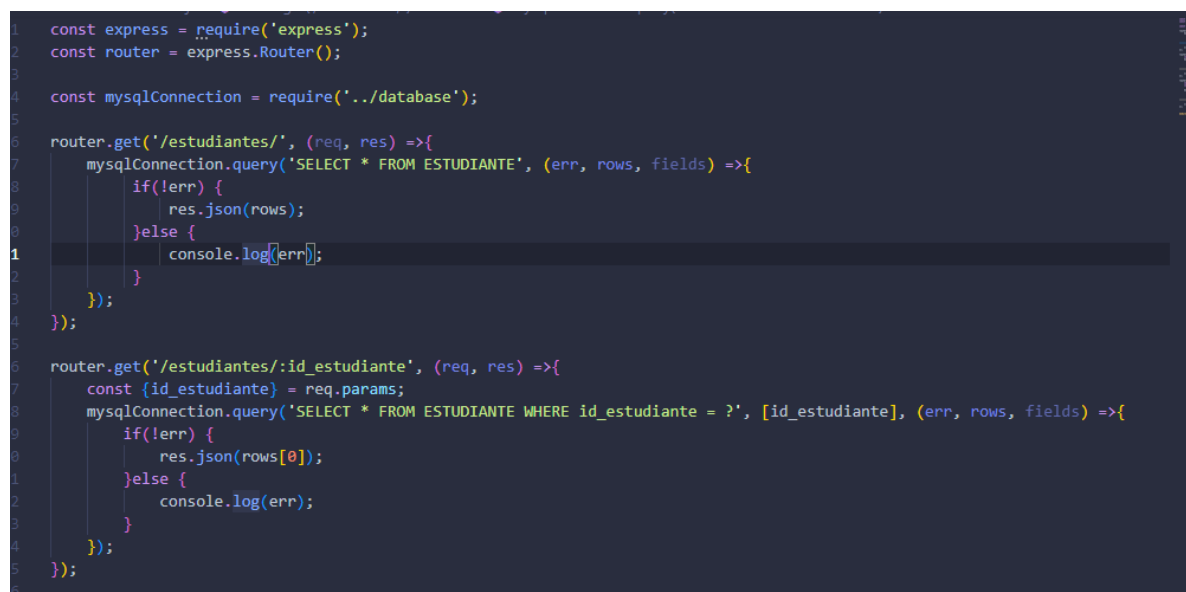
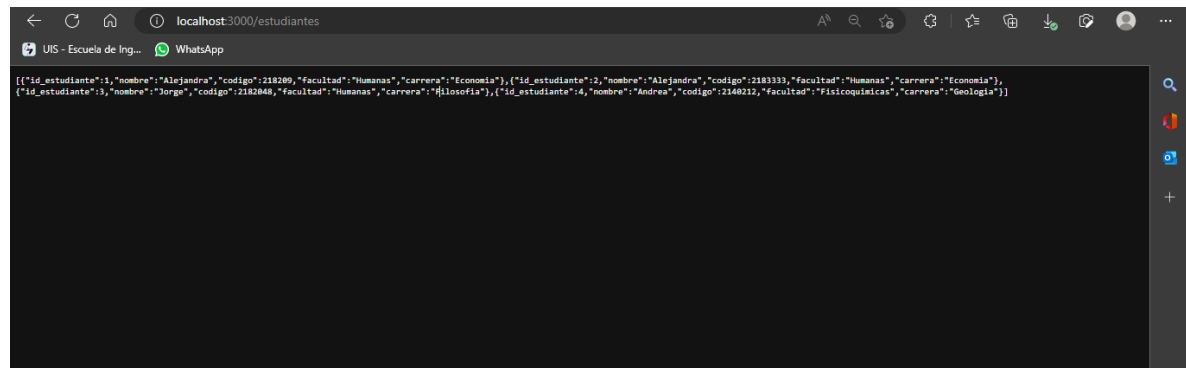
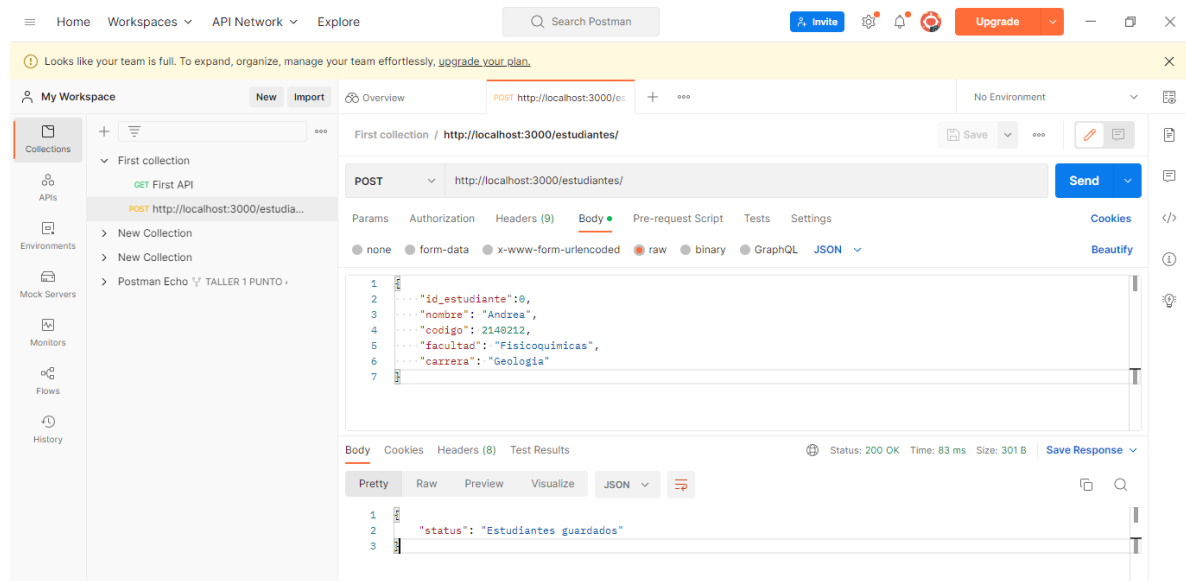


PRUEBAS – GET Y POST



```
Get Started package-lock.json JS chips.js JS database.js JS estudiantes.js X DB.sql JS clientes.js package.json
src > rutas > JS estudiantes.js > router.get('/estudiantes/') callback > mysqlConnection.query('SELECT * FROM ESTUDIANTE') callback
20     res.json(rows[0]);
21   }else {
22     console.log(err);
23   }
24   });
25   });
26
27   router.post('/estudiantes/', (req, res)=>{
28     const {id_estudiante, nombre, codigo, facultad, carrera} = req.body;
29     const query = `CALL agregarActualizarEstudiante(?, ?, ?, ?, ?)`;
30
31     mysqlConnection.query(query, [id_estudiante, nombre, codigo, facultad, carrera], (err, rows, fields) => {
32       if(!err){
33         res.json({status: 'Estudiantes guardados'});
34       }else {
35         console.log(err);
36       }
37     });
38   });
39 }
40
41 PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
npm ERR! This is probably not a problem with npm. There is likely additional logging output above.
npm ERR! A complete log of this run can be found in:
npm ERR! C:\Users\alero\AppData\Roaming\npm-cache\_logs\2022-11-07T16_00_11_236Z-debug.log
^C¿Desea terminar el trabajo por lotes (S/N)? N
PS C:\Users\alero\OneDrive\Escritorio\UIS\9 NOVENO SEMESTRE\Ingenieria-del-Software-3\Taller 1 - API REST y Mensajería Publish-Subscribe\mysql-nodejs-rest-api> npm start
> mysql-nodejs-rest-api@1.0.0 start C:\Users\alero\OneDrive\Escritorio\UIS\9 NOVENO SEMESTRE\Ingenieria-del-Software-3\Taller 1 - API REST y Mensajería Publish-Subscribe\mysql-nodejs-rest-api
> node src/index.js
Server en el puerto 3000
La base de datos está conectada

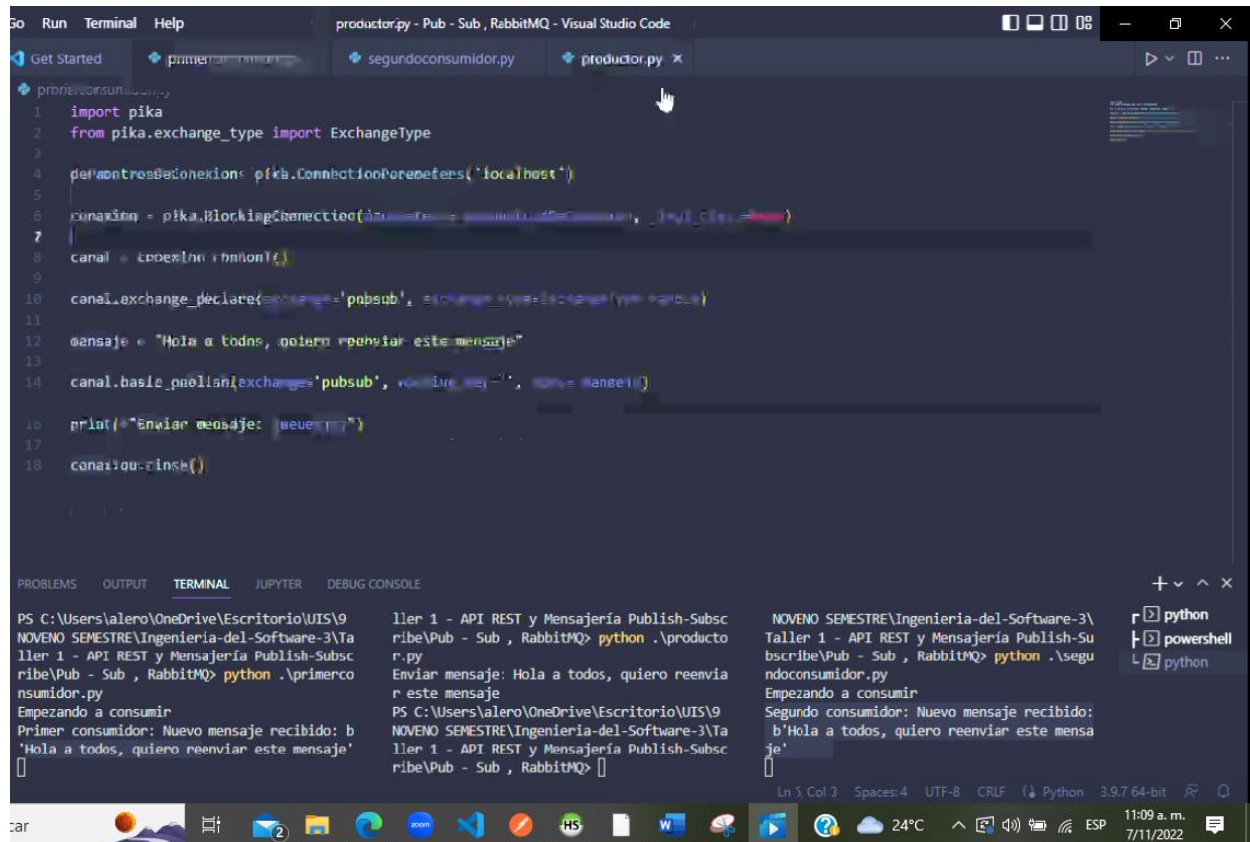
```

```
''
DROP TABLE estudiante;
CREATE TABLE `estudiante` (
  id_estudiante INT NOT NULL auto_increment,
  nombre VARCHAR(20),
  codigo INT,
  facultad VARCHAR(20),
  carrera VARCHAR(20),
  PRIMARY KEY (id_estudiante)
);

DROP PROCEDURE IF EXISTS agregarActualizarEstudiante;
DELIMITER $$
CREATE PROCEDURE agregarActualizarEstudiante(
  IN Nid_estudiante INT,
  IN Nnombre VARCHAR(20),
  IN Ncodigo INT,
  IN Nfacultad VARCHAR(20),
  IN Ncarrera VARCHAR(20)
)
BEGIN
  IF Nid_estudiante = 0 then
    INSERT INTO ESTUDIANTE (nombre, codigo, facultad, carrera)
    VALUES(Nnombre , Ncodigo, Nfacultad, Ncarrera);
    SET `Nid_estudiante` = LAST_INSERT_ID();
  ELSE
    UPDATE ESTUDIANTE
    SET nombre = Nnombre, codigo= Ncodigo, facultad=Nfacultad, carrera=Ncarrera
    WHERE id_estudiante = Nid_estudiante;
  END IF;
  SELECT Nid_estudiante AS 'id_estudiante';
END $$

```

PRUEBAS – Mensaje asíncrono con RabbitMQ



```
1 import pika
2 from pika.exchange_type import ExchangeType
3
4 def main():
5     connection = pika.ConnectionParameters('localhost')
6     channel = pika.BlockingChannel(connection)
7     channel.exchange_declare(exchange='pubsub', exchange_type=ExchangeType.queue)
8     message = "Hola a todos, quiero reenviar este mensaje"
9     channel.basic_publish(exchange='pubsub', routing_key='', body=message)
10    print("Enviar mensaje: ", message)
11    channel.close()
```

PS C:\Users\alero\OneDrive\Escritorio\UIS\9 NOVENO SEMESTRE\Ingeniería-del-Software-3\Taller 1 - API REST y Mensajería Publish-Subscribe\Pub - Sub, RabbitMQ> python .\productor.py

Enviar mensaje: Hola a todos, quiero reenviar este mensaje

PS C:\Users\alero\OneDrive\Escritorio\UIS\9 NOVENO SEMESTRE\Ingeniería-del-Software-3\Taller 1 - API REST y Mensajería Publish-Subscribe\Pub - Sub, RabbitMQ>

NOVENO SEMESTRE\Ingeniería-del-Software-3\Taller 1 - API REST y Mensajería Publish-Subscribe\Pub - Sub, RabbitMQ> python .\segundoconsumidor.py

Empezando a consumir

Segundo consumidor: Nuevo mensaje recibido: b'Hola a todos, quiero reenviar este mensaje'

Enlace repositorio: <https://github.com/ARS100/Ingenieria-del-Software-3/tree/main/Taller%201%20-%20API%20REST%20y%20Mensajer%3%ADa%20Publish-Subscribe>