|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| logo berri CIFP  CIFP Ciudad Jardín LHII | | | **Curso de Especialización:** Inteligencia artificial y Big Data | | | |
| **Modulo:**  Sistemas de Big Data  Big Data Aplicado | | | **Cód. Modulo:**  SBIG  BIGD |
| **Curso:** IABD | | | |
| **2ª Evaluación** | | | **Fecha**: 10/03/2024 | | | |
| **Cód. Prof:** PTI8 | | **Profesora:** Saioa Uribeetxebarria | |  | | |
|  | | | | | | |
| **Datos del Alumno** | | | | | | |
| **Nº** | **Nombre:** | | | | **Firma** | |
| **Apellido 1:** | | | | |
| **Apellido 2:** | | | | |

**Calificación:**

# EXAMEN

Para realizar los ejercicios necesitas:

* Ejercicios 1 y 2: Tu equipo (necesitas tener instalado pySpark)
* Ejercicios 3, 4: Una máquina virtual con:
* kafka (1 único broker)
* nifi: Usuario: nifi contraseña: nifinifinifi
* mongodb, mongodb-Compass

Para arrancar mongodb: sudo systemctl start mongod

Para comporbar el estado: sudo systemctl status mongod

Para parar mongodb: sudo service mongod stop

* En la carpeta personal se ha definido el entorno virtual: miEntorno que tiene instaladas las librerías que permiten crear con Python el productor de kafka.

Para activar el entorno virtual: source miEntorno/bin/activate

Para desactivar el entorno virtual: deactivate

Pero si quieres crear tu propio entorno virtual o trabajar sin entorno puedes hacerlo.

Si en algún momento alguna parte de un ejercicio no te sale no te bloquees y sigue adelante con esa parte simplificada. Por ejemplo, si no eres capaz de crear el productor Kafka que se pide crea otro cualquiera para poder seguir adelante con el ejercicio o envía mensajes al topic de kafka desde un productor de consola.

## EJERCICIO 1

Con el archivo Turismo.csv lleva a cabo las siguientes acciones utilizando Dataframes de SPARK.

En este ejercicio debes crear el archivo: Ejercicio1.ipynb con los pasos llevados a cabo y subirlo a moodle.

En el archivo que entregues que se vean todos los resultados de las operaciones.

1. Cargar los datos en un DataFrame de Spark infiriendo el esquema.
2. Mostrar el esquema del DataFrame y los primeros 5 registros.
3. Convertir la columna "FECHA" en formato de fecha (dd/MM/yyyy) y mostrar el esquema.
4. Listar las 3 visitas con la mayor puntuación de: SATISFACCION\_GLOBAL.
5. Contar cuántas visitas se realizaron en cada mes del año y mostrarlas en orden descendente.
6. Filtrar las visitas realizadas en septiembre y mostrar sólo los campos: VISITA y SATISFACCION\_GLOBAL.
7. Reemplazar valores nulos en la columna: LO\_PEOR con el valor: "Sin comentarios".
8. Crear una nueva columna llamada SATISFACCION\_PROMEDIO que sea el promedio de las 3 categorías de satisfacción: SATISFACCION\_INFOR, SATISFACCION\_RECORRIDO, SATISFACCION\_GUIA. Debes crear tu propia función de usuario y registrarla como una función de Spark. Con SQL y sin SQL.
9. Guardar el resultado final en formato .csv de tal forma que si el archivo ya existe que lo sobreescriba.

## EJERCICIO 2

Desarrolla una aplicación en PySpark Streaming que cuente la frecuencia de palabras en tiempo real.

La entrada de datos se proporcionará manualmente a través de un servidor de socket. Y la salida de datos se mostrará por consola.

1. Mostrar la cuenta de las palabras utilizando ventanas fijas de 2 minutos de duración.

Muestra los resultados sin que se trunquen para que se vea la información completa.

Codigo:

from pyspark.sql import SparkSession

from pyspark.sql.functions import window

from pyspark.sql.functions import explode, split

spark = SparkSession.builder.appName("Ejercicio2").getOrCreate()

dfLineas = spark.readStream \

.format("socket") \

.option("host", "localhost") \

.option("port", "9999") \

.option('includeTimestamp', 'true') \

.load()

dfPalabras = dfLineas.select(

explode(split(dfLineas.value, ' ')).alias('palabra'),

dfLineas.timestamp

)

windowedCounts = dfPalabras.groupBy(

window(dfPalabras.timestamp, "2 minutes"), dfPalabras.palabra

).count().orderBy('window')

palabrasQuery = windowedCounts.writeStream \

.format("console") \

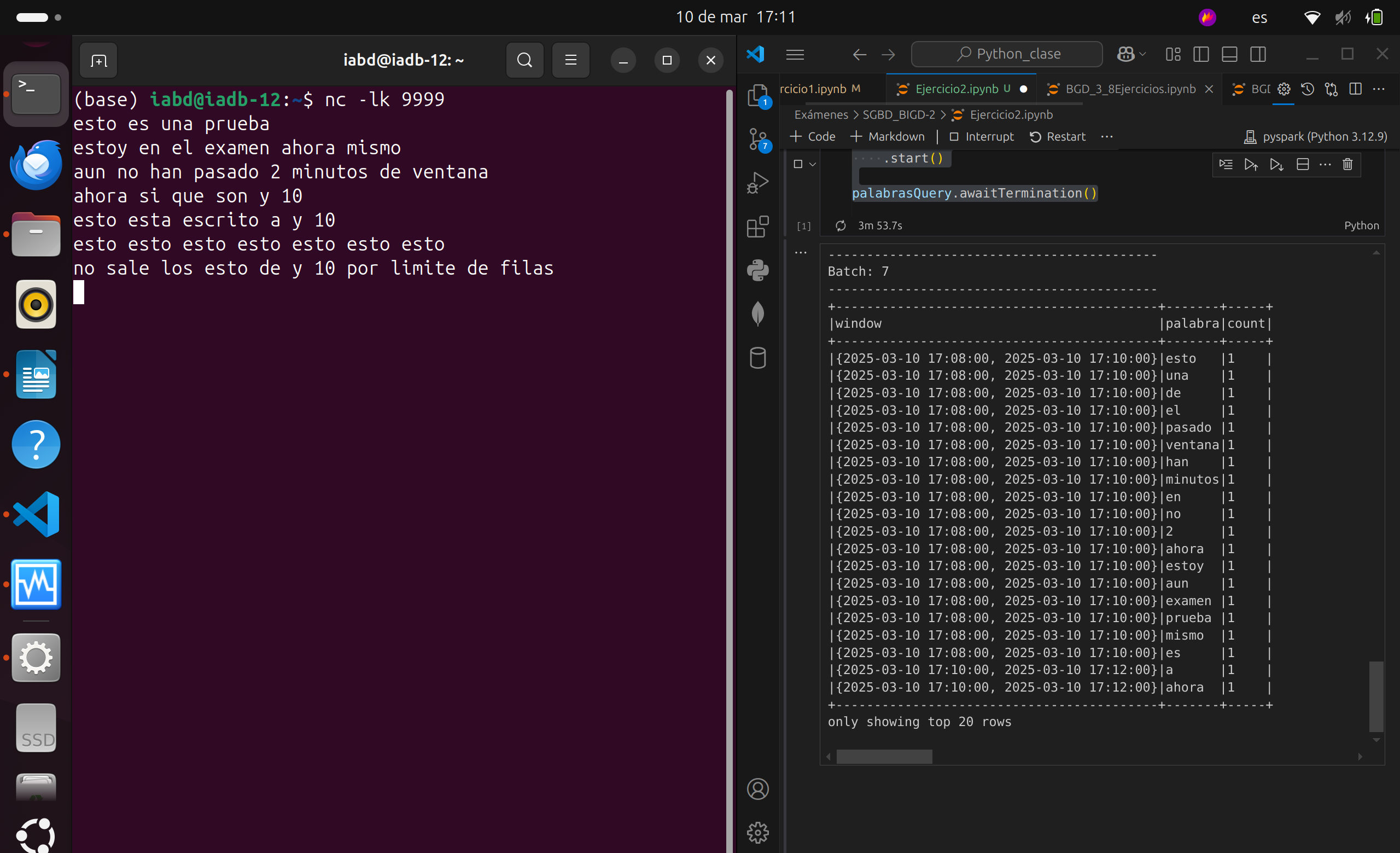
.outputMode("complete") \

.option('truncate', 'false') \

.start()

palabrasQuery.awaitTermination()

Funcionamiento:



## EJERCICIO 3

En la máquina virtual proporcionada que tiene Kafka (1 broker), nifi y MongoDB se pide:

Crear en nifi un flujo de datos que lea los datos de un topic de kafka y guarde esos datos en MongoDB, siguiendo los siguientes pasos:

1. Crear el topic de Kafka: nifi-flujo
2. Crear un productor con Python que cada 10 segundos envíe información sobre personas al topic: nifi-flujo.
3. Crear el flujo de nifi que lea los mensajes del topic de Kafka: nifi-flujo y los guarde en MongoDB en la base de datos: nifi y en la colección: misDatos.

(Necesitarás utilizar 2 procesadores)

1. Descarga el flujo creado en nifi a un archivo .json

Puedes apoyarte en el siguiente código para crear el productor de Kafka

***personas.py***

from faker import Faker

from json import dumps

from kafka import KafkaProducer

import time

fake = Faker()

def create\_person():

return {

'name': fake.name(),

'address': fake.address(),

'email': fake.email()

}

producer = KafkaProducer(value\_serializer=lambda m: dumps(m).encode('utf-8'),bootstrap\_servers=['localhost:9092'])

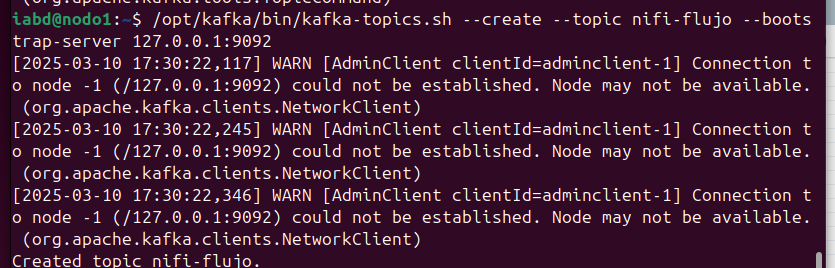
while True:

person = create\_person()

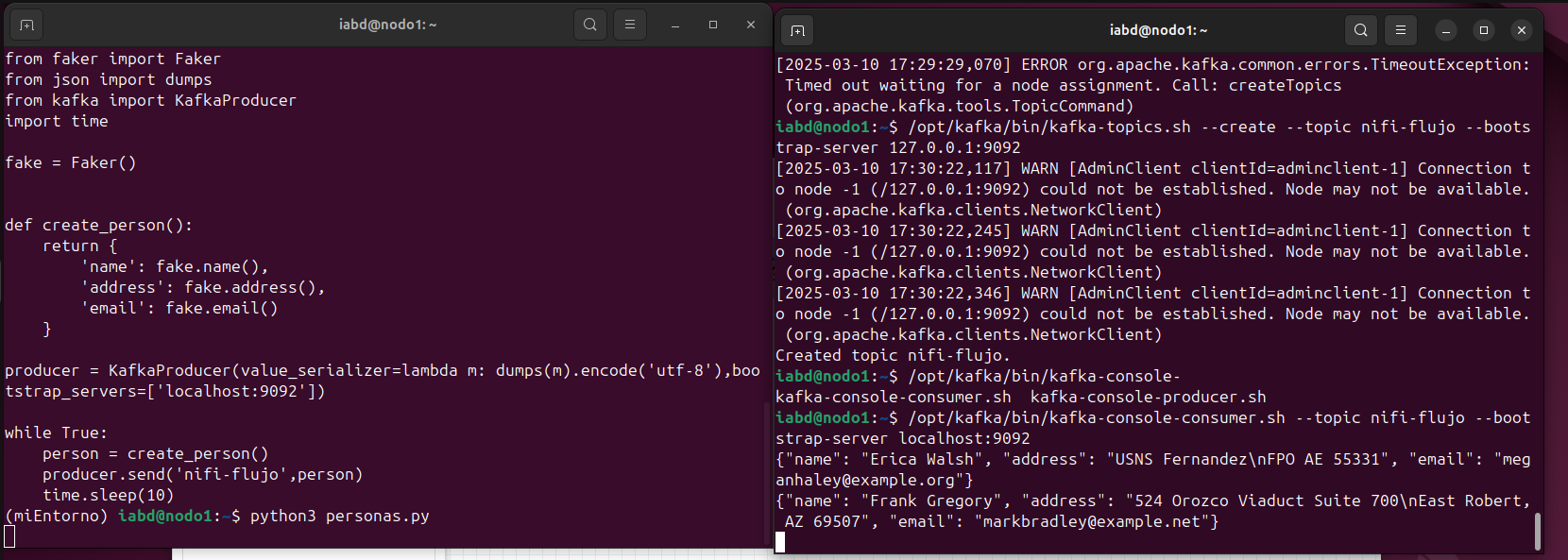
producer.send('nifi-flujo',person)

time.sleep(10)

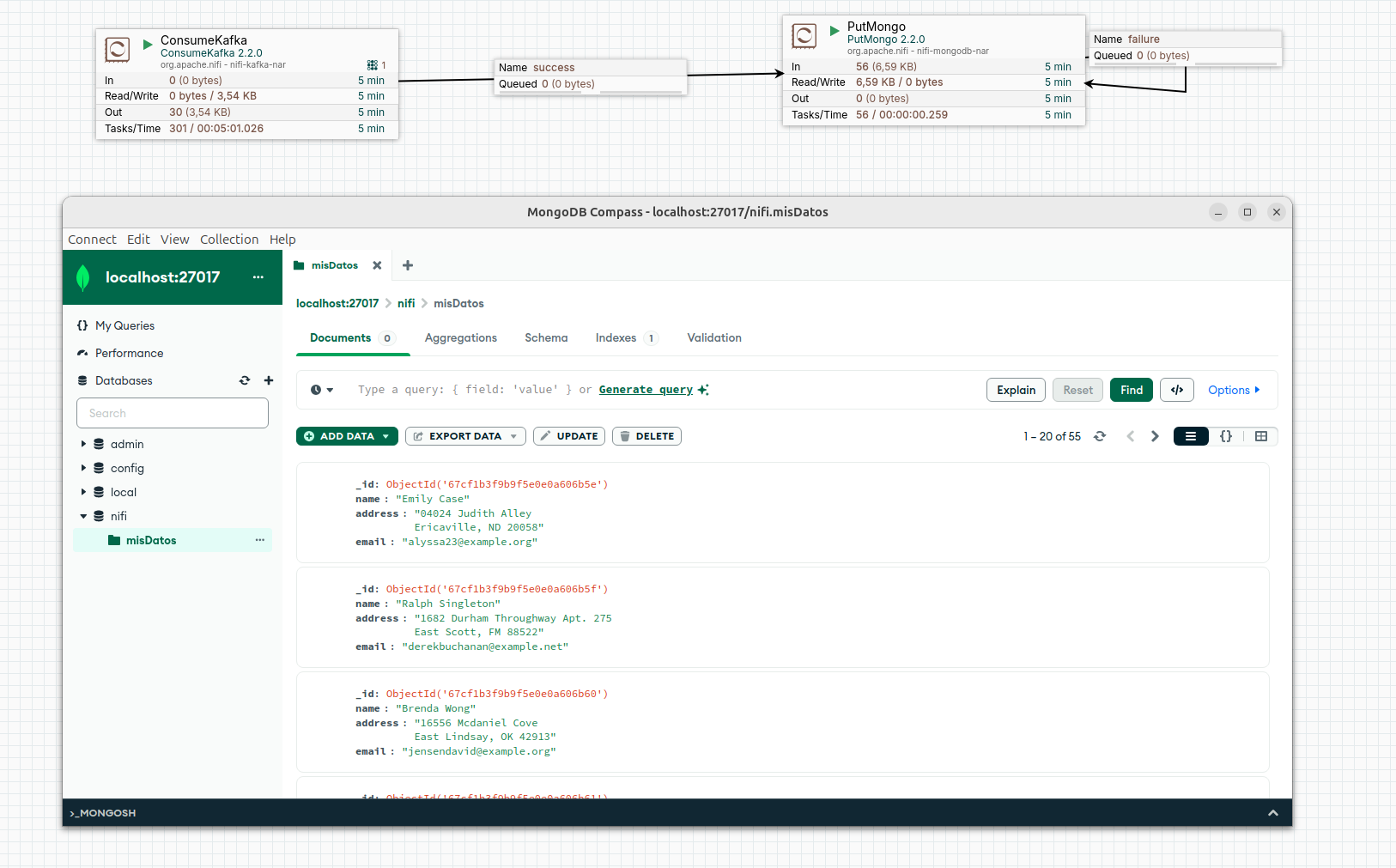
Creamos topic:



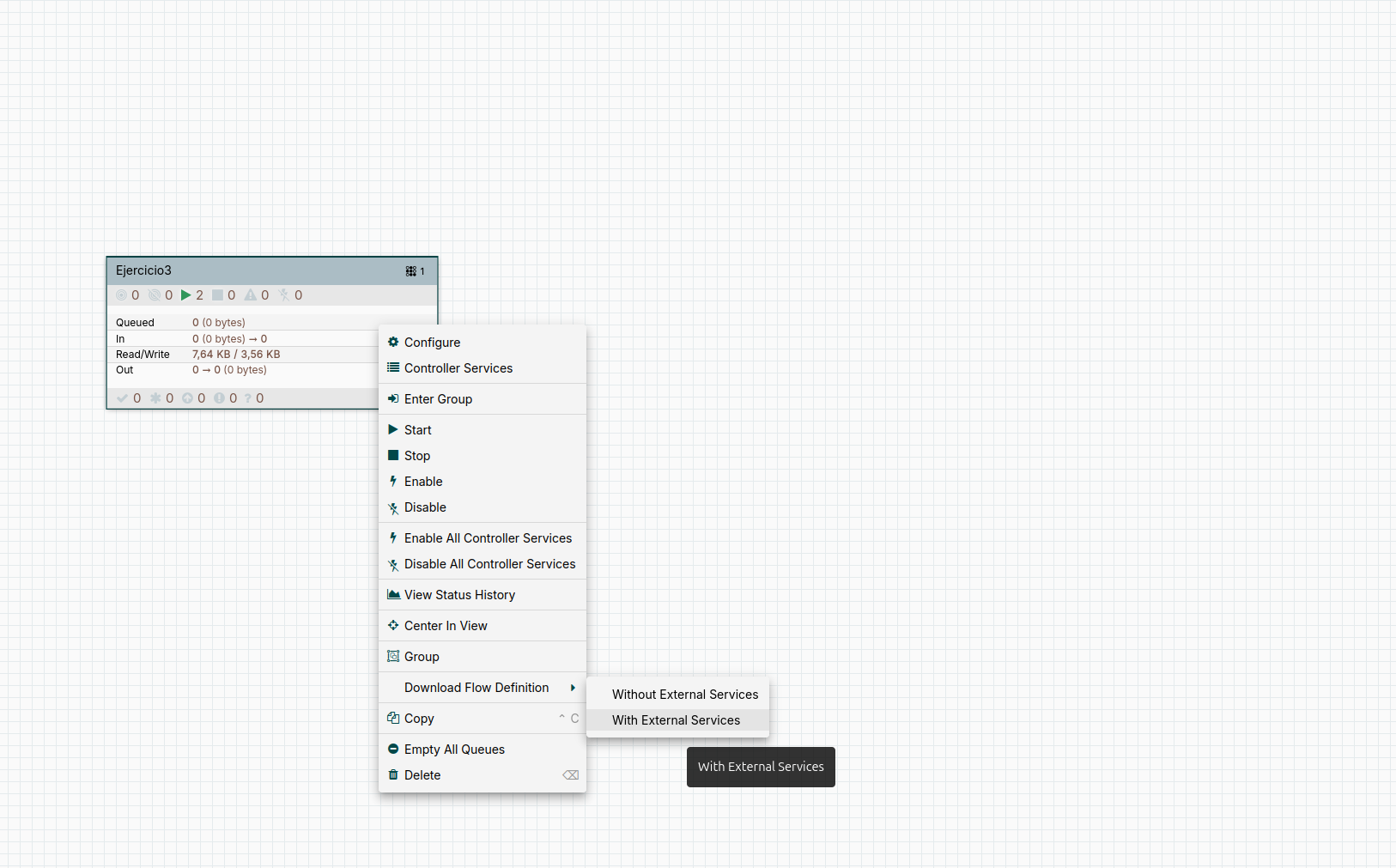
Lanzamos el personas.py y consumimos mensajes de prueba



Configuramos el flow de nifi:



No me va bien la conexión a la NAT en la VM pero saco captura de la descarga del json (haciendo el truco de meter todo en un process group)

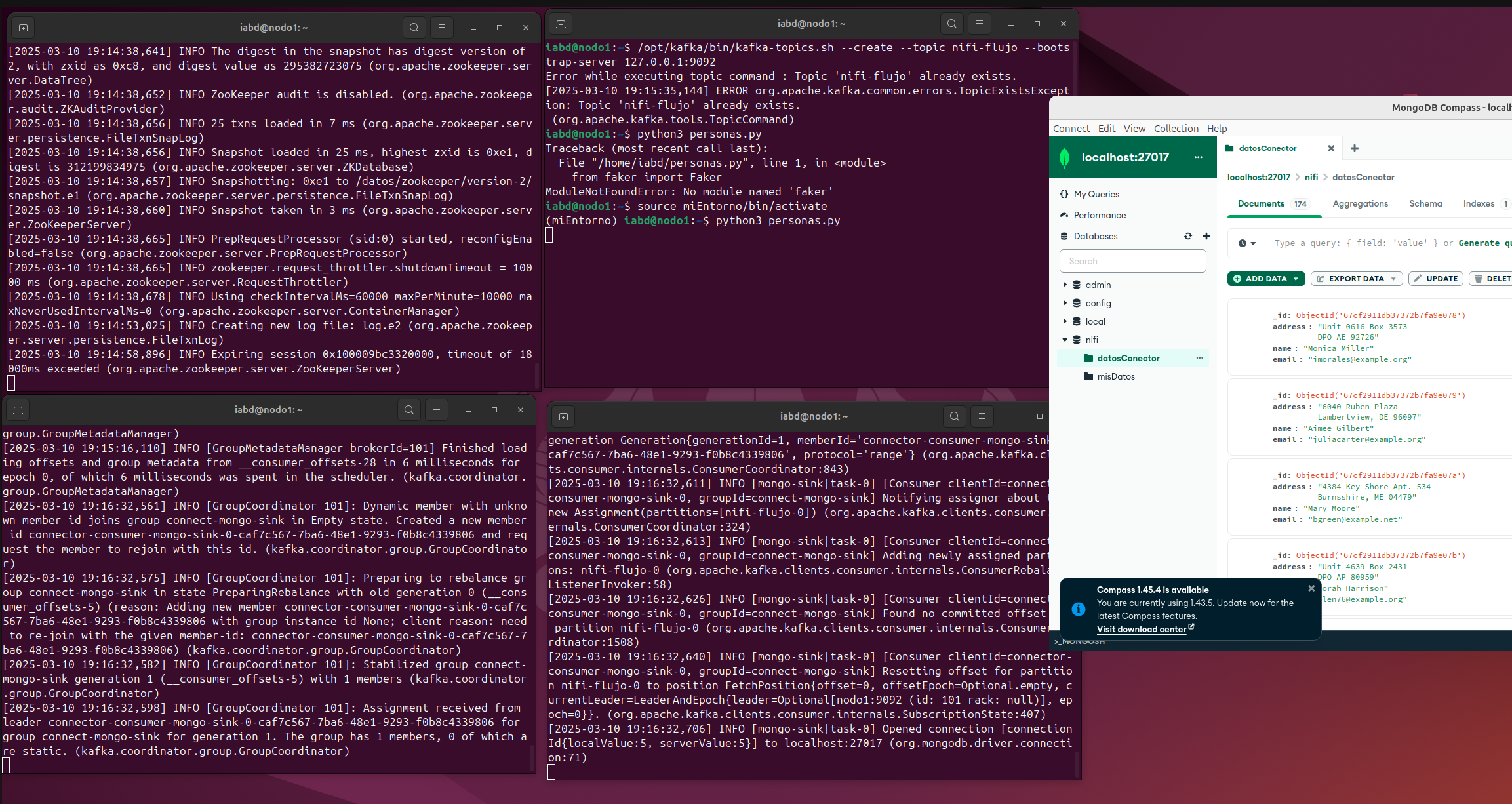


## EJERCICIO 4

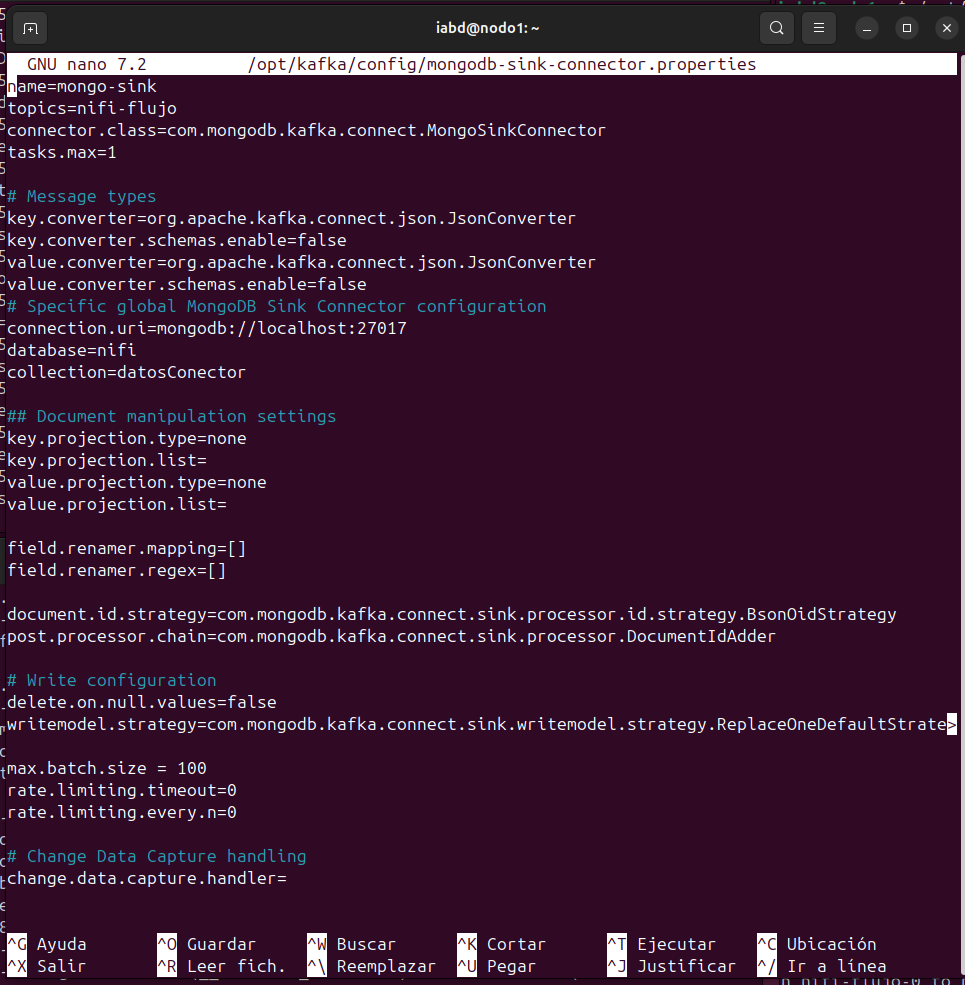
Utiliza un conector (Kafka Connect) para que consuma los mensajes del topic: nifi-flujo y los inserte en MongoDB de forma automática (base de datos: nifi, colección: datosConector). Aporta un pantallazo con los datos en MongoDB.

Utiliza el conector [MongoDB Connector (Source and Sink)](https://www.confluent.io/hub/mongodb/kafka-connect-mongodb) que encontrarás en https://www.confluent.io/hub/mongodb/kafka-connect-mongodb

MongoDB Connector corriendo:



He tenido que editar el properties:



# CRITERIOS DE EVALUACIÓN

|  |  |  |
| --- | --- | --- |
| **EJERCICIO 1** | **OBSERVACIONES** | **PUNTUACIÓN** |
| Apartado 1 |  | 0,25 |
| Apartado 2 |  | 0,25 |
| Apartado 3 |  | 0,25 |
| Apartado 4 |  | 0,25 |
| Apartado 5 |  | 0,25 |
| Apartado 6 |  | 0,25 |
| Apartado 7 |  | 0,25 |
| Apartado 8 |  | 0,5 |
| Apartado 9 |  | 0,25 |
|  | | **2,5** |

|  |  |  |
| --- | --- | --- |
| **EJERCICIO 2** | **OBSERVACIONES** | **PUNTUACIÓN** |
| PySpark conteo de palabaras ventanas fijas |  | 2,5 |
|  |  | **2,5** |

|  |  |  |
| --- | --- | --- |
| **EJERCICIO 3** | **OBSERVACIONES** | **PUNTUACIÓN** |
| Crear topic Kafka |  | 0,25 |
| Crear un productor kafka |  | 0,75 |
| Crear el flujo: Nifi-flujo |  | 1,0 |
| Comprobar datos en mongodb |  | 0,25 |
| Descarga el flujo a un archivo .json |  | 0,25 |
|  | | **2,5** |

|  |  |  |
| --- | --- | --- |
| **EJERCICIO 4** | **OBSERVACIONES** | **PUNTUACIÓN** |
| Descargar conector |  | 0,5 |
| Configurar y lanzar el conector |  | 1,75 |
| Comprobar los datos en mongodb |  | 0,25 |
|  | | **2,5** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EJERCICIO 1** | **EJERCICIO 2** | **EJERCICIO 3** | **EJERCICIO 4** | **TOTAL** |
| 2,5 | 2,5 | 2,5 | 2,5 | **10** |