الأكاديمية العربية الدولية
Arab International Academy

التعامل مع واجهة برمجة التطبيقات (API)

في لغة بايثون

Dealing with API in Python

الإستاذ ربيع ع. قبلاوي

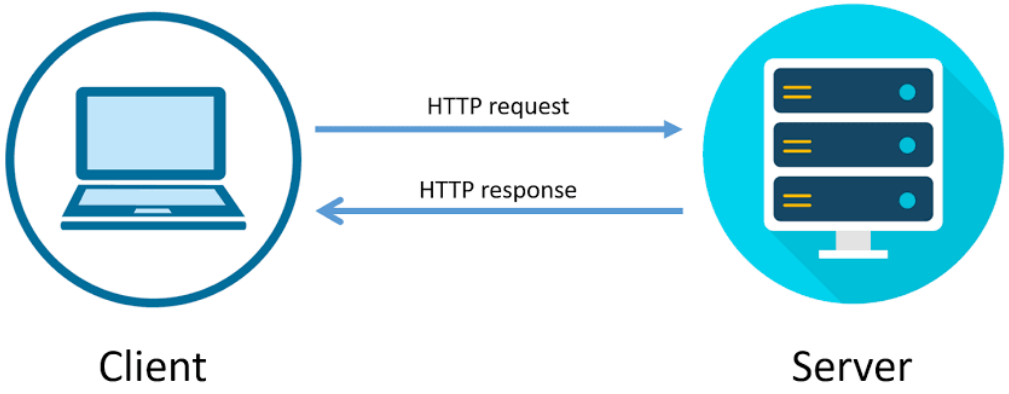
الأكاديمية العربية الدولية
Arab International Academy

في هذه الحصة سنتعلم

- التعرف على مفهوم HTTP وطلباته (GET و POST)
- التعرف على أداة Postman وطريقة استخدامها لاختبار الـ API
- تنفيذ طلبات GET و POST باستخدام لغة Python
- كيفية إنشاء حساب في OpenAI والحصول على API Key
- تنفيذ طلبات API باستخدام مكتبة openai في Python
- التعرف على منصة Google AI Studio وتجربة Gemini 1.5 مجاناً
- التعرف على منصة Hugging Face واستخدام مكتبة Transformers
- شرح مبسّط لمفاهيم Transformers و Embedding في الذكاء الاصطناعي

مقدمة عن HTTP

- ما هو HTTP؟ ولماذا نحتاجه للتواصل مع الخوادم؟
- "HTTP"هو طريقة يتواصل بها جهازك مع الخادم (server) ليطلب منه معلومات, نحتاجه لأنّه يساعدنا في إرسال الطلبات واستلام البيانات من المواقع.



ما هو Request GET؟

- Request GET هو طلب يُرسل من جهازك إلى الخادم (server) لطلب بيانات بدون تعديل أي شيء على الخادم.

مثال :

عندما تكتب رابط موقع في المتصفح مثل:

https://www.google.com

يقوم المتصفح بإرسال Request GET للسيرفر يطلب فيه تحميل الصفحة

السيرفر يرد عليك بالصفحة وتظهر أمامك.

ما هو Request POST ؟

- Request POST هو إرسال بيانات إلى السيرفر.

مثال :

عندما تقوم بتسجيل حساب جديد في موقع، وتضغط على زر "تسجيل"

يقوم المتصفح بإرسال Request POST إلى السيرفر يحتوي على بياناتك (الاسم، البريد، كلمة السر).

السيرفر يستقبل البيانات ويخزّنها أو يعالجها.

مقارنة بين GET و POST

GET	POST
لعرض بيانات من السيرفر	لإرسال بيانات إلى السيرفر
يظهر البيانات في الرابط	لا يظهر البيانات في الرابط

ما هي أكواد حالة HTTP؟

هي أرقام يرسلها الخادم لتوضيح نتيجة الطلب.

- 200: تم بنجاح
- 404: الصفحة غير موجودة
- 500: خطأ في الخادم (server)

تقييم

سؤال : ما الفرق بين GET و POST؟

أ. كلاهما يرسل بيانات

ب. GET يعرض بيانات و POST يرسل بيانات

ج. لا فرق

د. GET يستخدم للتحميل فقط

تقييم

سؤال : ما الفرق بين GET و POST؟

أ. كلاهما يرسل بيانات

ب. GET يعرض بيانات و POST يرسل بيانات

ج. لا فرق

د. GET يستخدم للتحميل فقط

الإجابة الصحيحة هي : ب. GET يعرض بيانات و POST يرسل بيانات

ما هو API؟

API (واجهة برمجة التطبيقات) هو مثل النادل

ينقل طلبك من التطبيق إلى الخادم

ثم يرجع لك بالبيانات.

مثال :

تطبيق الطقس يطلب حالة الجو من الخادم عبر API،

والخادم يرجّع النتيجة للتطبيق

مصطلحات أساسية يجب معرفتها

- **"Header (رأس الطلب)"** معلومات تُرفق مع الطلب، مثل مفتاح التفويض (Authorization).
- **"Endpoint (نقطة النهاية)"** عنوان URL الذي يُرسل إليه الطلب.

بالإضافة لمراجعة سريعة :

- **"API"** واجهة برمجية تسمح للبرامج بالتواصل مع خدمات أخرى عبر الإنترنت.
- **"Request (طلب)"** رسالة يُرسلها البرنامج إلى الخدمة لطلب بيانات أو تنفيذ أمر.
- **"Response (استجابة)"** رد الخدمة على الطلب المرسل.
- **"API Key (مفتاح API)"** رمز سري يخول المستخدم استخدام الخدمة ويضمن أمانها.
- **"POST (إرسال بيانات)"** نوع من طلبات HTTP يُستخدم لإرسال بيانات إلى الخدمة.

لماذا نستخدم Postman؟

Postman هي أداة تساعد المطوّرين في اختبار الـ API.

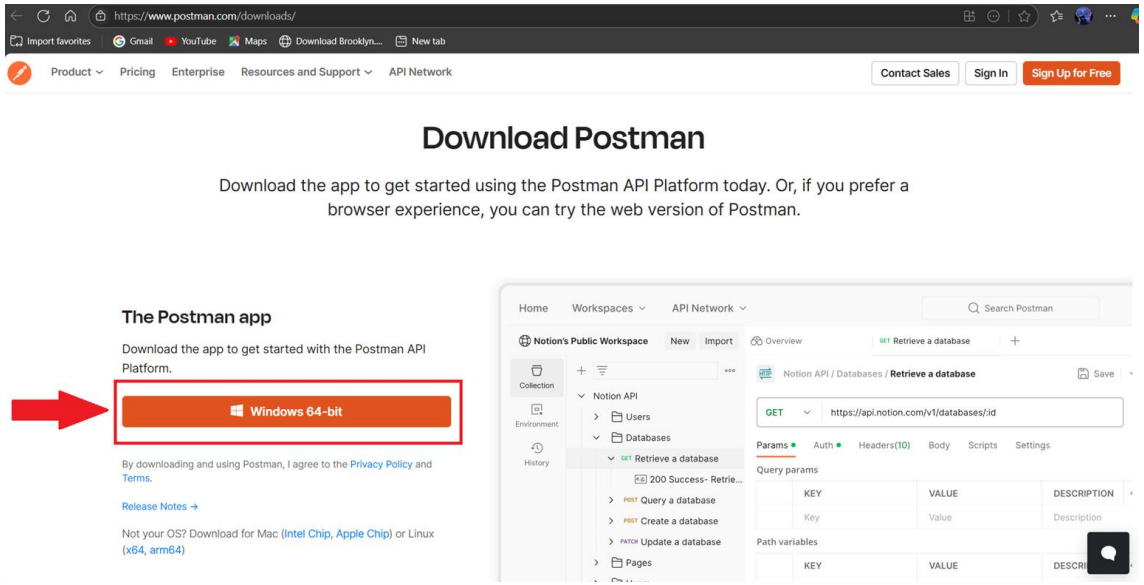
تُمكّنك من إرسال طلبات (مثل GET و POST) إلى الخادم ومشاهدة الرد مباشرة، بدون الحاجة لكتابة كود.

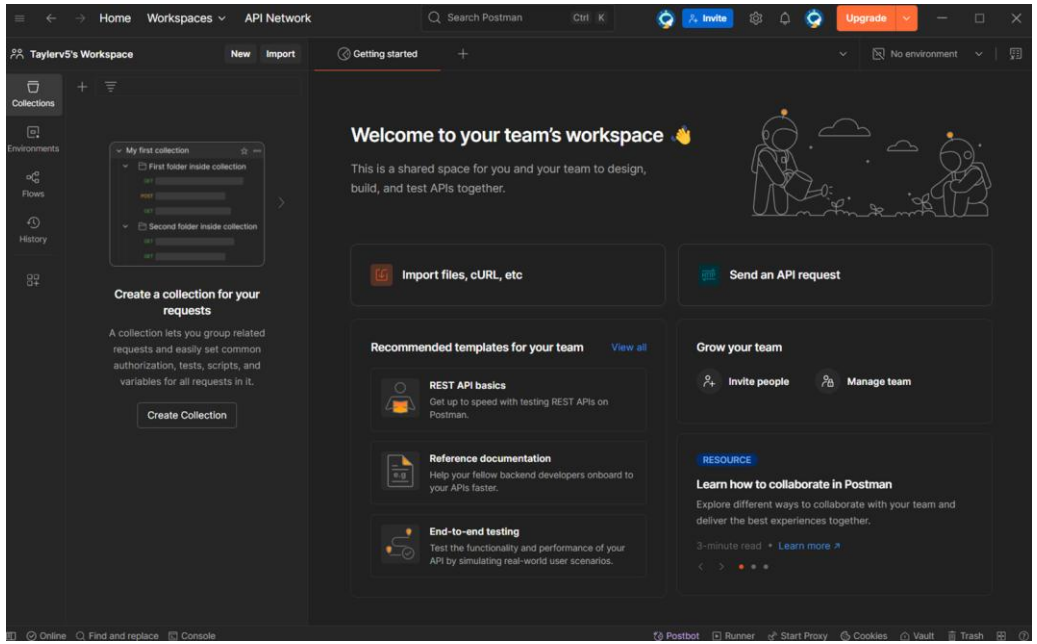
تثبيت Postman

يمكنك تحميل Postman من الموقع الرسمي:

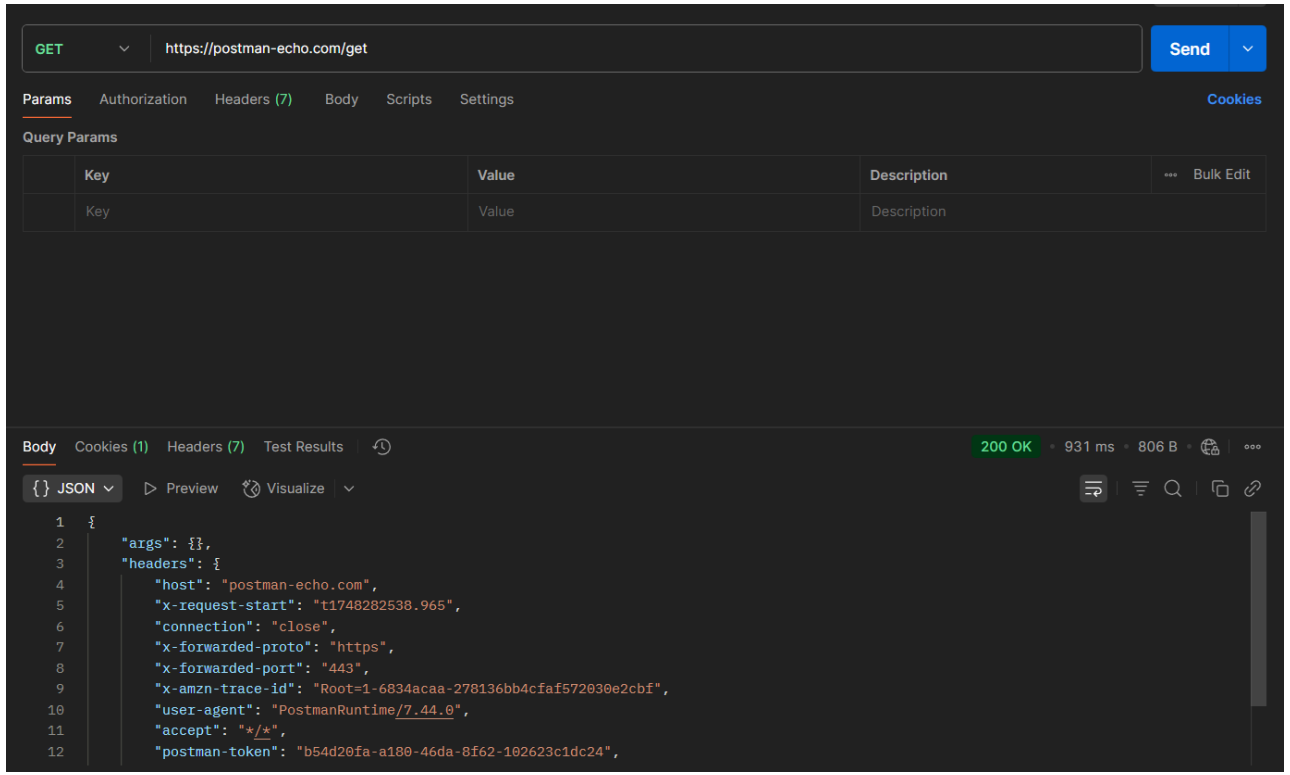
https://www.postman.com/downloads

بعد إكمال التثبيت ، وإنشاء حساب، ستظهر لك هذه الصفحة :





إرسال GET باستخدام Postman

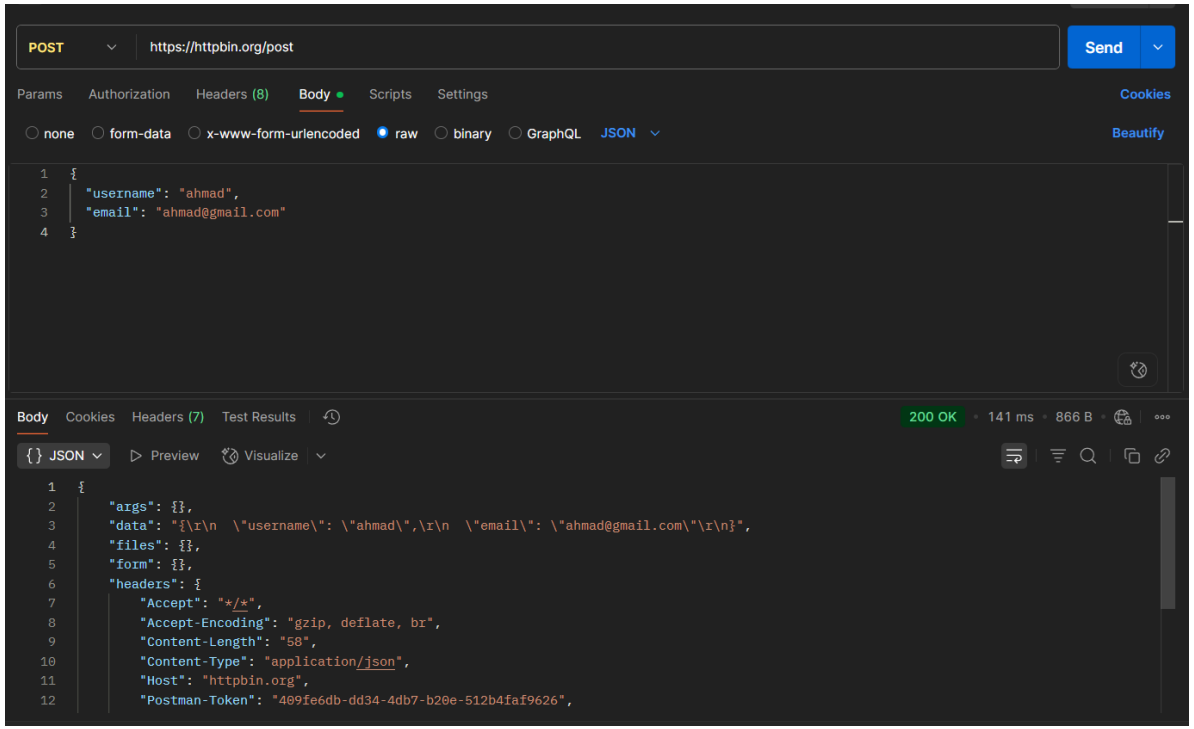


الخطوات:

- افتح تطبيق Postman.
- اضغط على New
- اختر HTTP
- اختر نوع الطلب GET من القائمة.
- أدخل رابط الـ API في خانة الرابط.
- اضغط على زر Send.
- ستظهر النتيجة في الأسفل > نجح الطلب بجلب معلومات من السيرفر .

إرسال POST باستخدام Postman

الخطوات:



- اختر نوع الطلب POST من القائمة.
- أدخل رابط الـ API في خانة الرابط.
- انتقل إلى تبويب Body.
- اختر raw ثم JSON.
- اكتب البيانات التي تريد إرسالها (مثال: الاسم والبريد).
- اضغط على زر Send.
- بعد ما أرسلنا POST Request، الخادم (السيرفر) استقبل هذه البيانات، وأعادها إلينا كجزء من الاستجابة، حتى نتمكن من رؤية ما تم إرساله بالفعل.

قراءة Response من Postman

- بعد إرسال الطلب في **Postman**، يظهر الرد (**Response**) في الجزء السفلي من الشاشة.
- **"Status Code"** رقم يوضح نتيجة الطلب (مثل **200** أو **404**).
- **"Body"** البيانات أو الرسالة التي أرسلها السيرفر، غالبًا تكون بصيغة **JSON**.

مثال :

Status200 :OK

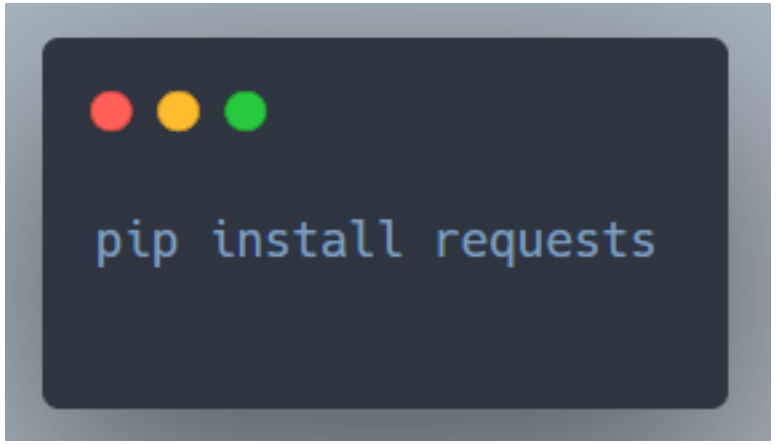
أمثلة API باستخدام Python

مكتبة Request

هي مكتبة في **بايثون** تُستخدم لإرسال الطلبات إلى السيرفر (GET, POST, ...)

طريقة التثبيت:

افتح الطرفية (**Terminal**) واكتب:



مثال GET في Python

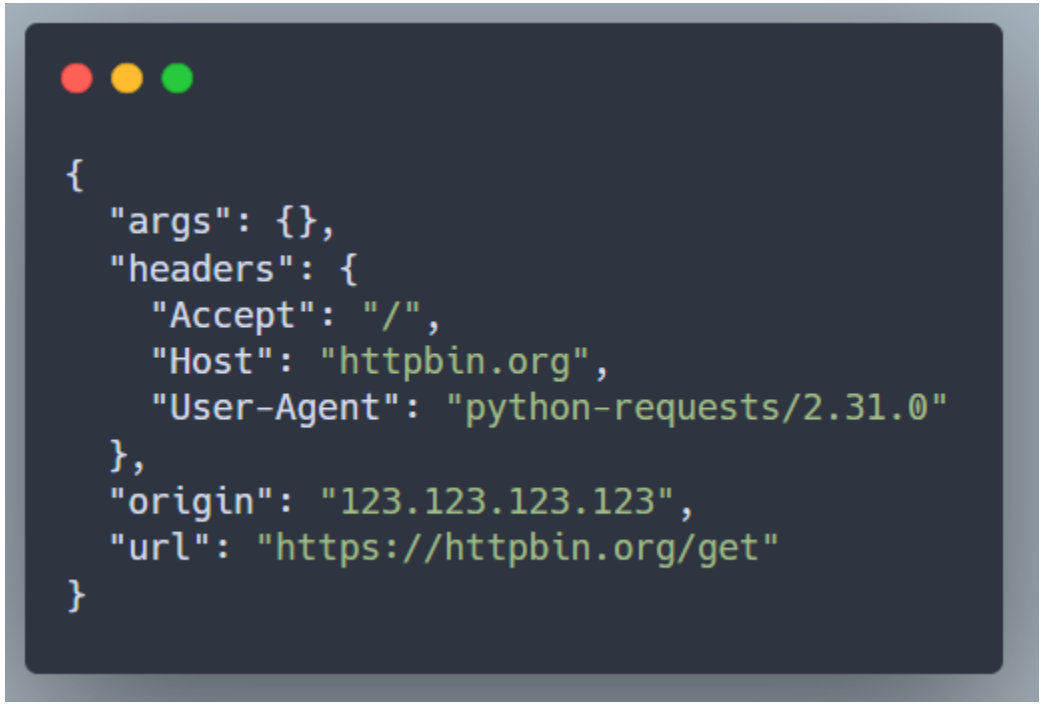
```python
import requests

response = requests.get("https://httpbin.org/get")
print(response.text)
```

- "Import request" استدعاء مكتبة Request.
- "requests.get()" إرسال طلب GET للرابط.
- "response.text" طباعة البيانات اللي رجعت من السيرفر.

```
{
  "args": {},
  "headers": {
    "Accept": "/",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.31.0"
  },
  "origin": "123.123.123.123",
  "url": "https://httpbin.org/get"
}
```

الكود يطلب بيانات من رابط API ويطبعها.

شرح النتيجة من الطلب



```
{
  "args": {},
  "headers": {
    "Accept": "/",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.31.0"
  },
  "origin": "123.123.123.123",
  "url": "https://httpbin.org/get"
}
```

- **"headers"** معلومات عن الجهاز والبرنامج اللذي أرسل الطلب.
- **"origin"** عنوان الـ IP للجهاز اللي أرسل الطلب.
- **"url"** الرابط اللي تم طلبه.
- **"args"** لو فيه بارامترات بالرابط (لا يوجد هنا).

مثال POST في Python

- "Import request" استدعاء مكتبة Request.
- "data" البيانات المُراد إرسالها.
- "requests.post()" إرسال POST مع البيانات بصيغه JSON إلى الرابط Request.
- "response.text" طباعة رد السيرفر.

```python
import requests

data = {
    "username": "ahmad",
    "email": "ahmad@example.com"
}

response = requests.post("https://httpbin.org/post", json=data)
print(response.text)
```

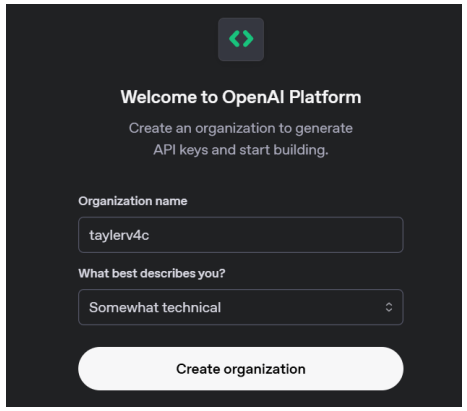
```
"json": {
  "username": "ahmad",
  "email": "ahmad@example.com"
}
```

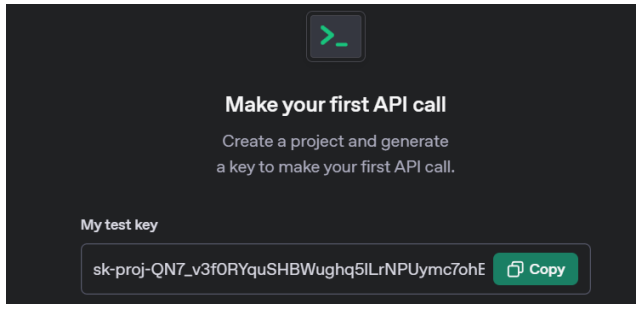
الكود يرسل بيانات تسجيل للسيرفر ويطبع النتيجة.

استخدام ال ChatGpt عن طريق ال Api

- **OpenAI** شركة متخصصة في تطوير ذكاء اصطناعي متقدم يقدم نماذج لغوية تساعد في كتابة النصوص، الترجمة، البرمجة، وغيرها.
- لاستخدام خدمات **OpenAI**، يجب التسجيل في موقعهم الرسمي والحصول على مفتاح API.
- هذا المفتاح هو رمز سري يُستخدم لإرسال الطلبات إلى خدمات **OpenAI**.
- تتوفر خطة مجانية محدودة تسمح بتجربة الخدمة، بعدها يكون الاستخدام مدفوع حسب الاستهلاك.
- بدون مفتاح API لا يمكن إرسال طلبات إلى نماذج **OpenAI**.

التسجيل في OpenAi والحصول على API Key





انتقل إلى موقع **OpenAi** الرسمي وقُم بإنشاء حساب جديد عبر الضغط على **SignUp.**

https://platform.openai.com/docs/overview

- بعد التسجيل في موقع OpenAI، تدخل إلى لوحة التحكم.
- لإنشاء مفتاح API، يُطلب منك اسم شركة أو مشروع.
- لا يشترط أن يكون الاسم حقيقيًا، يمكن اختيار أي اسم.
- تحصل على مفتاح API لاستخدامه في تطبيقاتك.
- احفظ المفتاح جيدًا ولا تشاركه مع أحد.

الآن يمكنك البدء بالكود

1- تجهيز طلب API

استيراد المكتبات :

نستخدم `requests` لإرسال الطلبات عبر الإنترنت، و`json` لتحويل البيانات إلى صيغة مناسبة للإرسال.

تحديد مفتاح API والرابط :

`api_key` هو رمز الأمان الخاص بك، و`url` هو عنوان خدمة OpenAI.

إعداد الرأس (Headers) :

يحتوي على معلومات مهمة مثل التفويض ومحتوى البيانات.

إعداد البيانات (Body) :

تشمل اختيار النموذج، النص المطلوب، والحد الأقصى لطول الرد.

```python
import requests
import json

api_key = "مفتاح API الخاص بك"
url = "https://api.openai.com/v1/completions"

headers = {
    "Authorization": f"Bearer {api_key}",
    "Content-Type": "application/json"
}

data = {
    "model": "text-davinci-003",
    "prompt": "اكتب جملة ترحيبية باللهجة الفلسطينية",
    "max_tokens": 50
}
```

2- إرسال الطلب والتعامل مع الرد

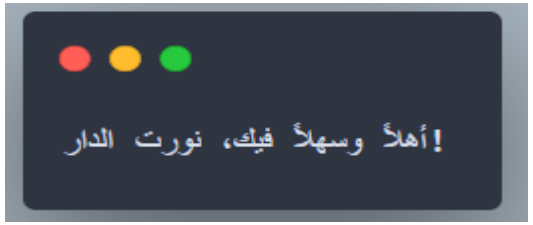
```python
response = requests.post(url, headers=headers, data=json.dumps(data))

if response.status_code == 200:
    print(response.json()['choices'][0]['text'].strip())
else:
    print("حدث خطأ:", response.status_code)
```

إرسال الطلب :

نستخدم طريقة **POST** لإرسال البيانات إلى الخادم.

فحص الرد :

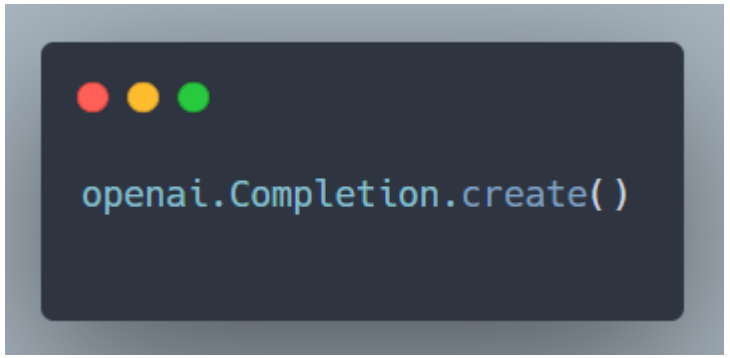
إذا كانت حالة الرد **200** يعني نجاح الطلب، نطبع النص الذي أعادته الخدمة.



التعامل مع الأخطاء :

إذا فشل الطلب، نطبع رمز الخطأ لنعرف المشكلة.

مكتبة OpenAi



- مكتبة جاهزة في Python تسهّل التعامل مع OpenAI API.
- تغني عن كتابة عنوان الرابط (Endpoint)، وHeaders، وBody يدويًا،
- المكتبة توفر لنا دوال (Functions) جاهزة نستخدمها مباشرة.

مثال :

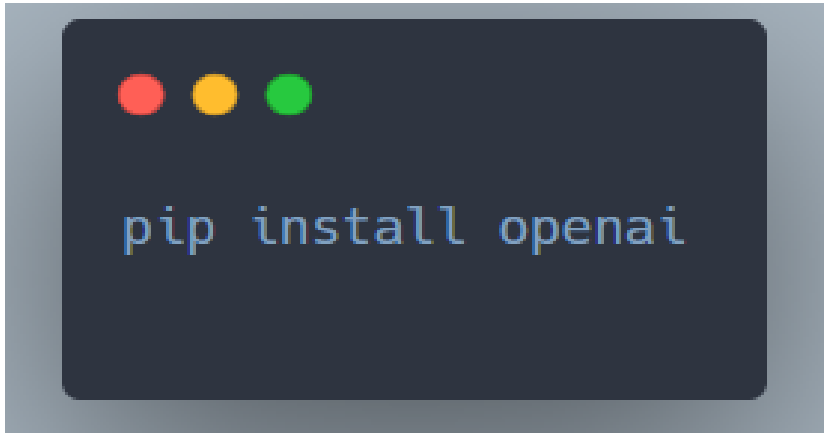
بدل كتابة طلب **POST** كامل، نستخدم :

ملاحظة :

المكتبة مخصّصة للتعامل مع OpenAI API فقط بينما requests يمكن استخدامها مع أي API آخر

تثبيت مكتبة OpenAi

لتثبيت مكتبة **openai** في بايثون، افتح الطرفية **(Terminal)** واكتب:



مثال على استخدام OpenAi

```python
import openai

openai.api_key = "مفتاح API الخاص بك"

response = openai.Completion.create(
  model="text-davinci-003",
  prompt="اكتب لي حكمة جميلة",
  max_tokens=50
)

print(response.choices[0].text)
```

الشرح :

- **"import openai"** استدعاء مكتبة openai.
- **"Completion.create()"** إرسال prompt (طلب نصي) إلى الموديل.
- **"txet.[0response.choices["** طباعة الرد.

الكود يرسل سؤال أو طلب لـ OpenAI، والموديل يرد بنص.

```
"الحياة قصيرة، فلا تهدرها في الحزن، وابتسم لكل شيء جميل من حولك."
```

التعامل مع Response ك Dictionary

عند استخدام مكتبة **openai**، تكون نتيجة الطلب بشكل **قاموس (Dictionary)**.

للوصول إلى النص الناتج، نستخدم :

```
print(response['choices'][0]['text'])
```

```
"افعل الخير مهما كان صغيرًا، فربما يكون كبيرًا عند الله."
```

response['choices'] . للوصول الى قائمة النتائج :

[0] : اختيار أول نتيجة في القائمة.

['text'] : جلب النص من النتيجة.

Gemini oidutS IA elgooG +2.5

ما هو 2.5Gemini؟

2.5Gemini هو نموذج لغوي متقدم من شركة **Google**.

يشبه **ChatGPT**، حيث يمكنه فهم النصوص، كتابة محتوى، تلخيص، والإجابة على الأسئلة.

Google AI Studio

Google AI Studio هو منصة من **Google** لتجربة واختبار نماذج الذكاء الاصطناعي مثل **Gemini** بطريقة سهلة ومباشرة عبر المتصفح.

يمكنك من خلالها :

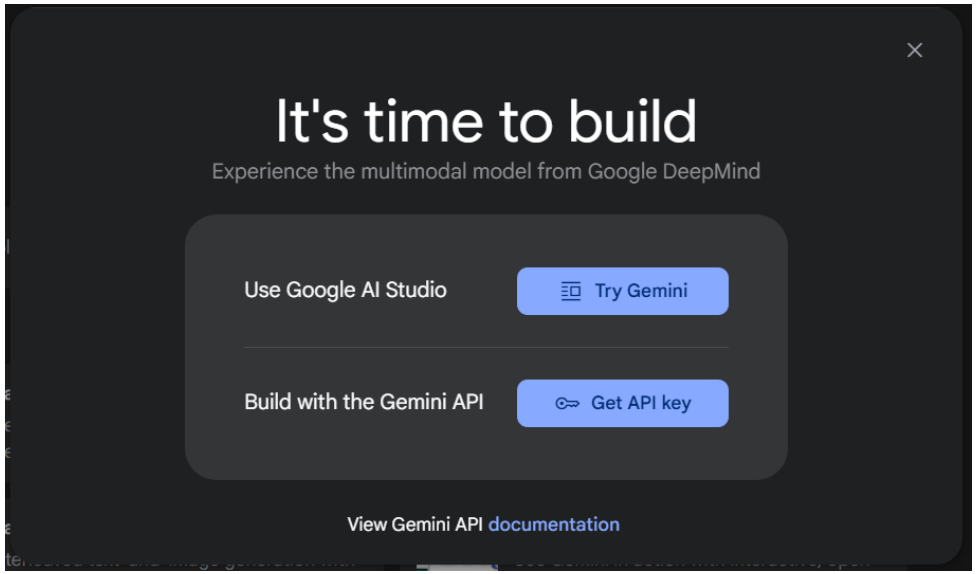
كتابة prompt (طلب نصي) والحصول على رد من النموذج.

اختبار قدرات النماذج المختلفة.

تجربة الذكاء الاصطناعي بدون كتابة كود.

الخلاصة: مكان على الإنترنت لتجربة ذكاء **Google** الاصطناعي بسهولة.

التسجيل في AI Studio



1. اذهب إلى الموقع: https://aistudio.google.com
2. اضغط على **Sign in** بحساب **Google** الخاص بك.
3. بعد التسجيل في الموقع , اضغط على **Try Gemini**
4. من القائمة الجانبية، اختر **Start a New Chat** أو **Start a New Prompt**.
5. اختر نموذج **Flash1.5 Gemini** أو أي نموذج مكتوب عليه **Free**.
6. بذلك يمكنك تجربة الذكاء الاصطناعي مجانًا.

Hugging Face

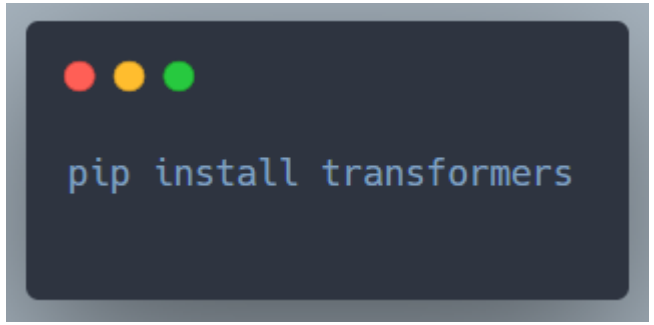
ما هو Hugging Face؟

Hugging Face هو موقع وأداة تساعد الناس على استخدام الذكاء الاصطناعي لفهم ومعالجة اللغة بطريقة سهلة. توفر مجموعة كبيرة من النماذج الجاهزة التي يمكن استخدامها لترجمة النصوص، أو تلخيصها، أو إنشاء نصوص جديدة، وغيرها من المهام. كما تساعد المطورين في بناء برامج ذكية بسرعة وبسهولة.

مكتبة Transformers

مكتبة **Transformers** هي أداة من **Hugging Face** تساعدك على استخدام نماذج الذكاء الاصطناعي الخاصة بفهم اللغة بسهولة في برامجك.Hugging Face هي الشركة والمنصة اللي طورت هذه المكتبة وقدمت نماذج جاهزة للاستخدام.

لتثبيت مكتبة **Transformers** اكتب في **Terminal** :



مثال تحميل نموذج

هذا المثال يوضح كيف تحمل نموذج جاهز من مكتبة Transformers باستخدام `pipeline`، ثم تطلب منه ايولد نص بناءً على جملة تبدأها.

```python
from transformers import pipeline

# تحميل نموذج جاهز لمعالجة النصوص (مثلاً للتوليد)
generator = pipeline('text-generation', model='gpt2')

# تجربة النموذج
result = generator("مرحباً، كيف حالك؟", max_length=30)
print(result)
```

- **pipeline** تسهل عليك استخدام النموذج.
- النموذج هنا اسمه **2gpt** وهو يولد نصوص.
- الكود يطبع النص الذي انتجه النموذج

```
[{'generated_text': 'مرحباً، كيف حالك؟ أنا بخير، شكراً لسؤالك. اليوم كان يوم جميل و...'}]
```

النتيجة تكون قائمة تحتوي على نص مولد من قبل النموذج مثلاً:

```
[{'generated_text': 'مرحباً، كيف حالك؟ أنا سعيد بمساعدتك اليوم...'}]
```

مكتبة Datasets

- هي مكتبة من Hugging Face تساعدك بالحصول على مجموعات بيانات جاهزة ومتنظمة بسهولة لاستخدامها في تدريب واختبار نماذج الذكاء الاصطناعي.

للعثور على النماذج الجاهزة على منصة Hugging Face Hub

1. ادخل إلى موقع https://huggingface.co/models
2. استخدم مربع البحث للبحث عن النموذج الذي تريده (مثل ترجمة، تصنيف، توليد نص).
3. يمكنك تصفية النتائج حسب اللغة، نوع النموذج، أو الاستخدام.
4. عند اختيار نموذج، تجد تفاصيل عنه وطريقة استخدامه مباشرة في الصفحة.
5. يمكنك تحميل النموذج أو استخدامه عبر مكتبة Transformers بسهولة.

Transformers (Encoding –Decoding –Embedding)

ما هو Transformer؟

Transformer هو نوع من نماذج الذكاء الاصطناعي صُمم لمعالجة وفهم النصوص بطريقة ذكية وفعّالة. يعتمد هذا النموذج على آلية تُسمّى **الانتباه (Attention)**، حيث يمكنه التركيز على جميع كلمات الجملة في نفس الوقت، ومعرفة العلاقة بين كل كلمة وأخرى.

يُستخدم Transformer في العديد من المهام مثل الترجمة، تلخيص النصوص، تصنيف المحتوى، وتوليد نصوص جديدة. وهو يُعد الأساس الذي بُنيت عليه أشهر نماذج الذكاء الاصطناعي الحديثة مثل BERT وGPT.

النصوصEncoding و الأرقام Decoding

- عند التعامل مع الذكاء الاصطناعي، لا يمكن للنماذج فهم النصوص بشكلها العادي، لذا يتم تحويل النصوص إلى أرقام. هذه العملية تُسمى **Encoding**.
  - هذا التحويل ضروري لأن الحاسوب يتعامل فقط مع الأرقام، وليس مع النصوص مباشرة.

- بعد أن يُحوِّل النموذج النص إلى أرقام لمعالجته، يمكن إعادة هذه الأرقام إلى نصوص مفهومة.
  - تسمى هذه العملية **Decoding**، وهي ببساطة تحويل الأرقام (tokens) إلى الكلمات الأصلية، بحيث يظهر النص من جديد بشكل طبيعي يمكن قراءته.

ما هو ال Embedding؟

هو تحويل الكلمات إلى أرقام بطريقة ذكية تساعد الحاسوب على فهم معناها, أي أن كل كلمة تُمثّل بمجموعة أرقام تعبر عن معناها وعلاقتها بالكلمات الأخرى, وبذلك يتمكّن النموذج من التمييز بين الكلمات التي تحمل معانٍ متقاربة.

شكرا لكم