
LangSmith
By
LangChain

Depuis l'arrivée de ChatGPT (OpenAI) pour le Grand Public en 2022, l'adoption des solutions d'IA Générative a été rapide, pour ne pas dire quasi immédiate.

D'autres acteurs ont mis à disposition leurs propres solutions depuis, donnant une impression de course à la nouveauté permanente, depuis maintenant 3 ans.

Actuellement, il ne se passe pas une semaine sans que l'on entende parler d'au moins un nouvel outil : un nouveau modèle (ChatGPT5, Claude Sonnet 4.1, mais encore de multiples modèles qui arrivent tous les jours sur des sites comme https://huggingface.co/ ou https://openrouter.ai/).

Les outils Low Code ou No Code sont aussi de la partie, avec des solutions comme Flowise, Langflow ou encore n8n, qui emportent l'adhésion de plus en plus d'utilisateurs, au vu des posts LinkedIn ou des vidéos Youtube sur ces sujets.

Et puis, il y a des noms de frameworks, plus techniques, qui sont là depuis longtemps, et avec lesquels les nouveautés ont intérêt à être compatibles. Des noms comme LangChain, ou encore LangSmith.

Nous ferons ici le focus sur LangChain et sur LangSmith.

LangChain est une librairie utilisable dans les mondes Python et Node. Elle permet d'interagir avec des modèles d'Intelligence Artificielle, utilisant l'IA dans les fonctionnalités même des futures applications. On parle ici d'Agents IA, et même d'Architecture Agentique, dès lors que plusieurs agents interagissent ensemble et sont même capables de se superviser entre eux, pour automatiser un Workflow de travail qui peut être conséquent.

LangSmith est un outil qui devrait plaire aux testeurs des applications agentiques construites avec LangChain ou ses équivalents. Cet outil permet à tout un chacun de se créer un projet qui va permettre de tracer les appels, et de voir si une chaîne d'IA qui est invoquée récupère bien les bonnes informations, renvoie bien une réponse dans le bon format attendu, utilise un nombre attendu de tokens, pour un coût qui n'explose pas (pour les modèles payants).

Voyons à présent comment cela fonctionne, en 3 étapes :

1- Avoir à disposition et exécuter du code (ici Python) qui utilise LangChain pour faire des appels à des modèles d'IA.
2- Obtenir les credentials LangSmith en créant un projet de traçage, et les configurer dans votre .env
3- Exécuter de nouveau et voir les traces de l'appel.

1. Code Python / LangChain à disposition

Nous avons à disposition le code Python suivant, qui demande au modèle OpenAI gpt-3.5-turbo-1106 de donner une liste de 10 synonymes du mot « content » :

```python
from langchain_openai import ChatOpenAI
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.output_parsers import StrOutputParser, CommaSeparatedListOutputParser, JsonOutputParser
from pydantic import BaseModel, Field


# Instantiate Model
llm = ChatOpenAI(
    model="gpt-3.5-turbo-1106",
    temperature=0.7
)
```

```python
def call_list_output_parser():
    # Prompt Template
    prompt = ChatPromptTemplate.from_messages(
        [
            ("system", "Génère une liste de 10 synonymes du mot suivant. Retourne le résultat sous forme de liste, avec des
            virgules comme séparateurs."),
            ("human", "{input}")
        ]
    )

    parser = CommaSeparatedListOutputParser()

    # Create LLM Chain
    chain = prompt | llm | parser

    return chain.invoke({"input": "content"})
```

```python
print(call_list_output_parser())
```

En exécutant ce code, on obtient la liste suivante :

['heureux', 'satisfait', 'comblé', 'joyeux', 'enchanté', 'ravi', 'épanoui', 'radieux', 'béat', 'réjoui']

Le résultat est ici intéressant et conforme à ce à quoi on pouvait s'attendre.

Par contre, ce résultat n'est pas tracé puisque présent seulement dans le terminal du développeur qui exécute ce code Python, et surtout on a du mal à savoir combien de TOKENS ont été utilisés, et quel a été le coût de consommation de notre API OpenAI (qui est payante…) utilisé lors de cette exécution.

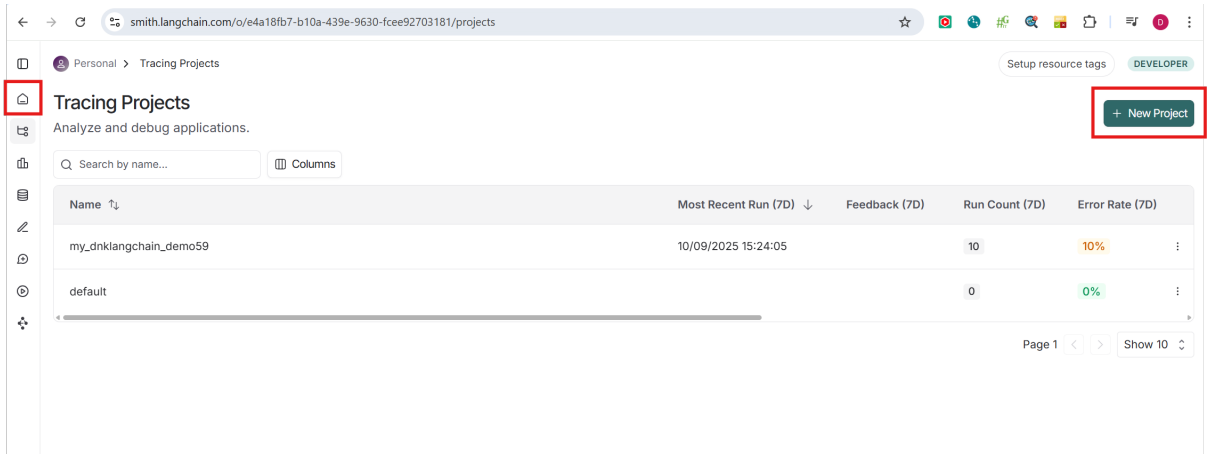
2. Création d'un projet de tracing dans LangSmith et récupération des credentials

Pour répondre aux questions précédemment posées, nous utilisons LangSmith, avec ce lien sur lequel nous nous sommes créés un compte (une adresse mail suffit, et nous n'aurons pas à payer quoi que ce soit) :

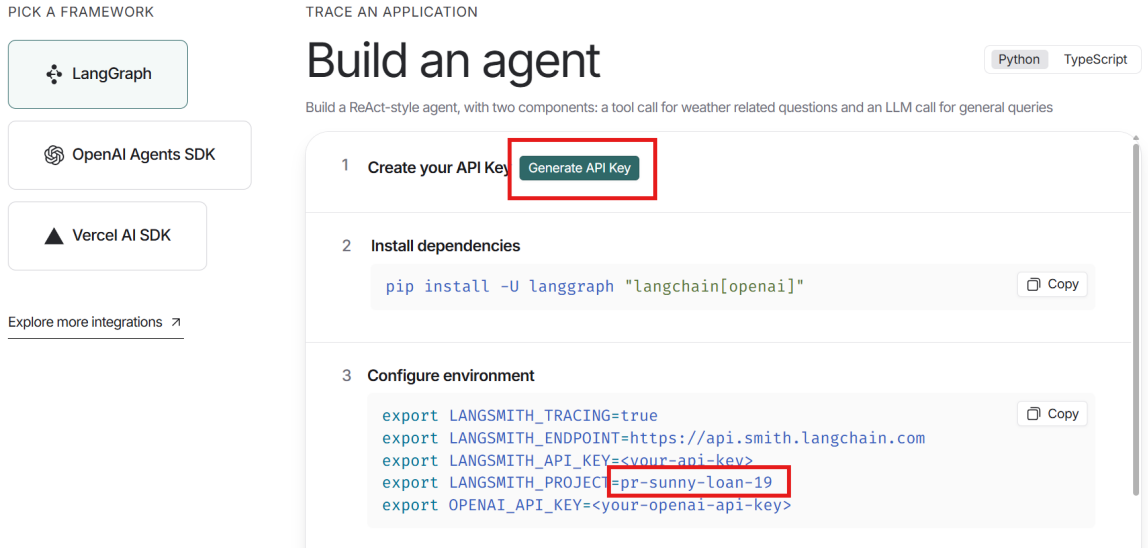
https://smith.langchain.com/

Sur cet outil , nous avons besoin de 2 choses exclusivement :

1- Créer un « Tracing Project »



2- Générer une API Key sur la page de création du « Tracing Project » :



Nous avons cliqué sur « Generate API Key » et sauvegardé la clé générée, qui est la suivante (cette clé a été révoquée depuis) :

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

Cette clé est prévue pour fonctionner avec le Tracing Project qui se nomme pr-sunny-loan-19 : ce nom a été donné au hasard au clic sur « New Project » dans l'étape précédente.

Nous sommes maintenant munis de nos 2 informations qui permettent de copier les 4 lignes suivantes dans le fichier .env de notre projet Python :

LANGCHAIN_TRACING_V2=true

LANGCHAIN_ENDPOINT="https://api.smith.langchain.com"

LANGCHAIN_API_KEY="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"

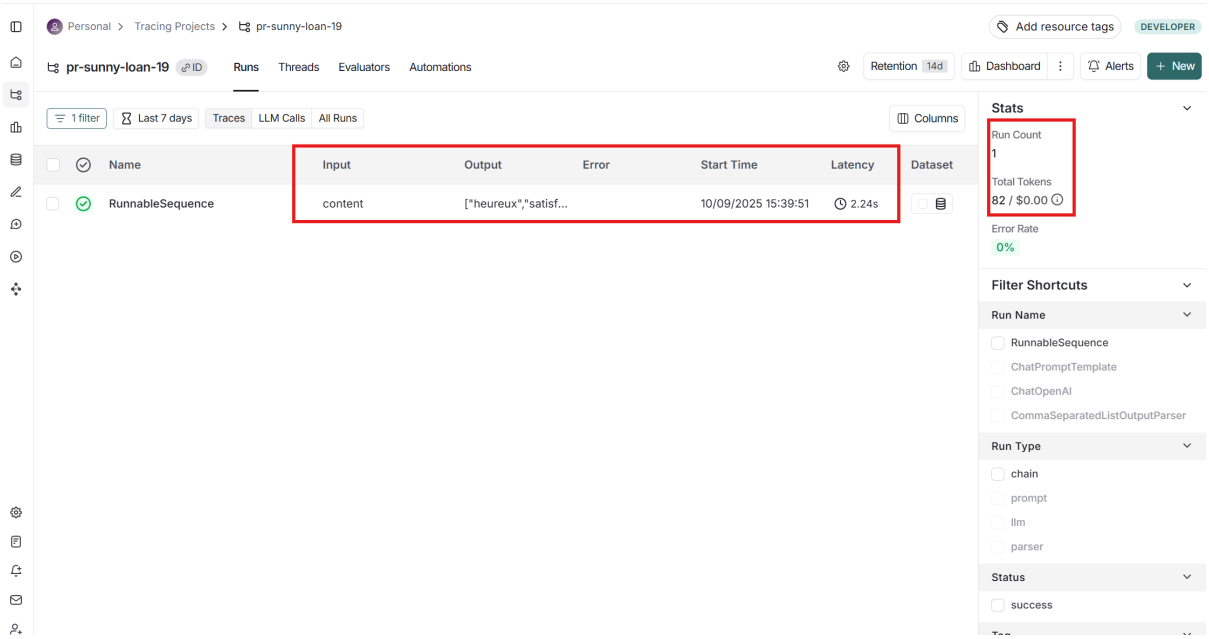
LANGCHAIN_PROJECT="pr-sunny-loan-19"

3. Visualisation des Traces d'appel dans LangSmith

A présent que les credentials sont renseignés dans .env (pour des raisons de sécurité), nous avons exécuté une seconde fois ce code, et nous avons obtenu la liste suivante de synonymes du mot « content », toujours dans le terminal du développeur qui exécute ce code :

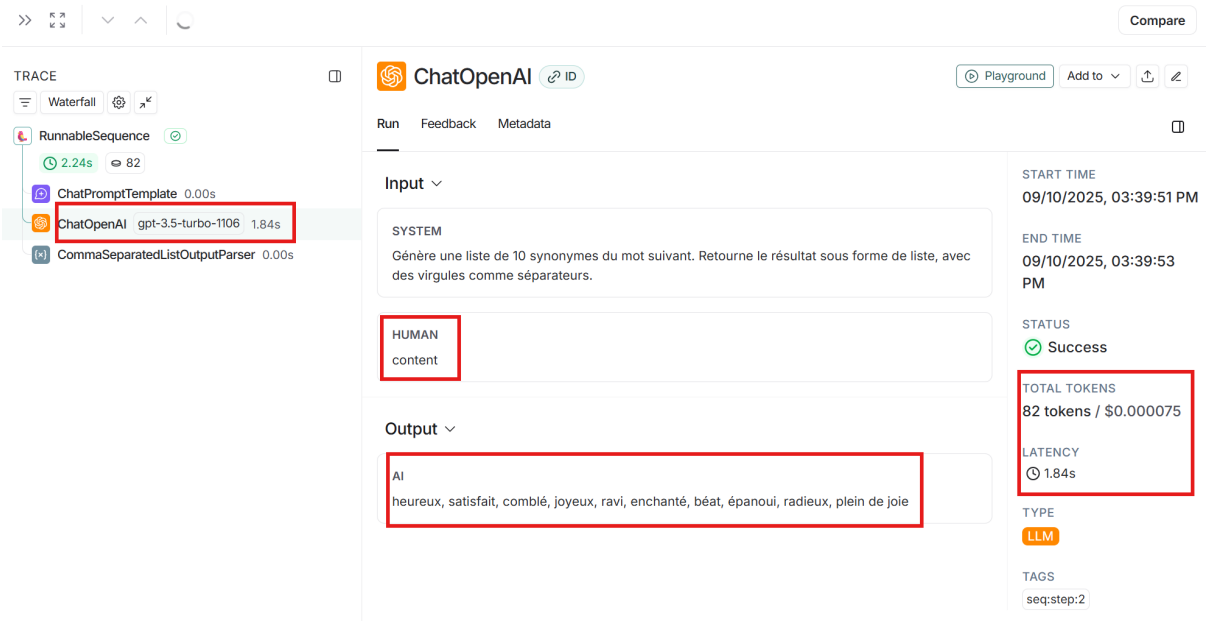
['heureux', 'satisfait', 'comblé', 'joyeux', 'ravi', 'enchanté', 'béat', 'épanoui', 'radieux', 'plein de joie']

Retournons sur LangSmith à présent que ce code a été exécuté :



On peut voir que le Tracing Project a été créé, et qu'un appel est ici tracé.

Au clic sur cet appel, l'écran suivant apparaît :



L'appel a été tracé, et plusieurs informations intéressantes sont disponibles :

- Le nom du modèle qui a été appelé (ici gpt-3.5-turbo-1106)
- Le prompt complet qui lui a été passé, et sa réponse
- Le nombre de tokens qui a été nécessaire (ici, 82 au total)
- Le coût de cet appel si le modèle utilisé est payant (ici, 0,000075$)
- Le temps de latence pour obtenir cette réponse (ici, 1,84 secondes)

Il est à noter que la durée de rétention de cet appel est de 14 jours avec une version gratuite de LangSmith.