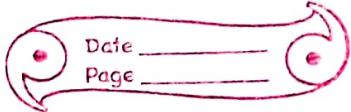


API (flask)



① API → An api is a Software intermediary that allows two or more application to talk to each other.

(Application Programming Interface)

② Types of API —

- Private - used within the organization.
- Partner - used within Business Partner.
- Public - used any third party developers.

③ `>>> from flask import Flask`
`>>> app = Flask(__name__)`

`>>> app = Flask(__name__)` this is a WSGI application
to connect between web server & web application

④ `@app.route('/')` → The decorator
`def welcome():`
`return "Hello world"`

WSGI → Web Server Gateway Interface

⇒ WSGI is the internal interface which will help the servers to connect with web applications.

Jinja2 - web Templating System

```
>>> from flask import Flask  
>>> app = Flask(__name__) # wsgi application  
>>> @app.route('/') # decorator for url  
>>> def welcome():  
>>>     return "Hi my name is"  
>>> @app.route('/member')  
>>> def member():  
>>>     return "what is the time"  
>>> if __name__ == "__main__":  
>>>     app.run(debug=True)
```

flask use werkzeug & jinja framework internally.

WSGI → Web Server gateway interface
→ used to call some function or request
→ to other program to run/execute

Q How will you differentiate between GET & POST

Ans) When we will send a data inside the browser it is GET (e.g. Google Search) and when we will send a data through body is POST (e.g. login in (log-in email))

HTTP is a protocol between programming language & a program (API) to call.

GET → Send data in URL + Query (visible)
POST → Send data in body, (people will not able to see it) (Secured - method)

~~HTTP has 200 & 500~~ → 200 = working fine.

200 in HTTP means working the URL,
500 in HTTP means Internal Server Error

```
>>> from flask import Flask, render_template, request,  
     jsonify
```

```
>>> app = Flask(__name__)
```

```
>>> @app.route('/via-postman', methods=['POST'])
```

```
>>> def math_operation():  
    if (request.method == 'POST'):  
        operation = request.json['operation']  
        num1 = int(request.json['num1'])  
        num2 = int(request.json['num2'])
```

```
        if operation == 'add':
```

```
            r = num1 + num2
```

```
            result = str(r)
```

```
    return jsonify(result)
```

```
>>> @app.route('/sums', methods=['POST'])
```

```
>>> def sum():
```

```
    if (request.method == 'POST'):
```

```
        a = request.json['num1']
```

```
        b = request.json['num1']
```

```
        c = request.json['num2']
```

```
    result = a * b * c
```

```
    return jsonify(result)
```

```
>>> if __name__ == "__main__":  
    app.run()
```

— MongoDB —

- # It is a NoSQL database in which we don't need a structured format like SQL to store it.
- # We have to install pymongo which is a driver and make a account in mongo db.
- # PyMongo - is the official Python driver that connects to and interact with MongoDB databases.

Code - (written in ES6V)

```

① >>> ! pip3 install pymongo [ES6V] # to install pymongo
② >>> # after this copy the code from the mongo db to
       connect with python.

>>> import pymongo
>>> client = pymongo.MongoClient("-----")
>>> db = client.test
>>> client.list_database_names() # to check the
       no. of list of databases.

```

	<u>MySQL</u>	<u>Mongo db</u>
# Database	Table	Collection
# Row / record	Document	Object

Date _____
Page _____

client

Code ④ db2 = Client("sudh") # to create database

Code ⑤ coll = db2['ineuron-collection'] # to create table/collection inside database

Code ⑥ dict1 = { 'name': 'sumay', 'password': '123456', 'email': 'sumay@123@gmail.com', 'company': 'iNeuron' }

If this is the data to store

Code ⑦ coll1.insertOne(dict1) # to pass the data to store in MongoDB.

Q What is document in MongoDB -

A) The single or multiple record is known as document and (is equivalent to one single record in MySQL which is table.)

To fetch a record in Python itself -

>> coll1.find()

Code ⑧ for i in coll1.find():
 print(i)] => to get all data from MongoDB

Code 9) `>>> for i in coll1.find({ "name": "raja" }):
 print(i)`

It will help to find the certain Key pair Value.

Code 10) H Now if we are get confused with name or the spell So we can get the value after giving `"$in"`.

`>>> for i in coll1.find({ "name": { "$in": ["raj", "raja", "ram"] } }):
 print(i)`

this will help to get the desire output from the MongoDB.

Code 11) `>>> for i in coll1.find({ "name": { "$in": ["sum", "sumay"] } },
 {"class": 1}):
 print(i)`

Note - `$in` - in the same list of values

`"$gt"` = greater than

`"$lt"` = lesser than

`"$lte"` = lesser than or equal to

`"$set"`, `"$inc"` = to update the value

to update or change -

Code 12 >>> coll1.update_many({ "name": "raju" }, { "\$set": { "name": "raja bro" } })

Code 13 >>> for i in coll1.find().limit(6):

print(i)

to get the 6 record only -

Code 14 # not greater than -

>>> coll1.find({ "salary": { "\$not": { "gt": 10000 } } })

Code 15 # not greater than equal to -

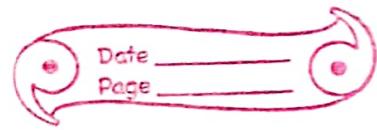
>>> coll1.find({ "salary": { "\$not": { "gte": 10000 } } })

Code 16 # Suppose if i have say that where there is
a Class 12, change the Section as a 'd' -

>>> coll1.find_one_and_update({ "class": "12" }, { "\$set": { "Sec": "d" } })

Code 17 # delete

>>> coll1.delete_many({ "class": "11" })



Q) In mongoDB we get a data in form of document which we have to pass through dictionary or JSON.

SQL with Python

Date _____
Page _____

» ! pip install mysql-connector-python

Ques 1 # to connect between Python & SQL

»»» import mysql.connector as connection

»»» mydb = connection.connect(host="localhost",
user="root", password="123456", use_pure=True)

»»» query = "SHOW DATABASES"

»»» cursor = mydb.cursor()

»»» cursor.execute(query)

»»» print(cursor.fetchall())

»»» except Exception as e:

»»» mydb.close()

»»» print(str(e))

— Starting from here —

Ques 2 # to see all database :-

»»» cursor.execute("SHOW DATABASES")

»»» cursor.fetchall()

Ques 3 : Now to connect with database (Py+SQL)

»»» conn = connection.connect(host="localhost",
user="root", password="123456", use_pure=True)

»»» cur = conn.cursor() # it is a pointer.

H Now whatever we will do with the help of pointer - ~~pointer~~ - ~~cursor~~ - cursor will give us

```
>>> cur.execute("Show databases")
>>> cur.fetchall()
```

Code 3 # to create new database and to see of fetch it -

```
>>> cur.execute("create database Sumay123")
>>> cur.execute("show database")
>>> cur.fetchall()
>>> conn.is_connected()
```

So cur is a pointer which creates a database Sumay123 now we will create table.

Now lets create creating a database then giving table name and then again closing the database.

Creating database -

Code 4 >>> conn = connection.connect(host='localhost', database='Sumay123', user='root', password="123456", port=3306)

code 5
- - -
```>>> cur = conn.cursor()  
>>> cur.execute("Show database")  
>> cur.fetchall()  
>>> conn.is\_connected() # to see the connection True / False

# After this create a table inside database -

code 6  
```>>> cur.execute("create table test (x1 INT(5), x2  
VARCHAR(20), x3 DATE)")

code 7
```>>> cur.close()

∴ This line we have make database and table  
and close also.

∴ Now we have to do again Code no 4 & 5  
because we have close it in code No 7 so  
we will do code no 4 & 5 again

code 8  
```>>> cur.execute("insert into test values(1452,'suman',  
"2023-03-15")")

```>>> conn.commit()

→ Result our test is a table in the database SumanDB

Note :- to save data into database or table we  
have to pass commit() of the connection  
- So, conn is connecting our,

# to check the record (table name only)  
cursor.execute ("select \* from test")

code 9 => cur.execute (''select \* from test'')  
=> cur.fetchall ()

code 10 => to see each row in list as row with list

=> cur.execute (''select \* from test'')

{= > for i in cur.fetchall (): print (i)}

code 11 # To See particular column - ( ) what we do

=> cur.execute ("Select x1,x2 from test")

=> forming in cur.fetchall () and add  
print (i)