API (Flask)

Date
Page

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 with in the Organization.
- Partner - used with in their Business Partners.
- Public - used any third party developers.

③ >>> from flask import Flask
(Package name) (class) (collection of function)
>>> app = Flask(__name__)

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

④ @app.route('/') # 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 application.

Jinja2 - Web Templating System

```
>>> from flask import Flask
>>> app = Flask(__name__) # wsgi application
>>> @app.route('/')      # decorater 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
⇒ use to call some function or request to other program to run/execute

Q How will you differntiate between get & post

Ans) When we will send a data inside the browser it is GET (eg- google search) and when we will send a data through body is POST (eg-login and (log-in email)

http is a protocal between programming language & a portmal (API) to call.

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

~~http/200~~ ~~200~~
200 in http is mean 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('/sumas', methods=['POST'])
>>> def sum():
        if (request.method == 'POST'):
            a = request.json['num0']
            b = request.json['num1']
            c = request.json['num2']
        result = a*b*c
        return jsonify(result)
>>> if __name__ == '__main__':
        app.run()
```

- MongoDB -

It is a No SQL database in which we don't need a structer formate 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 connect to and interact with MongoDB databases.

Code —

(Srv)

① >>> pip install pymongo[srv] # to install Pymongo

② >>> # after this copy the code from the mongodb to connect with Python.

>>> import pymongo
>>> client = pymongo.MongoClient("...")
>>> db = client.test

③ >>> client.list_database_names() # to check the no. of list of databases.

#	MySQL	Mongo db
	Database	Data base
	table	Collection
	row / record	Document.

client

Coding ④ db2 = Client["sudh"] # to create database

Coding ⑤ coll1 = db2["ineuron_collection"] # to create table/collection/inside database.

Coding ⑥ dict1 = {'name': 'sumay',
'email': 'sumay@123gmail.com',
'company': 'iNeuron.ai'
}

this is the data to store

Coding ⑦ coll1.insert_one(dict1) # to pass the data to store in Mongo DB.

Q) what is document in MongoDB -

Ans) The Single or multiple record is known as document and (is equivalent to one Single record to MySQL which is table.)

To fetch a record in Python itself -

>>> coll1.find()

Coding ⑧ >>> for i in coll1.find():
print(i)] ⇒ to get all data from Monogodb

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

It will help to find the certain key pair value.

Code (10) # 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": ['sun', "sunay"]},
"class": "11"}):
print(i)

Note :- "$in" = in
"$gt" = greater than
"$lt" = lesser than
"$lte" = lesser than or equal to
"$set", "$$et" = 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":1000}}})

Code (16) # Suppose if i have say that where then 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"})

Date
Page

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

SQL with Python

```
>>> ! pip install mysql-connector-python
```

ode1 # to connect between Python & SQL -

```
>>> import mysql.connector as connection
>>> try:
>>>     mydb = connection.connect(host="localhost",
connect database      user="root", passwd="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 —

ode2 # to see all database :-

```
>>> cursor.execute("SHOW DATABASE")
>>> cursor.fetchall()
```

ode3 : How to connect with database (Put+SQL)

```
>>> conn = connection.connect(host="localhost",
connection database)   user="root", passwd="123456", use_pure=True)
>>> cur = conn.cursor() # it is a pointer
```

Now whatever we will do with the help of pointer—

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

Code 3 # to create new database and to see/fetch it—

```
>>> cur.execute("create database Sumanji23")
>>> cur.execute("show database")
>>> cur.fetchall()

>>> conn.is_connected()
```

∴ So cur is a pointer which creates a database Sumanji23 now we will created table.

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

Creating database—

Code 4 >>> Conn connection.connect(host='localhost', database="Sumanji23", user="root", password="12456", use-pure

code5
```
>>> cur = conn.cursor()
>>> cur.execute("show database")
>>> cur.fetchall()
>>> conn.is_connected() # to esee the connection True/False
```

After this create a table inside database -

code6 `>>> cur.execute("create a table test(x1 INT(5), x2 VARCHAR(20), x3 DATE)")`

code7 `>>> cur.close()`

∴ This time 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

→ table name

code8
```
>>> cur.execute("insert into test values(1452,'sumanj',
                "2023-03-15")")
>>> conn.commit()
```
→ Remember our test is a table in the database Sumanj13

Note :- to save data into database of table we have to pass commit() of the connection -so . conn is connection over.

to check the record

Code 9 >>> cur.execute ('select * from test') → table name only
>>> cur.fetchall()

Code 10 # to see in loop:
>>> cur.execute ('Select * from test')
>>> for i in cur.fetchall():
>>> print (i)

Code 11 # To see particular columns -
>>> cur.execute ("Select x1, x2 from test")
>>> for i in cur.fetchall():
print (i)