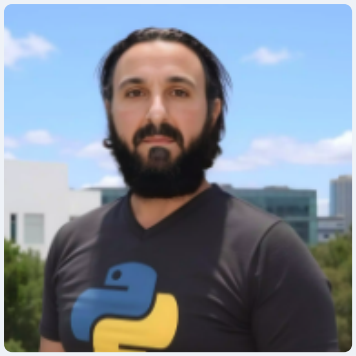


CONTACT

France - Remote

rachid.jeffali@gmail.com

+33 7 56 88 61 64

LinkedIn · github.com/jrachid

Badges Credly

CERTIFICATIONS

Langage Python
Éditions ENI · RS6701 France Compétences · février 2026

Tosa Python 3 — 913/1000, niveau Expert
Isograd · novembre 2024

PCAP – Certified Associate in Python Programming
Python Institute · PCAP-31-03 · août 2023

PCEP – Certified Entry-Level Python Programmer
Python Institute · PCEP-30-02 · juin 2023

AWS Certified Developer – Associate
DVA-C02 · en préparation

COMPÉTENCES TECHNIQUES

Architecture
DDD, Architecture Hexagonale, Clean Architecture, monolithe modulaire, conception d'API

Backend
TypeScript/Node.js, Python, Kotlin/JVM, Go, FastAPI, Django, Spring Boot

Serverless
Firebase Cloud Functions v2, Firestore, Firebase Auth & Hosting, Google Cloud Secret Manager

RACHID JEFFALI

Backend Architect • Architecture logicielle • SaaS

Conception backend • DDD • Performance & Scalabilité • Engineering Quality • IA Engineering

À PROPOS

Backend Architect avec plus de 8 ans d'expérience dans la conception et le développement de **plateformes SaaS et d'architectures backend**.

Expertise en **DDD, architecture hexagonale, conception d'API, traitement asynchrone, concurrence, PostgreSQL, tests automatisés et CI/CD**, avec une approche centrée sur la **simplicité architecturale, la maintenabilité, la performance et la fiabilité en production**.

Montée en charge de **5 à plus de 1 000 opérations simultanées**, frontières applicatives vérifiées automatiquement, industrialisation de systèmes backend en **Python, TypeScript/Node.js et Kotlin/JVM**. Auteur de bibliothèques open source publiées sur PyPI, et de workflows d'ingénierie augmentés par **LLM** encadrés par des garde-fous exécutables.

IA ENGINEERING / AI-AUGMENTED DEVELOPMENT

- **Harnais d'agents de code versionné dans le dépôt** (Claude Code, Codex, Gemini) : 17 règles d'architecture et de qualité, 12 skills, 4 agents spécialisés (diagnostic sur données de production, triage des tests instables, sondage du contrat d'une API tierce, exécution des tests) et 9 hooks qui rendent le processus **exécutable plutôt que confié à la mémoire de l'agent**.
- **Discipline « test rouge d'abord » imposée aux agents** : le test qui échoue précède le code, les scénarios validés deviennent intouchables, et chaque tranche passe une **revue adverse multi-modèles** avant fusion.
- **Garde-fous automatisés sur le code généré** : tests d'architecture à cliquet (seuils qui ne peuvent que baisser), blocage des requêtes non bornées, journal des tests instables tiré de l'API de la CI (582 jobs analysés), audit de 167 pipelines.
- **Contexte mesuré, pas supposé** : six façons de donner le schéma de la base aux agents comparées (pg_dump 40 s, export Hibernate 1,4 s puis 0,08 s en cache, exploration libre 155 s et non déterministe) ; graphe de code (MCP) pour l'exploration des dépôts.
- **Le LLM comme brique produit** : jugements typés et probabilisés plutôt que texte libre (cgu-checklist), pipelines planifiés d'agents, skills publiés en plugin Claude Code (documentation dont chaque affirmation est vérifiée dans le code, SEO branché sur Google Search Console).

EXPÉRIENCE PROFESSIONNELLE

Full-Stack Engineer & Architecte Serverless | ASSUR POWER (Freelance)

Septembre – Octobre 2026
Courtage en assurance construction B2B | Remote

- Conception et livraison de **5 plateformes d'acquisition de leads** (décennale photovoltaïque, entreprise étrangère, décennale résiliée, constructeur de maisons, entreprise en redressement) : site public **Next.js 16 / TypeScript** en export statique, back-office et API serverless.

Data & Messaging
PostgreSQL, PostGIS, SQL, SQLAlchemy, Hibernate/JPA, Apache Kafka, pandas, traitement asynchrone

Performance & Fiabilité
concurrence, Java Virtual Threads, verrous optimistes et distribués, observabilité, disjoncteurs

Tests & Qualité
TDD, BDD, tests d'architecture, tests de mutation, Pytest, JUnit, Playwright, Testcontainers, ArchUnit

Frontend
Next.js, React, HTMX, Tailwind CSS, ECharts, MapLibre

DevOps & Cloud
Docker, GitLab CI/CD, GitHub Actions, Linux, Cloudflare, AWS (EC2, Lambda, S3, RDS, IAM, API Gateway)

IA
Claude Code, Codex, Gemini, MCP, agents, skills, hooks, scikit-learn, spaCy

LANGUES

Français : langue maternelle

Anglais : professionnel, à l'aise à l'écrit comme à l'oral

FORMATION

Licence Sciences Physiques
Université de Lille

DUT Mesures Physiques
Université de Lille

- Back-end **Node.js 22 sur Cloud Functions v2** (europe-west1) branché derrière Firebase Hosting : réception des demandes, concurrence et nombre d'instances bornés, secrets dans **Google Secret Manager**, paramètres de déploiement typés.
- **API REST d'administration authentifiée** : jeton Firebase ID vérifié côté serveur, e-mail vérifié et liste blanche, workflow de statuts tracé (qui, quand), notes horodatées, filtres et export CSV.
- Sécurité en profondeur : **Firestore fermé au navigateur** (règles en refus total, accès par le SDK d'administration seulement), CORS par liste d'origines, middleware anti-abus (JSON seul, 64 Ko max, **limitation de débit par IP** avec Retry-After), champ-piège anti-robot, validation et échappement côté serveur.
- **Zéro perte de lead** : la demande est persistée avant la notification e-mail, un échec du fournisseur est tracé sur le document au lieu de faire échouer l'envoi ; préremplissage des données légales par le SIREN via l'API publique recherche-entreprises.
- Tests unitaires et **de bout en bout sur l'émulateur Firebase** (Auth, Firestore, Functions) avec le test runner natif de Node, et guide de déploiement pour remettre chaque site sur le compte du client.

Stack : TypeScript, Node.js 22, Next.js 16, React 19, Tailwind CSS 4, Firebase (Hosting, Auth, Firestore, Cloud Functions v2), Google Cloud Secret Manager, Resend, node:test

Backend Architect & Full-Stack Engineer | XPONE

Août 2025 – Aujourd'hui

Startup SaaS B2B — plateforme d'automatisation de la prospection multicanale | Remote

- Conception de l'**architecture backend en monolithe modulaire** structurée selon le **DDD** ; frontières entre modules vérifiées par **ArchUnit** et par des tests à cliquet, sans grand chantier de refonte.
- Refonte du modèle de concurrence de l'enrichissement : transaction coupée en trois et **Java Virtual Threads**, connexion PostgreSQL tenue **3–5 s → ~10 ms**, capacité portée de **5 à plus de 1 000 opérations simultanées**.
- Moteur de campagnes : envoi côté serveur via une API de messagerie tierce, quota d'envoi partagé par **tourniquet pondéré** (fonction pure, 87 % de score de mutation), quotas et heures calmes anti-bannissement, détection des réponses par prospect.
- Garde-fous anti-rafale en défense en profondeur côté base : **verrou optimiste, contraintes UNIQUE, verrou distribué par job, plafond par destinataire** ; audit complet de la concurrence et des transactions.
- Fiabilité en production : résolution d'un OOM (requête de 5,7 min sans LIMIT → **statement_timeout, pagination, test qui interdit les listes non bornées**), dimensionnement mémoire de la JVM en conteneur, disjoncteurs sur les API externes.
- Sécurité avant lancement : WebSocket/STOMP authentifié par **JWT au CONNECT** au lieu d'un en-tête de confiance, contrôle de propriété, crédits d'enrichissement réservés, capturés et libérés hors transaction.
- Stratégie de qualité : **~6 000 tests backend et 1 000+ tests front**, tests d'acceptation multi-drivers sur un **DSL métier** (un scénario, trois niveaux d'exécution), intégrés au pipeline GitLab CI/CD.
- Livraison continue mesurée selon les **indicateurs DORA** (90 jours) : **182 déploiements, soit ~14 par semaine**, délai médian du commit à la production de **58 min**, taux d'échec des changements de 15 %, rétablissement médian en 3,5 h.

- Industrialisation du développement augmenté par **LLM** : harnais d'agents, revues multi-modèles et garde-fous automatisés (voir section IA Engineering).

Stack : Kotlin, Spring Boot, Hibernate/JPA, PostgreSQL, HTMX, Pebble, WebSocket/STOMP, Docker, GitLab CI/CD, Playwright, Testcontainers, ArchUnit

Créateur & Architecte Full-Stack | DataHexa — donnees-france.fr

Janvier 2026 – Aujourd'hui

Projet personnel en production — données publiques françaises par commune

- Architecture d'une plateforme transformant des **sources de données publiques hétérogènes** (INSEE, DVF, OFGL, RNE…) en **indicateurs territoriaux comparables** : pipelines d'ingestion Python, API FastAPI sur **PostgreSQL/PostGIS**, application Next.js.
- Méthodologie statistique défendable : **score sécurité robuste** (log, winsorisation, z-score par voisinage, shrinkage bayésien des petites communes), **score budgétaire explicatif**, prix au m² DVF corrigés des ventes multi-lots (~33 % des lignes, 17 625 → 1 975 €/m²).
- Front-end **TypeScript / Next.js (App Router) / React 19** d'environ 18 000 lignes, typage strict vérifié en CI : pages communes et thématiques, cartes MapLibre, graphiques ECharts, sitemap dynamique, images de partage et données structurées générées côté serveur.
- Génération de **35 000+ fiches communes** en rendu incrémental à la demande puis préchauffées, avec route de revalidation ; **gestionnaire de cache Node.js écrit sur mesure** (LRU + TTL sur volume borné à 2 Gio, isolé par build) après une saturation disque en production.
- **Migration de domaine et de marque sans perte de référencement** : redirections 301, ancien sitemap servi, données structurées, gel de contenu et répétition générale ; sitemap de ~33 700 URL.
- Exploitation : Docker sur VPS derrière **Cloudflare**, origine fermée au trafic qui contourne le CDN, politique d'accès des robots IA, suivi Search Console automatisé.

Stack : TypeScript, Node.js, Next.js, React 19, Tailwind CSS, ECharts, MapLibre, Python, FastAPI, pandas, PostgreSQL/PostGIS, Docker, Cloudflare

Backend Engineer | Wassati

Septembre 2019 – Décembre 2025

Lille, France

- Conception du backend d'une plateforme **SaaS d'analyse automatisée de contenus**, couvrant la collecte, le preprocessing, le scoring et l'exposition des résultats.
- Structuration des traitements afin de **séparer les étapes de collecte et d'analyse** et permettre leur exécution asynchrone.
- Mise en place d'**Apache Kafka** pour découpler les traitements et distribuer les charges de collecte et d'analyse.
- Conception d'API backend permettant l'exploitation des résultats produits par les pipelines de scoring.
- Développement et maintien en production de pipelines d'analyse multilingues **français, anglais et espagnol**.

Stack : Python, FastAPI, Apache Kafka, scikit-learn, spaCy, Pandas

Formateur Développement Logiciel | Organismes de formation & entreprises

2021 – 2025

France & Remote

- Conception et animation de formations en **développement backend, architecture logicielle, qualité du code et tests automatisés** auprès de développeurs et d'équipes techniques.
- Formation et accompagnement pour différents organismes et grands comptes, notamment **Orange et CANAL+**.
- Accompagnement des développeurs dans la **structuration d'applications**, la séparation des responsabilités et l'application de pratiques favorisant maintenabilité et testabilité.
- Conception de projets et exercices techniques reproduisant des problématiques d'ingénierie proches de situations professionnelles réelles.
- Intervention sur plusieurs écosystèmes permettant d'aborder les principes d'architecture indépendamment du langage ou du framework utilisé.

Technologies enseignées : Python, TypeScript, Node.js, React, architecture logicielle, tests automatisés

OPEN SOURCE

seedgraph — bibliothèque Python publiée sur PyPI · github.com/jrachid/seedgraph

Remplit une base de test avec un **graphe d'objets SQLAlchemy cohérent en un seul appel** (« 3 utilisateurs, 2 articles chacun, 3 commentaires par article »), puis vérifie chaque clé étrangère contre la ligne visée. Valeurs réalistes et reproductibles, unicité respectée même face aux lignes existantes, intégration pytest sans configuration, mode async. Chaque garantie est couverte par un test nommé sur SQLite et PostgreSQL ; documentation en ligne et plugin Claude Code qui apprend la bibliothèque aux agents.

tabcustody — bibliothèque et CLI Python publiées sur PyPI · github.com/jrachid/tabcustody

Avec les modèles tabulaires en contexte (TabPFN, TabICL, TabDPT), **le fichier du modèle contient les données d'entraînement**. tabcustody le prouve sans exécuter le fichier : analyse des pickle, joblib et archives, code de sortie utilisable comme **porte de CI**, rapport JSON. Mécanisme mesuré avant d'écrire la bibliothèque : 300 lignes sur 300 retrouvées dans chacun des quatre modèles testés, 0 sur 300 pour XGBoost.

cgu-checklist · GitHub

Extension navigateur et CLI qui lisent des conditions générales et en tirent une check-list, **chaque point justifié par la clause exacte**, via un modèle qui rend des jugements typés et probabilisés.

grab · GitHub

Gestionnaire de téléchargements en **Go sans dépendance** : morceaux téléchargés en parallèle, pause et reprise exacte après coupure, flux vidéo HLS, extension Chrome.

claude-skills · GitHub

Plugin Claude Code installable : skill de documentation de base de code dont **chaque affirmation est vérifiée dans le code**, avec ses règles d'écriture.