

# **Dal prompt al prodotto: un chatbot con Gradio e deploy su Hugging Face Spaces**

Stefano D'Urso



# Stefano D'Urso



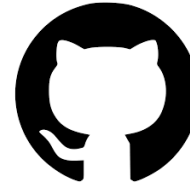
[Stefano D'Urso | LinkedIn](#)



[Schools'in! Education pills 4u!](#)

## Attualmente...

- Docente @ ITIS G. Vallauri – Velletri
- PhD Student @ Universitas Mercatorum
- Docente @ Università Telematica Pegaso
- Co-Founder DyslexIA s.r.l.



[paisleypark3121/VelletriDevDemo](#)

<https://tinyurl.com/VelletriDEVdemo>



# Large Language Models




## Cosa sono

Modelli di IA addestrati su enormi quantità di testo per comprendere e generare linguaggio naturale.




## Applicazioni

Chatbot, traduzione automatica, generazione di testi, assistenti virtuali.

## Tecnologia alla base

-  Deep Neural Networks
-  Architettura Transformer
-  Meccanismi di attenzione per il contesto

## Caratteristiche principali

-  **Tokenizzazione:** il testo è suddiviso in unità (token)
-  **Dataset massivi:** libri, articoli, pagine web
-  **Generazione contestuale:** output basato sui token precedenti

# Large Language Models

## Principio base

Prevede la **parola successiva** in una sequenza di testo.




## Tecnica

- Analisi di enormi quantità di dati testuali
- Apprendimento delle probabilità di co-occorrenza delle parole

## Natura del modello

Non “ragiona” come un umano → imita **schemi linguistici** basati su probabilità.

## Sfide principali

-  **Allucinazioni**: può generare risposte non veritiere
-  **Bias nei dati**: riflette i pregiudizi del dataset
-  **Ragionamento complesso**: difficoltà con deduzioni logiche profonde

# Hugging Face

 **Piattaforma open-source** per AI, con focus su NLP e LLM




 Riferimento grazie a:

-  **Libreria Transformers**
-  **Hub di modelli pre-addestrati**
-  **Servizi API per l'AI**






# Hugging Face

## Hugging Face Hub

-  **Modelli pre-addestrati** (GPT, LLaMA, Mistral, Claude...)
-  **Dataset pubblici** pronti per l'addestramento
-  **Spaces** → app interattive con Gradio/Streamlit

## Libreria *Transformers*


-  Caricare modelli con poche righe di codice
-  Eseguire inferenza (testo, classificazione, traduzione...)
-  Fare fine-tuning su dataset personalizzati


## Hugging Face Spaces

- Creazione rapida di **interfacce AI**
- Supporto a Gradio & Streamlit
- Condivisione e test di modelli in modo interattivo



 **Libreria Python open-source** per creare **interfacce web interattive**

 **Semplice e veloce:** demo e test di modelli AI con poche righe di codice





 **Accessibile via web:** locale o condivisa online

 **Compatibile con Hugging Face Spaces**





### **Caratteristiche principali**

-  Creazione di interfacce personalizzate con widget (testo, immagini, audio, video)
-  Condivisione immediata tramite link
-  Compatibile con **TensorFlow, PyTorch, scikit-learn, Hugging Face**
-  Personalizzazione con CSS

### **Gradio Playground**

- Esegui e prova modelli **direttamente online**
- Senza bisogno di configurazioni complesse





⚙️ Installazione di Gradio  
`pip install gradio`

Verifica installazione:  
`import gradio as gr`  
`print(gr.__version__)`

🚀 Prima interfaccia Gradio  
`import gradio as gr`

```
def greet(name):  
    return f"Ciao, {name}!"
```

```
iface = gr.Interface(fn=greet, inputs="text",  
outputs="text")  
iface.launch()
```



127.0.0.1:7860

name

Stefano

Clear Submit

output

Ciao, Stefano!

Flag





## **Gradio con API Key**

## **Accesso a modelli remoti**

È possibile collegare Gradio a servizi esterni (OpenAI, Hugging Face, Google Gemini...) tramite **API Key**.

## **Semplicità**

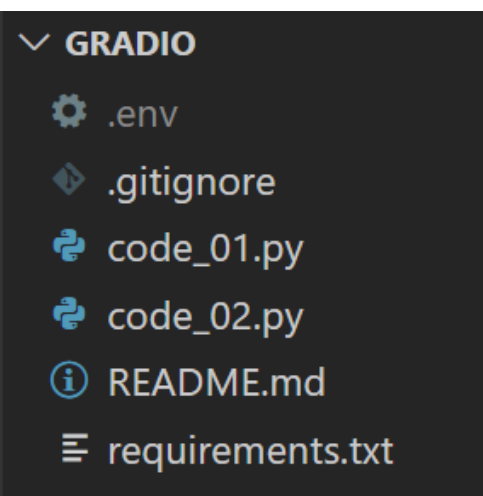
- Si inserisce la chiave nel file .env
- Si carica nel codice con dotenv
- Gradio gestisce l'interfaccia → input/output immediati

## **Esempi tipici**

- Chatbot con modelli OpenAI
- Traduzione automatica
- Generazione di testo o immagini



# GitHub Project



<https://tinyurl.com/VelletriDEVdemo>





# GitHub Project – README.md

```
---  
title: Velletri Dev Demo  
emoji: 🧪  
colorFrom: indigo  
colorTo: purple  
sdk: gradio  
sdk_version: "4.44.0"  
app_file: code_02.py  
pinned: false  
---
```

Demo Gradio per test LLM con API Key da Secrets.

## ATTENZIONE

- Mantenere il nome del file  
MAIUSCOLO e mantenere la stessa  
struttura per garantire un corretto  
deploy su HuggingFace Space



# GitHub Project – code\_02.py

```
import os
from dotenv import load_dotenv
import gradio as gr
from openai import OpenAI

load_dotenv()

api_key = os.getenv("OPENAI_API_KEY")

client = OpenAI(api_key=api_key)

def chat_with_llm(prompt):
    response = client.chat.completions.create(
        model="gpt-4.1-nano",
        messages=[{"role": "user", "content": prompt}]
    )
    return response.choices[0].message.content
```

```
iface = gr.Interface(
    fn=chat_with_llm,
    inputs="text",
    outputs="text",
    title="Chat con LLM via OpenAI API"
)

iface.launch()
```





# Esecuzione in locale

## Chat con LLM via OpenAI API

prompt

mi chiamo stefano

Clear

Submit

output

Ciao Stefano! Come posso aiutarti oggi?

Flag

# Deploy su Hugging Face Spaces

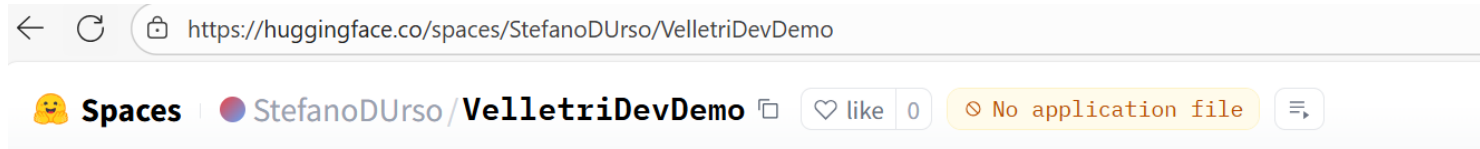
## Creazione di uno space su Hugging Face

- Vai su 🤗 **Hugging Face Spaces**
- Clicca “**New Space**” → scegli **Gradio** come SDK





# Deploy su Hugging Face Spaces



**Get started with your gradio Space!**

Your new space has been created, follow these steps to get started (or read the full [documentation](#))

Start by cloning this repo by using:

**HTTPS**

SSH

```
# When prompted for a password, use an access token with write permissions.  
# Generate one from your settings: https://huggingface.co/settings/tokens  
git clone https://huggingface.co/spaces/StefanoDUrso/VelletriDevDemo
```

**CLI**

```
# Make sure hf CLI is installed: pip install -U "huggingface_hub[cli]"  
hf download StefanoDUrso/VelletriDevDemo --repo-type=space
```



# Deploy su Hugging Face Spaces

## 🔑 Creazione di un Access Token su Hugging Face

- Vai su 😊 **Hugging Face** → **Settings** → **Access Tokens**  
👉 <https://huggingface.co/settings/tokens>
- Crea un nuovo token con:
  - **Type:** *Write*
  - **Scope:** (puoi lasciare “all spaces” oppure specificare)
- Conservare il token generato



# Deploy su Hugging Face Spaces



[Models](#) [Datasets](#) [Spaces](#) [Community](#) [Docs](#) [Pricing](#) [⌵](#)



**Stefano D'Urso**

StefanoDUrso

Profile

Account

Authentication

Organizations

Billing

**Access Tokens**

SSH and GPG Keys

## < Create new Access Token

### Token type

Fine-grained [Read](#) **[Write](#)**

ⓘ This cannot be changed after token creation.

### Token name

This token has read and write access to all your and your orgs resources and can make calls to Inference Providers on your behalf.

Create token



# Deploy su Hugging Face Spaces

## Creazione Variabili di Ambiente

- Vai sul tuo Space → **Settings** → **Variables and secrets**
- Clicca **New secret**
  - **Name** → OPENAI\_API\_KEY
  - **Value** → la tua chiave sk-...
  - Salva



# Deploy su Hugging Face Spaces

https://huggingface.co/spaces/StefanoDUrso/VelletriDevDemo/settings

1 GB

**Variables and secrets** ⓘ

New variable New secret

Variables Public

No variables

Secrets Private

No secrets



# Deploy su Hugging Face Spaces

## Push su GitHub

```
git push -u origin main
```

## Autenticazione su Hugging Face tramite Access Token

```
huggingface-cli login --token hf_AccessToken
```

## Aggiungi Hugging Face come secondo remote

```
git remote add hf https://huggingface.co/spaces/<username>/<space>
```

È possibile verificare che il remote è stato aggiunto tramite:

```
git remote -v
```

## Push su HuggingFace

```
git push hf main
```



# Deploy su Hugging Face Spaces



## Chat con LLM via OpenAI API

prompt

ciao!

Clear

Submit

output

Ciao! Come posso aiutarti oggi?

*Crazie*