PYTHON & DATA : API, INTELLIGENCE ARTIFICIELLE

Intégrer des LLM dans une application avec l’API OpenAI ou Anthropic

Apprenez à intégrer des LLM dans vos applications avec l’API OpenAI ou l’API Anthropic, choisie lors de l’analyse du besoin : prompts structurés, sorties JSON, conversations, function calling, RAG, streaming et coûts.

DURÉE	TARIF	NIVEAU	LANGUE	GROUPE	FORMAT
3j (21h)	**1 275 € / participant** Inter-entreprise. Intra : sur devis, au plus 1 275 € / participant	**Intermédiaire**	**Français**	**1-6**	**Formation** Présentiel ou à distance

1 PUBLIC VISÉ

- Développeurs web et Python qui veulent intégrer des LLM dans leurs applications ou leurs outils internes
- Développeurs freelances ou en agence qui veulent proposer des fonctionnalités d’IA à leurs clients

2 PRÉREQUIS

- Appeler une API REST avec une clé d’authentification, en Python (requests ou httpx) et en JavaScript (fetch, async / await), et lire sa réponse JSON
- Aucune connaissance en machine learning ni en mathématiques n’est requise

3 OBJECTIFS PÉDAGOGIQUES

- Appeler l’API du fournisseur retenu (OpenAI ou Anthropic) en Python et en JavaScript, en réglant la température et en gérant les erreurs et les nouvelles tentatives
- Rédiger un system prompt structuré (rôle, contraintes, format de sortie) et y appliquer les techniques zero-shot, few-shot et chain-of-thought pour une tâche métier
- Obtenir des sorties JSON ou balisées valides et les exploiter dans le code
- Gérer une conversation sur plusieurs tours en conservant l’historique utile
- Faire appeler des fonctions par le modèle (function calling) et construire un RAG simple sur ses propres documents
- Intégrer un LLM dans un backend FastAPI ou Express avec réponse en streaming, sans exposer la clé API côté client
- Estimer le coût d’un usage et proposer des optimisations

4 PROGRAMME DÉTAILLÉ

Jour 1 (7h) : Comprendre les LLM et rédiger des prompts

Module 1 : Appeler les API des LLM (3h30)

Comment fonctionne un LLM (1h30)

- Qu’est-ce qu’un LLM : prédiction du token suivant, pas de « compréhension »

- La fenêtre de contexte : taille, conséquences, ce que le modèle « voit »
- Les tokens : ce que c'est, comment les compter, pourquoi ils comptent dans la facturation
- Température et top-p : réponses plus variées ou plus constantes
- Différences pratiques entre les modèles : grands modèles et modèles légers, qualité, coût et latence
- Les hallucinations : pourquoi elles se produisent, comment les limiter
- Exercice d'entraînement : observer l'effet de la température sur des réponses répétées

L'API du fournisseur retenu (2h)

- Le fournisseur, OpenAI ou Anthropic, est choisi lors de l'analyse du besoin ; toute la formation se fait avec son API
- Modèles disponibles et tarification par token ; l'endpoint principal : Responses (/v1/responses) chez OpenAI, Messages (/v1/messages) chez Anthropic
- Créer ses clés API, les ranger dans des variables d'environnement, bonnes pratiques de sécurité
- Le SDK officiel du fournisseur, en Python et en JavaScript
- Gérer les erreurs : limite de requêtes (429), délai dépassé, nouvelles tentatives
- TP évalué : appeler l'API retenue en Python et en JavaScript avec la même consigne, sur un grand modèle et un modèle léger, régler la température, comparer les réponses et relancer automatiquement un appel refusé pour limite de requêtes

Module 2 : Rédiger des prompts efficaces (3h30)

Les rôles et la structure d'un prompt (1h30)

- Les rôles system, user et assistant
- Écrire un bon system prompt : persona, contraintes, format de sortie, ton
- L'ordre des instructions : ce qui est lu en premier et en dernier
- Délimiteurs et structure : balises XML, markdown, numérotation
- Exercice d'entraînement : une même tâche, trois system prompts différents

Zero-shot, few-shot, chain-of-thought et tâches courantes (2h)

- Zero-shot : une instruction claire sans exemple
- Few-shot : guider le modèle avec des exemples d'entrée et de sortie ; choisir des exemples variés, avec des cas limites et un format constant
- Chain-of-thought : demander au modèle de raisonner étape par étape avant de répondre
- Tâches courantes : classification, résumé, transformation en JSON, génération de code, correction et traduction
- TP évalué : traitement de tickets support : system prompt structuré, classification en few-shot, résumé avec chain-of-thought, résultats comparés à une version zero-shot

Jour 2 (7h) : Sorties structurées, conversations, outils et RAG

Module 3 : Structurer les sorties et gérer les conversations (3h30)

Obtenir du JSON et du texte balisé fiables (1h30)

- Pourquoi les LLM produisent parfois du JSON invalide
- Instructions de format dans le prompt : schéma explicite, exemples, contraintes
- Les sorties structurées de l'API : imposer un schéma JSON à la réponse
- Valider et lire la réponse : json.loads, gestion des erreurs, nouvelle demande automatique
- Sorties balisées : quand préférer XML à JSON, balises personnalisées, lecture avec re ou xml.etree
- Exercice d'entraînement : extraire des données structurées d'e-mails non formatés

Conversations sur plusieurs tours (2h)

- Construire et maintenir l'historique messages[]
- Quand l'historique dépasse la fenêtre de contexte : supprimer les anciens tours, résumer le passé
- Enrichir le system prompt à la volée : données de l'utilisateur, données métier, date et heure
- Modèles de prompts avec f-strings et Jinja2 ; ranger les prompts hors du code et les versionner
- TP évalué : chatbot de support technique qui répond en JSON valide exploité par le code et conserve l'historique utile sur plusieurs tours, en tronquant les anciens tours

Module 4 : Function calling et RAG (3h30)

Function calling (1h30)

- Le principe : le modèle décide d'appeler une fonction, il ne l'exécute pas
- Déclarer des outils : nom, description, paramètres en JSON Schema
- Intercepter l'appel, exécuter la vraie fonction, renvoyer le résultat au modèle
- Enchaîner plusieurs appels d'outils ; limiter les risques : délai, nombre maximal d'itérations, validation des paramètres
- Exercice d'entraînement : faire appeler par le modèle une API météo fictive

RAG simplifié (2h)

- Pourquoi un LLM ne « connaît » pas vos données métier ; l'idée du RAG
- Embeddings : ce que c'est, comment les générer : modèle d'embeddings d'OpenAI, ou de Voyage AI avec Anthropic, qui n'en propose pas
- Similarité cosinus ; stocker les embeddings avec numpy, introduction à ChromaDB
- Le pipeline : indexation, recherche, prompt augmenté, réponse ; découper les documents en morceaux
- Limites et pièges : morceaux trop longs, mauvaise récupération, hallucination malgré le contexte
- TP évalué : FAQ qui répond à partir d'une documentation Markdown avec un RAG simple et fait appeler par le modèle un outil de recherche dans un catalogue fictif

Jour 3 (7h) : Intégration dans une application et projet de synthèse

Module 5 : Intégrer un LLM dans une application web et maîtriser son coût (3h)

Endpoint en streaming et interface (1h45)

- Créer un endpoint /chat qui relaie les appels à l'API du LLM, avec FastAPI ou Express
- Streaming des réponses avec le SDK du fournisseur ; Server-Sent Events vers le navigateur
- Afficher les tokens au fur et à mesure avec EventSource ou fetch et ReadableStream ; indicateur de chargement, bouton d'annulation
- Ne jamais exposer une clé API côté client
- Exercice d'entraînement : interface de chat minimaliste connectée à un endpoint fourni

Sécurité et coûts (1h15)

- Limiter les appels par utilisateur ; modérer les entrées ; tracer les appels et leurs coûts
- La facturation : tokens d'entrée et de sortie, prix par modèle ; calculer le coût d'un usage avant de développer
- Réduire le coût : choisir le modèle selon la tâche, raccourcir les prompts, mettre les prompts en cache
- TP évalué : endpoint FastAPI ou Express qui streame les réponses d'un assistant sans exposer la clé, puis estimation du coût par appel et deux optimisations chiffrées

Module 6 : Projet de synthèse, évalué (4h)

Réalisation de la mini-application (2h30)

- Réalisation individuelle d'une fonctionnalité d'IA de bout en bout, au choix dans une liste (assistant de rédaction, analyseur de contrat, générateur de tests unitaires, FAQ, résumé de tickets)
- Étape 1 : concevoir le prompt et l'architecture
- Étape 2 : endpoint backend (FastAPI ou Express) avec system prompt structuré et versionné, et au moins une technique avancée : function calling, RAG, streaming ou conversation sur plusieurs tours
- Étape 3 : interface minimaliste et estimation du coût par appel

Présentation et retour (1h30)

- Présentation et revue collective
- Retour individuel du formateur sur les choix de prompting et d'architecture

Le contenu peut être ajusté selon le niveau, les prérequis et les besoins des participants.

5 COMPÉTENCES VISÉES

- Appeler les API des LLM et rédiger des prompts structurés pour des tâches métier
- Exploiter les réponses d'un LLM dans une application : sorties structurées, conversations sur plusieurs tours
- Étendre un LLM avec des outils et ses propres documents : function calling et RAG
- Intégrer un LLM dans une application web en streaming et en maîtriser le coût

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou à distance, en classe virtuelle. Le formateur alterne entre méthode démonstrative, méthode interrogative (analyse des sorties du modèle et des choix de prompting) et méthode active (exercices d'entraînement, TP évalués et projet de synthèse).

7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique mis à disposition des apprenants
- Scripts Python et JavaScript annotés avec exercices et corrections par module
- Environnement de développement : VS Code + Python 3.12+ + Node.js 22+ + un compte chez le fournisseur retenu (OpenAI ou Anthropic) par participant, avec quelques euros de crédit ; avec Anthropic, un compte Voyage AI pour les embeddings
- Pour le distanciel : visioconférence (Microsoft Teams ou équivalent), partage d'écran, chat en direct
- Supports remis par e-mail et sur la plateforme pédagogique LaPolaris (ressources, émargement)

En formation intra-entreprise sur le site du client, celui-ci s'assure et s'engage à disposer de toutes les ressources matérielles nécessaires à son bon déroulement (salle, postes de travail, logiciels, accès à Internet), conformément aux prérequis indiqués dans le programme et aux moyens prévus dans la convention.

8 MODALITÉS D'ÉVALUATION

- Questionnaire de positionnement individuel renseigné par chaque participant avant le démarrage de la formation, analysé par le formateur et donnant lieu, le cas échéant, à un ajustement du déroulé
- Exercices d’entraînement réalisés par chaque participant sur son poste, corrigés en séance
- Évaluation des acquis par les TP évalués, réalisés individuellement sur poste : un par module, puis le projet de synthèse décrit dans le dernier module du programme ; chaque objectif est évalué dans au moins un TP
- Grille de suivi renseignée par le formateur pour chaque participant et pour chacun des objectifs de l’action, au regard des critères d’évaluation ci-dessous
- Questionnaire d’appréciation renseigné par les participants en clôture de formation, et enquête à froid dans un délai de trois à six mois

9 CRITÈRES D'ÉVALUATION

- Appels qui aboutissent vers l’API retenue en Python et en JavaScript ; température choisie selon la tâche ; limite de requêtes et délai dépassé gérés par de nouvelles tentatives
- System prompt avec rôle, contraintes et format de sortie ; technique zero-shot, few-shot ou chain-of-thought choisie selon la tâche, résultats comparés
- Sorties JSON ou balisées valides, lues par le code sans erreur, réponse invalide redemandée automatiquement
- Historique conservé sur plusieurs tours, anciens tours tronqués ou résumés quand la fenêtre de contexte l’exige
- Fonction appelée par le modèle avec les bons paramètres et son résultat renvoyé ; réponse du RAG appuyée sur les passages récupérés dans les documents fournis
- Endpoint backend qui streame une réponse affichée au fil de l’eau ; clé API seulement côté serveur
- Coût par appel calculé à partir des tokens d’entrée et de sortie, avec au moins deux optimisations chiffrées

10 INDICATEURS DE RÉSULTATS

Les indicateurs de résultats de cette formation (taux de satisfaction et taux de réussite) seront publiés sur sa fiche en ligne à l’issue des premières sessions réalisées.

11 MODALITÉS DE VALIDATION

Une attestation de fin de formation est remise à chaque participant à l’issue du parcours. Elle mentionne **l’intitulé** et la **nature de l’action**, sa **durée** en heures, sa **période de réalisation**, **les objectifs de l’action** et le **niveau d’atteinte** de chacun d’eux, exprimé en trois valeurs : **atteint**, **partiellement atteint**, **non atteint**. **Une attestation d’assiduité**, établie à partir des émargements, est délivrée dans les mêmes conditions.

Formation non certifiante : elle ne délivre pas de certification enregistrée au RNCP ou au Répertoire spécifique.

12 SUIVI ET ACCOMPAGNEMENT

- Émargement électronique signé par chaque participant et par le formateur à chaque demi-journée, en présentiel comme à distance
- Traçabilité des activités pédagogiques réalisées
- Attestation d’assiduité délivrée en fin de formation
- Suivi individuel via les QCM de fin de journée, les exercices corrigés et les TP évalués

13 CONDITIONS D'ACCÈS

Formation accessible sans prérequis administratif particulier. Cette formation n’est pas éligible aux financements publics ou mutualisés, notamment au compte personnel de formation et à une prise en charge par un opérateur de compétences.

14 DÉLAIS D'ACCÈS

Inscription possible jusqu’à 15 jours ouvrés avant le début de la session pour permettre la préparation logistique et, si besoin, l’adaptation du contenu au contexte client. Pour toute demande spécifique, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations peuvent être accessibles aux personnes en situation de handicap. La question d’un besoin d’aménagement (matériel, temporel ou pédagogique) est posée lors de l’analyse du besoin et du questionnaire de positionnement ; vous pouvez aussi nous le signaler à tout moment, y compris pendant la formation, afin d’étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

LaPolaris

TÉL. +33762584798
EMAIL contact@lapolaris.fr
WEB lapolaris.fr