Local rest API created with Python and Flask connected by MySQL DB

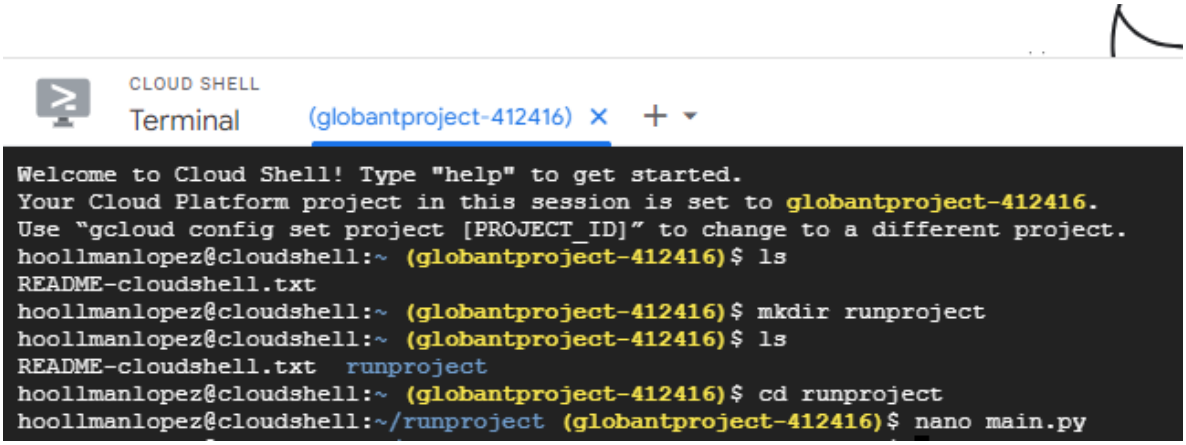
Python version: `python:3.10.13-bookworm`

MySQL: Clever Cloud

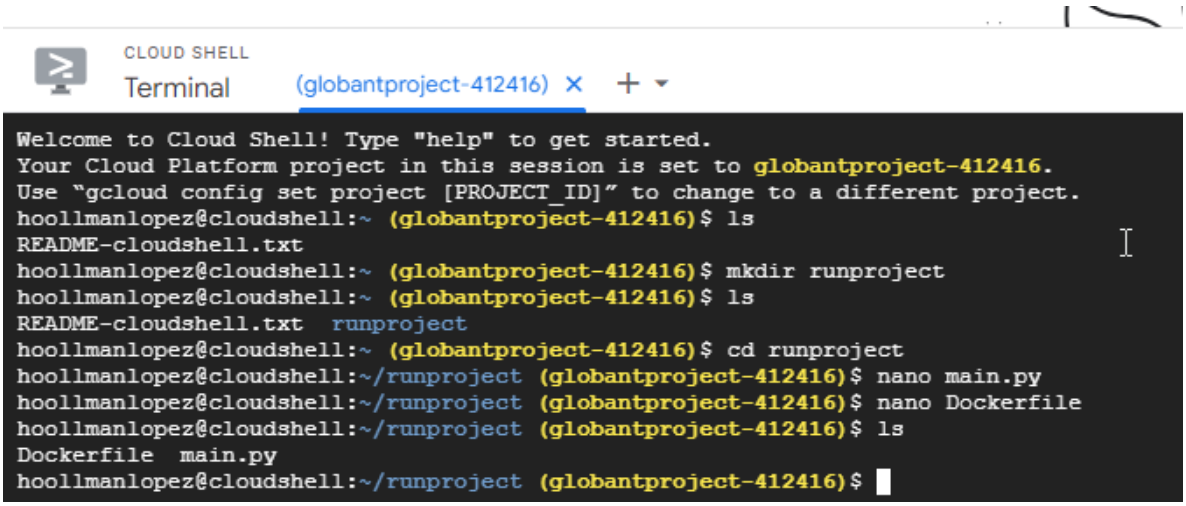
Cloud Environment: GCP

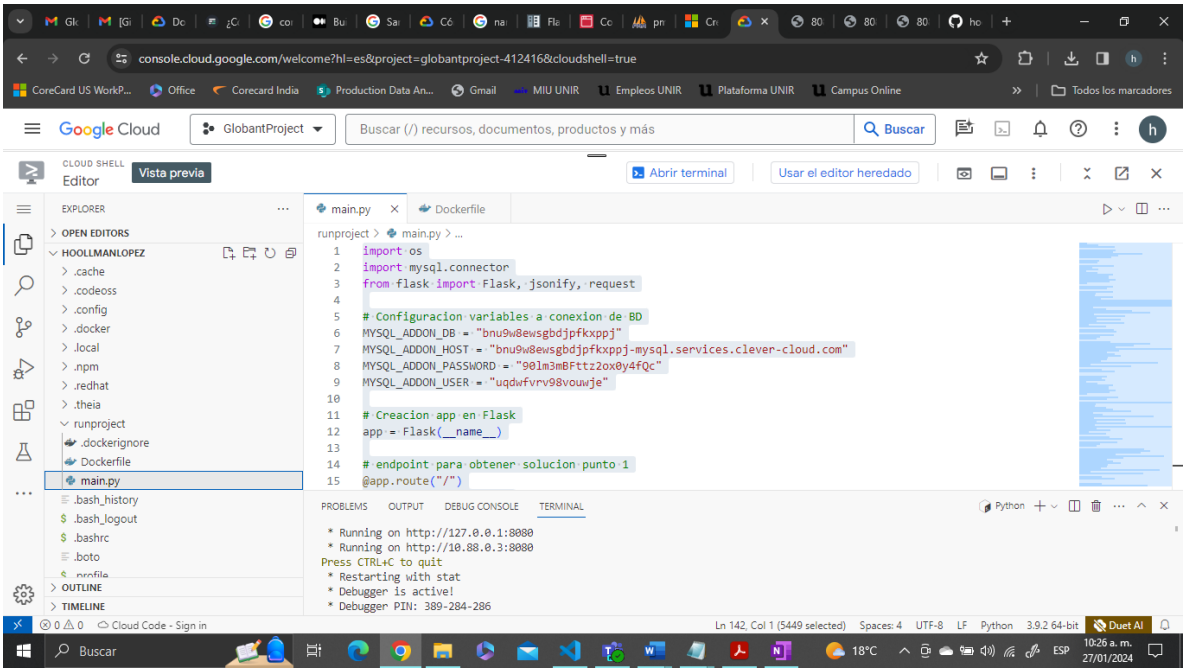
Project created with name: Globantproject

Main.py created



Docker file created





Code to create Docker file to Containerize application

#Code Docker

```
FROM python:3.10.13-bookworm

#Allow statements and log messages to immediately appear in the Knative logs
ENV PYTHONUNBUFFERED True

#Copy local code to the container image.
ENV APP_HOME /app
WORKDIR $APP_HOME
COPY . ./

#Install production dependencies.
RUN pip install Flask gunicorn

CMD exec gunicorn --bind :$PORT --workers 1 --threads 8 --timeout 0 main:app
```

Code in Python to crated API with Python – Flask – Mysql

#Code API in Python with Flask

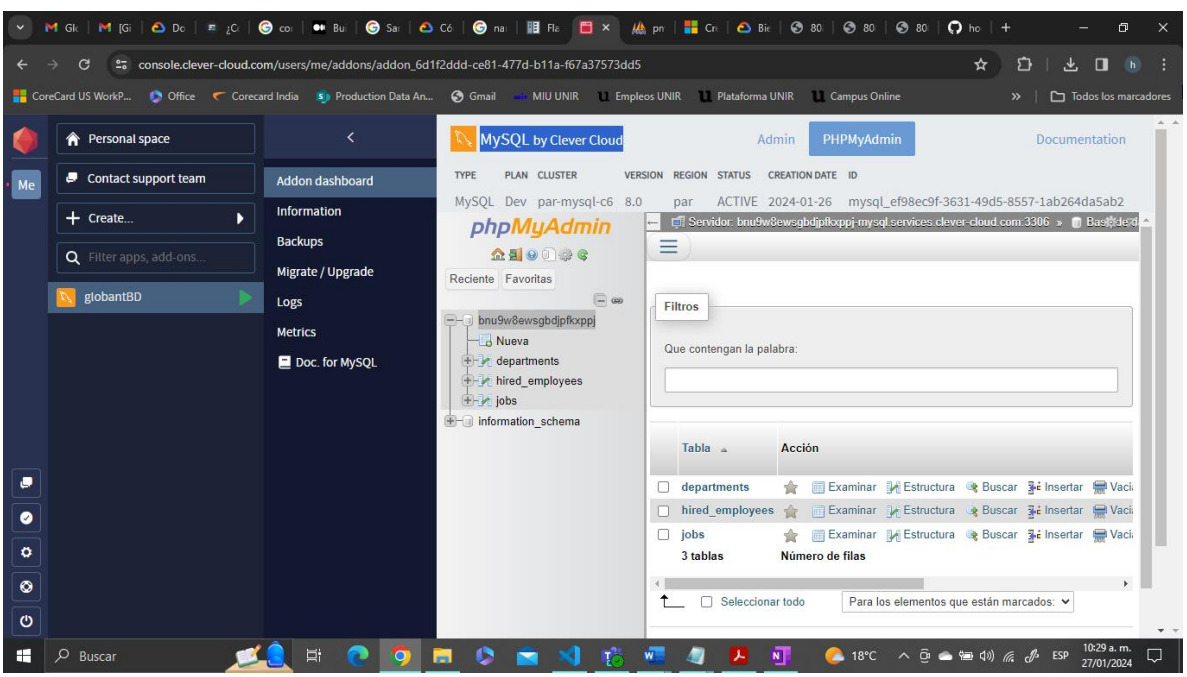
```
import os

import mysql.connector
from flask import Flask, jsonify, request

# Configuracion variables a conexion de BD
MYSQL_ADDON_DB = "bnu9w8ewsgbdjpfkxppj"
MYSQL_ADDON_HOST = "bnu9w8ewsgbdjpfkxppj-mysql.services.clever-cloud.com"
MYSQL_ADDON_PASSWORD = "90lm3mBFttz2ox0y4fQc"
MYSQL_ADDON_USER = "uqdwfvrv98vouwje"

# Creacion app en Flask
app = Flask(__name__)

# endpoint para obtener solucion punto 1
@app.route("/")
def employees_hired():
    try:
        # Conectar BD
        mydb = mysql.connector.connect(
            host=MYSQL_ADDON_HOST,
            user=MYSQL_ADDON_USER,
            password=MYSQL_ADDON_PASSWORD,
            database=MYSQL_ADDON_DB
        )

        # Cursor para ejecutar consulta
        mycursor = mydb.cursor()

        # Ejecucion de Consulta Punto 1
        query = """
            SELECT
                departments.department,
                jobs.job,
                SUM(CASE WHEN EXTRACT(MONTH FROM hired_employees.datetime) BETWEEN 1 AND 3 THEN 1
ELSE 0 END) AS Q1,
                SUM(CASE WHEN EXTRACT(MONTH FROM hired_employees.datetime) BETWEEN 4 AND 6 THEN 1
ELSE 0 END) AS Q2,
                SUM(CASE WHEN EXTRACT(MONTH FROM hired_employees.datetime) BETWEEN 7 AND 9 THEN 1
ELSE 0 END) AS Q3,
                SUM(CASE WHEN EXTRACT(MONTH FROM hired_employees.datetime) BETWEEN 10 AND 12 THEN 1
ELSE 0 END) AS Q4
```

```
            FROM
                hired_employees
                JOIN jobs ON hired_employees.job_id = jobs.id
                JOIN departments ON hired_employees.department_id = departments.id
            WHERE
                YEAR(hired_employees.datetime) = 2021
            GROUP BY
                departments.department, jobs.job
            ORDER BY
                departments.department, jobs.job asc
        """
        mycursor.execute(query)

        # Obtener resultados de consulta
        departamentos = mycursor.fetchall()

        # Cerrar cursor y  conexion a BD
        mycursor.close()
        mydb.close()

        # Si se accede mostrar datos
        if request.headers.get("User-Agent") in request.headers.get("User-Agent"):
            output = "<h1>Datos de los departamentos:</h1><br>"
            output += "<table border='1'>"
            output +=
"<tr><th>Department</th><th>Job</th><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr>"
            for departamento in departamentos:
                output += "<tr>"
                for item in departamento:
                    output += f"<td>{item}</td>"
                output += "</tr>"
            output += "</table>"
            return output
        # Si  accede desde un cliente que acepta JSON, devolvemos los datos en formato JSON
        else:
            return jsonify({"departamentos": departamentos})

    except Exception as e:
        return jsonify({"error": str(e)})


# endpoint para el segundo punto
@app.route("/<path:path>")
def employees_by_department():
    try:
```

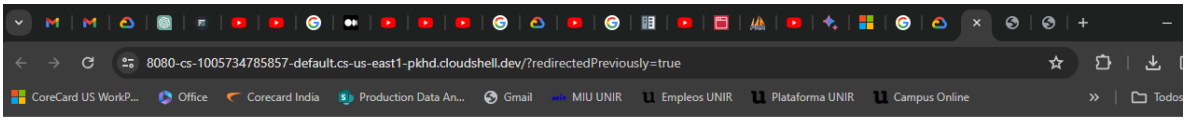
```
        # Conectar BD
        mydb = mysql.connector.connect(
            host=MYSQL_ADDON_HOST,
            user=MYSQL_ADDON_USER,
            password=MYSQL_ADDON_PASSWORD,
            database=MYSQL_ADDON_DB
        )

        # Crear cursor para ejecutar consulta
        mycursor = mydb.cursor()

        # Ejecutar consulta para obtener la lista de ids, nombres y numero de empleados contratados
de cada departamento
        query = """
            SELECT
                departments.id,
                departments.department,
                COUNT(hired_employees.id) AS total_employees
            FROM
                departments
                JOIN hired_employees ON departments.id = hired_employees.department_id
            WHERE
                YEAR(hired_employees.datetime) = 2021
            GROUP BY
                departments.id, departments.department
            HAVING
                COUNT(hired_employees.id) > (SELECT AVG(dept_employees.total_employees) FROM (SELECT
departments.id, COUNT(hired_employees.id) AS total_employees FROM departments JOIN hired_employees
ON departments.id = hired_employees.department_id WHERE YEAR(hired_employees.datetime) = 2021 GROUP
BY departments.id) AS dept_employees)
            ORDER BY
                total_employees DESC
        """
        mycursor.execute(query)

        # resultados de la consulta
        department_employees = mycursor.fetchall()

        # Cerramos el cursor y conexion a BD
        mycursor.close()
        mydb.close()

        # Si se accede desde un navegador, mostramos los datos en una tabla
        if request.headers.get("User-Agent") in request.headers.get("User-Agent"):
            output = "<h1>Employees hired of each department:</h1><br>"
```

```
            output += "<table border='1'>"
            output += "<tr><th>Department ID</th><th>Department</th><th>Total Employees</th></tr>"
            for department in department_employees:
                output += "<tr>"
                for item in department:
                    output += f"<td>{item}</td>"
                output += "</tr>"
            output += "</table>"
            return output
        # Si se accede desde un cliente que acepta JSON, devolvemos los datos en formato JSON
        else:
            return jsonify({"department_employees": department_employees})

    except Exception as e:
        return jsonify({"error": str(e)})


if __name__ == "__main__":
    app.run(debug=True, host="0.0.0.0", port=int(os.environ.get("PORT", 8080)))
```

DB created in Clever cloud with Files attached in exercise



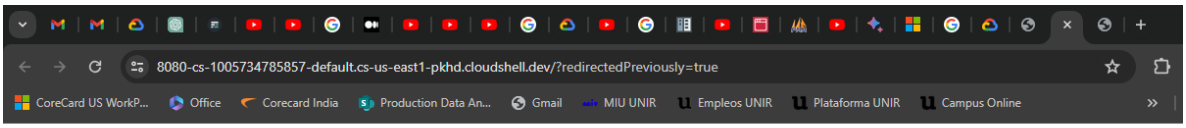
Result 1st endpoint



Datos de los departamentos:

Department	Job	Q1	Q2	Q3	Q4
Accounting	Account Representative IV	1	0	0	0
Accounting	Actuary	0	1	0	0
Accounting	Analyst Programmer	0	0	1	0
Accounting	Budget/Accounting Analyst III	0	1	0	0
Accounting	Cost Accountant	0	1	0	0
Accounting	Database Administrator III	0	0	0	1
Accounting	Desktop Support Technician	0	0	1	0
Accounting	Food Chemist	1	0	0	0
Accounting	Graphic Designer	0	1	0	0
Accounting	Health Coach III	0	0	0	1
Accounting	Health Coach IV	0	0	1	0
Accounting	Help Desk Technician	0	0	1	0
Accounting	Junior Executive	0	0	1	0
Accounting	Legal Assistant	0	0	1	1
Accounting	Media Manager III	0	1	0	0
Accounting	Programmer Analyst IV	0	1	1	1
Accounting	Programmer III	0	0	1	0
Accounting	Research Assistant IV	0	0	1	0
Accounting	Sales Representative	1	0	1	0
Accounting	Senior Cost Accountant	0	0	0	1

Result 2nd endpoint



Employees hired of each department:

Department ID	Department	Total Employees
8	Support	221
5	Engineering	208
6	Human Resources	204
7	Services	204
4	Business Development	187
3	Research and Development	151
9	Marketing	143