Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

BDI

Name: Kartik Jolapara **Branch:** Computer Engineering

SAP ID: 60004200107 **Batch:** B1

EXPERIMENT NO. 8
MongoDB

AIM: Perform CRUD Operations using MongoDB.

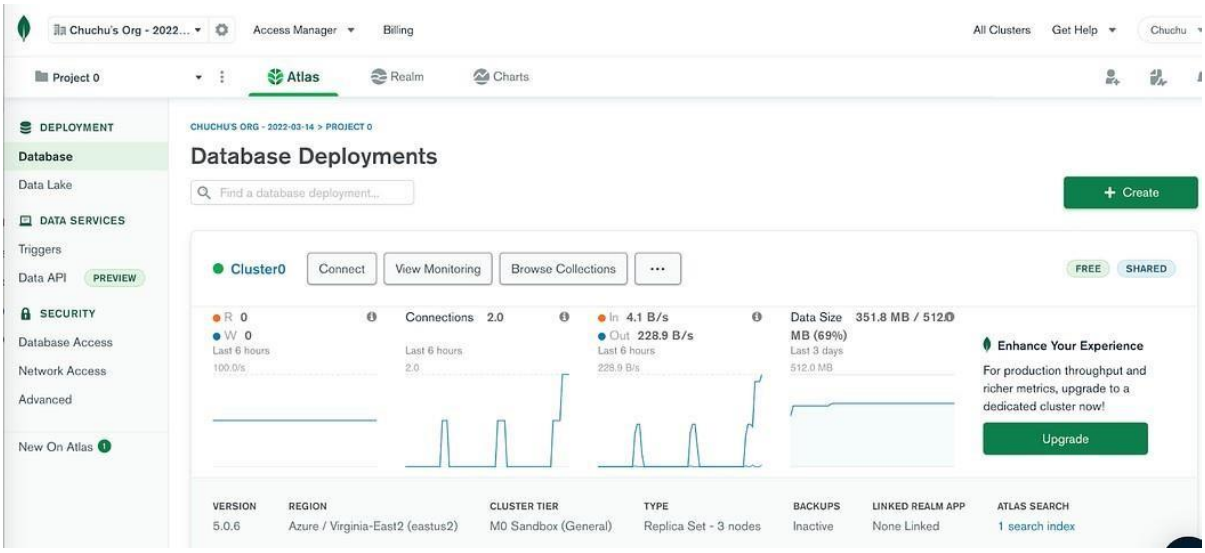
THEORY:

MongoDB is a document database that has a flexible schema for storing necessary data. A document is a record where the data is stored as key-value pairs in MongoDB. When retrieving information from the database, it can be in a JSON format while initially, each record is a document in BSON format. A JSON is a data format that can be used to store and retrieve data which is stored as key-value pairs. BSON is the binary representation of the data which is just a binary JSON.

MongoDB ensures scalability. It makes it easy to store data such that programmers have an easier time working with it because of its flexible schema. It is built to scale up quickly. It is easy to store unstructured and structured data because of the JSON like format that is used to store documents. MongoDB can handle high volumes of data which is needed having a large database of movies that can be streamed to the user in a movie streaming scenario.

Using mongo compass:

MongoDB Compass is a good GUI tool for managing mongoDB collections and doing data queries. It is free to use and can be run on macOS, Windows, and Linux.



A sample data set was created and used in order to test the functionality of the MongoDB database.

1. To show databases present in MongoDB Atlas. **`show dbs;`**

```
BDA          80.00 KiB
BDA1          8.00 KiB
admin       120.00 KiB
config       60.00 KiB
```

```
local       160.00 KiB
myDB          8.00 KiB


  poulations 56.00 KiB
test         80.00 KiB
```

2. **To select the database. use admin;**

```
'switched to db admin'
```

3. **To show all collections present in database.**

```
show collections;
```

```
employees Student
system.version
```

4. **To create a new collection.**

```
db.createCollection('Movie');
```

```
{ ok: 1 }
```

5. **To insert a single row in collection.**

```
db.Movie.insertOne({ID:1,Title:"Terminator",Director:"Sanjay",Year:1985,
Bud get:25000000});
```

```
{ acknowledged: true,
insertedId: ObjectId("64302a42c27158d639b83646") }
```

```
db.Movie.insertOne({ID:2,Title:"DDLJ",Director:"Karan",Year:1989,Budget:
200
00000});
```

```
{ acknowledged: true,
insertedId: ObjectId("64302a86c27158d639b83647") }
```

6. **To display values in collection.**

```
db.Movie.find({ID:2});
```

```
{ _id: ObjectId("64302a86c27158d639b83647"),

ID: 2,
Title: 'DDLJ',
Director: 'Karan',
Year: 1989,
Budget: 20000000 }
```

```
db.Movie.find();
```

```
{ _id: ObjectId("64302a42c27158d639b83646"),
ID: 1,
Title: 'Terminator',
Director: 'Sanjay',
Year: 1985,
Budget: 25000000 }
{ _id: ObjectId("64302a86c27158d639b83647"),
ID: 2,
Title: 'DDLJ',
Director: 'Karan',
Year: 1989,
Budget: 20000000 }
```

7. To insert multiple rows in collection.

```
db.Movie.insertMany([{ID:3,Title:"Dabang",Director:"salman",Year:2012,Budget:30000000},{ID:4,Title:"Bagban",Director:"Dharmendra",Year:2010,Budget:150 00000}]);
```

```
{ acknowledged: true, insertedIds:
{ '0': ObjectId("64302b79c27158d639b83648"), '1':
ObjectId("64302b79c27158d639b83649") } }
```

8. To display values in collection. `db.Movie.find();`

```
{ _id: ObjectId("64302a42c27158d639b83646"),

ID: 1,
Title: 'Terminator',
Director: 'Sanjay',
Year: 1985,
Budget: 25000000 }
{ _id: ObjectId("64302a86c27158d639b83647"),
ID: 2,
Title: 'DDLJ',   Director:
'Karan',

Year: 1989,
Budget: 20000000 }
{ _id: ObjectId("64302b79c27158d639b83648"),
ID: 3,
Title: 'Dabang',
Director: 'salman',
Year: 2012,
Budget: 30000000 }
{ _id: ObjectId("64302b79c27158d639b83649"),
ID: 4,
Title: 'Bagban',
Director: 'Dharmendra',
Year: 2010,
Budget: 15000000 }
```

8. To update value in the collection.

```
db.Movie.updateOne({'Title':'Terminator'},{$set:{'Title':'Robot'}}); { acknowledged: true,

insertedId: null,   matchedCount:
1,   modifiedCount: 1,
upsertedCount: 0 }
db.Movie.find();
```

```
{ _id: ObjectId("64302a42c27158d639b83646"), ID:
1,
Title: 'Robot', Director:
'Sanjay',
Year: 1985,
Budget: 25000000 }
{ _id: ObjectId("64302a86c27158d639b83647"),
ID: 2,
Title: 'DDLJ',
Director: 'Karan',
Year: 1989,
Budget: 20000000 }
{ _id: ObjectId("64302b79c27158d639b83648"),
ID: 3,
Title: 'Dabang',
Director: 'salman',
Year: 2012,
Budget: 30000000 }
{ _id: ObjectId("64302b79c27158d639b83649"),
ID: 4,
Title: 'Bagban',
Director: 'Dharmendra',
Year: 2010,
Budget: 15000000 }
```

```
db.Movie.find({$and:[{Year:{$lt:"2015-01-01"}},{Budget: {$gt:500000}}]})
```

9. To find maximum budget movies group by year.

```
db.Movie.aggregate([{$group : {_id : "$Year", num_tutorial : {$max :
"$Budget"}}}])
```

```
{ _id: 1985, num_tutorial: 25000000 }

{ _id: 1989, num_tutorial: 20000000 }
{ _id: 2010, num_tutorial: 15000000 }
{ _id: 2012, num_tutorial: 30000000 }
```

CONCLUSION: We have successfully executed CRUD operations in MongoDB Atlas.