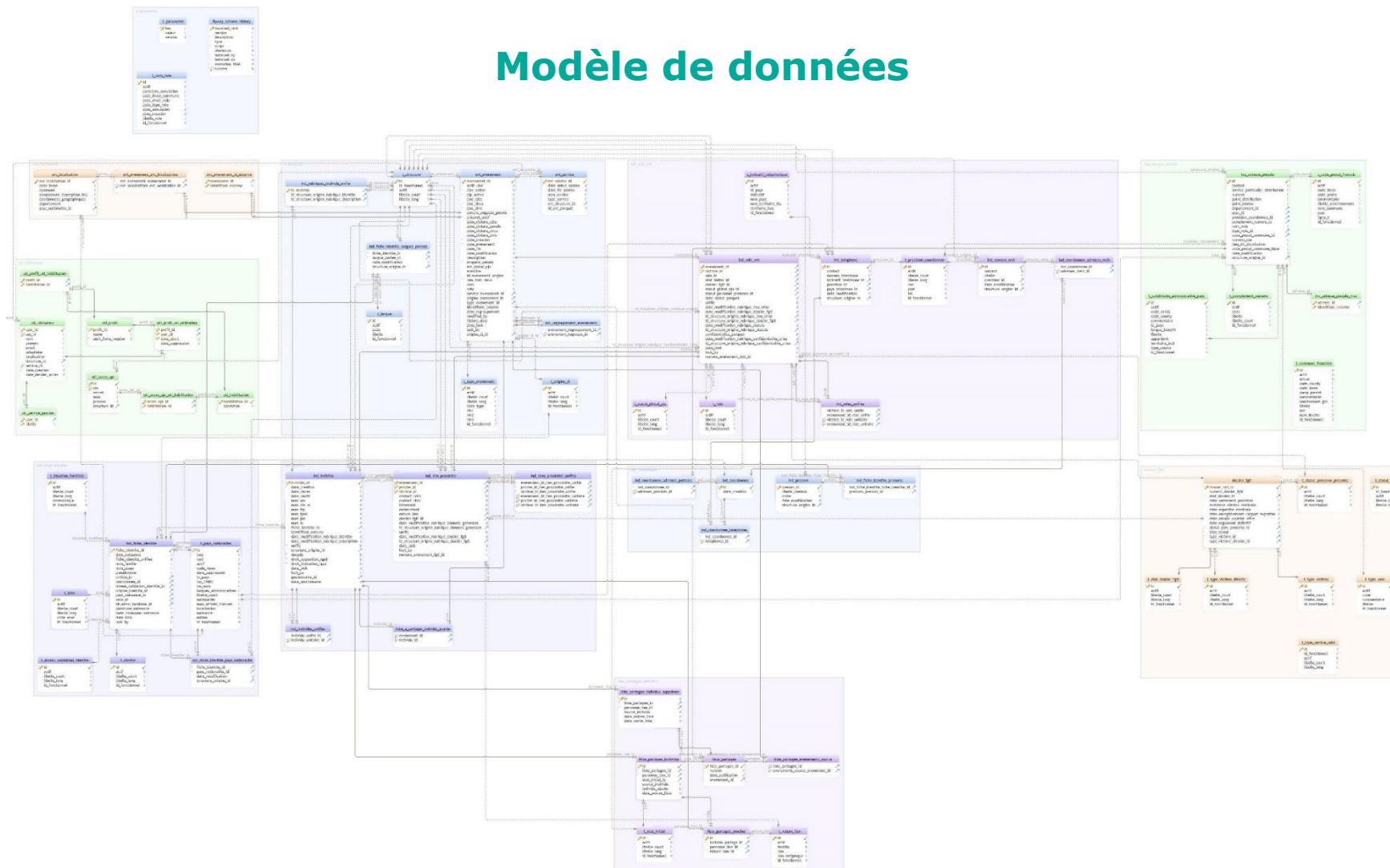
Cette première expérience m'a permis de découvrir le monde du développement dans une entreprise dédiée au numérique et d'interagir avec les acteurs du projet qui ont enrichi ma vision et m'ont conforté dans l'idée de faire carrière dans cette branche.

Enfin, elle s'est soldée par un retour positif de l'équipe ce qui a amené à une proposition de CDI avant le terme de cette période.

Ayant accepté cette proposition, je peux aborder la suite avec sérénité.

Annexes

1- Modèle de données.....	61
2 - Ecran réalisé	62
3- Diagramme de classe Evènement	63
4 - Diagramme de classe Individu	64
5 - Diagramme de classe Référentiel.....	65



1- Modèle de données

 MINISTÈRE DE LA JUSTICE
Liberté Egalité Fraternité



SIVAC Système d'Information interministériel des Victimes d'Attentats et de Catastrophes

Mercredi 27 juillet 2022 09:30 

Données confidentielles | Utilisateur : Alice Champ
Durée limite de conservation des impressions : 6 mois



 ACCUEIL

 ÉVÉNEMENTS

 INDIVIDUS

 LISTES

Liste globale des victimes

Listes partagées à republier

 IMPORT

AC 
Alice Champ
BAVPA

Listes partagées à republier

Listes > Listes partagées à republier

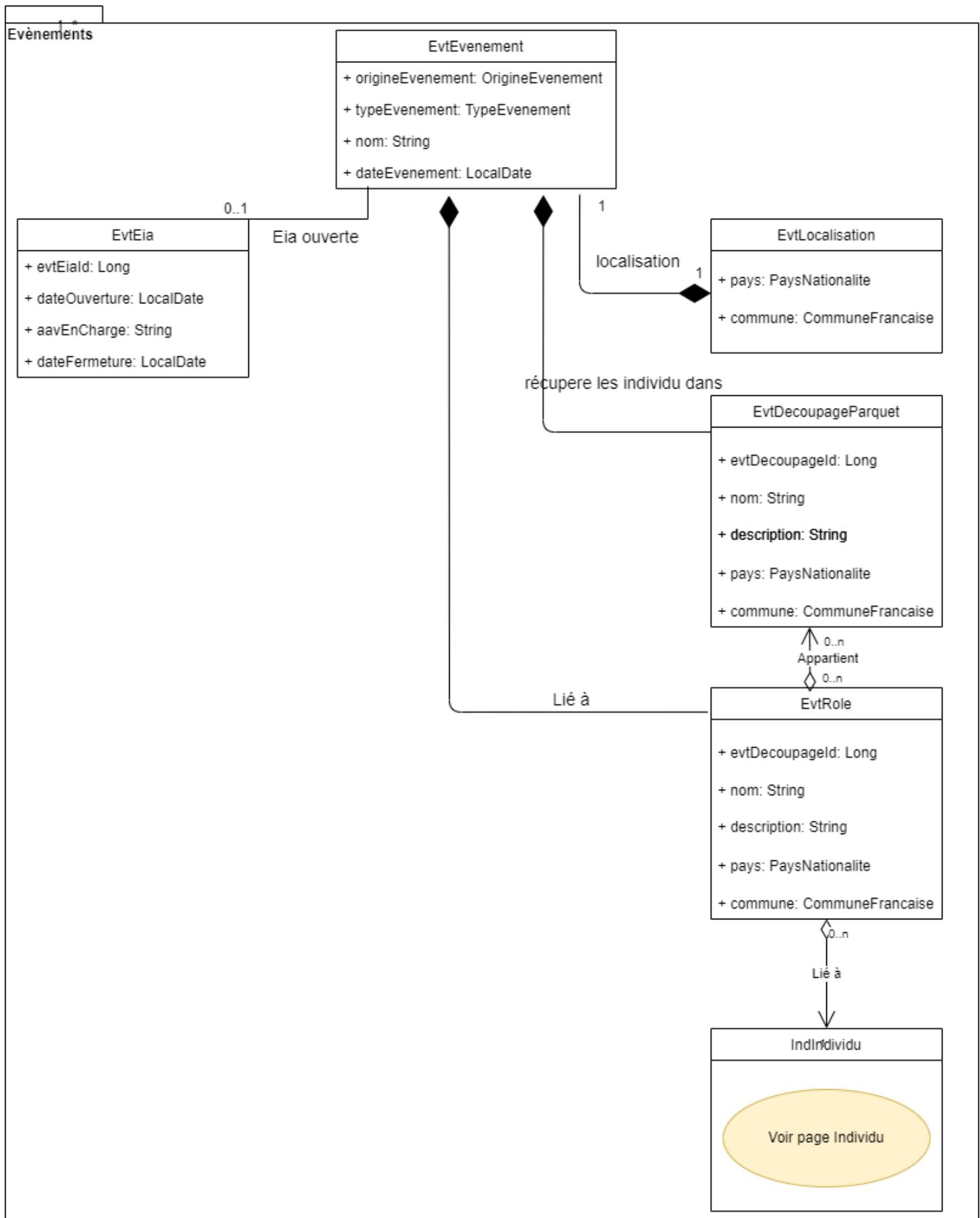
Générée le 07/07/2022 à 22:57:33 | 83 listes

Résultats par page 10 ▾ 1 – 10 sur 83 | < < > >|

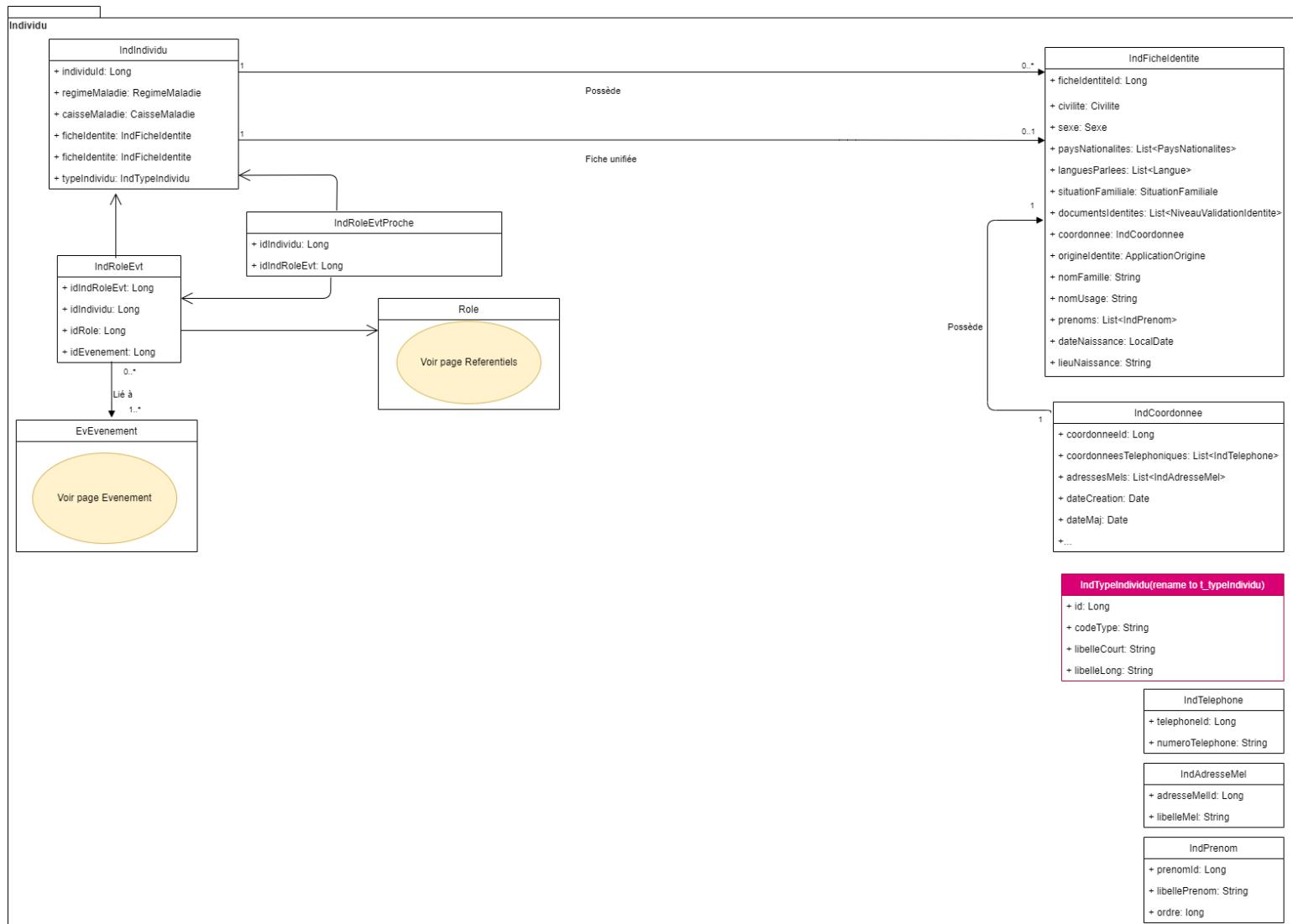
Evénement de rattachement	Date événement	Nombre d'individus ajoutés	Version de la liste partagée	Date de publication	Republiée	Lien vers la liste à partager
EG NICE (157)	14/07/2016	1955	V8	01/04/2021	Non	
STRASBOURG (108)	10/12/2018	57	V2	27/05/2021	Non	
CONDE-SUR-SARTHE-PRISON (135)	04/03/2019	1	V2	21/04/2021	Non	
NIGER-KOURE (89)	09/08/2020	6	V2	28/07/2021	Oui	
EG test MSL 2 (969)	24/03/2021	1	V5	09/11/2021	Non	
EG test MSL 3 (945)	31/03/2021	2	V5	25/05/2021	Non	
test MSL 5 (171)	01/04/2021	4	V1	01/04/2021	Non	
EG JFL ESSAI 1 (643)	07/04/2021	5	V1	07/09/2021	Oui	

2021 © Ministère de la Justice | Application SIVAC 1.14.0-SNAPSHOT | [Plan du site](#) | Accessibilité : Non conforme

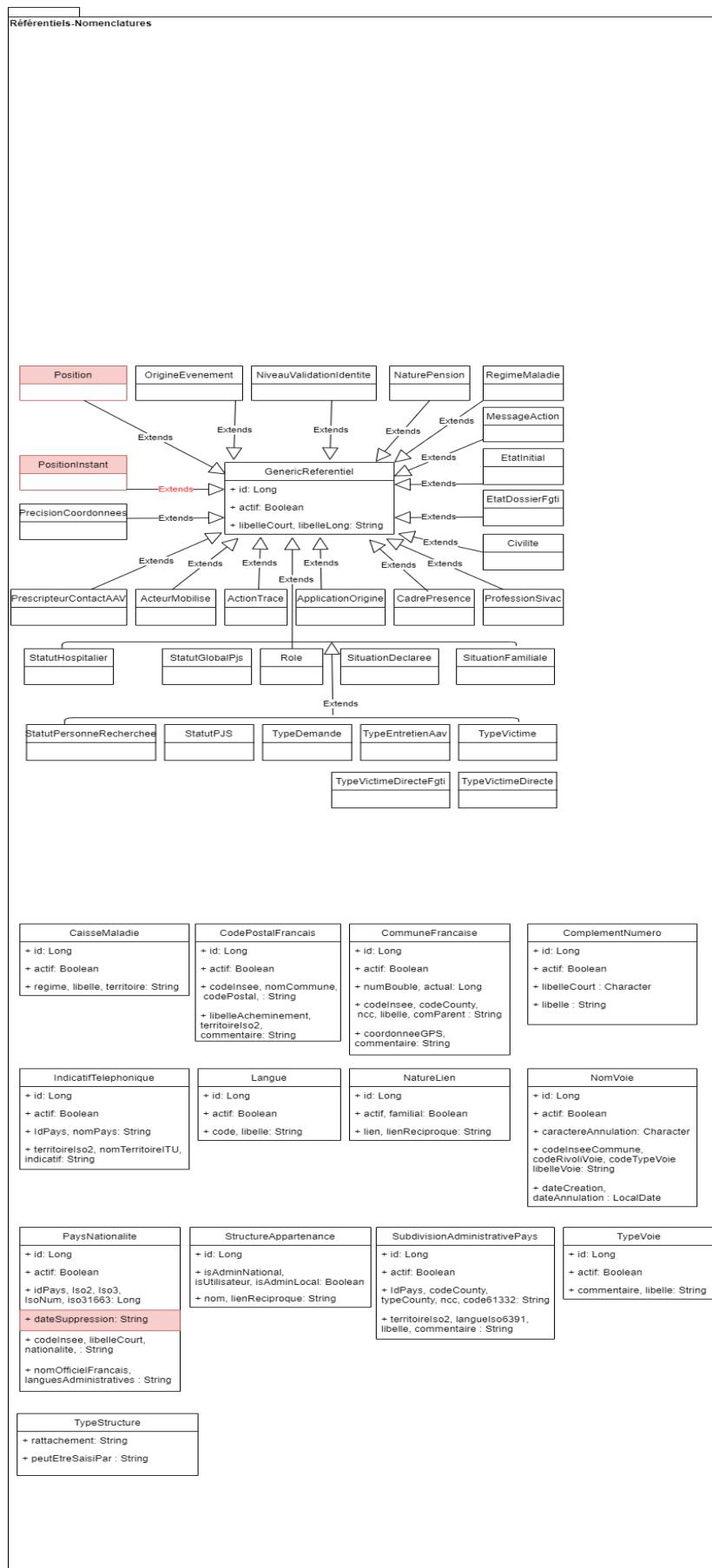
2 - Ecran réalisé



3- Diagramme de classe Evènement



4 - Diagramme de classe Individu



5 - Diagramme de classe Référentiel