

Nombre: Celic Gabriel Hernández Archundia.		Matrícula: 2877240
Nombre del curso: Infraestructura para Big Data		Nombre del profesor: Miguel de Jesús Martínez Felipe
Módulo II		Actividad: Avance 2 Evidencia 1
Fecha: 12/03/2024		
Bibliografía: Apache Ambari Cloudera. (2023, July 18). Cloudera. https://es.cloudera.com/products/open-source/apache-hadoop/apache-ambari.html		

Capturas Base de Datos Azure

Inicio - Microsoft Azure

https://portal.azure.com/#home

Microsoft Azure

Buscar recursos, servicios y documentos (G+)

Copilot

AL02877240@tecmleni...
UNIVERSIDAD TECMILENIO (UT...)

Servicios de Azure

Crear un recurso

Máquinas virtuales

Cuentas de almacenamiento

Todos los recursos

Centro de inicio rápido

App Services

SQL Database

Azure Cosmos DB

Servicios de Kubernetes

Más servicios

Recursos

Reciente Favorito

Nombre	Tipo	Última consulta
Celcic	Máquina virtual	hace 2 semanas
storagecelcic	Cuenta de almacenamiento	hace 2 semanas
Celcic_group	Grupo de recursos	hace 2 semanas
Celcic_group_02032235	Grupo de recursos	hace 1 mes
Azure subscription 1	Suscripción	hace 1 mes

Ver todo

Navegar

https://portal.azure.com/#create/hub

Celcic - Microsoft Azure

https://portal.azure.com/#@utm.edu.mx/resource/subscriptions/379f8ac8-a7c3-4809-a286-e197de246e3c/resour...

Microsoft Azure

Buscar recursos, servicios y documentos (G+)

Copilot

AL02877240@tecmleni...
UNIVERSIDAD TECMILENIO (UT...)

Inicio >

Celcic

Máquina virtual

Buscar

Conectar Iniciar Reiniciar Detener Hibernar (versión preliminar) Captura Eliminar Actualizar ...

Información general

- Registro de actividad
- Control de acceso (IAM)
- Etiquetas
- Diagnosticar y solucionar problemas

Conectar

- Conectar
- Bastión

Redes

- Configuración de red
- Equilibrio de carga
- Grupos de seguridad de la aplicación
- Administrador de red

Propiedades

Máquina virtual

Nombre del equipo	Celcic
Sistema operativo	Linux
Editor de imagen	canonical
Oferta de imagen	0001-com-ubuntu-server-focal
Plan de imagen	20_04-lts-gen2
Generación de VM	V2
Arquitectura de VM	x64
Hibernación	Deshabilitado
Grupo host	-
Host	-
Grupo con ubicación por proximidad	-
Estado de ubicación	N/D
Grupo de reserva de capacidad	-
Tipo de controladora de disco	SCSI

Redes

Dirección IP pública	20.38.41.3 (Interfaz de red celcic709_z1)
Dirección IP pública (IPv6)	-
Dirección IP privada	10.1.0.4
Dirección IP privada (IPv6)	-
Red virtual/subred	Celcic-vnet/default
Nombre DNS	Configurar

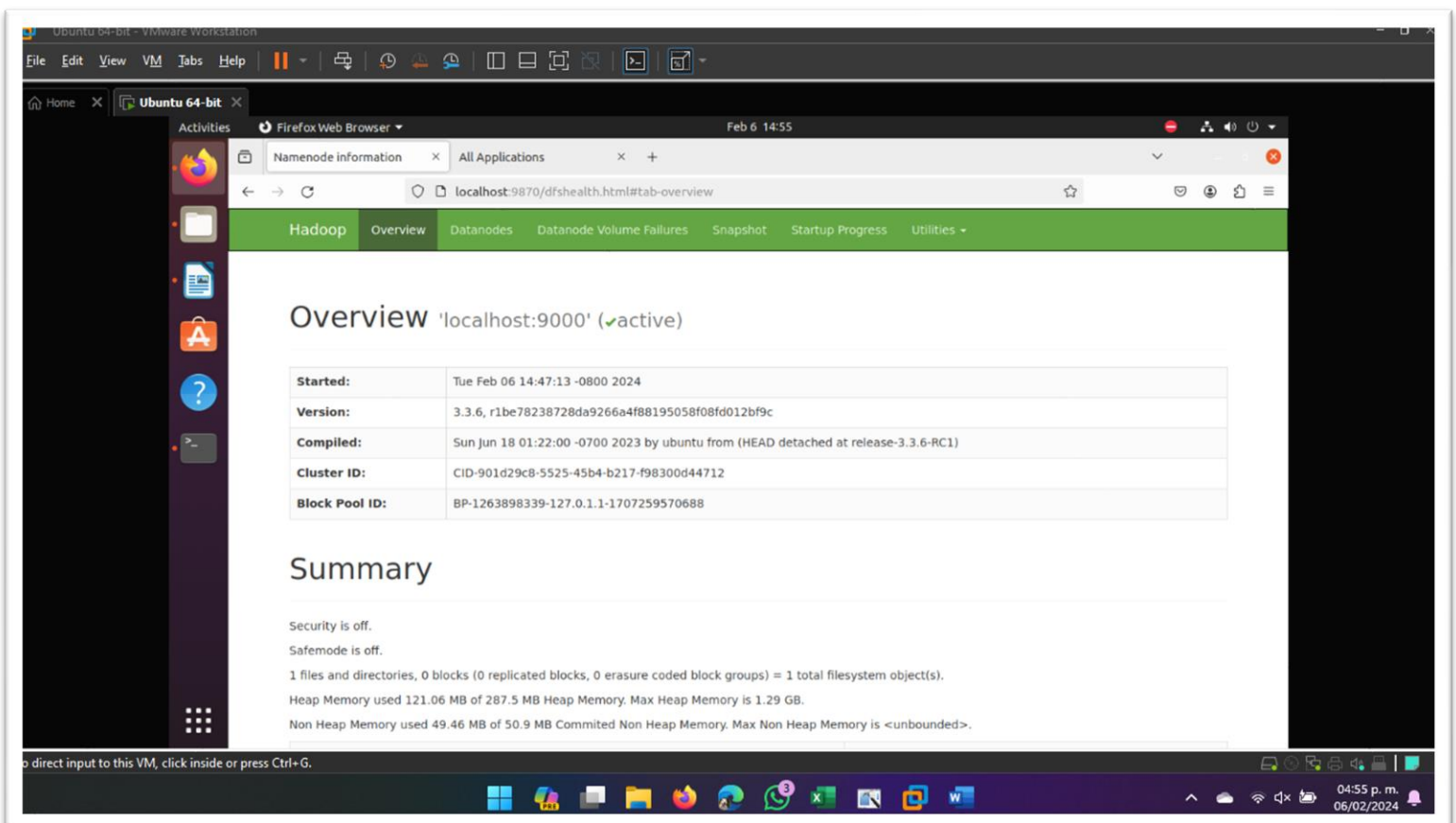
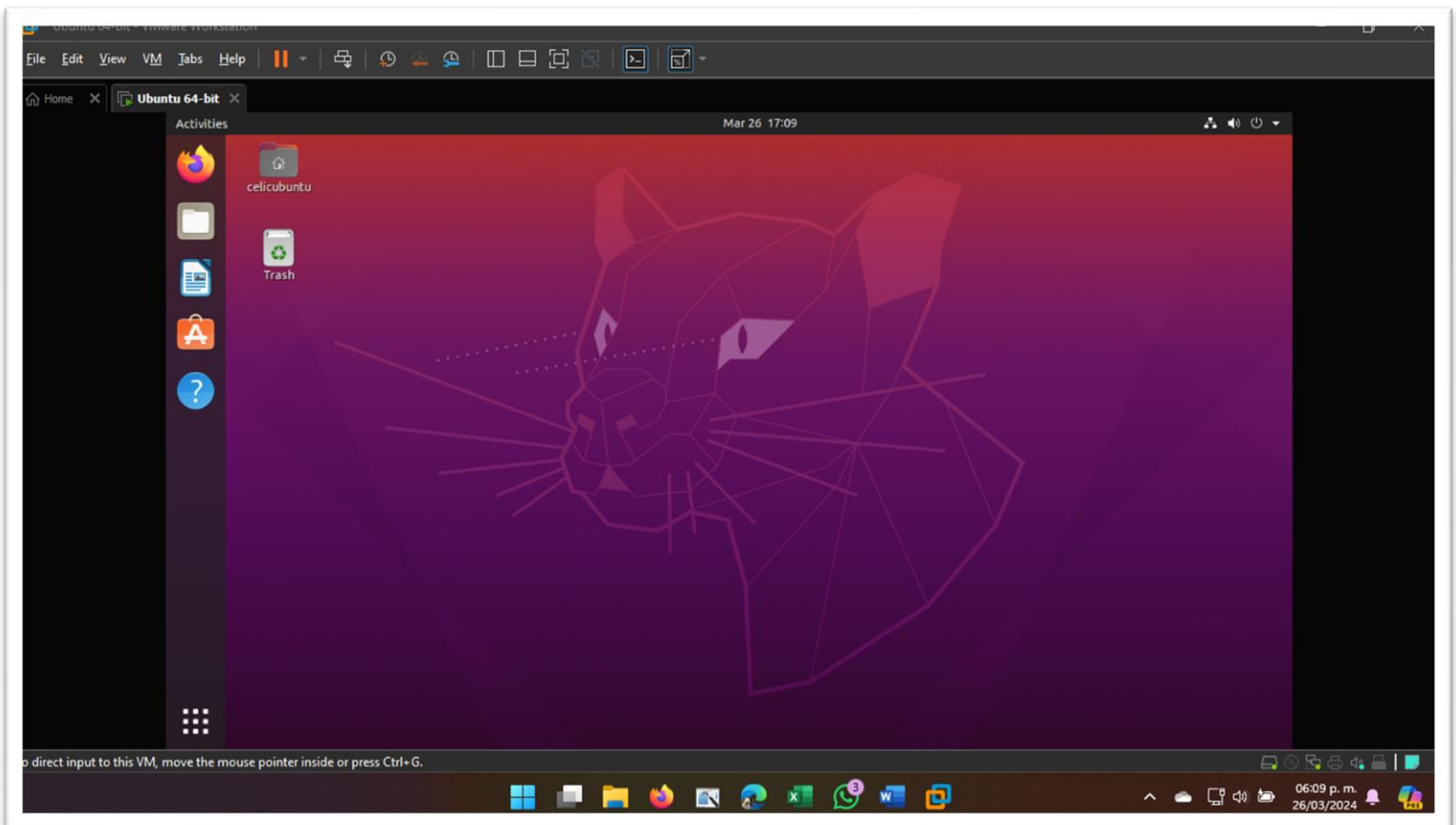
Tamaño

Tamaño	Standard B1s
vCPU	1
RAM	1 GiB

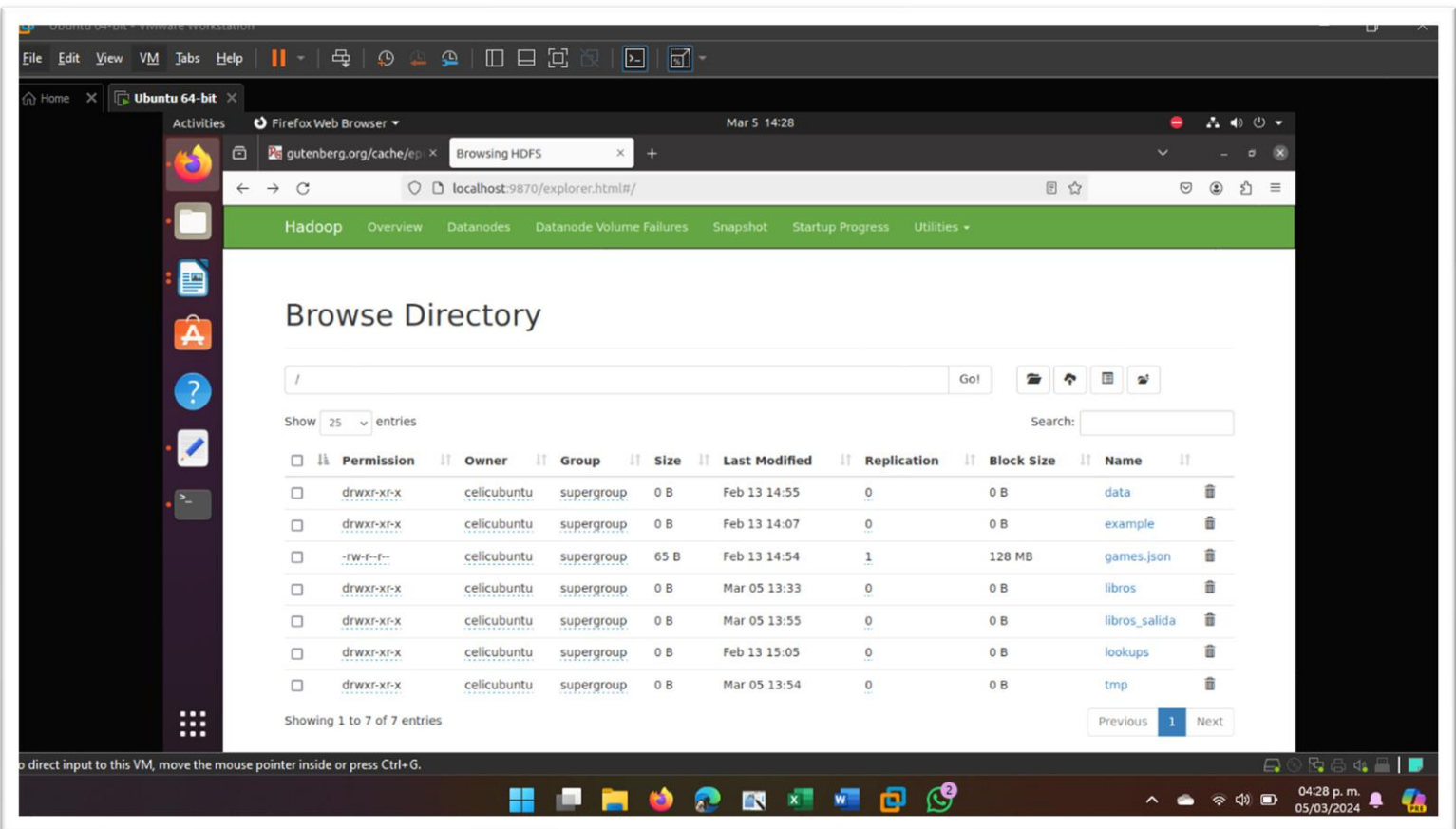
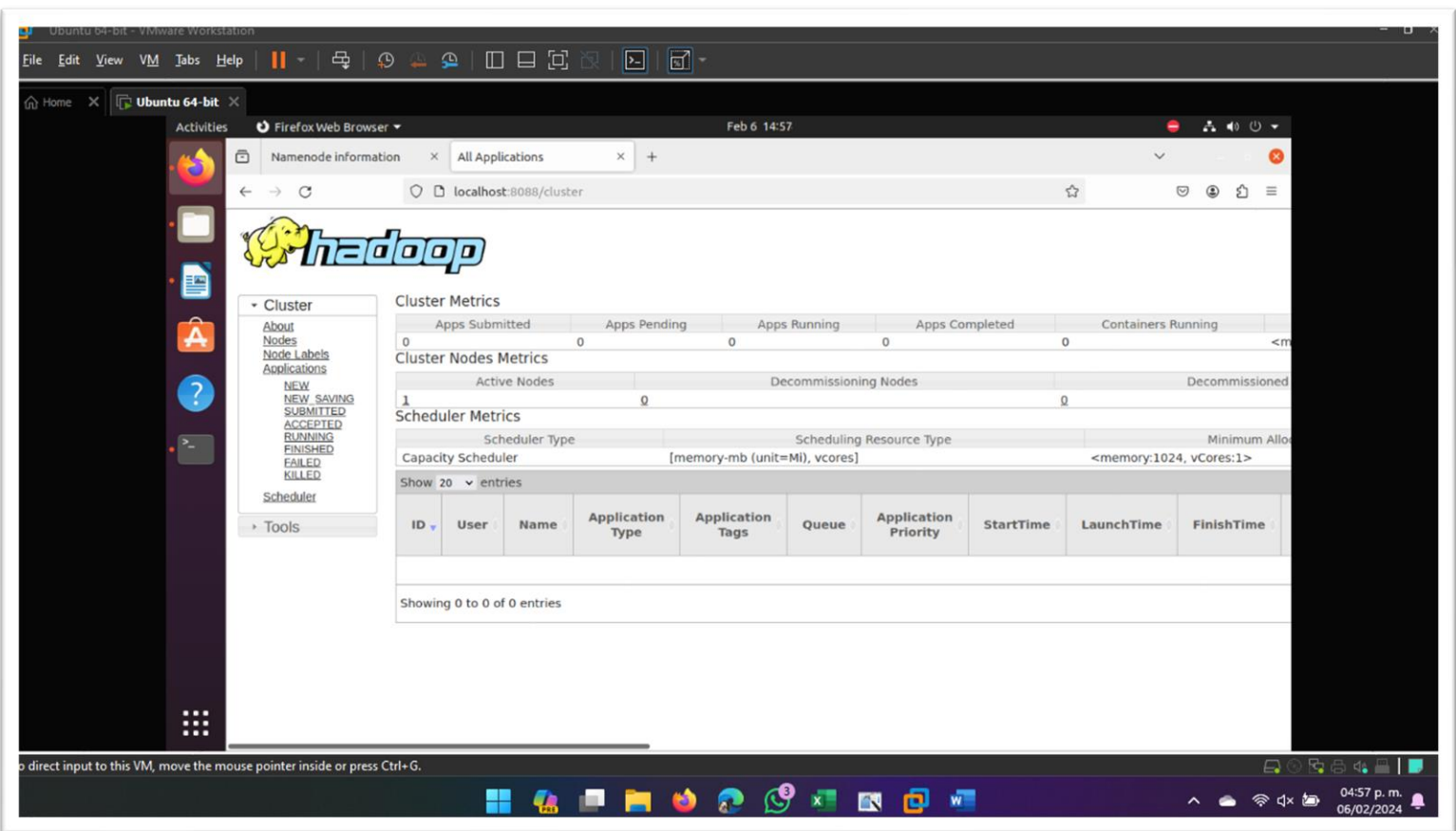
Disco

Disco del SO	Celcic_disk1_923a4d2cc9fd48bc9d517dfbaa7c7e84
Cifrado en el host	Deshabilitado
Azure Disk Encryption	No habilitado

Capturas del Overview y Ubuntu



Capturas Clúster Hadoop



Investigación Ambari View

Antes de hablar de *Ambari View* hay que decir qué es *Apache Ambari*; esta es una plataforma de gestión y administración de clústeres de datos de código abierto que simplifica la implementación, monitorización y administración de entornos de big data. En este caso, *Apache Ambari View* es una característica de *Apache Ambari* que permite a los usuarios personalizar y extender la interfaz de usuario de *Ambari* para incluir funcionalidades específicas de sus aplicaciones o servicios. Se puede decir que es una forma de crear paneles y vistas personalizadas dentro de la interfaz de usuario de *Ambari*.

Con *Ambari View* los desarrolladores pueden construir y agregar nuevos componentes visuales a la interfaz de usuario de *Ambari*; esto permite a los usuarios interactuar con diferentes aspectos de sus clústeres de datos de una manera más intuitiva y específica según sus necesidades.

Algunos ejemplos de lo que se puede lograr con *Ambari View* incluyen:

1. **Paneles de control personalizados:** Los desarrolladores pueden crear paneles de control personalizados que proporcionan métricas y visualizaciones específicas para los servicios o aplicaciones que se están ejecutando en el clúster. Por ejemplo, un panel de control de Hadoop podría mostrar el estado de los nodos del clúster, la utilización de recursos y la actividad de MapReduce.
2. **Visualizaciones de datos avanzadas:** *Ambari View* permite a los desarrolladores integrar visualizaciones de datos avanzadas dentro de la interfaz de usuario de *Ambari*. Esto podría incluir gráficos interactivos, diagramas de dispersión, mapas de calor y otras visualizaciones que ayuden a los usuarios a comprender mejor los datos que están siendo procesados por el clúster.
3. **Herramientas de gestión específicas:** Los desarrolladores pueden crear herramientas de gestión específicas que simplifican tareas comunes de administración y operaciones en el clúster. Por ejemplo, una vista podría proporcionar una interfaz para la configuración y administración de tablas en *HBase* o *Hive*.

Capturas MongoDB

MongoDB Compass - localhost:27017/estudianteDB.estudiantes

Connect Edit View Collection Help

localhost:27017

My Queries estudianteDB estudiantes Databases

estudianteDB.estudiantes

0 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } or [Generate query](#) Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

1 - 11 of 11

<pre>{ "_id": ObjectId("65f0e1fc9a30cd58f1ce0afb"), "Nombre": "Celica", "Apellido": "Hernández", "Edad": 21, "Calificación promedio": 98.5 }</pre>
<pre>{ "_id": ObjectId("65f0e1fc9a30cd58f1ce0afc"), "Nombre": "Emilio", "Apellido": "Sonck", "Edad": 21, "Calificación promedio": 90 }</pre>
<pre>{ "_id": ObjectId("65f0e1fc9a30cd58f1ce0afd"), "Nombre": "Jade", "Apellido": "Ullolita", "Edad": 20, "Calificación promedio": 95 }</pre>

>_MONGOSH

09:45 p. m. 12/03/2024

MongoDB Compass - localhost:27017/estudianteDB.collectionEstudiante.estudiantes

Connect Edit View Collection Help

localhost:27017

My Queries estudianteDB estudiantes Databases collectionEstudiante.estudia...

estudianteDB.collectionEstudiante.estudiantes

11 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } or [Generate query](#) Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

1 - 11 of 11

<pre>{ "_id": ObjectId("65f0dfc89a30cd58f1ce0af0"), "Nombre": "Celica", "Apellido": "Hernández", "Edad": 21, "Calificación promedio": 98.5 }</pre>
<pre>{ "_id": ObjectId("65f0dfc89a30cd58f1ce0af1"), "Nombre": "Emilio", "Apellido": "Sonck", "Edad": 21, "Calificación promedio": 90 }</pre>
<pre>{ "_id": ObjectId("65f0dfc89a30cd58f1ce0af2"), "Nombre": "Jade", "Apellido": "Ullolita", "Edad": 20, "Calificación promedio": 95 }</pre>

>_MONGOSH

09:46 p. m. 12/03/2024


```
4_coach_details_mongodb_crud x +
localhost:8888/notebooks/4_coach_details_mongodb_crud.ipynb
jupyter 4_coach_details_mongodb_crud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
+ X Copy Paste Run Code
JupyterLab Python 3 (ipykernel)

[11]: docsEstudiantes = [
      { "Nombre": "Celic", "Apellido": "Hernández", "Edad": 21, "Calificación promedio": 98.5 },
      { "Nombre": "Emilio", "Apellido": "Sonck", "Edad": 21, "Calificación promedio": 90 },
      { "Nombre": "Jade", "Apellido": "Ullolita", "Edad": 20, "Calificación promedio": 95 },
      { "Nombre": "Marco", "Apellido": "Polo", "Edad": 21, "Calificación promedio": 93 },
      { "Nombre": "Luis", "Apellido": "Lecham", "Edad": 20, "Calificación promedio": 99.1 },
      { "Nombre": "Diego", "Apellido": "Salsa", "Edad": 20, "Calificación promedio": 90.2 },
      { "Nombre": "Diego", "Apellido": "Malerva", "Edad": 21, "Calificación promedio": 97 },
      { "Nombre": "Miguel", "Apellido": "Mendoza", "Edad": 22, "Calificación promedio": 96 },
      { "Nombre": "Mariana", "Apellido": "González", "Edad": 21, "Calificación promedio": 94.6 },
      { "Nombre": "Regina", "Apellido": "Ferreira", "Edad": 21, "Calificación promedio": 98 },
      { "Nombre": "Benito", "Apellido": "Camargo", "Edad": 21, "Calificación promedio": 89.5 }
    ]

    # Create Database:
    dbEstudiantes = client.estudianteDB

    # Create Collection:
    collectionEstudiante = dbEstudiantes["estudiantes"]

[6]: # Dummy documents:
     test_docs = [
       { "first_name": "test_1", "gender": "F", "shift": "AM", "ph_num": 234564980, "trains_for": ["tennis", "badminton"], "manager": "Jack" },
       { "first_name": "test_2", "gender": "M", "shift": "AM", "ph_num": 987654321, "trains_for": ["valleyball", "basketball"], "manager": "Amin" },
       { "_id": 14, "first_name": "test_3", "gender": "M", "shift": "AM", "ph_num": 987654321, "trains_for": ["valleyball", "basketball"], "manager": "Amin" }
     ]

[7]: # Dummy Document:
     test_doc = { "_id": 14, "first_name": "test_3", "gender": "M", "shift": "AM", "ph_num": 987654321, "trains_for": ["valleyball", "basketball"], "manager": "Amin" }
```

```
4_coach_details_mongodb_crud x +
localhost:8888/notebooks/4_coach_details_mongodb_crud.ipynb
jupyter 4_coach_details_mongodb_crud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
+ X Copy Paste Run Code
JupyterLab Python 3 (ipykernel)

To insert documents to the above collection 'coach_details':

[7]: # To insert one document:
     collection.insert_one(doc_1)

[7]: InsertOneResult(1, acknowledged=True)

[6]: # To insert multiple documents:
     collection.insert_many(docs)

[6]: InsertManyResult([ObjectId('65f0dc3b9a30cd58f1ce0ae7'), ObjectId('65f0dc3b9a30cd58f1ce0ae8'), ObjectId('65f0dc3b9a30cd58f1ce0ae9'), ObjectId('65f0dc3b9a30cd58f1ce0aea'), ObjectId('65f0dc3b9a30cd58f1ce0aeb'), ObjectId('65f0dc3b9a30cd58f1ce0aec'), ObjectId('65f0dc3b9a30cd58f1ce0aed'), ObjectId('65f0dc3b9a30cd58f1ce0aee'), ObjectId('65f0dc3b9a30cd58f1ce0aef')], acknowledged=True)

[12]: # Insertar a mis estudiantes
      collectionEstudiante.insert_many(docsEstudiantes)

[12]: InsertManyResult([ObjectId('65f0e1fc9a30cd58f1ce0afb'), ObjectId('65f0e1fc9a30cd58f1ce0afc'), ObjectId('65f0e1fc9a30cd58f1ce0afd'), ObjectId('65f0e1fc9a30cd58f1ce0afe'), ObjectId('65f0e1fc9a30cd58f1ce0aff'), ObjectId('65f0e1fc9a30cd58f1ce0b00'), ObjectId('65f0e1fc9a30cd58f1ce0b01'), ObjectId('65f0e1fc9a30cd58f1ce0b02'), ObjectId('65f0e1fc9a30cd58f1ce0b03'), ObjectId('65f0e1fc9a30cd58f1ce0b04'), ObjectId('65f0e1fc9a30cd58f1ce0b05')], acknowledged=True)

* [10]: # To insert test documents:
        collection.insert_many(test_docs)

[10]: InsertManyResult([ObjectId('655bf02cc1a00c9ba5e668e9'), ObjectId('655bf02cc1a00c9ba5e668ea'), 14], acknowledged=True)

To fetch/read documents:

[11]: # To fetch specific document by passing ID:
```