PROGRAMME DE FORMATION

RAG EN PRODUCTION — DE L'EMBEDDING À L'AGENT INTELLIGENT

FICHE SYNTHÉTIQUE DE LA FORMATION

Intitulé	RAG en Production — De l'embedding à l'agent intelligent
Durée	4 jours (28 heures)
Référence	RAG-PROD-4J
Tarif	2 500 € HT / personne
Modalité	Présentiel ou distanciel
Public	Développeurs Python, data scientists, ML engineers, tech leads
Prérequis	Python intermédiaire, usage d'une API LLM, notions de NLP
Effectif	8 participants maximum
Attestation	Attestation de fin de formation Pythonia
Lieu	255 boulevard Pereire, 75017 Paris ou à distance (Zoom)

1. Public visé

Cette formation s'adresse aux professionnels techniques souhaitant concevoir, déployer et maintenir des systèmes RAG (Retrieval-Augmented Generation) en production : développeurs Python confirmés, data scientists, ML engineers, ingénieurs IA, architectes logiciels et tech leads chargés d'intégrer des capacités d'IA générative dans des applications métier.

2. Prérequis

- Maîtrise de Python intermédiaire (fonctions, classes, gestion de dépendances)
- Expérience préalable avec au moins une API LLM (OpenAI, Anthropic Claude, Mistral…)
- Notions de base en NLP (tokenisation, embeddings, similarité sémantique)
- Connaissance d'un environnement de développement (VS Code, Git, terminal)
- Anglais technique en lecture (documentation, articles de recherche)

Un test de positionnement est proposé avant l'entrée en formation afin de vérifier l'adéquation du parcours avec le profil du stagiaire.

3. Objectifs pédagogiques

À l'issue de cette formation, le stagiaire sera capable de :

- Comprendre l'architecture d'un système RAG et ses cas d'usage en entreprise (O1)
- Choisir et implémenter une stratégie d'embedding et de chunking adaptée (O2)
- Sélectionner et configurer un vector store pertinent (Chroma, Qdrant, pgvector…) (O3)
- Mettre en œuvre des techniques de retrieval avancées : hybride, reranking, HyDE (O4)
- Concevoir des architectures RAG agentiques avec routing et multi-step reasoning (O5)
- Traiter des documents complexes : PDF, tableaux, images (RAG multimodal) (O6)

- Évaluer objectivement la qualité d'un système RAG avec RAGAS et DeepEval (O7)
- Observer et monitorer un RAG en production (LangSmith, Langfuse) (O8)
- Sécuriser un système RAG : prompt injection, PII, exfiltration de données (O9)
- Optimiser la latence et les coûts : caching, batching, modèles hybrides (O10)
- Déployer un RAG avec FastAPI et Docker en tenant compte du scaling (O11)

4. Compétences visées

Code	Compétence
O1 – O3	Fondamentaux RAG, embeddings, vector stores
O4 – O5	Retrieval avancé et architectures agentiques
O6	RAG multimodal et documents complexes
O7 – O8	Évaluation, observabilité et monitoring
O9 – O10	Sécurité, optimisation des coûts et de la latence
O11	Déploiement et mise en production

5. Programme détaillé

JOUR 1 — Fondamentaux du RAG et premier pipeline (7h)

Objectifs pédagogiques visés : O1, O2, O3

Matin (3h30) — Comprendre et positionner le RAG

- Pourquoi le RAG : limites des LLM, hallucinations, knowledge cutoff (O1)
- RAG vs fine-tuning : coûts, maintenance, cas d'usage (O1)
- Typologie des cas d'usage : Q&A; documentaire, support client, copilote métier
- Architecture d'un pipeline RAG : ingestion, indexation, retrieval, génération
- Panorama des embeddings : OpenAI, Cohere, BGE, E5, modèles open source (O2)
- Critères de choix : dimension, langue, coût, latence, performance MTEB

Méthodes : exposé interactif, démonstrations live sur l'API OpenAI et Claude, comparaison d'embeddings en direct.

Après-midi (3h30) — Vector stores et premier pipeline

- Comparatif des vector stores : Chroma, Qdrant, Pinecone, pgvector, Weaviate (O3)
- Critères de sélection : self-hosted vs managed, scalabilité, filtres métier
- Stratégies de chunking : fixe, récursif, sémantique, par structure de document (O2)
- Pièges du chunking : perte de contexte, overlap, granularité
- Construction d'un pipeline avec LangChain et LlamaIndex
- Comparaison des deux frameworks : forces, faiblesses, cas d'usage

Travaux pratiques :

- TP1 : Générer et comparer des embeddings (OpenAI vs modèle open source) sur un corpus métier
- TP2 : Implémenter trois stratégies de chunking et mesurer leur impact qualitatif
- TP3 (projet fil rouge) : Premier RAG fonctionnel sur un corpus de documents internes (contrats, FAQ, documentation technique)

Évaluation formative : correction collective, identification des limites du premier pipeline, tour de table de fin de journée.

JOUR 2 — Retrieval avancé : la différence POC / production (7h)

Objectifs pédagogiques visés : O4

Matin (3h30) — Recherche hybride et reranking

- Limites de la recherche purement vectorielle : recall, entités, jargon métier
- Recherche lexicale avec BM25 : principes et implémentation
- Recherche hybride : fusion BM25 + dense, pondération, RRF (Reciprocal Rank Fusion) (O4)
- Reranking avec cross-encoders : pourquoi et quand l'utiliser
- Intégration de Cohere Rerank et de modèles open source (bge-reranker)
- Mesure du gain : précision, MRR, NDCG

Méthodes : démonstrations comparatives, live coding, analyse d'exemples de défaillance.

Après-midi (3h30) — Query transformation et contexte enrichi

- Query transformation : pourquoi réécrire la requête utilisateur
- HyDE (Hypothetical Document Embeddings) : principe et mise en œuvre
- Multi-query retrieval : générer plusieurs variantes et fusionner

- Step-back prompting : remonter à la question générale
- Contextual Retrieval (technique Anthropic) : enrichir chaque chunk de son contexte
- Metadata filtering : combiner recherche sémantique et filtres structurés

Travaux pratiques :

- TP1 : Implémenter une recherche hybride BM25 + dense et mesurer le gain
- TP2 : Ajouter un reranker Cohere et comparer les résultats
- TP3 (challenge) : Benchmarker 5 stratégies de retrieval sur un dataset annoté fourni, produire un rapport de performance

Évaluation formative : QCM mi-parcours de 20 questions sur les concepts de retrieval, analyse des résultats du benchmark.

JOUR 3 — RAG avancé, agentique et multimodal (7h)

Objectifs pédagogiques visés : O5, O6

Matin (3h30) — RAG agentique

- Du RAG linéaire au RAG agentique : quand et pourquoi (O5)
- Routers : diriger la requête vers la bonne source de données
- Self-querying : l'agent génère ses propres filtres de metadata
- Multi-step reasoning : décomposition de requêtes complexes
- RAG conversationnel : gestion de la mémoire et du contexte multi-tours
- Outils d'orchestration : LangGraph, LlamaIndex Workflows
- Génération avec citations structurées et traçabilité

Méthodes : démonstrations d'architectures agentiques, analyse de logs d'exécution, live coding.

Après-midi (3h30) — RAG multimodal et Graph RAG

- Documents complexes : PDF avec tableaux, images, schémas (O6)
- Outils de parsing avancé : Unstructured, LlamaParse, Docling
- RAG multimodal : embeddings d'images, description automatique par VLM
- Introduction au Graph RAG : principes de Microsoft GraphRAG
- Quand choisir Graph RAG vs RAG classique
- Knowledge graphs : extraction d'entités et de relations

Travaux pratiques :

- TP1 : Construire un agent RAG avec routing entre plusieurs sources (documentation, base de données, FAQ)
- TP2 : Traiter un corpus de PDF techniques contenant tableaux et schémas
- TP3 (mini-projet) : Assembler un RAG conversationnel multi-tours avec mémoire et citations

Évaluation formative : correction collective, tour de table, préparation du projet final.

JOUR 4 — Évaluation, production et sécurité (7h)

Objectifs pédagogiques visés : O7, O8, O9, O10, O11

Matin (3h30) — Évaluation et observabilité

- Pourquoi l'évaluation est critique : sans métriques, pas d'amélioration (O7)
- Construire un dataset d'évaluation : annotation manuelle vs synthétique
- RAGAS : faithfulness, answer relevancy, context precision, context recall
- DeepEval et Phoenix : alternatives et spécificités
- Évaluation LLM-as-a-judge : forces, limites, biais
- Observabilité en production : LangSmith, Langfuse, Arize (O8)
- Traces, spans, feedback utilisateur, détection de régression

Méthodes : démonstration complète d'un pipeline d'évaluation, analyse de cas réels.

Après-midi (3h30) — Sécurité, optimisation et déploiement

- Sécurité : prompt injection, jailbreaks, exfiltration de données sensibles (O9)
- Gestion des PII : détection, anonymisation, conformité RGPD
- Guardrails : Llama Guard, NeMo Guardrails, validations côté application
- Optimisation des coûts : caching sémantique, batching, modèles en cascade (O10)
- Latence : streaming, parallélisation du retrieval, pré-calcul

- Déploiement : FastAPI + Docker, architecture scalable (O11)
- Considérations DevOps : CI/CD, versioning des index, A/B testing

Travaux pratiques — Projet final :

- Chaque participant finalise son RAG (choix d'architecture, retrieval, évaluation)
- Mise en place d'un pipeline d'évaluation RAGAS reproductible
- Déploiement en API FastAPI avec endpoint sécurisé
- Soutenance individuelle : présentation du RAG, démonstration, analyse des métriques

Évaluation sommative : soutenance du projet final (20 min par participant) et QCM final de 30 questions couvrant l'ensemble du programme.

6. Méthodes et moyens pédagogiques

MÉTHODES MOBILISÉES

- **Apports théoriques :** exposés interactifs avec diaporama, démonstrations live sur API réelles (Claude, OpenAI, Cohere)
- **Mise en pratique intensive :** TP individuels sur datasets métier fournis, projet fil rouge sur les 4 jours
- **Pédagogie par la comparaison :** chaque concept est démontré en comparant plusieurs approches (framework, modèle, stratégie)
- **Alternance théorie/pratique :** ratio 30 % théorie / 70 % pratique
- **Projet final intégrateur :** chaque participant repart avec un RAG déployé et évalué, réutilisable en contexte professionnel
- **Retours personnalisés :** soutenance individuelle avec feedback détaillé sur les choix d'architecture

MOYENS TECHNIQUES

- Salle de formation équipée : vidéoprojecteur, paperboard, connexion Wi-Fi haut débit
- Poste informatique par stagiaire avec VS Code, Python 3.11+ et extensions préinstallées
- Environnement cloud fourni pendant la formation (accès GPU et vector stores managés)
- Clés API fournies pour OpenAI, Anthropic Claude et Cohere pendant toute la durée de la formation
- En distanciel : Zoom avec partage d'écran, tableau blanc collaboratif, salons de groupe
- Supports de cours, notebooks Jupyter et code source remis aux stagiaires
- Template de RAG production-ready prêt à forker, livré en fin de formation

7. Modalités d'évaluation

Avant l'entrée en formation, chaque participant complète un test de positionnement évaluant son niveau Python et ses acquis en IA générative.

POSITIONNEMENT À L'ENTRÉE

- Test de positionnement en ligne (Python, API LLM, NLP) avant la formation
- Entretien téléphonique avec le formateur en cas de doute sur le niveau

ÉVALUATIONS EN COURS DE FORMATION

- **Évaluation formative :** QCM mi-parcours de 20 questions (Jour 2), corrections collectives quotidiennes
- **Évaluation sommative :** soutenance du projet final + QCM final de 30 questions (Jour 4)
- **Critères de réussite :** obtenir 60 % au QCM final et présenter un projet fonctionnel
- **Attestation de fin de formation :** remise à l'issue de la formation, mentionnant les objectifs pédagogiques, la durée et les résultats

SUIVI POST-FORMATION

- Accès au code source, notebooks et template production-ready
- Invitation à la communauté Pythonia des anciens stagiaires

SATISFACTION

- Questionnaire de satisfaction en fin de formation (formateur, contenu, logistique, atteinte des objectifs)

- Questionnaire à froid à 3 mois sur l'impact en situation professionnelle

8. Accessibilité aux personnes en situation de handicap

Pythonia s'engage à rendre ses formations accessibles aux personnes en situation de handicap. Notre référent handicap est disponible pour étudier les aménagements nécessaires :

- Adaptation des supports pédagogiques (taille de police, contraste, format audio)
- Aménagement des conditions de passage (temps majoré, pauses adaptées)
- Orientation vers nos partenaires spécialisés : Agefiph, Cap Emploi, MDPH

Contact référent handicap : contact@pythonia.fr – 06 66 06 38 59

9. Engagement et suivi

- **Suivi de l'assiduité :** feuille d'émargement signée par demi-journée
- **Prévention des abandons :** suivi individualisé, adaptation du rythme, relance en cas d'absence
- **Réclamations :** toute réclamation peut être adressée par email à contact@pythonia.fr. Elle sera traitée sous 48h ouvrées.