



Mongo DB Usuarios y Roles



www.ute.edu.ec



Se basa en varios principios:

Autenticación

Verifica la identidad de usuarios. Métodos:

- ✓ Autenticación interna (nombre de usuario y contraseña).
- ✓ LDAP.
- ✓ Kerberos.
- ✓ Autenticación por certificados x.509.

Autorización

Controla qué puede hacer cada usuario:

- ✓ MongoDB usa un sistema basado en roles (RBAC).
- ✓ Roles como read, readWrite, dbAdmin, entre otros.
- ✓ Se pueden crear roles personalizados.



Determinan las acciones que un usuario puede realizar en la base de datos. A continuación, se presenta una tabla con los principales roles incorporados y sus respectivos permisos:

Rol	Descripción	Permisos Clave
read	Permite leer datos de todas las colecciones no sistemáticas y de la colección system.js.	find, listCollections, listIndexes
readWrite	Incluye todos los permisos del rol read y permite modificar datos en todas las colecciones no sistemáticas y en la colección system.js.	insert, update, delete, find, listCollections, listIndexes
dbAdmin	Permite realizar tareas administrativas como la creación de índices, la recopilación de estadísticas y la gestión de esquemas. No incluye permisos para gestionar usuarios o roles.	createCollection, dropCollection, createIndex, dropIndex, collStats, dbStats
userAdmin	Permite crear y modificar roles y usuarios en la base de datos actual.	createUser, dropUser, grantRole, revokeRole



Role	Descripción	Permisos Clave
clusterAdmin	Proporciona acceso administrativo a todo el clúster. Incluye la capacidad de gestionar y monitorear el estado del clúster.	addShard, replSetConfigure, serverStatus, top, listDatabases
readAnyDatabase	Permite leer datos de todas las bases de datos, excepto las bases de datos system. Este rol solo se define en la base de datos admin.	find, listCollections, listIndexes en todas las bases de datos
readWriteAnyDatabase	Incluye todos los permisos del rol readAnyDatabase y permite modificar datos en todas las bases de datos. Este rol solo se define en la base de datos admin.	insert, update, delete, find, listCollections, listIndexes en todas las bases de datos
dbAdminAnyDatabase	Proporciona las mismas capacidades que dbAdmin, pero para todas las bases de datos. Este rol solo se define en la base de datos admin.	createCollection, dropCollection, createIndex, dropIndex, collStats, dbStats en todas las bases de datos
userAdminAnyDatabase	Permite crear y modificar roles y usuarios en todas las bases de datos. Este rol solo se define en la base de datos admin.	createUser, dropUser, grantRole, revokeRole en todas las bases de datos
root	Proporciona acceso completo a todas las operaciones y recursos. Es el rol de superusuario.	Todos los permisos de los roles anteriores combinados

Url:

<https://www.mongodb.com/try/download/shell>

Version

2.5.5



Platform

Windows x64 (10+)



Package

zip



Download



Copy link

More Options



Habilitar la
autenticación
en MongoDB

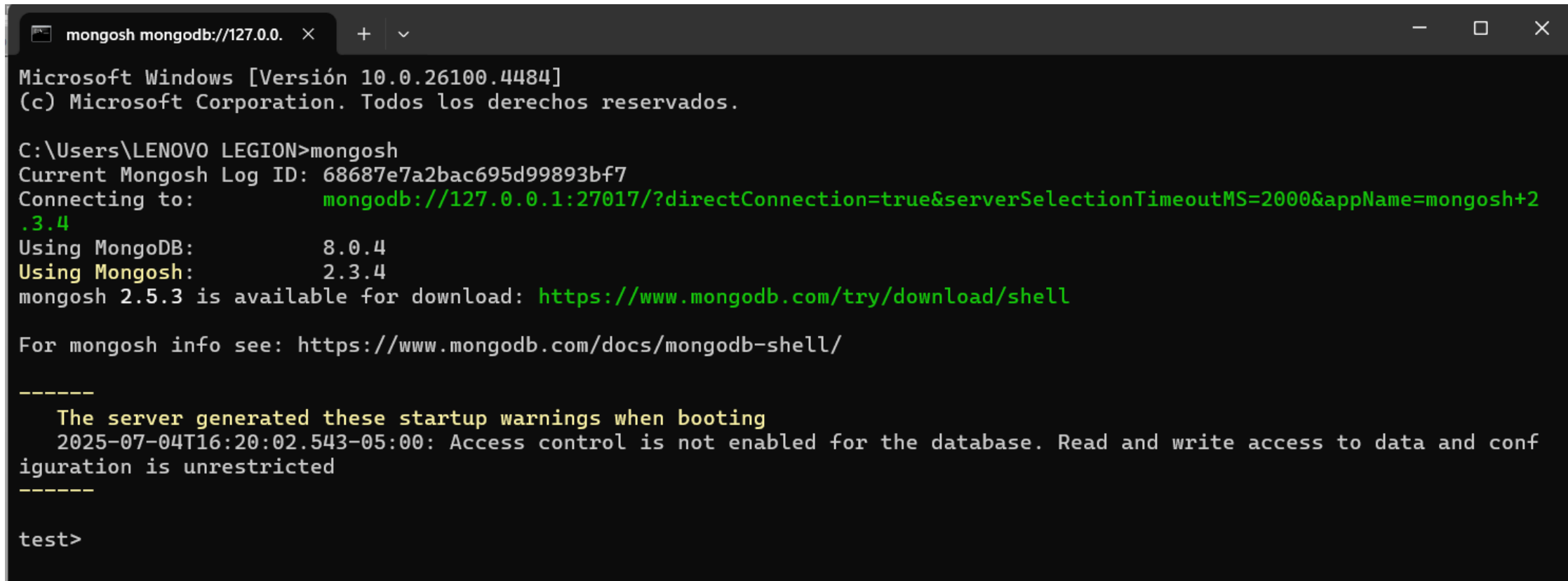
Usuarios con
permisos
específicos

Roles
personalizados

Mejores
prácticas de
seguridad

1. Presionamos Windows + R
2. cmd
3. Digitamos mongosh

C:\Program Files\MongoDB\Server\8.0\bin



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.4
Microsoft Windows [Versión 10.0.26100.4484]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\LENOVO LEGION>mongosh
Current Mongosh Log ID: 68687e7a2bac695d99893bf7
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.4
Using MongoDB:      8.0.4
Using Mongosh:       2.3.4
mongosh 2.5.3 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

-----
  The server generated these startup warnings when booting
  2025-07-04T16:20:02.543-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test>
```

Opción: 1

Creación de Usuario root

```
db.createUser({
  user: "root",
  pwd: "sa.1",
  roles: [{ role: "root", db: "admin" }]
})
```

Opción: 2

```
db.createUser({ user: "root", pwd: "sa.1", roles: [{ role: "root", db: "admin" }] })
```

```
test> use admin
switched to db admin
admin> show collections
system.roles
system.users
system.version
admin> db.createUser({
... user:"root",
... pwd:"sa.1",
... roles:[{role:"root",db:"admin"}]
... })
{ ok: 1 }
admin>
```



```
test> use admin
switched to db admin
admin> show collections
system.roles
system.users
system.version
admin>
```

use admin

show collections

Antes:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# Where and how to store data.
storage:
  dbPath: C:\Program Files\MongoDB\Server\8.0\data

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: C:\Program Files\MongoDB\Server\8.0\log\mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 127.0.0.1

#processManagement:

#security:

#operationProfiling:

#replication:

#sharding:

## Enterprise-Only Options:

#auditLog:
```

Despues:

```
# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# Where and how to store data.
storage:
  dbPath: C:\Program Files\MongoDB\Server\8.0\data

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: C:\Program Files\MongoDB\Server\8.0\log\mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 127.0.0.1

#processManagement:

security:
  authorization: enabled
#security:

#operationProfiling:

#replication:

#sharding:

## Enterprise-Only Options:

#auditLog:
```

Servicios (locales)

MongoDB Server (MongoDB)

[Detener](#) el servicio
[Reiniciar](#) el servicio

Descripción:
MongoDB Database Server
(MongoDB)

Nombre

McAfee Framework Host
McAfee Software Update
McAfee WebAdvisor
McpManagementService
MessagingService_20fdc6b
Microsoft Account Sign-in Assistant
Microsoft Edge Elevation Service (MicrosoftEdgeElevationService)
Microsoft Edge Update Service (edgeupdate)
Microsoft Edge Update Service (edgeupdatem)
Microsoft Office Click-to-Run Service
Microsoft Passport
Microsoft Storage Spaces SMP
Modo incrustado
Módulos de creación de claves de IPsec para IKE y AuthIP
MongoDB Server (MongoDB)

Descripción	Estado	Tipo de inicio	Iniciar sesión como
McAfee Fra...	En ejecu...	Automático	Sistema local
		Manual	Sistema local
McAfee Web...	En ejecu...	Automático	Sistema local
Servicio de a...		Manual	Sistema local
El servicio au...		Manual (desen...	Sistema local
Enables user...		Manual (desen...	Sistema local
Mantiene ac...		Manual	Sistema local
Mantiene ac...		Automático (in...	Sistema local
Mantiene ac...		Manual (desen...	Sistema local
Administra l...	En ejecu...	Automático	Sistema local
Ofrece aisla...		Manual (desen...	Sistema local
Host service ...		Manual	Servicio de red
El servicio de...		Manual (desen...	Sistema local
El servicio IK...		Manual (desen...	Sistema local
MongoDB D...	En ejecu...	Automático	Servicio de red

Si sale este error asegúrate :

1. Que este bien escrito la instrucción de security
2. Cierra todas las ventanas de MongoDB

```
# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: C:\Program Files\MongoDB\Server\8.0\log\mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 127.0.0.1

#processManagement:

#security:
security:
  authorization: enabled
```

Servicios

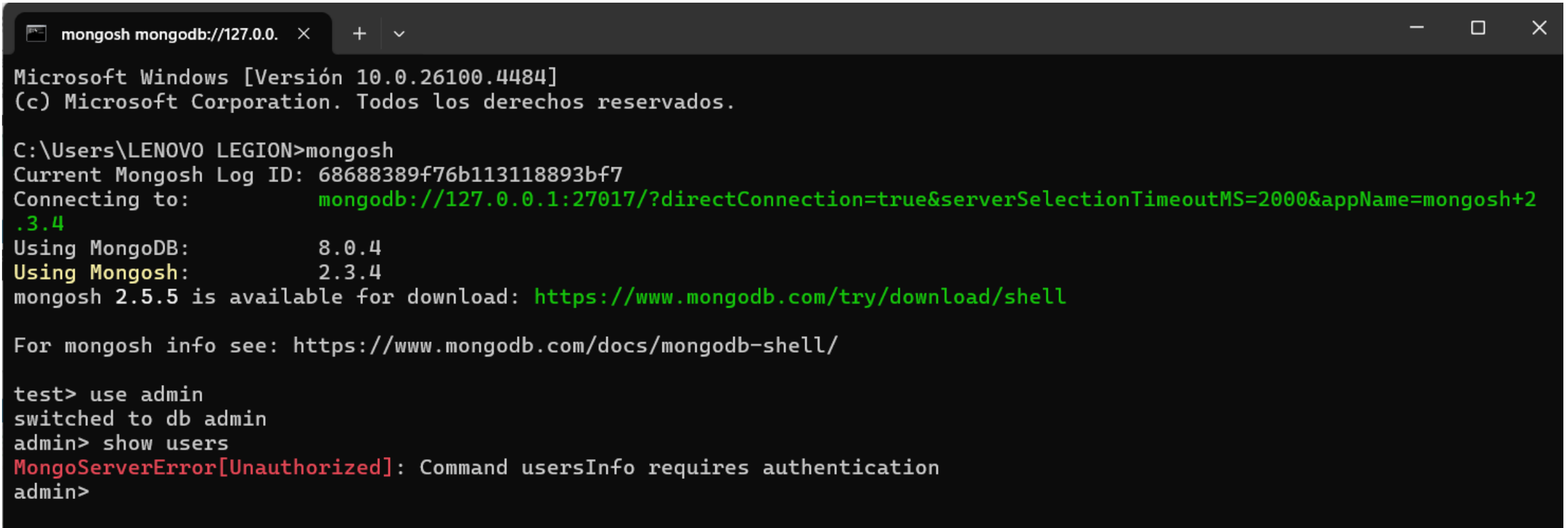


Windows no pudo iniciar el servicio MongoDB Server (MongoDB) en Equipo local.

Error 1053: El servicio no respondió a tiempo a la solicitud de inicio o de control.

Aceptar

Al momento de logearse nos tiene que solicitar las credenciales tal como se muestra en la imagen:



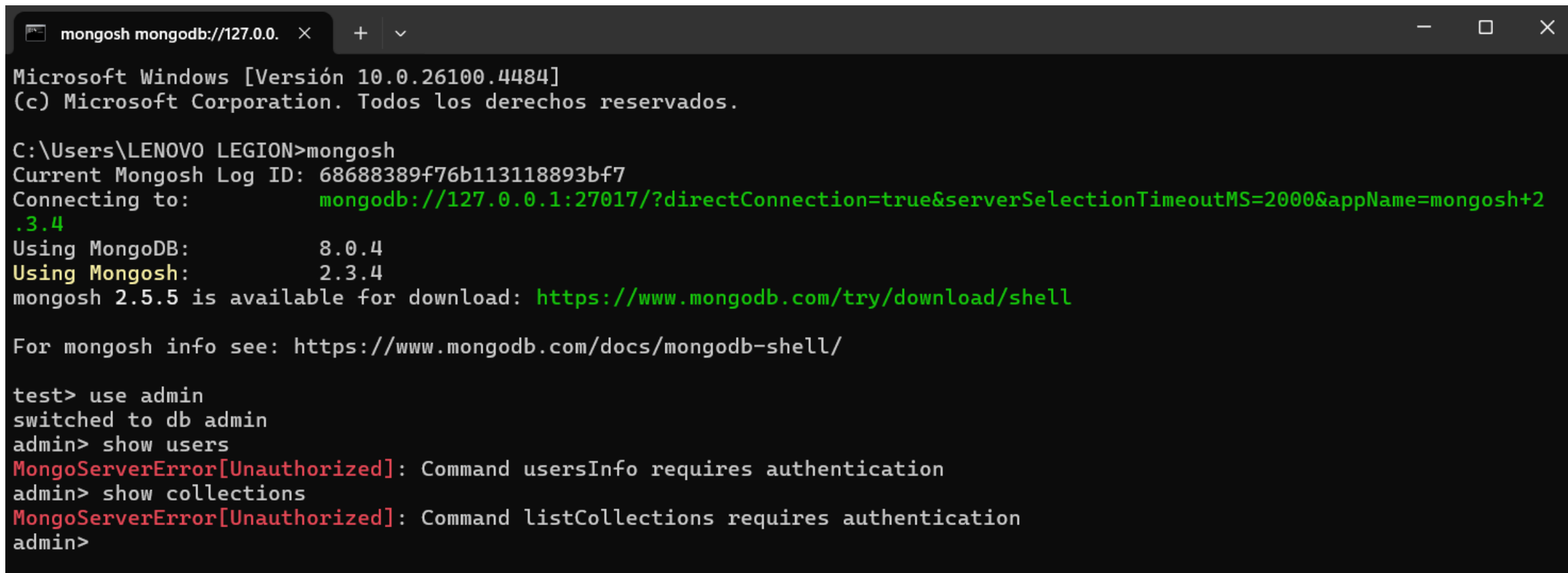
```
Microsoft Windows [Versión 10.0.26100.4484]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\LENOVO LEGION>mongosh
Current Mongosh Log ID: 68688389f76b113118893bf7
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.4
Using MongoDB:      8.0.4
Using Mongosh:       2.3.4
mongosh 2.5.5 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

test> use admin
switched to db admin
admin> show users
MongoServerError[Unauthorized]: Command usersInfo requires authentication
admin>
```

Al momento de logearse nos tiene que solicitar las credenciales tal como se muestra en la imagen:



```
Microsoft Windows [Versión 10.0.26100.4484]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\LENOVO LEGION>mongosh
Current Mongosh Log ID: 68688389f76b113118893bf7
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.4
Using MongoDB:      8.0.4
Using Mongosh:       2.3.4
mongosh 2.5.5 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

test> use admin
switched to db admin
admin> show users
MongoServerError[Unauthorized]: Command usersInfo requires authentication
admin> show collections
MongoServerError[Unauthorized]: Command listCollections requires authentication
admin>
```

Ingreso de credenciales:

```
test> use admin
switched to db admin
admin> show users
MongoServerError[Unauthorized]: Command usersInfo requires authentication
admin> show collections
MongoServerError[Unauthorized]: Command listCollections requires authentication
admin> db.auth("root","sa.1")
{ ok: 1 }
admin> show collections
system.roles
system.users
system.version
admin>
```

Visualización de usuarios:

```
admin> show users
[
  {
    _id: 'admin.cristian_lector',
    userId: UUID('d6fa2a93-88da-4f1f-9a23-24c59522d701'),
    user: 'cristian_lector',
    db: 'admin',
    roles: [ { role: 'read', db: 'sri' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'admin.root',
    userId: UUID('efb801b9-88e3-4aaf-bdd5-ebdc501ce12c'),
    user: 'root',
    db: 'admin',
    roles: [ { role: 'root', db: 'admin' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  }
]
admin>
```

Crear la colección de conductores

```
admin> use amt
switched to db amt
amt> show collections
conductores
vehiculos
amt> db.conductores.find()
[
  { _id: 101, nombre: 'Cristian', licencia: 'Tipo B' },
  { _id: 102, nombre: 'Andrea', licencia: 'Tipo C' },
  { _id: 103, nombre: 'Juan', licencia: 'Tipo A' }
]
amt>
```

Crear usuario para la colección amt (Permisos de Lectura)

```
amt> db.createUser({ user:"usuario_lector", pwd:"Lector123", roles:[{role:"read",db:"amt"}]})
{ ok: 1 }
amt>
```

```
db.createUser({ user:"usuario_lector",
pwd:"Lector123",
roles:[{role:"read",db:"amt"}]})
```

```
amt> db.auth("usuario_lector","Lector123")
```

Visualización de usuarios en la colección amt:

```
amt> show users
[
  {
    _id: 'amt.usuario_lector',
    userId: UUID('f86c5757-0f3a-40d8-b928-e64e41c81ac1'),
    user: 'usuario_lector',
    db: 'amt',
    roles: [ { role: 'read', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  }
]
amt>
```

Probamos el usuario lector en la collection amt:

```
amt> db.auth("usuario_lector","Lector123")
{ ok: 1 }
amt> show users
MongoServerError[Unauthorized]: not authorized on amt to execute command { usersInfo: 1, lsid: { id: UUID("43e17e7c-351b-4778-8d94-c1e6b4e3ae0f") }, $db: "amt" }
amt> |
```

```
amt> show collections
conductores
vehiculos
amt> db.conductores.find()
[
  { _id: 101, nombre: 'Cristian', licencia: 'Tipo B' },
  { _id: 102, nombre: 'Andrea', licencia: 'Tipo C' },
  { _id: 103, nombre: 'Juan', licencia: 'Tipo A' }
]
amt>
```


Probamos el usuario lector en la collection amt:

```
amt> db.usuarios.insertOne({nombre:"Pero",licencia:"Tipo G"})|
```

El mensaje que nos muestra es correcto porque solos puede realizar consultas no (operaciones crud) en amt:

```
amt> db.usuarios.insertOne({nombre:"Pero",licencia:"Tipo G"})
MongoServerError[Unauthorized]: not authorized on amt to execute command { insert: "usuarios", documents: [ { nombre: "Pero", licencia: "Tipo G", _id: ObjectId('68688953f76b113118893bf8') } ], ordered: true, lsid: { id: UUID("43e17e7c-351b-4778-8d94-c1e6b4e3ae0f") }, $db: "amt" }
amt>
```

Ingreso de credenciales

```
switched to db admin
admin> db.auth("root","sa.1")
{ ok: 1 }
admin>
```

Cambio de Base de Datos

```
amt> db.createUser({ user:"Editor",
... pwd:"Editor123",
... roles:[{role:"readWrite",db:"amt"}]
... })
```

Consultar los usuarios de la Base de Datos

```
amt> db.createUser({ user:"Editor",
... pwd:"Editor123",
... roles:[{role:"readWrite",db:"amt"}]
... })
{ ok: 1 }
amt> show users
[
  {
    _id: 'amt.Editor',
    userId: UUID('1b0cd1b8-70fb-47f4-98e5-37997977b3db'),
    user: 'Editor',
    db: 'amt',
    roles: [ { role: 'readWrite', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'amt.usuario_lector',
    userId: UUID('f86c5757-0f3a-40d8-b928-e64e41c81ac1'),
    user: 'usuario_lector',
    db: 'amt',
    roles: [ { role: 'read', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  }
]
amt>
```

```
amt> db.auth("Editor","Editor123")
```

```
amt> db.conductores.find()
[
  { _id: 101, nombre: 'Cristian', licencia: 'Tipo B' },
  { _id: 102, nombre: 'Andrea', licencia: 'Tipo C' },
  { _id: 103, nombre: 'Juan', licencia: 'Tipo A' }
]
amt>
```

Realizamos la prueba con el usuario y validamos que si se pueda relzar operaciones crud

```
amt> db.conductores.insertOne({nombre:"Pero",licencia:"Tipo G"})
{
  acknowledged: true,
  insertedId: ObjectId('68688c6df76b113118893bf9')
}
amt>
```

Resultado

```
amt> db.conductores.find()
[
  { _id: 101, nombre: 'Cristian', licencia: 'Tipo B' },
  { _id: 102, nombre: 'Andrea', licencia: 'Tipo C' },
  { _id: 103, nombre: 'Juan', licencia: 'Tipo A' },
  {
    _id: ObjectId('68688c6df76b113118893bf9'),
    nombre: 'Pero',
    licencia: 'Tipo G'
  }
]
amt> |
```

Visualización de la collections conductores

```
amt> db.vehiculos.find()
[
  { _id: 1, placa: 'ABC123', marca: 'Toyota', conductor_id: 101 },
  { _id: 2, placa: 'XYZ789', marca: 'Chevrolet', conductor_id: 102 },
  { _id: 3, placa: 'LMN456', marca: 'Hyundai', conductor_id: 103 }
]
amt>
```

Roles

```
amt> show roles
[
  {
    role: 'read',
    db: 'amt',
    isBuiltin: true,
    roles: [],
    inheritedRoles: []
  },
  {
    role: 'readWrite',
    db: 'amt',
    isBuiltin: true,
    roles: [],
    inheritedRoles: []
  },
  {
    role: 'userAdmin',
    db: 'amt',
    isBuiltin: true,
    roles: [],
    inheritedRoles: []
  }
]
```

Rol personalizado

```
amt> db.createRole({
... role:"insert_clientes",
... privileges:[{
... resource:{db:"amt",
... collection:"vehiculos",
... actions:["insert"]}
... }],roles:[]})
```

Creación de rol específico

```
amt> db.createRole({ role:"insert_vehiculos", privileges:[{ resource:{db:"amt", collection:"vehiculos"}, actions:["insert"]}], roles:[]})
```

```
{
  _id: 'amt.insert_clientes',
  role: 'insert_clientes',
  db: 'amt',
  roles: [],
  isBuiltin: false,
  inheritedRoles: []
},
{
  _id: 'amt.insert_vehiculos',
  role: 'insert_vehiculos',
  db: 'amt',
  roles: [],
  isBuiltin: false,
  inheritedRoles: []
}
]
amt>
```

Creación de usuario gestor de vehículos

```
amt> db.createUser({  
... user:"gestor_vehiculos",  
... pwd:"gestor",  
... roles:[{role:"insert_vehiculos",db: "amt"}]  
... })
```

Creación de rol específico gestor_vehiculos

```
amt> db.createUser({  
... user:"gestor_vehiculos",  
... pwd:"gestor",  
... roles:[{role:"insert_vehiculos",db: "amt"}]  
... })  
{ ok: 1 }  
amt>
```

Visualización de usuarios.

```
amt> show users
[
  {
    _id: 'amt.Editor',
    userId: UUID('1b0cd1b8-70fb-47f4-98e5-37997977b3db'),
    user: 'Editor',
    db: 'amt',
    roles: [ { role: 'readWrite', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'amt.gestor_vehiculos',
    userId: UUID('5ac2a0f5-5379-4ccc-9afb-10dbd6876b9f'),
    user: 'gestor_vehiculos',
    db: 'amt',
    roles: [ { role: 'insert_vehiculos', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'amt.usuario_lector',
    userId: UUID('f86c5757-0f3a-40d8-b928-e64e41c81ac1'),
    user: 'usuario_lector',
    db: 'amt',
    roles: [ { role: 'read', db: 'amt' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  }
]
amt>
```


Creación de usuario gestor de vehículos

```
amt> db.auth("gestor_vehiculos","gestor")|
```

```
amt> db.auth("gestor_vehiculos","gestor")  
{ ok: 1 }  
amt>
```

Validation de que este usuario solo permite leer y no insertar vehiculos

```
amt> db.auth("gestor_vehiculos","gestor")  
{ ok: 1 }  
amt> db.vehiculos.find()  
MongoServerError[Unauthorized]: not authorized on amt to execute command { find: "vehiculos", filter: {}, lsid: { id: UUIID("7185acf0-39c1-4b1a-b859-90601644c89d") }, $db: "amt" }  
amt>
```

Visualización de la collection vehículos

```
amt> db.vehiculos.find()  
[  
  { _id: 1, placa: 'ABC123', marca: 'Toyota', conductor_id: 101 },  
  { _id: 2, placa: 'XYZ789', marca: 'Chevrolet', conductor_id: 102 },  
  { _id: 3, placa: 'LMN456', marca: 'Hyundai', conductor_id: 103 }  
]  
amt>
```

```
amt> db.auth("gestor_vehiculos","gestor")
{ ok: 1 }
amt> db.vehiculos.find()
MongoServerError[Unauthorized]: not authorized on amt to execute command { find: "vehiculos", filter: {}, lsid: { id: UUID("7185acf0-39c1-4b1a-b859-90601644c89d") }, $db: "amt" }
amt> db.conductores.find()
MongoServerError[Unauthorized]: not authorized on amt to execute command { find: "conductores", filter: {}, lsid: { id: UUID("7185acf0-39c1-4b1a-b859-90601644c89d") }, $db: "amt" }
amt> db.vehiculos.insertOne({id:4,placa:"PCT897",marca:"ISUZU",conductor_id:101}
... )
{
  acknowledged: true,
  insertedId: ObjectId('6868981ff76b113118893bfa')
}
amt> db.vehiculos.find()
MongoServerError[Unauthorized]: not authorized on amt to execute command { find: "vehiculos", filter: {}, lsid: { id: UUID("7185acf0-39c1-4b1a-b859-90601644c89d") }, $db: "amt" }
amt>
```



Gracias