Création d'une base de données pour une application de gestion d'étudiants

Objectif :

Développer la partie "*Back-end*" d'une application en ligne permettant de gérer une liste d'étudiants.

Travail demandé :

1. Tester les 4 cas d'utilisation du **CRUD** avec les commandes **SQL** décrites dans le cours dans un "bac à sable" (https://mysql-sandbox.nullx.me/)
2. Reprendre ces tests avec **MySQL** en local. On utilisera l'interface graphique de **phpMyAdmin** (http://127.0.0.1/phpmyadmin)
3. Après avoir défini un utilisateur spécifique à la base de données pour un accès externe (port **3306**), coder une application en langage **Python** pour accéder à ces données.
4. Faire évoluer cette application pour qu'elle soit capable de répondre à des requêtes **HTTP** et renvoyer les résultats demandés en **JSON**
   (cf doc du TP front-end : Utilisation d'une API REST)
   Cela nécessite l'installation du serveur Flask pour Python

Librairies utilisées :

1. *mysql.connector* : Connexion à la base de données (https://www.w3schools.com/python/python_mysql_getstarted.asp)

```python
import mysql.connector

mydb = mysql.connector.connect(
  host = "127.0.0.1",
  user = "root",
  password = "",
  database = "ciel2025"
)

cursor = mydb.cursor()
request = "SELECT * FROM etudiant"
cursor.execute(request)
result = cursor.fetchall()

for record in result:
  print(record)
```

flask : Serveur web

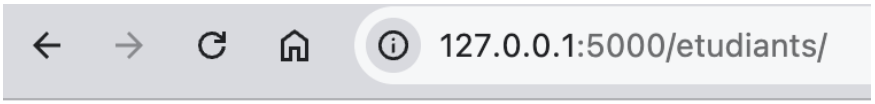
Il faut tout d'abord créer un environnement virtuel pour votre application : https://code.visualstudio.com/docs/python/environments

Une fois *flask* installé dans cet environnement virtuel, on peut coder un programme simple avec 2 routes (URL) :

```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return 'Page d\'accueil'

@app.route('/etudiants/')
def about():
    return 'Page etudiants'
```

Après avoir lancé **Flask** et l'exécution du programme en Python, on peut se connecter à ce serveur en **local** sur le port **5000** :

Page d'accueil



Page etudiants

Intégration finale :

Pour réaliser notre API, l'application backend va utiliser flask pour router les requêtes http vers des fonctions spécifiques, et mysql pour exécuter dans chacune de ces fonctions la requête SQL appropriée.

Ainsi, pour récupérer la totalité de la table etudiant dans un flux JSON, on routera la requête HTTP http://127.0.0.1:5000/etudiants vers la fonction ***getEtudiants()*** qui présente maintenant les données sous forme de dictionnaire (clé:valeur) :

```python
@app.route('/etudiants', methods=['GET'])
def getEtudiants():
    etudiants = []
    request = "SELECT * FROM etudiant"
    cursor.execute(request)
    result = cursor.fetchall()
    for row in result:
        etudiant = {
            "idetudiant": row[0],
            "nom": row[1],
            "prenom": row[2],
            "email": row[3],
            "telephone": row[4]
            }
        etudiants.append(etudiant)
    return jsonify(etudiants), 201
```

Pour ne récupérer les données que d'un seul étudiant, nous écrirons une nouvelle fonction ***getEtudiants(id)*** qui reçoit en paramètre l'id de cet étudiant, qui sera appelée par la requête HTTP http://127.0.0.1:5000/etudiant/{id}

```python
@app.route('/etudiant/<int:id>', methods=['GET'])
def getEtudiant(id):
    request = "SELECT * FROM etudiant WHERE idetudiant = " + str(id)
    print (request)
    cursor.execute(request)
    row = cursor.fetchone()
    etudiant = {
        "idetudiant": row[0],
        "nom": row[1],
        "prenom": row[2],
        "email": row[3],
        "telephone": row[4]
    }
    return jsonify(etudiant)
```

*** Document à compléter ***