

DATABASE SYSTEMS AND CLOUD COMPUTING

3rd Homework Assignment

Due on: January 9, 2023

In this homework, you will implement the REST API (swagger.json) in Python using the following modules:

- eventlet
- Flask
- flask-cors
- flask-socketio
- **pymongo** (You will use MongoDB in the homework)
- requests
- schedule
- websockets
- flask-swagger-ui
- kafka-python

You can use the following template for the implementation:

```
from flask import Flask, jsonify, request
from flask_cors import CORS
from flask_socketio import SocketIO
from pymongo import MongoClient
from flask_swagger_ui import get_swaggerui_blueprint

"""
Flask/SocketIO/CORS configuration
"""

api = Flask(__name__)
api.config["DEBUG"] = True
cors = CORS(api)
socketio = SocketIO(api, cors_allowed_origins="*")

"""
mongodb
pymongo configuration
"""

mongo_client = MongoClient("mongodb://localhost:27017")
hr_db = mongo_client["crm"] # use crm
customers = hr_db.customers # collection name: customers

SWAGGER_URL = '/api/docs'
API_URL = '/static/swagger.json'

apiDoc = get_swaggerui_blueprint(
    SWAGGER_URL,
    API_URL,
    config={
        'app_name': "CRM"
    }
)
```

```
api.register_blueprint(apiDoc, url_prefix=SWAGGER_URL)

fields = [
    "_id", "firstName", "lastName", "photo", "email", "birthYear", "phones"
]

@api.route("/crm/api/v1/customers", methods=["GET"])
def get_customers():
    pass
    # TODO: get customers by pagination
    # TODO: return customers in the page as JSON

@api.route("/crm/api/v1/customers/<identity>", methods=["GET"])
def get_employee_by_identity(identity):
    pass
    # TODO: get customer by identity
    # TODO: return found customer as JSON

@api.route("/crm/api/v1/customers", methods=["POST"])
def add_customer():
    # TODO: create new customer
    # TODO: create an event and send it through Websocket and Kafka
    # TODO: Event: {"eventType": "CUSTOMER_ACQUIRED", "eventData": customer}
    return jsonify({"status": "ok"})

@api.route("/crm/api/v1/customers/<identity>", methods=["PUT"])
def update_customer(identity):
    # TODO: update customer
    return jsonify({"status": "ok"})

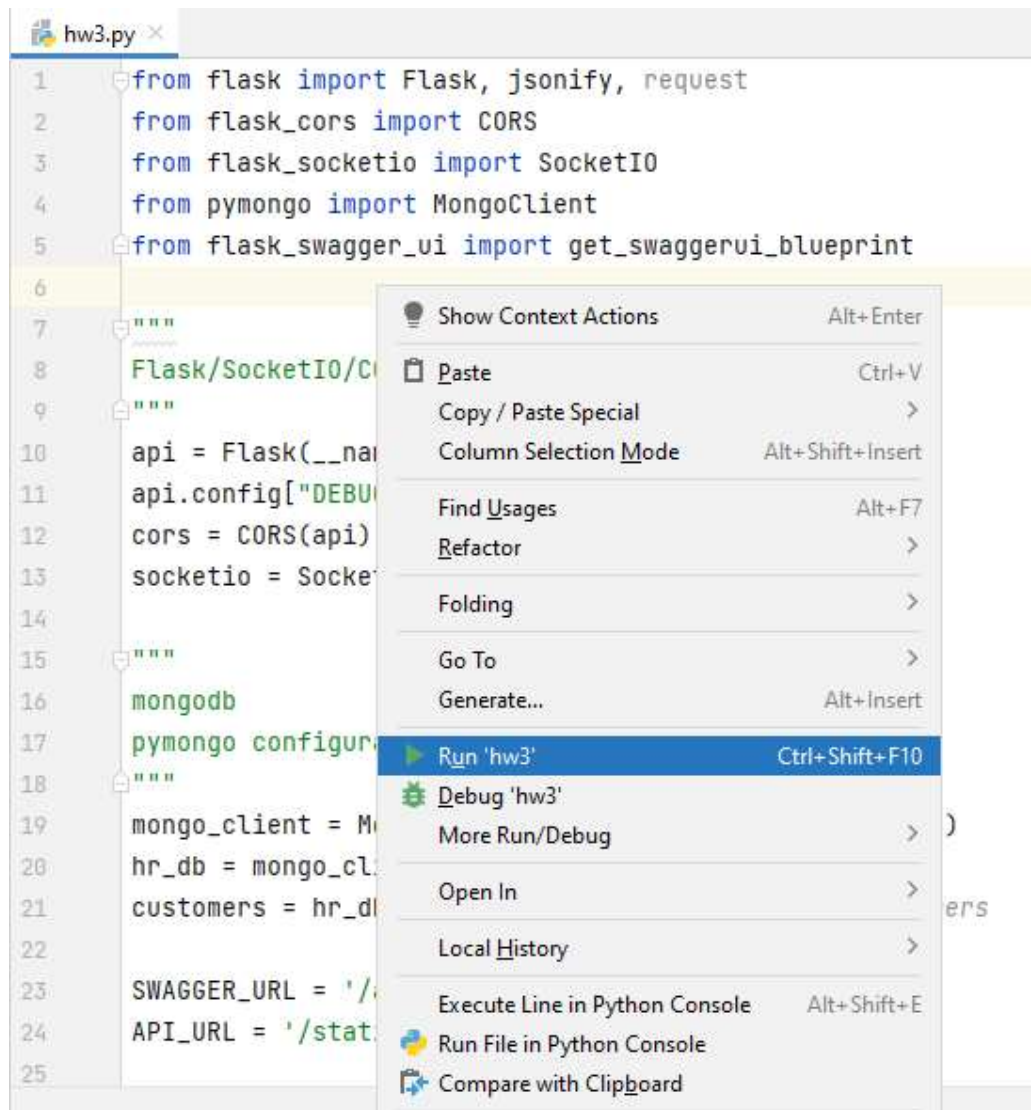
@api.route("/crm/api/v1/customers/<identity>", methods=["DELETE"])
def remove_customer(identity):
    # TODO: release a customer by identity
    # TODO: create an event and send it through Websocket and Kafka
    # TODO: Event: {"eventType": "CUSTOMER_RELEASED", "eventData": customer}
    # TODO: return removed customer as JSON
    pass

socketio.run(api, port=8100)
```

Task-1: Pull the template project from the following GitHub repository:

<https://github.com/deepcloudlabs/ain3003-22.23-database.systems.and.cloud.computing/tree/main/homeworks/hw3/project>

Task-2: Run the Python script h3.py:



```
1 from flask import Flask, jsonify, request
2 from flask_cors import CORS
3 from flask_socketio import SocketIO
4 from pymongo import MongoClient
5 from flask_swagger_ui import get_swaggerui_blueprint
6
7 """
8 Flask/SocketIO/Configuration
9 """
10 api = Flask(__name__)
11 api.config["DEBUG"] = True
12 cors = CORS(api)
13 socketio = SocketIO(api)
14
15 """
16 mongodb
17 pymongo configuration
18 """
19 mongo_client = MongoClient('mongodb://localhost:27020/')
20 hr_db = mongo_client.hr
21 customers = hr_db.customers
22
23 SWAGGER_URL = '/static/swagger.json'
24 API_URL = '/static/swagger.json'
25
```

Task-3: After you run the script (hw3.py), open the following URL in a Web Browser:

<http://localhost:8100/api/docs>

The screenshot shows a web browser displaying the Swagger API documentation for the CRM API (version 1.0.0). The base URL is localhost:8100/crm/api/v1. The API is described as a CRM Application API with MIT license. A dropdown menu shows the selected scheme as HTTP. The main section is titled 'Customers API for customer' and lists several endpoints under the 'customers' group:

- POST** /customers
- GET** /customers: Get all customers in system
- GET** /customers/{identityNo}: Get customer with given Identity No
- DELETE** /customers/{identityNo}: Delete customer with given Identity No
- PUT** /customers/{identityNo}: Update customer
- PATCH** /customers/{identityNo}: Update customer

Task-4: After you run the script (hw3.py), open the following URL in a Web Browser:

<http://localhost:8100/static/index.html>.

Implement the REST API endpoints and try to use the web application:

Customer

Identity No

First Name

Last Name

E-mail


Birth Year

Photo ?

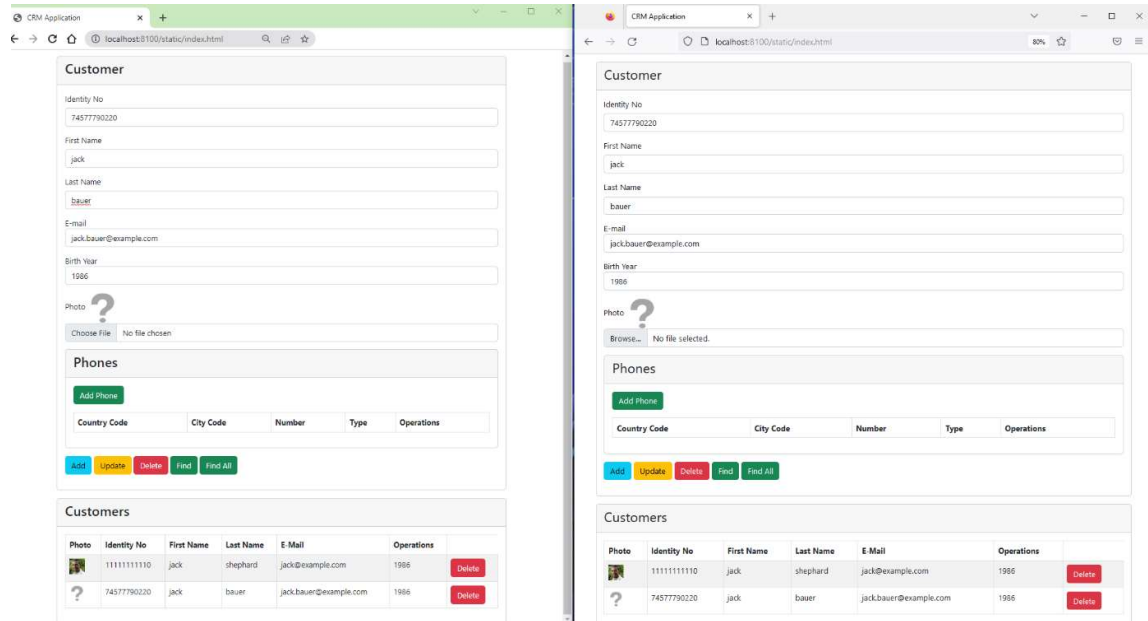
Phones

Country Code	City Code	Number	Type	Operations

Customers

Photo	Identity No	First Name	Last Name	E-Mail	Operations	
	11111111110	jack	shephard	jack@example.com	1986	<input type="button" value="Delete"/>
?	74577790220	jack	bauer	jack.bauer@example.com	1986	<input type="button" value="Delete"/>

Task-5: Make sure that events are published in the backend by opening the web application in two different Web Browsers:



Add a new customer and remove a customer using the web application. Make sure that UI in the other Web Browser also updates the table automatically.

IMPORTANT

- Academic dishonesty, including but not limited to cheating, plagiarism, and collaboration, is unacceptable and subject to disciplinary action. Any student found guilty will have a grade of F. Assignments are due in class on the due date. Late assignments will generally not be accepted. Any exception must be approved. Approved late assignments are subject to a grade penalty.