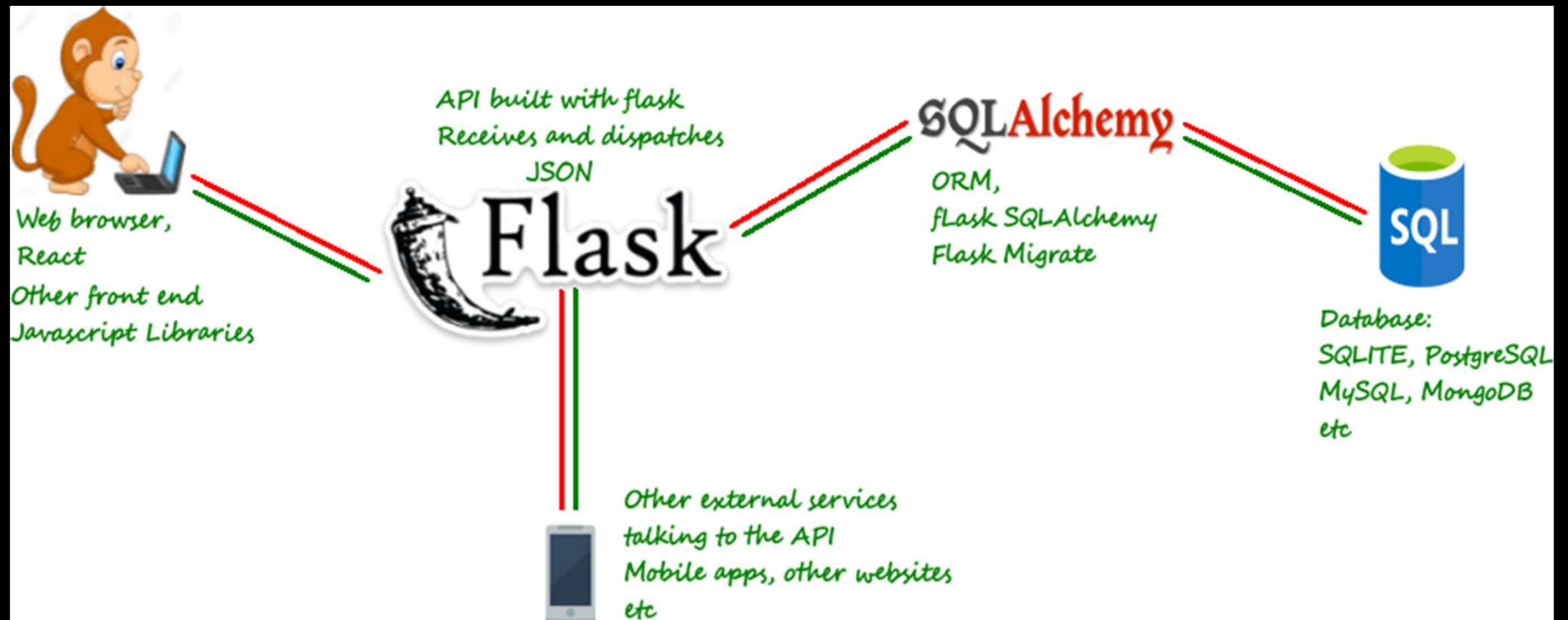




RECAP

APPLICATION TIER PYTHON
FLASK



FLASK RESTAPI COMPLETE OVERVIEW

TYPICALL CLEINTS OFF APPLICATION TIER [FLASK REST API]

- WEB APPLICATION [EITHER FROM SELF PHONES OR PC'S]
- SELF PHONE APPLICATIONS
- OTHER APPLICATION TIERS [BACKEND APPLICATIONS]

BUT HOW TO TEST THE APPLICATION TIER DURING DEVELOPMENT ?



END TO END TESTING OF APPLICATION TIER [FLASK REST API]

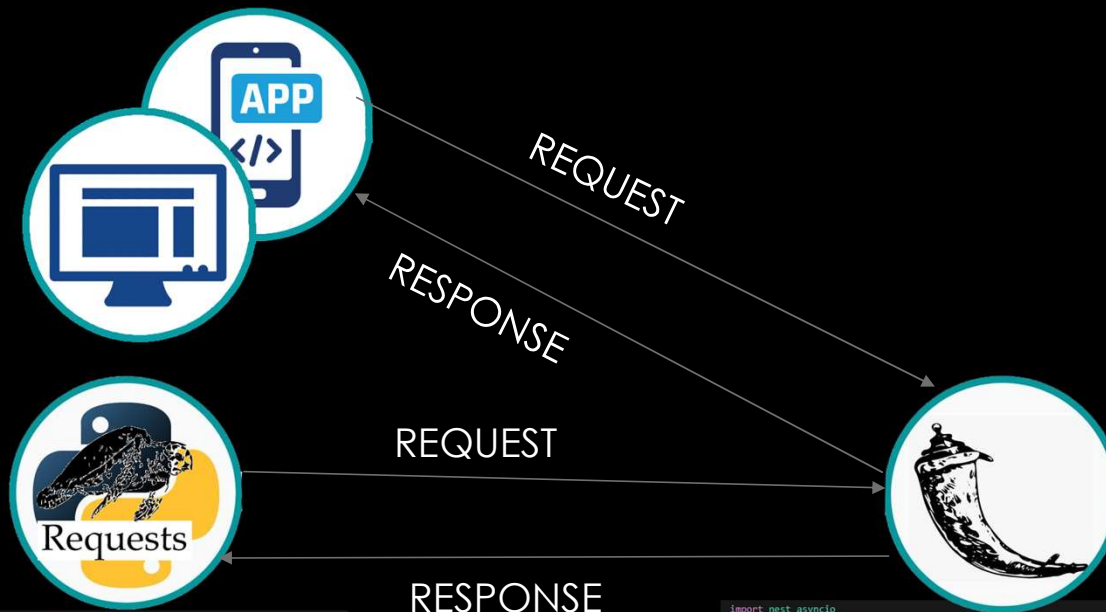
1. PYTHON REQUESTS
LIBRARY

2. SWAGGER

3. POSTMAN



FLASK RESTAPI



```
# Create comment /comment method POST
comment_post_url = f"{BASE_URL}/comment"
comment_payload = {"id": 3, "content": "some", "post_id": 1}

response = requests.post(url=comment_post_url, json=comment_payload)

status_code = response.status_code

if status_code == HTTPStatus.BAD_REQUEST:
    display(response.json())

if status_code == HTTPStatus.CONFLICT:
    display(response.json())

if status_code == HTTPStatus.CREATED:
    display(response.json())

# get posts /comment method GET
comment_id = 3
comment_get_url = f"{BASE_URL}/comment/{comment_id}"

response = requests.get(url=comment_get_url)
```

```
import nest_asyncio
from flask import Flask, jsonify, request
from werkzeug.serving import run_simple

nest_asyncio.apply()

app = Flask(__name__) # Application Tier app

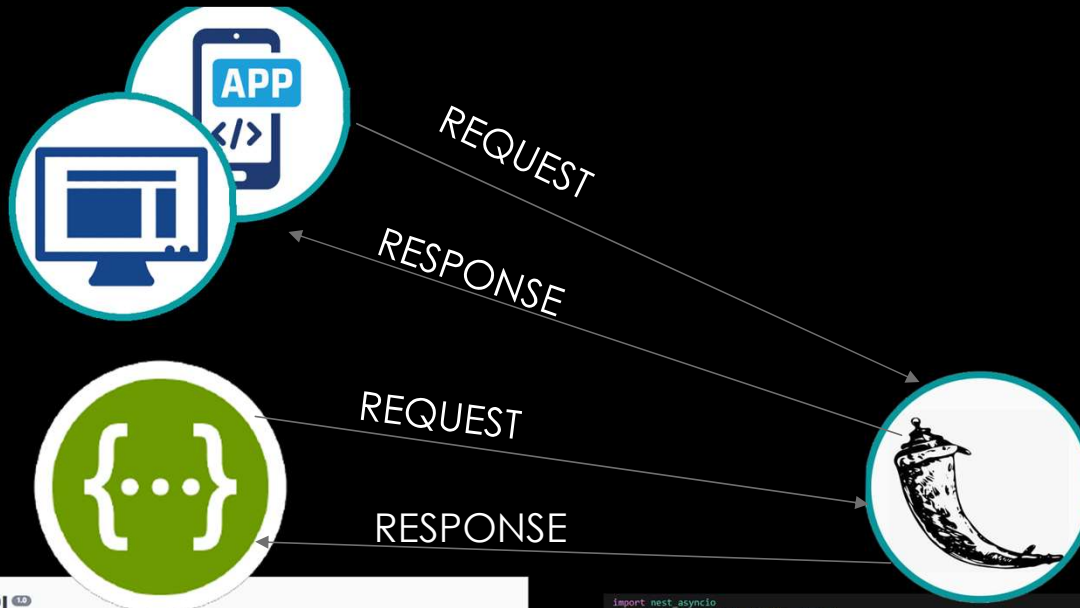
# http://localhost:5000/hello
@app.route("/hello", methods=["GET"])
def hello_world():
    message_data = {"message": "Hello, World!"}

    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} -> jsonify(data) = {"message": "Hello, World!"}

# http://localhost:5000/greet/adam -> greet(name="adam") # we call this endpoint with path variable
@app.route("/greet/<string:name>", methods=["GET"])
def greet(name: str):
    message_data = {"message": f"Hello, {name}!"}
    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} -> jsonify(data) = {"message": "Hello, Adam!"}

# http://localhost:5000/query
# http://localhost:5000/query/name=adam
# http://localhost:5000/query/name=adam&age=37
@app.route("/query", methods=["GET"])
def query():
```

FLASK RESTX REST API



Swagger enabled API

[Base URL: /]
<http://localhost:5000/swagger.json>

A simple Quotes API

quotes Quote CRUD operations

POST /quotes/ Create a new quote

GET /quotes/ List all quotes

DELETE /quotes/{id} Delete a quote given its identifier

GET /quotes/{id} Fetch a quote given its identifier

PUT /quotes/{id} Update a quote given its identifier

```
import nest_asyncio
from flask import Flask, jsonify, request
from werkzeug.serving import run_simple

nest_asyncio.apply()

app = Flask(__name__) # Application Tier app

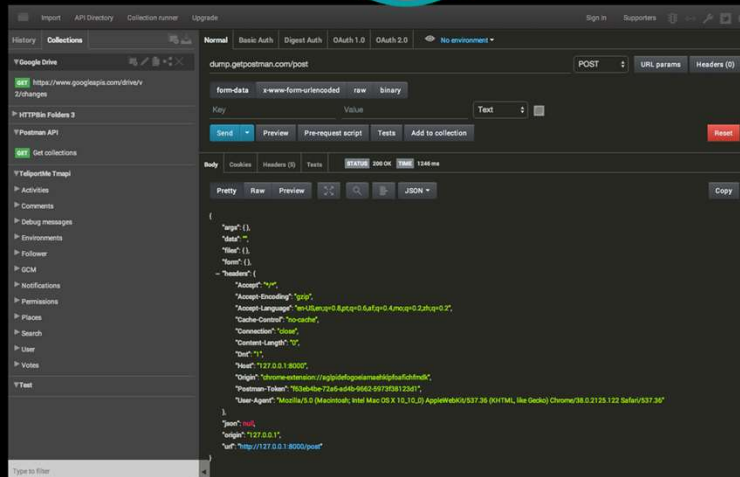
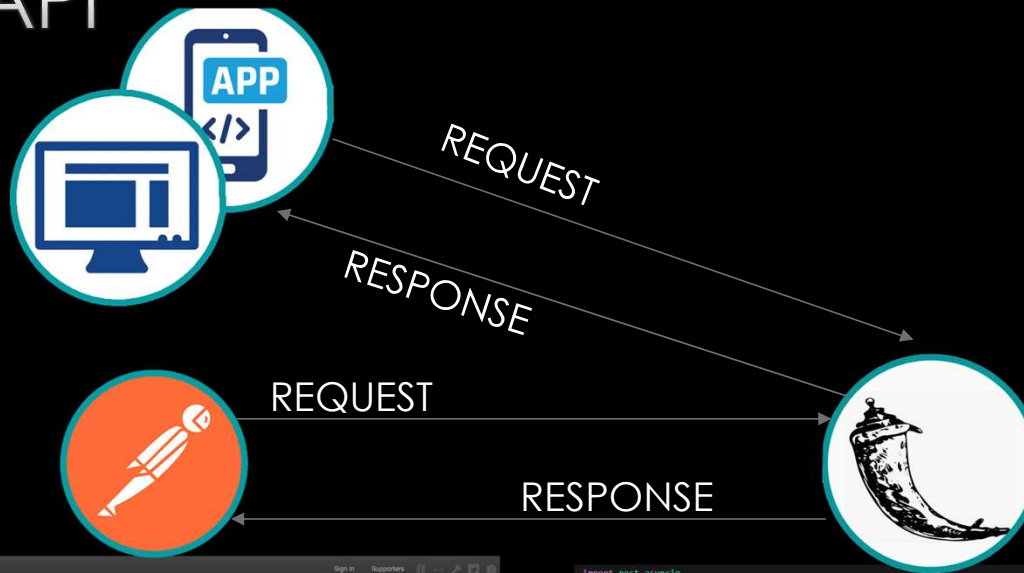
#http://localhost:5000/hello
@app.route("/hello", methods=["GET"])
def hello_world():
    message_data = {"message": "Hello, World!"}

    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} -> jsonify(data) = {"message": "Hello, World!"}

# http://localhost:5000/greet/adam -> greet(name="adam") # we call this endpoint with path variable
@app.route("/greet/<string:name>", methods=["GET"])
def greet(name: str):
    message_data = {"message": f"Hello, {name}!"}
    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} -> jsonify(data) = {"message": "Hello, Adam!"}

# http://localhost:5000/query
# http://localhost:5000/query/name=adam
# http://localhost:5000/query/name=adam&age=37
@app.route("/query", methods=["GET"])
def query():
```

FLASK RESTAPI



```
import nest_asyncio
from flask import Flask, jsonify, request
from werkzeug.serving import run_simple

nest_asyncio.apply()

app = Flask(__name__) # Application Tier app

@app.route("/hello", methods=["GET"])
def hello_world():
    message_data = {"message": "Hello, World!"}
    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} => jsonify(data) = {"message": "Hello, World!"}

# http://localhost:5000/greet/adam => greet(name="adam") # we call this endpoint with path variable
@app.route("/greet/<string:name>", methods=["GET"])
def greet(name: str):
    message_data = {"message": f"Hello, {name}!"}
    return jsonify(message_data), 200 # data = {"message": "Hello, World!"} => jsonify(data) = {"message": "Hello, World!"}

# http://localhost:5000/query
# http://localhost:5000/query?name=adam
# http://localhost:5000/query?name=adam&age=37
@app.route("/query", methods=["GET"])
def query():
    name = request.args.get("name", "Guest") # if name is none then name = Guest
    age = request.args.get("age")

    data = {"message": f"Hello, {name}!"}

    if age: # if age is not none
        data = {"message": f"Hello, {name}! You are {age} years old."}

    return jsonify(data), 200
```