

Projet RNR – Site Grand Public

Documentation d'Architecture Technique

Version 1.5 du 10/12/2025

Historique des versions

Version	Date	Auteur	Description
1.0	12/02/2019	Déborah KASSABI	Version initiale du document
1.1	01/04/2019	Déborah KASSABI	Mise à jour du document dans le cadre de la migration JBOSS 7/JAVA 8
1.2	10/09/2021	Déborah KASSABI	Mise à jour du document dans le cadre de la migration JBOSS 7.3.6/JAVA 11
1.3	16/01/2023	Bastian CHARLET	Mise à jour du document Tomcat 10.0.24 / JDK1
1.4	04/07/2024	Déborah KASSABI	Mise à jour du document au sujet de la fusion des services transfert et import
1.5	10/12/2025	Julien Le Fur	Mise à jour du document lors de la migration Angular version 17

Visas

Rôle	Nom	Date et visa
Auteur	Déborah Kassabi	12/02/2019
Valideur Nat System	Jean-Luc Tholozan	
Valideur ABM	David Vitte	

Table des matières

1	Introduction	4
2	Architecture de l'application	4
2.1	Architecture logique	4
2.2	Pile technique	5
2.2.1	Backend.....	5
2.2.2	Frontend.....	6
2.3	Structure du projet.....	6
2.4	Poste client	6
2.4.1	Architecture matérielle.....	6
2.4.2	Architecture Angular.....	10
3	Déploiement et infrastructure	11
3.1	Conteneurisation	11
3.2	Orchestration.....	11
3.3	Configuration	11
4	Les services	12
4.1	Service keygen	12
4.2	Service import.....	12

1 INTRODUCTION

Ce document décrit l'architecture technique de l'application **POPP Grand Public**. L'application permet la gestion des demandes via une interface publique. Elle se compose d'un Front-End Angular et d'un Back-End Java Spring Boot. Particularité notable : le Back-End ne communique pas en direct avec une base de données applicative persistante pour le stockage métier, mais génère des flux XML traités par un système tiers.

La présentation de la plate-forme technique se décompose en 2 phases

- Architecture applicative
- Les services

Décrivant l'architecture logique et matérielle pour chacune.

Nous nous plaçons d'un point de vue client, serveur et poste de développement.

2 ARCHITECTURE DE L'APPLICATION

2.1 Architecture logique

L'application est découpée en deux tiers principaux :

- **Frontend (Application Public)** : Application Single Page (SPA) développée en Angular. Elle est servie aux utilisateurs via un navigateur web.
- **Backend (API)** : Application Java Spring Boot exposant des services REST. Elle reçoit les données du Frontend, les valide, et génère des fichiers XML.

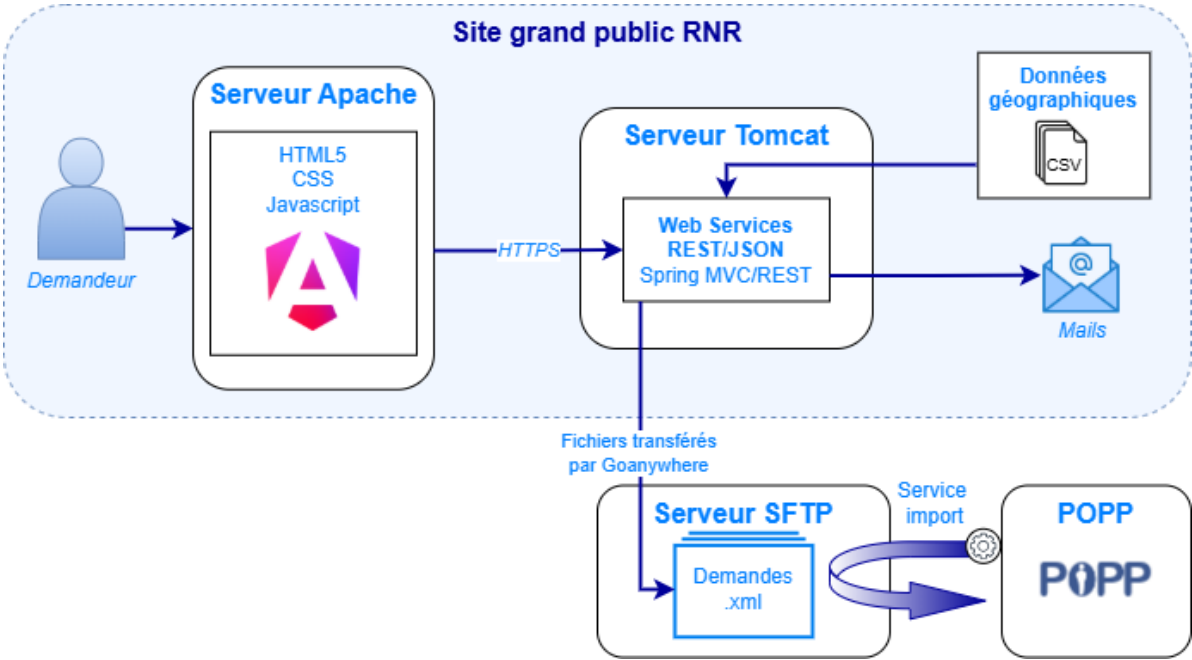


Figure 1: Architecture logique

Flux de données principal :

1. L'utilisateur saisit une demande sur le Frontend.
2. Le Frontend envoie un DTO JSON au Backend via REST.
3. Le Backend transforme ce DTO en XML (via JAXB).
4. Le fichier XML est chiffré et déposé sur un serveur de fichiers via GoAnywhere.
5. Une notification par email est envoyée.

2.2 Stack technique

2.2.1 Backend

Le backend est développé en JAVA avec le framework Spring Boot.

Technologie	Version	Usage
Java	17	Langage de programmation
Spring Boot	3.3.5	Framework applicatif
Tomcat	10.1	Conteneur de servlet
Maven	3+	Outil de build
itextpdf	5.5.13.3	Librairie de gestion des documents
jakarta.xml.bind	4.0	JAXB pour XML
jakarta.mail	2.1.3	Envoi d'emails

2.2.2 Frontend

Le frontend est une Single Page Application (SPA) développée avec Angular.

Technologie	Version	Usage
Angular	17.3.1	Framework Frontend
PrimeNG	17.17.0	Bibliothèque de composants UI
RxJS	7.8.0	Programmation réactive
PrimeIcons	7.0.0	Librairie PrimeNG d'icônes
Bootstrap	5.3.3	Framework css
Typescript	5.3.2	

2.3 Structure du projet

Module	Description
biomedecine-popp-gd-public-bo	Module contenant une couche d'exposition REST contenant les points d'entrées de l'application Backend ainsi que le code source Java.
biomedecine-popp-gd-public-angular	Code source de l'application Angular
rnr-integrator	Module indépendant contenant le process de construction du livrable.

2.4 Poste client

L'application RNR est un site grand public accessible depuis un poste banalisé avec un navigateur Internet (Internet Explorer, Chrome ou Mozilla Firefox).

Le navigateur affiche la page Angular correspondant à sa demande via une route (URL). De manière générale, à chaque fonctionnalité de l'application correspond une route Angular qui charge le composant.

2.4.1 Architecture matérielle

2.4.1.1 Poste de développement

Composant	Nom	Version
OS	Windows/Linux/MacOS	
IDE	IntelliJ / Eclipse / Visual Studio	

JAVA	JDK	17
Outil de build	Maven	3+
Service de transfert de fichier	SFTP	
Framework	Spring Boot	3.0.1

2.4.1.2 Les données

L'application Front Office ne dispose pas de Base de données à proprement parler. Des fichiers de données sont utilisés pour les listes de données de références. Les informations saisies par un utilisateur lors de sa demande sont stockées dans un fichier XML

2.4.1.2.1 Fichier de données de référence

Les fichiers de références des pays, communes et codes postaux sont de type csv (séparateur « ; ») avec les structures suivantes :

- Fichier des pays :

Donnée	Type	Description
idPays	Integer	Identifiant du pays
nom	String	Nom du pays

- Fichier des départements :

Donnée	Type	Description
IdDep	Integer	Identifiant du département
Code	Integer	Code du département
nom	String	Nom du département

- Fichier des communes :

Donnée	Type	Description
IdCom	Integer	Identifiant de la commune
IdDep	Integer	Identifiant du département
Libelle	String	Nom de la commune
Cp	Integer	Code postal de la commune

- Fichier des code postaux :

Donnée	Type	Description
IdCom	Integer	Identifiant de la commune

Libelle	String	Nom du pays
---------	--------	-------------

Les fichiers de références géographiques sont générés par requête sur les tables de références de l'application POPP. Les requêtes d'extraction des données sont précisées dans le document d'installation.

2.4.1.2.2 Fichier demande RNR

La saisie d'un formulaire de demande sur le site grand public entraîne la création d'un fichier XML dont la structure est la suivante :

Balise	Balise mère	Type	Obligatoire	Commentaire
demande		nœud	O	Élément principal de la demande
type_demande	demande	string (enum)	O	Type de la demande 3 valeurs possibles: - inscription - modification - annulation
num_dossier	demande	integer(19)	N*	Numéro du dossier *Obligatoire les demandes de modification et d'annulation
date_creation	demande	dateTime	O	Date de création de la demande
nom	demande	string(100)	O	Nom du demandeur
nom_usuel	demande	string(100)	O	Nom usuel du demandeur
prenom1	demande	string(100)	O	Prénom du demandeur
prenom2	demande	string(100)	N	Prénom du demandeur
prenom3	demande	string(100)	N	Prénom du demandeur
sexe	demande	string(enum)	N*	Sexe du demandeur *Obligatoire pour les demandes d'inscription et de modification 2 valeurs possibles: M et F
date_naissance	demande	dateTime	N*	Date de naissance du demandeur *Obligatoire pour les demandes d'inscription et de modification

ville_naissance	demande	String(100)	N*	Ville de naissance du demandeur *Obligatoire pour les demandes d'inscription et de modification
dep_naissance	demande	string(40)	N	Département de naissance du demandeur
pays_naissance	demande	integer(5)	N	Pays de naissance du demandeur
adresse	demande	string(100)	N*	Adresse du demandeur *Obligatoire pour les demandes d'inscription et de modification
ville	demande	string(100)	N*	Ville du demandeur *Obligatoire pour les demandes d'inscription et de modification
cp	demande	string(10)	N*	Code postal du demandeur *Obligatoire pour les demandes d'inscription et de modification. obligatoire pour l'étranger ?
pays	demande	integer(5)	N*	Pays du demandeur *Obligatoire pour les demandes d'inscription et de modification
mail	Demande	string(100)	N*	Mail du demandeur *Obligatoire pour les demandes d'inscription et de modification
therapeutique	demande	nœud	N	Nœud pour les choix thérapeutique
tous_organes	therapeutique	boolean	N	Choix de refus thérapeutique sur tous les organes
foie	therapeutique	boolean	N	Choix de refus thérapeutique sur le foie. Absent si "Tous les organes" choisi.
reins	therapeutique	boolean	N	Choix de refus thérapeutique sur les reins. Absent si "Tous les organes" choisi.
cœur	therapeutique	boolean	N	Choix de refus thérapeutique sur le cœur. Absent si "Tous les organes" choisi.
poumons	therapeutique	boolean	N	Choix de refus thérapeutique sur les poumons. Absent si "Tous les organes" choisi.
pancreas	therapeutique	boolean	N	Choix de refus thérapeutique sur le pancreas. Absent si "Tous les organes" choisi.
intestins	therapeutique	boolean	N	Choix de refus thérapeutique sur les intestins. Absent si "Tous les organes" choisi.

tous_tissus	therapeutique	boolean	N	Choix de refus thérapeutique sur tous les tissus
cornees	therapeutique	boolean	N	Choix de refus thérapeutique sur les cornées. Absent si "Tous les tissus" choisi.
peau	therapeutique	boolean	N	Choix de refus thérapeutique sur la peau. Absent si "Tous les tissus" choisi.
vaisseaux	therapeutique	boolean	N	Choix de refus thérapeutique sur les va. Absent si "Tous les tissus" choisi.
valves	therapeutique	boolean	N	Choix de refus thérapeutique sur les valves. Absent si "Tous les tissus" choisi.
os_tendons_cartilages	therapeutique	boolean	N	Choix de refus thérapeutique sur les os, les tendons et les cartilages. Absent si "Tous les tissus" choisi.
scientifique	demande	boolean	N	Choix de refus pour la recherche scientifique
autopsie	demande	boolean	N	Choix de refus pour l'autopsie
pièces_jointes	demande	nœud	O	Nœud pour les pièces jointes
pj	pièces_jointes	base64Binary	O	Fichier pièce jointe en base64 Deux pj maximum possibles
type_confirmation	demande	string(enum)	N*	Type de confirmation souhaitée par le demandeur. 2 valeurs possibles: - mail - courrier *Obligatoire pour les demandes d'inscription et de modification
conditions_generales	demande	Boolean	O	Confirmation de lecture des conditions générales

2.4.2 Architecture Angular

2.4.2.1 Vision et objectifs Angular

L'objectif est de fournir une interface utilisateur réactive, moderne et maintenable, séparée du Backend. L'utilisation d'Angular 17 permet de bénéficier des dernières performances et fonctionnalités (Signaux, Control Flow, etc.).

2.4.2.2 Architecture générale

L'application suit une architecture dite en standalone, à l'inverse de l'architecture modulaire des applications Angular ayant une version antérieure à 17.

- La configuration de l'ensemble du projet est gérée par le fichier `app.config.ts`
- Chaque composant gère lui-même ses dépendances.

2.4.2.3 Composants et directives

Utilisation extensive de la librairie PrimeNg pour les composants riches (Tableaux, Modales, Calendriers, champs des formulaires) et de bootstrap pour l'utilisation de certaines icônes.

2.4.2.4 Gestion de l'État

Gestion de l'état principalement locale aux composants ou via des services (RxJS ou signaux) pour les données partagées.

2.4.2.5 Communication avec le Backend

Communication via HttpClient d'Angular. Les appels API ciblent les contrôleurs Spring Boot.

2.4.2.6 Performances

- Lazy Loading des composants pour réduire le bundle initial.
- Build de production optimisé (AOT, Minification).

3 DÉPLOIEMENT ET INFRASTRUCTURE

3.1 Conteneurisation

Utilisation du plugin Maven Jib (`com.google.cloud.tools:jib-maven-plugin`) pour construire des images Docker optimisées sans nécessité de Docker Engine local pour le build.

- Image de base : `tomcat:10.1.34-jre17`
- L'application est déployée sous forme de WAR dans le conteneur Tomcat.

3.2 Orchestration

Le déploiement est géré via des scripts d'intégration CI/CD (visible via `.gitlab-ci.yml`), orchestrant la construction des images et leur déploiement sur les environnements cibles.

3.3 Configuration

La configuration est externalisée :

- Fichier `application.properties`
- Variables d'environnement pour les chemins sensibles (répertoire de dépôt XML, clés de chiffrement).

4 LES SERVICES

4.1 Service keygen

Il s'agit d'un service (.jar) en Java 11 de génération de clés de chiffrement RSA (2048) et permet de générer les clés privée et publique utilisées pour le chiffrement et le déchiffrement des fichiers de données demandeurs.

- La clé publique est utilisée par le site grand public pour le chiffrement des données.
- La clé privée est utilisée par le service java exécuté sur l'environnement de l'application POPP.

4.2 Service import

Le processus de transfert des demandes depuis le serveur du RNR vers le serveur de POPP est réalisé par GoAnywhere (interne ABM).

Le processus d'import des demandes dans l'application POPP est réalisé par le service (.jar) d'import en Java 11.

Celui-ci doit est installé sur le serveur applicatif de POPP (ABM).

