INSCRIPTION SERVICE

Documentation Technique

Portail de Suivi du Doctorat

Architecture Microservices

Information	Valeur
Microservice	inscription-service
Port	8082
Base de données	MySQL - inscription_db
Framework	Spring Boot 3.x
Date	01/01/2026

TABLE DES MATIÈRES

1. PRÉSENTATION GÉNÉRALE DU MICROSERVICE

1.1 Rôle Fonctionnel

Le **inscription-service** est le microservice responsable de la gestion complète du cycle de vie des inscriptions et réinscriptions des doctorants au sein du portail doctoral. Il gère également le système de dérogations pour les doctorants dépassant la durée normale de 3 ans de thèse.

1.2 Responsabilités Principales

- **Gestion des Campagnes d'Inscription** : Création, activation et suivi des périodes d'inscription universitaires
- **Premières Inscriptions** : Traitement des nouvelles demandes d'inscription au doctorat
- **Réinscriptions** : Gestion du renouvellement annuel des inscriptions
- **Système de Dérogations** : Workflow complet pour les prolongations au-delà de 3 ans (4ème, 5ème, 6ème année)
- **Vérification d'Éligibilité** : Contrôle automatique des prérequis pour la réinscription
- **Publication d'Événements** : Notification des autres services via Kafka lors des changements d'état

1.3 Cas d'Usage Couverts

Acteur	Cas d'Usage
Candidat	Soumettre une première inscription
Doctorant	Effectuer une réinscription annuelle
Doctorant	Demander une dérogation de prolongation
Directeur	Valider/Rejeter une réinscription
Directeur	Valider/Rejeter une demande de dérogation
Admin	Gérer les campagnes d'inscription
Admin	Traiter les premières inscriptions
Admin	Approuver/Refuser les dérogations

1.4 Communication avec les Autres Microservices

Service	Type	Description

user-service	REST (Feign)	Récupération des informations utilisateur (doctorant, directeur)
notification-service	Kafka (Async)	Envoi d'événements pour déclencher les emails

2. ARCHITECTURE INTERNE

2.1 Organisation du Code

Le microservice suit une architecture en couches classique Spring Boot, avec une séparation claire des responsabilités :

Couche	Package	Rôle
Controllers	controllers	Exposition des endpoints REST
Services	services	Logique métier et orchestration
Repositories	repositories	Accès aux données (JPA)
Entities	entities	Modèles de données JPA
DTOs	dto	Objets de transfert de données
Events	events	Événements Kafka
Clients	clients	Clients Feign pour appels inter-services
Config	config	Configuration Spring
Enums	enums	Énumérations métier

2.2 Flux Métier Principal - Inscription

Le workflow d'inscription suit un processus de validation en plusieurs étapes :

```
Première Inscription :
BROUILLON → EN_ATTENTE_ADMIN → ADMIS (ou REJETE_ADMIN)

Réinscription :
BROUILLON → EN_ATTENTE_DIRECTEUR → EN_ATTENTE_ADMIN → ADMIS (ou REJETE)
```

2.3 Flux Métier - Dérogation

Le système de dérogation suit un workflow à deux niveaux de validation :

```
EN_ATTENTE_DIRECTEUR → EN_ATTENTE_ADMIN → APPROUVEE (ou REFUSEE)
```

2.4 Règles Métier - Durée du Doctorat

Règle	Valeur	Description

Durée normale	3 ans	Période standard sans dérogation requise
Durée maximale	6 ans	Limite absolue, aucune prolongation possible après
4ème année	Dérogation	Première demande de prolongation PED
5ème année	Dérogation	Deuxième demande de prolongation
6ème année	Dérogation	Dernière année possible (ultime chance)

3. FICHIERS DE CONFIGURATION

3.1 InscriptionServiceApplication.java

Rôle : Point d'entrée principal du microservice Spring Boot.

Annotations utilisées :

- **@SpringBootApplication** : Active l'auto-configuration Spring Boot, le scan de composants et la configuration
- **@EnableFeignClients** : Active les clients Feign pour la communication REST inter-services

3.2 application.properties

Rôle : Fichier de configuration centralisé du microservice.

Paramètre	Valeur	Description
server.port	8082	Port d'écoute du service
spring.datasource.url	jdbc:mysql://localhost:3306/inscription_db	URL de connexion MySQL
spring.jpa.hibernate.ddl-auto	update	Mise à jour automatique du schéma
eureka.client.service-url.defaultZone	http://localhost:8761/eureka/	URL du serveur Eureka
spring.kafka.bootstrap-servers	localhost:9092	Adresse du broker Kafka
spring.kafka.producer.acks	all	Garantie de livraison des messages
feign.circuitbreaker.enabled	true	Active le circuit breaker pour Feign

3.3 SecurityConfig.java

Rôle : Configuration de la sécurité Spring Security.

Fonctionnalités :

- Désactivation de CSRF (API REST stateless)
- Configuration CORS pour autoriser Angular (localhost:4200)
- Autorisation de toutes les requêtes (mode développement)
- Support des méthodes GET, POST, PUT, DELETE, OPTIONS

3.4 KafkaConfig.java

Rôle : Configuration du producteur Kafka pour la publication d'événements.

Beans créés :

- **ProducerFactory** : Factory pour créer des producteurs Kafka avec sérialisation JSON

- **KafkaTemplate** : Template pour envoyer des messages vers les topics Kafka

3.5 KafkaTopics.java

Rôle : Classe utilitaire définissant les noms des topics Kafka utilisés.

Constante	Valeur	Usage
INSCRIPTION_CREATED	inscription-created	Nouvelle inscription créée
INSCRIPTION_STATUS_CHANGED	Inscription-status-changed	Changement de statut
INSCRIPTION_SUBMITTED	inscription-submitted	Inscription soumise
REINSCRIPTION_REMINDER	reinscription-reminder	Rappel de réinscription

4. MODÈLE DE DONNÉES (ENUMS & ENTITIES)

4.1 Énumérations (Enums)

4.1.1 StatutInscription.java

Rôle : Définit les états possibles d'une inscription dans le workflow.

Valeur	Description
BROUILLON	Inscription créée mais pas encore soumise
EN_ATTENTE_ADMIN	Soumise, en attente de validation par l'administrateur
EN_ATTENTE_DIRECTEUR	Validée par admin, en attente du directeur (réinscription)
ADMIS	Validée par toutes les parties, le candidat devient doctorant
REJETE_ADMIN	Refusée par l'administrateur
REJETE_DIRECTEUR	Refusée par le directeur de thèse

4.1.2 StatutDerogation.java

Rôle : Définit les états du workflow de dérogation à deux niveaux.

Valeur	Description
EN_ATTENTE_DIRECTEUR	Étape 1 : En attente de validation par le directeur
EN_ATTENTE_ADMIN	Étape 2 : Validée par directeur, en attente admin
EN_ATTENTE	Legacy : Pour compatibilité ascendante
APPROUVEE	Dérogation accordée
REFUSEE	Dérogation refusée (par directeur ou admin)
EXPIREE	Dérogation expirée (date dépassée)
ANNULEE	Annulée par le doctorant

4.1.3 TypeDerogation.java

Rôle : Énumération enrichie définissant les types de dérogation avec métadonnées.

Attributs : libelle (String), anneeAutorisee (int)

Méthode utilitaire : pourAnnee(int) - Retourne le type de dérogation pour une année donnée

Valeur	Libellé	Année
PROLONGATION_4EME_ANNEE	Prolongation 4ème année	4
PROLONGATION_5EME_ANNEE	Prolongation 5ème année	5
PROLONGATION_6EME_ANNEE	Prolongation 6ème année (dernière)	6
SUSPENSION_TEMPORAIRE	Suspension temporaire	0
AUTRE	Autre motif	0

4.1.4 TypeInscription.java & TypeDocument.java

Enum	Valeurs	Description
TypeInscription	PREMIERE_INSCRIPTION, REINSCRIPTION	Distingue 1ère année des suivantes
TypeDocument	DIPLOME, CV, LETTRE_MOTIVATION, ATTESTATION_TRAVAIL, AUTRE	Types de pièces jointes

4.2 Entités JPA

4.2.1 Inscription.java

Rôle : Entité principale représentant une demande d'inscription au doctorat.

Table : inscriptions

Champ	Type	Description
id	Long (PK)	Identifiant unique auto-généré
doctorantId	Long	Référence vers le candidat/doctorant (user-service)
directeurId	Long	Référence vers le directeur de thèse
campagne	Campagne (FK)	Campagne d'inscription associée
typeInscription	TypeInscription	PREMIERE_INSCRIPTION ou REINSCRIPTION
statut	StatutInscription	État actuel dans le workflow
sujetThese	String	Sujet de la thèse proposé
laboratoireAccueil	String	Laboratoire de rattachement
documents	List<Document>	Pièces jointes (CV, diplômes...)
dateSoumission	LocalDateTime	Date de soumission de la demande
commentaireDirecteur	String	Remarques du directeur
commentaireAdmin	String	Remarques de l'administrateur

4.2.2 Derogation.java

Rôle : Entité représentant une demande de dérogation pour prolongation du doctorat.

Table : derogations

Champ	Type	Description
id	Long (PK)	Identifiant unique
doctorantId	Long	Doctorant demandeur
directeurId	Long	Directeur de thèse assigné
typeDerogation	TypeDerogation	Type de prolongation demandée

statut	StatutDerogation	État dans le workflow à 2 niveaux
motif	String	Justification de la demande
anneeDemandee	Integer	Année de doctorat demandée (4, 5 ou 6)
dateValidationDirecteur	LocalDateTime	Date de validation par le directeur
commentaireDirecteur	String	Avis du directeur
dateDecision	LocalDateTime	Date de décision finale
dateExpiration	LocalDate	Date d'expiration de la dérogation

Méthodes utilitaires : estValide(), estEnAttente(), estTraitee(), getEtapeWorkflow()

4.2.3 Campagne.java

Rôle : Entité représentant une période d'inscription universitaire.

Table : campagnes

Champ	Type	Description
id	Long (PK)	Identifiant unique
titre	String	Nom de la campagne
anneeUniversitaire	String	Ex: "2024-2025"
dateOuverture	LocalDate	Début de la période d'inscription
dateFermeture	LocalDate	Fin de la période d'inscription
active	Boolean	Indique si c'est la campagne courante

Méthode : isOuverte() - Vérifie si la campagne est actuellement ouverte

4.2.4 Document.java

Rôle : Entité représentant un document attaché à une inscription.

Table : documents

Champs principaux : id, inscription (FK), documentServiceId (référence externe), typeDocument, nomFichier, uploadedAt

5. COUCHE REPOSITORY

Les repositories étendent JpaRepository et fournissent l'accès aux données via Spring Data JPA.

5.1 InscriptionRepository.java

Rôle : Accès aux données des inscriptions.

Méthode	Description
findByDoctorantId()	Inscriptions d'un doctorant
findByDirecteurId()	Inscriptions supervisées par un directeur
findByStatut()	Filtrage par statut
findByCampagneId()	Inscriptions d'une campagne
findByDirecteurIdAndStatut()	Inscriptions d'un directeur avec un statut donné
findByTypeInscriptionAndStatut()	Filtrage combiné type + statut

5.2 DerogationRepository.java

Rôle : Accès aux données des dérogations avec requêtes métier.

Méthode	Description
findByDoctorantIdOrderByDateDemandeDesc()	Dérogations d'un doctorant (triées)
findByDirecteurIdAndStatut()	Dérogations pour un directeur avec statut
findByStatutOrderByDateDemandeAsc()	Dérogations par statut (FIFO)
hasDerogationValideByStatut() [JPQL]	Vérifie existence dérogation valide
hasDerogationEnCoursByStatuts() [JPQL]	Vérifie si demande en cours existe

5.3 CampagneRepository.java & DocumentRepository.java

Repository	Méthodes Principales
CampagneRepository	findByActiveTrue(), findByAnneeUniversitaire()
DocumentRepository	findByInscriptionId()

6. OBJETS DE TRANSFERT (DTOs)

6.1 UserDTO.java

Rôle : Représentation d'un utilisateur provenant du user-service.

Champs principaux : id, username (matricule), email, nom, prenom, role, directeurId, titreThese, anneeThese, nbPublications, nbConferences, heuresFormation

6.2 DemandeDerogationDTO.java

Rôle : DTO d'entrée pour créer une nouvelle demande de dérogation.

Champ	Validation	Description
doctorantId	@NotNull	ID du doctorant demandeur
directeurId	Optionnel	ID du directeur (auto-récupéré si absent)
typeDerogation	@NotNull	Type de prolongation demandée
motif	@NotBlank	Justification obligatoire

6.3 DecisionDerogationDTO.java

Rôle : DTO pour enregistrer une décision sur une dérogation.

Champs : derogationId (@NotNull), approuver (@NotNull Boolean), decideurId (@NotNull), commentaire (optionnel)

6.4 EligibiliteReinscriptionDTO.java

Rôle : DTO de sortie pour le résultat de la vérification d'éligibilité à la réinscription.

Champ	Type	Description
eligible	boolean	Le doctorant peut-il se réinscrire ?
anneeActuelle	int	Année de doctorat en cours
prochaineAnnee	int	Année si réinscription
derogationRequise	boolean	Une dérogation est-elle nécessaire ?
derogationObtenue	boolean	La dérogation a-t-elle été accordée ?
typeDerogationRequise	TypeDerogation	Type de dérogation à demander

message	String	Message explicatif pour l'utilisateur
anneesRestantes	int	Années avant la limite de 6 ans
enPeriodeAlerte	boolean	Alerte si 5ème ou 6ème année

7. ÉVÉNEMENTS KAFKA

Le microservice utilise Apache Kafka pour publier des événements consommés par le notification-service.

7.1 BaseEvent.java

Rôle : Classe abstraite parente de tous les événements.

Champs communs : eventId (UUID), timestamp (LocalDateTime), source ("inscription-service")

Méthode : initializeEvent() - Initialise les champs avec valeurs par défaut

7.2 InscriptionCreatedEvent.java

Rôle : Événement publié lors de la création d'une nouvelle inscription.

Topic Kafka : inscription-created

Données : inscriptionId, doctorantId, doctorantEmail, doctorantNom, doctorantPrenom, sujetThese, directeurTheseEmail, campagneId, status

7.3 InscriptionStatusChangedEvent.java

Rôle : Événement publié lors d'un changement de statut d'une inscription.

Topic Kafka : inscription-status-changed

Données : inscriptionId, doctorantId, doctorantEmail, oldStatus, newStatus, sujetThese, commentaire, validatedBy

8. COMMUNICATION INTER-SERVICES (FEIGN)

8.1 UserServiceClient.java

Rôle : Interface Feign pour communiquer avec le user-service.

Annotation : @FeignClient(name = "USER-SERVICE", fallback = UserServiceClientFallback.class)

Méthode	Endpoint	Description
getUserById(Long id)	GET /api/users/{id}	Récupère un utilisateur par ID
getDoctorantsByDirecteur(Long id)	GET /api/users/directeur/{id}/doctorants	Liste des doctorants d'un directeur

8.2 UserServiceClientFallback.java

Rôle : Classe de fallback appelée en cas d'échec de communication avec user-service.

Comportement :

- getUserById() : Retourne un UserDTO avec valeurs par défaut ("Utilisateur Inconnu")
- getDoctorantsByDirecteur() : Retourne une liste vide

Avantage : Le service reste fonctionnel même si user-service est indisponible (résilience).

9. COUCHE SERVICE (LOGIQUE MÉTIER)

9.1 CampagneService / CampagneServiceImpl

Rôle : Gestion du cycle de vie des campagnes d'inscription.

Méthode	Description
createCampagne()	Crée une nouvelle campagne (désactive les autres si active)
updateCampagne()	Met à jour une campagne existante
deleteCampagne()	Supprime une campagne
getCampagneActive()	Retourne la campagne actuellement active
activerCampagne()	Active une campagne (désactive les autres)

9.2 InscriptionService / InscriptionServiceImpl

Rôle : Orchestration du workflow d'inscription multi-étapes.

Méthode	Transition	Description
create()	BROUILLON	Crée une nouvelle inscription
soumettre()	BROUILLON → EN_ATTENTE_*	Soumet l'inscription pour validation
validerParDirecteur()	EN_ATTENTE_DIR → EN_ATTENTE_ADMIN	Validation directeur
rejeterParDirecteur()	EN_ATTENTE_DIR → REJETE_DIR	Rejet par directeur
validerParAdmin()	EN_ATTENTE_ADMIN → ADMIS	Validation finale admin
rejeterParAdmin()	EN_ATTENTE_ADMIN → REJETE_ADMIN	Rejet par admin

Workflow conditionnel : La méthode soumettre() envoie vers EN_ATTENTE_ADMIN pour les premières inscriptions, et vers EN_ATTENTE_DIRECTEUR pour les réinscriptions.

9.3 DerogationService / DerogationServiceImpl

Rôle : Gestion complète du workflow de dérogation à deux niveaux de validation.

Méthode	Acteur	Description
demanderDerogation()	Doctorant	Crée une demande (statut: EN_ATTENTE_DIRECTEUR)

validerParDirecteur()	Directeur	Valide et passe à EN_ATTENTE_ADMIN
refuserParDirecteur()	Directeur	Refuse la demande
approuverDerogation()	Admin	Approuve la dérogation (APPROUVEE)
refuserDerogation()	Admin	Refuse la dérogation (REFUSEE)
enrichirDerogation()	Interne	Ajoute les infos utilisateur via Feign

9.4 DoctoratDureeService

Rôle : Service métier implémentant les règles de durée du doctorat selon le cahier des charges.

Méthode	Description
calculerAnneeDoctorat()	Calcule l'année actuelle basée sur la date de 1ère inscription
verifierEligibiliteReinscription()	Retourne un EligibiliteReinscriptionDTO complet
peutSeReinscrire()	Booléen simple d'éligibilité
anneesRestantes()	Nombre d'années avant la limite de 6 ans
estEnPeriodeAlerte()	True si en 5ème ou 6ème année
getMessageAlerte()	Message d'alerte contextuel pour l'utilisateur

9.5 InscriptionEventPublisher

Rôle : Publication des événements vers Kafka pour notification asynchrone.

Méthode	Topic	Déclencheur
publishInscriptionCreated()	inscription-created	Nouvelle inscription
publishInscriptionStatusChanged()	inscription-status-changed	Changement de statut
publishInscriptionSubmitted()	inscription-submitted	Soumission

10. COUCHE CONTROLLER (API REST)

10.1 CampagneController.java

Base URL : /api/campagnes

Méthode	Endpoint	Description
POST	/	Créer une campagne
GET	/	Lister toutes les campagnes
GET	/{id}	Obtenir une campagne par ID
GET	/active	Obtenir la campagne active
PUT	/{id}	Mettre à jour une campagne
PUT	/{id}/activer	Activer une campagne
DELETE	/{id}	Supprimer une campagne

10.2 InscriptionController.java

Base URL : /api/inscriptions

Méthode	Endpoint	Description
POST	/	Créer une inscription
GET	/	Lister toutes les inscriptions
GET	/{id}	Obtenir une inscription par ID
GET	/doctorant/{id}	Inscriptions d'un doctorant
GET	/directeur/{id}	Inscriptions d'un directeur
GET	/statut/{statut}	Filtrer par statut
PUT	/{id}/soumettre	Soumettre une inscription
PUT	/{id}/valider-directeur	Validation par directeur
PUT	/{id}/rejeter-directeur	Rejet par directeur
PUT	/{id}/valider-admin	Validation par admin

PUT	/{id}/rejeter-admin	Rejet par admin
GET	/directeur/{id}/reinscriptions-en-attente	Réinscriptions en attente directeur
GET	/admin/premieres-inscriptions-en-attente	Premières inscriptions en attente

10.3 DerogationController.java

Base URL : /api/derogations

Méthode	Endpoint	Description
GET	/eligibilite/{doctorantId}	Vérifier éligibilité réinscription
GET	/annee/{doctorantId}	Obtenir l'année de doctorat
POST	/	Créer une demande de dérogation
GET	/doctorant/{id}	Dérogations d'un doctorant
GET	/directeur/{id}	Dérogations pour un directeur
PUT	/{id}/valider-directeur	Validation par directeur
PUT	/{id}/refuser-directeur	Refus par directeur
GET	/en-attente-admin	Dérogations en attente admin
PUT	/{id}/approuver	Approbation par admin
PUT	/{id}/refuser	Refus par admin
GET	/stats	Statistiques des dérogations

11. QUESTIONS / RÉPONSES POTENTIELLES

Q1 : Pourquoi avoir séparé les premières inscriptions et les réinscriptions dans le workflow ?

R : Les premières inscriptions nécessitent une validation admin pour l'assignation du directeur, tandis que les réinscriptions requièrent d'abord l'avis du directeur qui connaît l'avancement du doctorant. Cela reflète la réalité organisationnelle du processus doctoral.

Q2 : Comment le système gère-t-il la règle des 6 ans maximum ?

R : Le DoctoratDureeService calcule l'année actuelle du doctorant et bloque toute réinscription au-delà de 6 ans (DUREE_MAXIMALE_ANNEES = 6). Pour les années 4, 5 et 6, une dérogation approuvée est obligatoire.

Q3 : Que se passe-t-il si le user-service est indisponible ?

R : Grâce au pattern Circuit Breaker et à la classe UserServiceClientFallback, le service continue de fonctionner avec des valeurs par défaut. Les dérogations et inscriptions peuvent être créées, mais les informations utilisateur seront marquées comme 'Inconnu'.

Q4 : Pourquoi utiliser Kafka plutôt que des appels REST synchrones pour les notifications ?

R : Kafka permet un découplage complet entre inscription-service et notification-service. Les événements sont persistés et peuvent être rejoués. Cela améliore la résilience et permet au service de notification de traiter les messages à son rythme.

Q5 : Comment est garantie l'unicité de la campagne active ?

R : La méthode activerCampagne() et createCampagne() appellent desactiverToutesCampagnes() avant d'activer la nouvelle campagne. Ainsi, une seule campagne peut être active à la fois.

Q6 : Pourquoi le workflow de dérogation a-t-il deux niveaux de validation ?

R : Le directeur de thèse valide d'abord car il connaît le contexte et l'avancement du doctorant. L'admin valide ensuite pour s'assurer de la conformité administrative. Cela garantit une décision éclairée et conforme au règlement.

12. CONCLUSION

Le **inscription-service** est un microservice central du portail doctoral, responsable de la gestion complète du cycle de vie des inscriptions et du système de dérogations.

Points Forts de l'Architecture :

- **Workflow Multi-Étapes** : Gestion différenciée des premières inscriptions et réinscriptions avec validation à plusieurs niveaux
- **Règles Métier Encapsulées** : DoctoratDureeService centralise toutes les règles de durée du doctorat
- **Communication Asynchrone** : Utilisation de Kafka pour les notifications, garantissant découplage et résilience
- **Résilience** : Fallback Feign pour continuer à fonctionner même si user-service est indisponible
- **API REST Complète** : Endpoints bien structurés pour chaque acteur (doctorant, directeur, admin)
- **Conformité au Cahier des Charges** : Implémentation fidèle des règles de durée normale (3 ans) et maximale (6 ans)

Valeur dans l'Architecture Globale :

Ce microservice constitue le point d'entrée administratif pour les doctorants. Il s'intègre parfaitement dans l'écosystème via Eureka (découverte de services), communique avec user-service (Feign) pour les données utilisateur, et publie des événements vers notification-service (Kafka) pour informer les parties prenantes des évolutions de leur dossier.