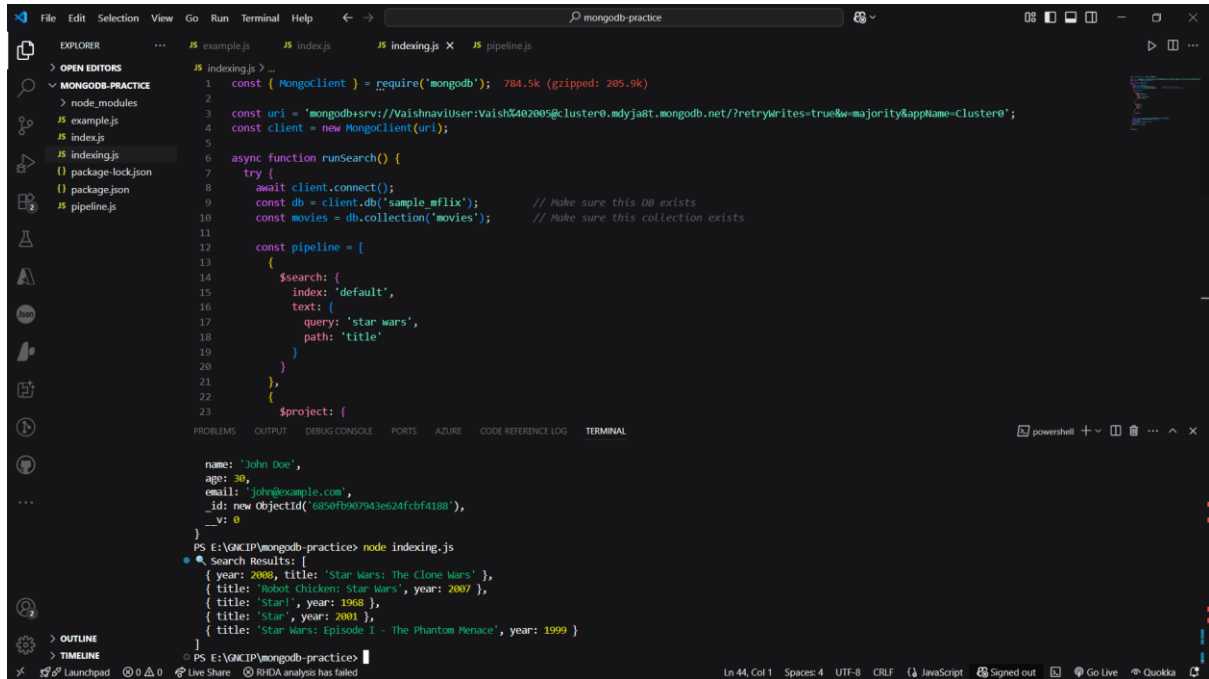


# June 17, 2025

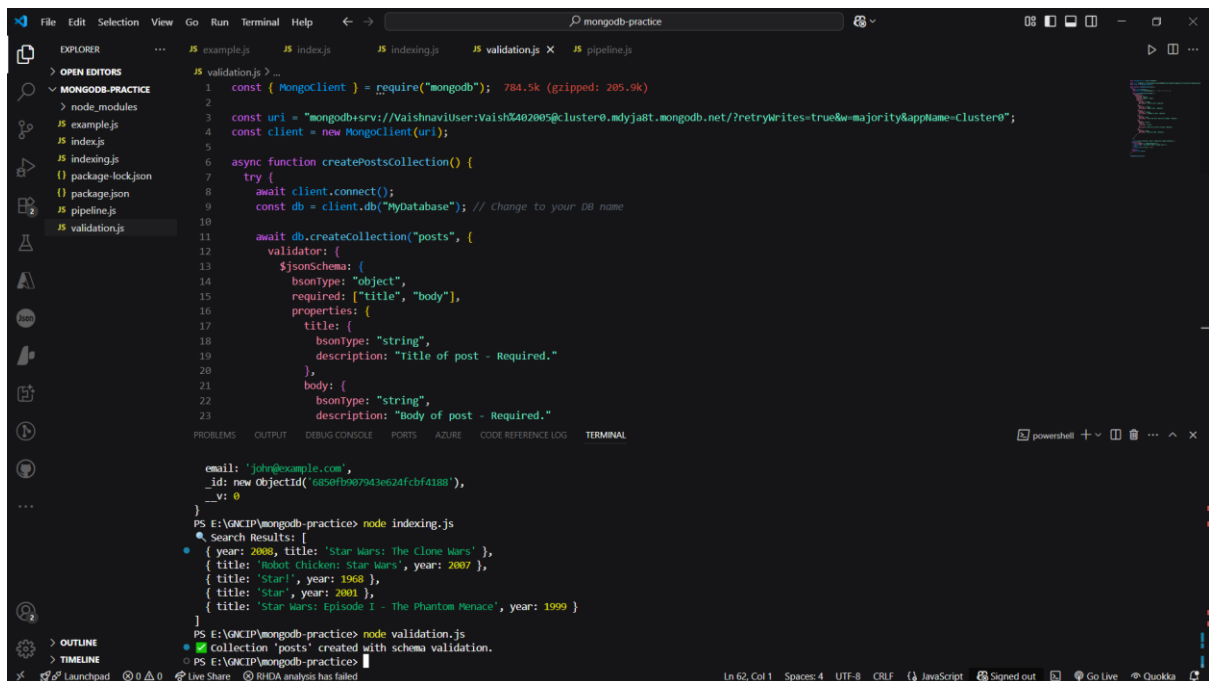
## SESSION 1



The screenshot shows a VS Code editor with the file explorer on the left. The file explorer shows a project named 'MONGODB-PRACTICE' with files: example.js, index.js, indexing.js, package-lock.json, package.json, and pipeline.js. The 'indexing.js' file is open in the editor. The code in 'indexing.js' is as follows:

```
1 const { MongoClient } = require('mongodb'); // 784.5k (gripped: 285.9k)
2
3 const uri = "mongodb+srv://VaishnaviUser:Vaish402005@cluster0.mdyja8t.mongodb.net/?retryWrites=true&majority=&appName=Cluster0";
4 const client = new MongoClient(uri);
5
6 async function runSearch() {
7   try {
8     await client.connect();
9     const db = client.db('sample_mflix'); // Make sure this DB exists
10    const movies = db.collection('movies'); // Make sure this collection exists
11
12    const pipeline = [
13      {
14        $search: {
15          index: 'default',
16          text: {
17            query: 'star wars',
18            path: 'title'
19          }
20        }
21      },
22      {
23        $project: {
24
25          name: 'John Doe',
26          age: 30,
27          email: 'john@example.com',
28          _id: new ObjectId('6850fb907943e624fcbf4188'),
29          _v: 0
30        }
31      }
32    ];
33
34    PS E:\GKIP\mongodb-practice> node indexing.js
35
36 Search Results: [
37   { year: 2008, title: 'Star Wars: The Clone Wars' },
38   { title: 'Robot Chicken: Star Wars', year: 2007 },
39   { title: 'Star!', year: 1968 },
40   { title: 'Star', year: 2001 },
41   { title: 'Star Wars: Episode I - The Phantom Menace', year: 1999 }
42 ]
43
44 PS E:\GKIP\mongodb-practice>
```

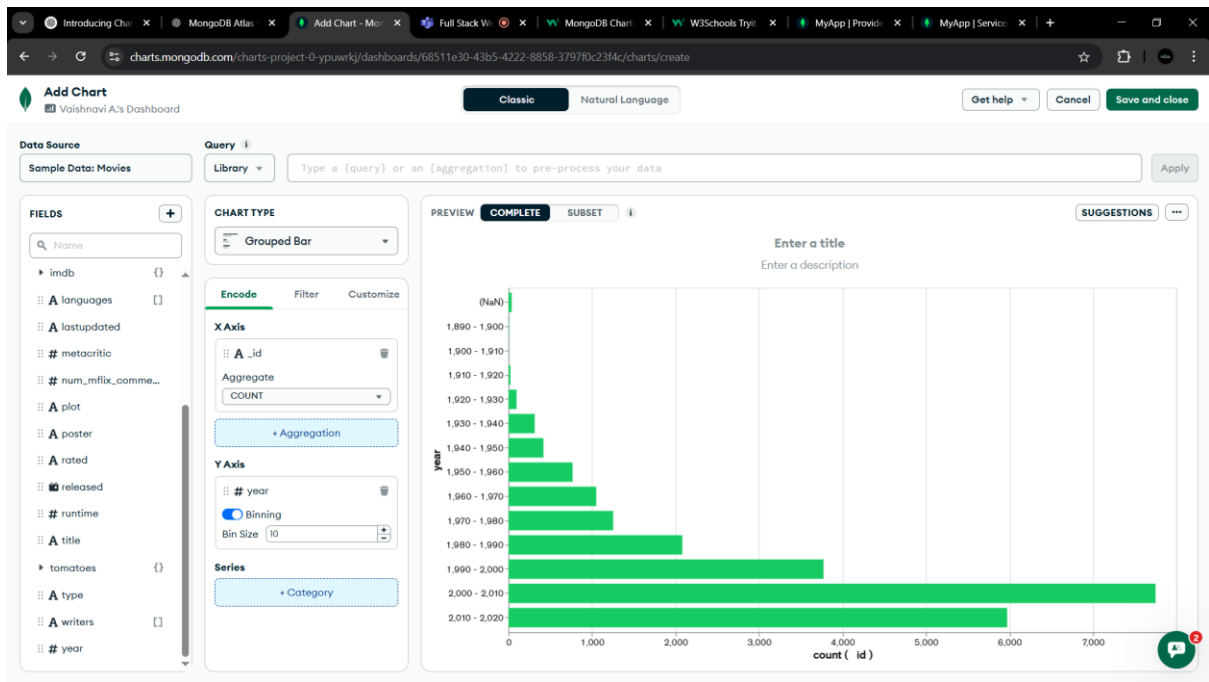
The terminal output shows the search results for the query 'star wars' in the 'sample\_mflix' database, 'movies' collection. The results are an array of objects with 'year' and 'title' properties.



The screenshot shows a VS Code editor with the file explorer on the left. The file explorer shows a project named 'MONGODB-PRACTICE' with files: example.js, index.js, indexing.js, validation.js, package-lock.json, package.json, and pipeline.js. The 'validation.js' file is open in the editor. The code in 'validation.js' is as follows:

```
1 const { MongoClient } = require("mongodb"); // 784.5k (gripped: 285.9k)
2
3 const uri = "mongodb+srv://VaishnaviUser:Vaish402005@cluster0.mdyja8t.mongodb.net/?retryWrites=true&majority=&appName=Cluster0";
4 const client = new MongoClient(uri);
5
6 async function createPostsCollection() {
7   try {
8     await client.connect();
9     const db = client.db("MyDatabase"); // Change to your DB name
10
11     await db.createCollection("posts", {
12       validator: {
13         $jsonSchema: {
14           bsonType: "object",
15           required: ["title", "body"],
16           properties: {
17             title: {
18               bsonType: "string",
19               description: "Title of post - Required."
20             },
21             body: {
22               bsonType: "string",
23               description: "Body of post - Required."
24             }
25           }
26         }
27       }
28     });
29
30     email: 'john@example.com',
31     _id: new ObjectId('6850fb907943e624fcbf4188'),
32     _v: 0
33   }
34
35 PS E:\GKIP\mongodb-practice> node indexing.js
36
37 Search Results: [
38   { year: 2008, title: 'Star Wars: The Clone Wars' },
39   { title: 'Robot Chicken: Star Wars', year: 2007 },
40   { title: 'Star!', year: 1968 },
41   { title: 'Star', year: 2001 },
42   { title: 'Star Wars: Episode I - The Phantom Menace', year: 1999 }
43 ]
44
45 PS E:\GKIP\mongodb-practice> node validation.js
46
47 collection 'posts' created with schema validation.
48
49 PS E:\GKIP\mongodb-practice>
```

The terminal output shows the search results for the query 'star wars' in the 'sample\_mflix' database, 'movies' collection. The results are an array of objects with 'year' and 'title' properties. The output also shows the successful creation of the 'posts' collection with schema validation.



## SESSION 2

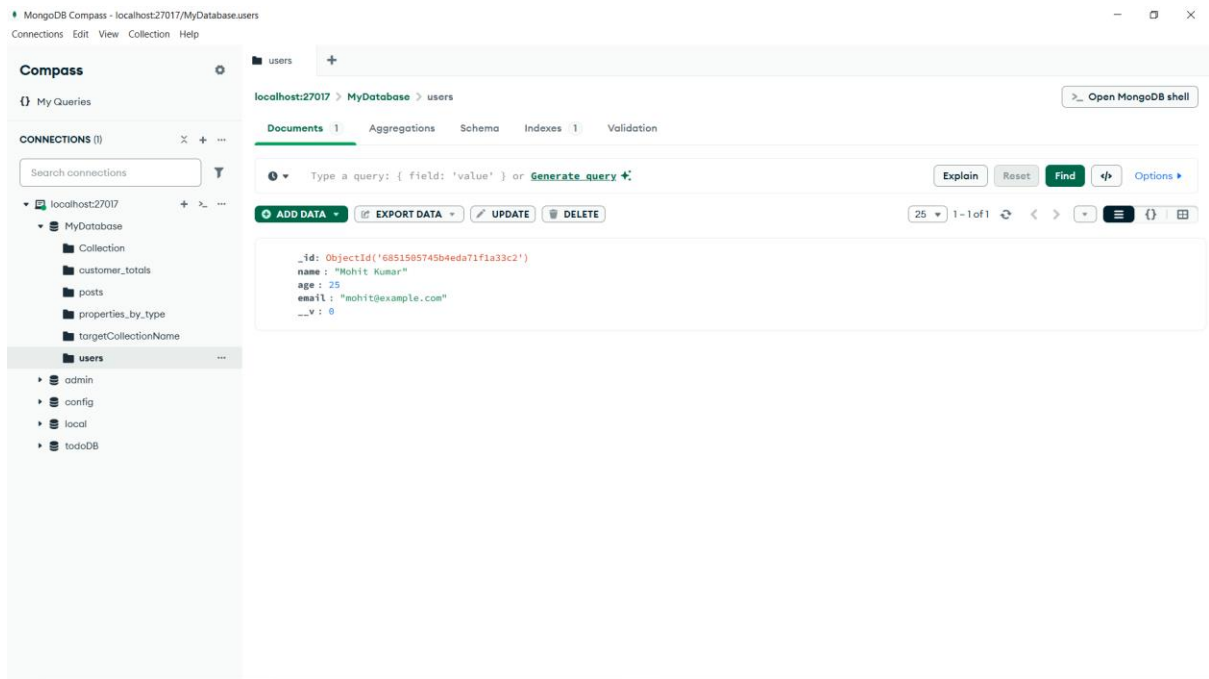
```

File Edit Selection View Go Run Terminal Help
mu-node-app-mongoose

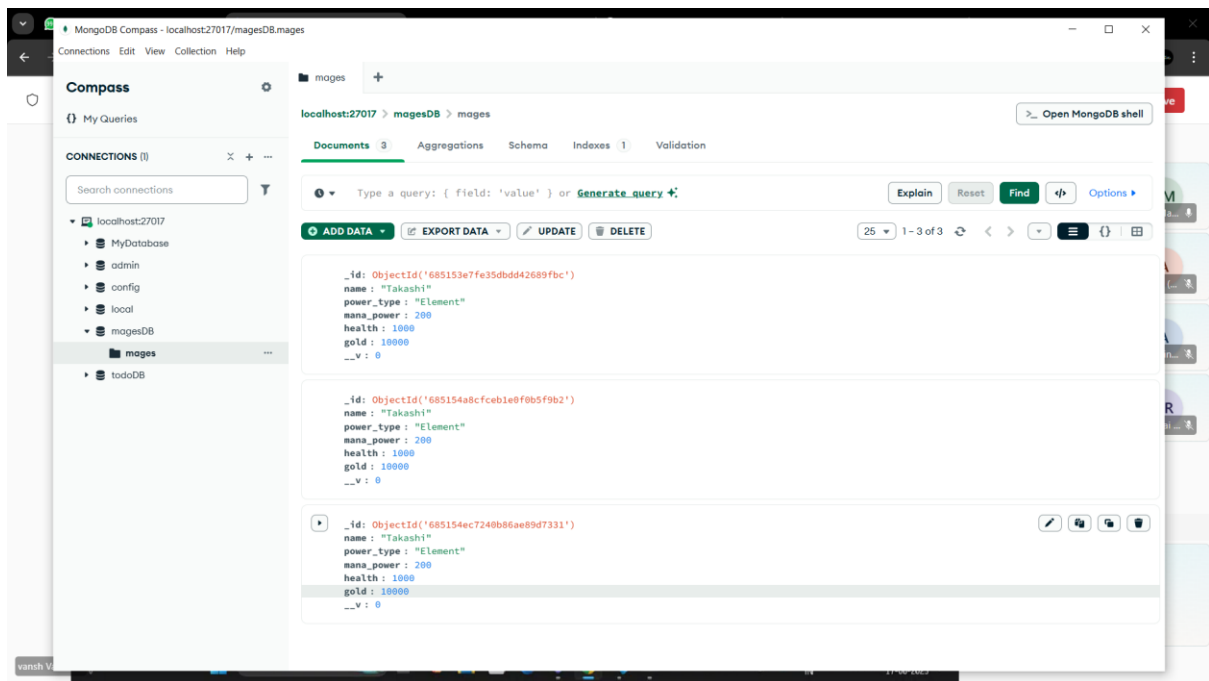
server.js
15 });
16
17 // Create a model
18 const User = mongoose.model('User', userSchema);
19
20 // Insert a document
21 const user = new User({
22   name: 'Mohit Kumar',
23   age: 25,
24   email: 'mohit@example.com',
25 });
26
27 user.save()
28   .then(() => console.log('User saved'))
29   .catch((err) => console.error('Error:', err));

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
node v22.16.0
PS E:\GKIP\mu-node-app-mongoose> node server.js
(node:5328) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
(node:5328) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
User saved
  
```

```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
server.js x .env
server.js > ...
1 const mongoose = require('mongoose'); 580.5k (gzipped: 145.1k)
2
3 mongoose.connect('mongodb://localhost:27017/MyDatabase')
4 .then(() => console.log("MongoDB connected"))
5 .catch(err => console.error("Connection error:", err));
6
7 const userSchema = new mongoose.Schema({
8   name: String,
9   age: Number,
10  email: String,
11 });
12
13 const User = mongoose.model('User', userSchema);
14
15 const user = new User({
16   name: 'Mohit Kumar',
17   age: 25,
18   email: 'mohit@example.com',
19 });
20
21 user.save()
22 .then(() => console.log('User saved'))
23 .catch(err => console.error("Error:", err));
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
node + v 16.16.0
PS E:\GKIP\mu-node-app-mongoose> node server.js
(node:5328) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
(node:5328) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
User saved
PS E:\GKIP\mu-node-app-mongoose> node server.js
MongoDB connected
User saved
```



```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
server.js x env
10 const mage_1 = new Mage({
21   power_type: 'Element',
22   mana_power: 200,
23   health: 1000,
24   gold: 10000
25 });
26
27 mage_1.save()
28 .then(() => console.log("Mage saved"))
29 .catch((err) => console.error("Error:", err))
30 .finally(() => mongoose.connection.close()); // this closes the app properly
31
32
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
> node server.js
PS E:\GKIP\mu-node-app-mongoose> npm start
> mu-node-app-mongoose@1.0.0 start
> node server.js
PS E:\GKIP\mu-node-app-mongoose> node server.js
PS E:\GKIP\mu-node-app-mongoose> npm start
> mu-node-app-mongoose@1.0.0 start
> node server.js
Mage saved
PS E:\GKIP\mu-node-app-mongoose>
Launchpad 0 0 0 Live Share Ln 32, Col 1 Spaces 4 UTF-8 CRLF JavaScript Signed out Go Live Quokka
```



```
server.js
14
15
16 // Calling Schema class
17 const Schema = mongoose.Schema;
18
19 // Creating Structure of the collection
20 const collection_structure = new Schema({
21   name: String,
22   marks: Number,
23 })
24
25 // Creating collection
26 const collections = mongoose.model("GFG", collection_structure)
27
28 collections.create({
29   name: 'Ragavpp',
30   marks: 13,
31 })
32 .then((ans) => {
33   console.log("Document inserted")
34 })
35
36 .catch((err) => {
37
38 })
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

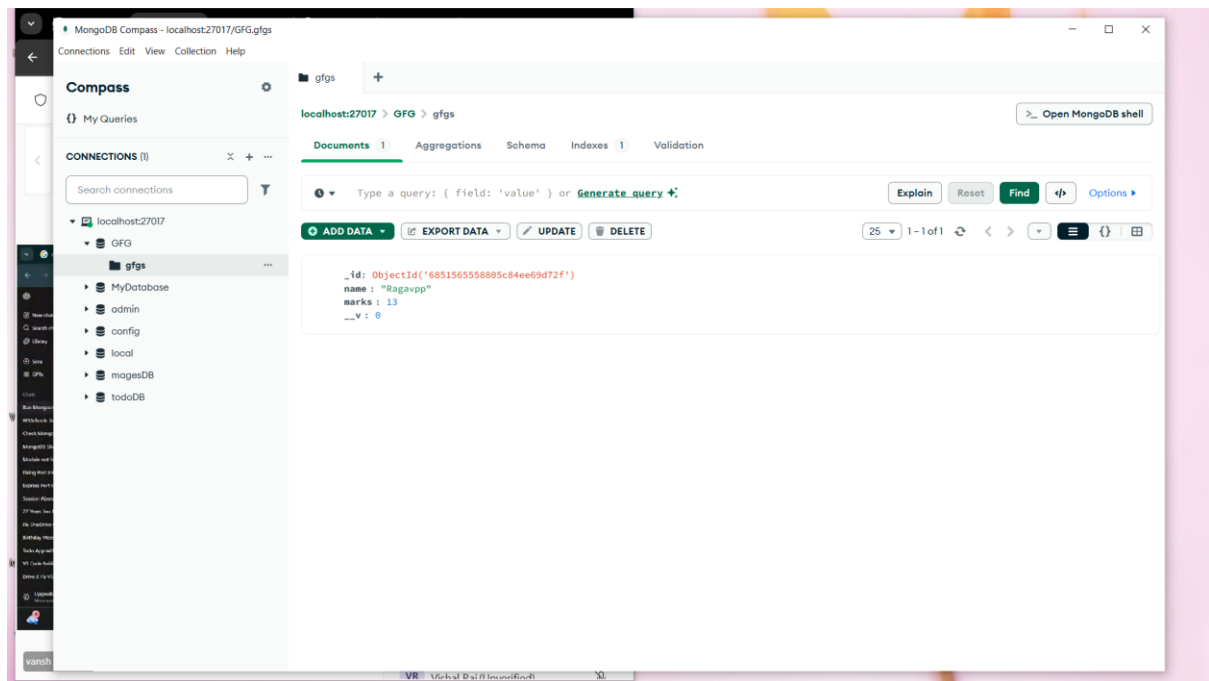
```
PS E:\GKIP\mu-node-app-mongoose> node server.js
PS E:\GKIP\mu-node-app-mongoose> npm start

> mu-node-app-mongoose@1.0.0 start
> node server.js

Mage saved
PS E:\GKIP\mu-node-app-mongoose> npm start

> mu-node-app-mongoose@1.0.0 start
> node server.js

Connected successfully
Document inserted
```



```
23 const collection_structure = new Schema({
37 })
38
39 // Creating collection
40 const collections = mongoose.model("GFG", collection_structure)
41
42 collections.create({
43   name: "Sam Snehl",
44   marks: 88,
45   mobile: 9338948473,
46 })
47 .then((res) => {
48   console.log("Document inserted")
49 })
50 .catch((err) => {
51   console.log(err);
52 })
53 }
```

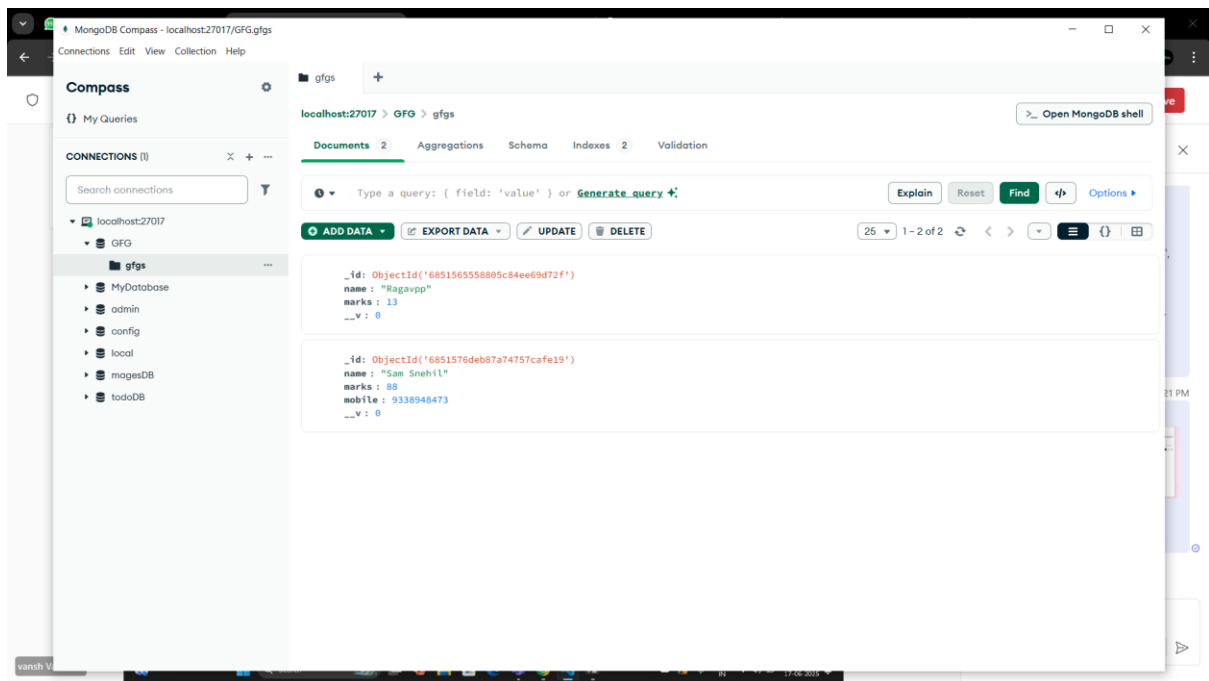
mu-node-app-mongoose@1.0.0 start  
node server.js

Mage saved  
PS E:\GMCIP\muu-node-app-mongoose> npm start

mu-node-app-mongoose@1.0.0 start  
node server.js

connected successfully  
Document inserted  
PS E:\GMCIP\muu-node-app-mongoose> node server2.js

connected Successful  
Document inserted



The screenshot shows the VS Code editor with the file `app.js` open. The code defines a `main` function that sets up a Mongoose connection, defines an `Animal` schema with `name` and `type` fields, and inserts two documents: `{ name: 'bond', type: 'dog' }` and `{ name: 'cavine', type: 'cat' }`. It then uses `findSimilarTypes` to find similar animals and logs the result. The terminal shows the command `node app.js` being executed, resulting in the output: `Similar animals: [ { _id: new ObjectId('685159eb5ad6affa16ef0645'), name: 'bond', type: 'dog', __v: 0 } ]`.

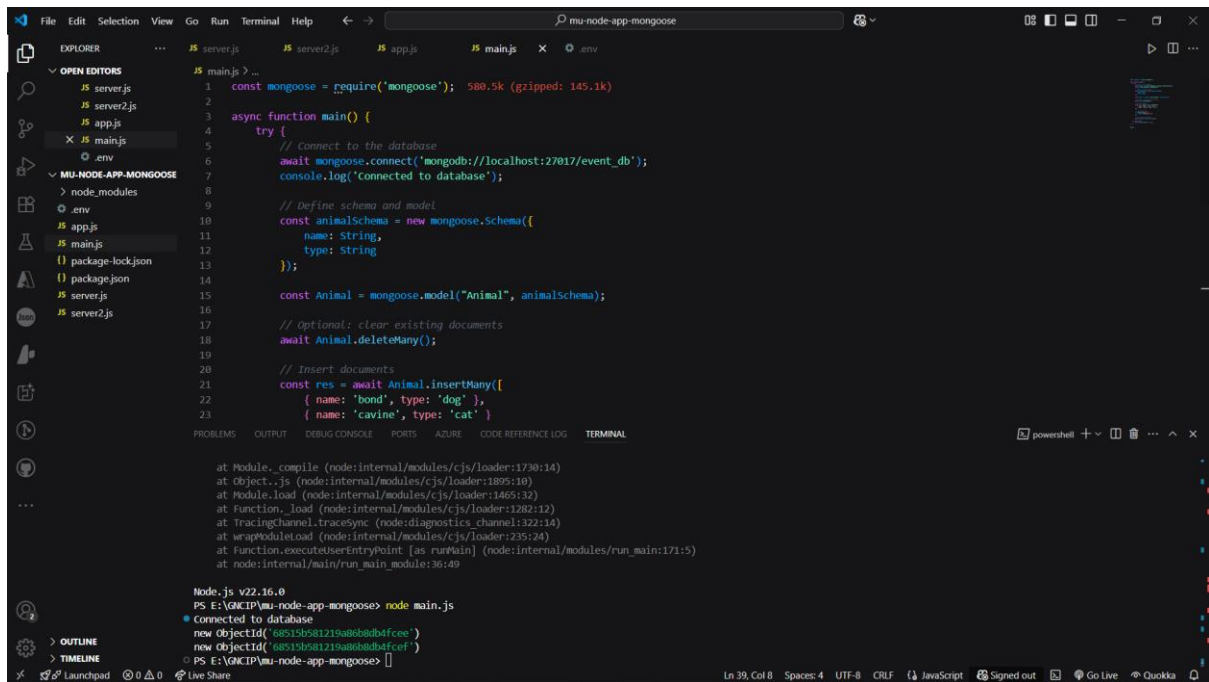
```
3 async function main() {
18
19   const Animal = mongoose.model("Animal", animalSchema);
20
21   const animals = [
22     { name: 'bond', type: 'dog' },
23     { name: 'cavine', type: 'cat' }
24   ];
25
26   await Animal.deleteMany(); // optional: clear previous data
27   await Animal.insertMany(animals);
28
29   const dog = new Animal({ type: 'dog' });
30   const similar = await dog.findSimilarTypes();
31
32   console.log("Similar animals:", similar);
33   mongoose.connection.close(); // close connection after all work is done
34
35   } catch (err) {
36     console.error("Connection failed:", err);
37   }
38 }
39
```

at Function.find (E:\GNCIP\mu-node-app-mongoose\node\_modules\mongoose\lib\model.js:2084:11)  
at model.findSimilarTypes (E:\GNCIP\mu-node-app-mongoose\app.js:13:57)  
at main (E:\GNCIP\mu-node-app-mongoose\app.js:30:13)  
at process.processTicksAndRejections (node:internal/process/task\_queues:105:5)  
PS E:\GNCIP\mu-node-app-mongoose> node app.js  
Connected to database  
Similar animals: [  
 {  
 \_id: new ObjectId('685159eb5ad6affa16ef0645'),  
 name: 'bond',  
 type: 'dog',  
 \_\_v: 0  
 }  
]

The screenshot shows the VS Code editor with the file `app.js` open. The code continues from the previous screenshot, adding a `cat` instance and finding similar cats. The terminal shows the command `node app.js` being executed, resulting in the output: `Similar animals: [ { _id: new ObjectId('68515a29fd622c290ba52e5'), name: 'cavine', type: 'cat', __v: 0 } ]`.

```
26   };
27
28   // Optional: clean existing data
29   await Animal.deleteMany();
30
31   // Insert multiple documents
32   await Animal.insertMany(animals);
33
34   // Create a new cat instance to test the method
35   const cat = new Animal({ type: 'cat' });
36
37   const similarCats = await cat.findSimilarTypes();
38   console.log("Similar animals:", similarCats);
39
40   // Close DB connection
41   await mongoose.connection.close();
42
43   } catch (err) {
44     console.error("Error:", err);
45   }
46 }
47
```

type: 'dog',  
\_\_v: 0  
}  
]  
PS E:\GNCIP\mu-node-app-mongoose> node app.js  
Connected to database  
Similar animals: [  
 {  
 \_id: new ObjectId('68515a29fd622c290ba52e5'),  
 name: 'cavine',  
 type: 'cat',  
 \_\_v: 0  
 }  
]



```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
```

EXPLORER

- server.js
- server2.js
- app.js
- main.js
- env
- MU-NODE-APP-MONGOOSE
  - node\_modules
  - env
  - app.js
  - main.js
  - package-lock.json
  - package.json
  - server.js
  - server2.js

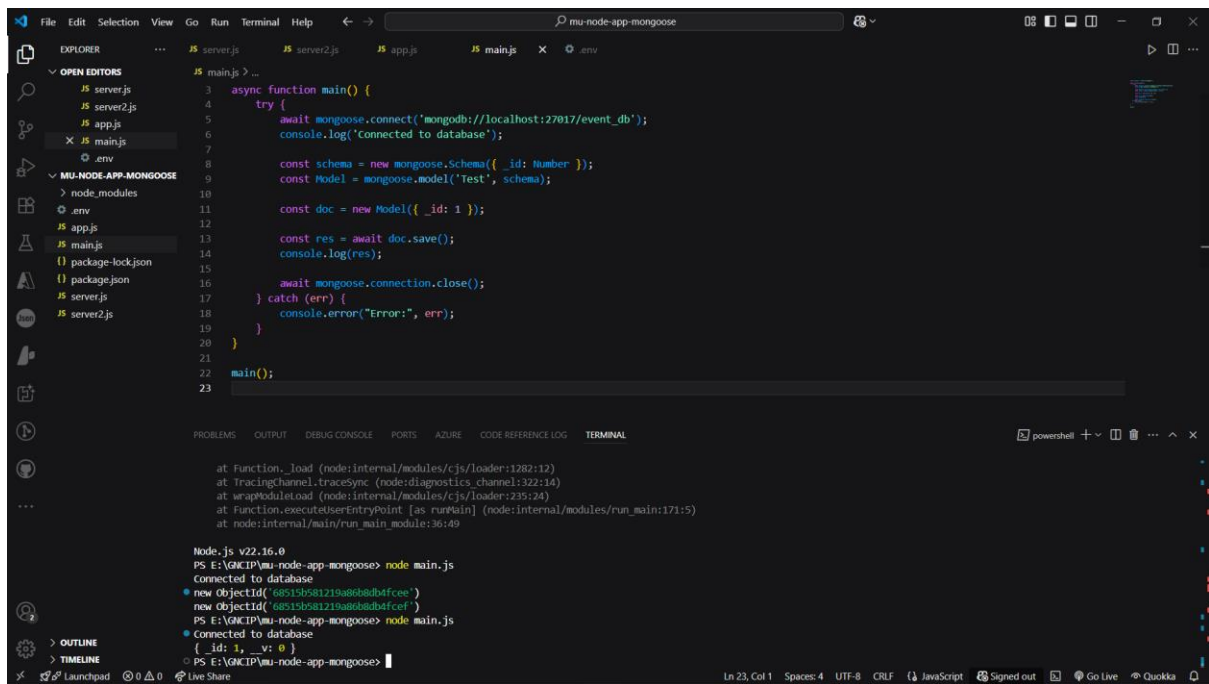
main.js

```
1 const mongoose = require('mongoose'); 580.5k (gzipped: 145.1k)
2
3 async function main() {
4   try {
5     // Connect to the database
6     await mongoose.connect('mongodb://localhost:27017/event_db');
7     console.log("Connected to database");
8
9     // Define schema and model
10    const animalSchema = new mongoose.Schema({
11      name: String,
12      type: String,
13    });
14
15    const Animal = mongoose.model("Animal", animalSchema);
16
17    // Optional: clear existing documents
18    await Animal.deleteMany({});
19
20    // Insert documents
21    const res = await Animal.insertMany([
22      { name: 'bond', type: 'dog' },
23      { name: 'cavine', type: 'cat' } ]
24    );
25  } catch (err) {
26    console.error("Error:", err);
27  }
28}
29
30main();
```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```
at Module._compile (node:internal/modules/cjs/loader:1730:14)
at Object.<.> (node:internal/modules/cjs/loader:1895:10)
at Module.load (node:internal/modules/cjs/loader:1465:32)
at Function.<.> (node:internal/modules/cjs/loader:1282:12)
at TracingChannel.traceSync (node:diagnostics_channel:322:14)
at wrapModuleLoad (node:internal/modules/cjs/loader:235:24)
at Function.<.> [as runMain] (node:internal/modules/run_main:171:5)
at node:internal/main/run_main_module:36:49

Node.js v22.16.0
PS E:\GNCIP\mu-node-app-mongoose> node main.js
Connected to database
new ObjectId('68515b581219a86b8db4fcee')
new ObjectId('68515b581219a86b8db4fcef')
PS E:\GNCIP\mu-node-app-mongoose>
```



```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
```

EXPLORER

- server.js
- server2.js
- app.js
- main.js
- env
- MU-NODE-APP-MONGOOSE
  - node\_modules
  - env
  - app.js
  - main.js
  - package-lock.json
  - package.json
  - server.js
  - server2.js

main.js

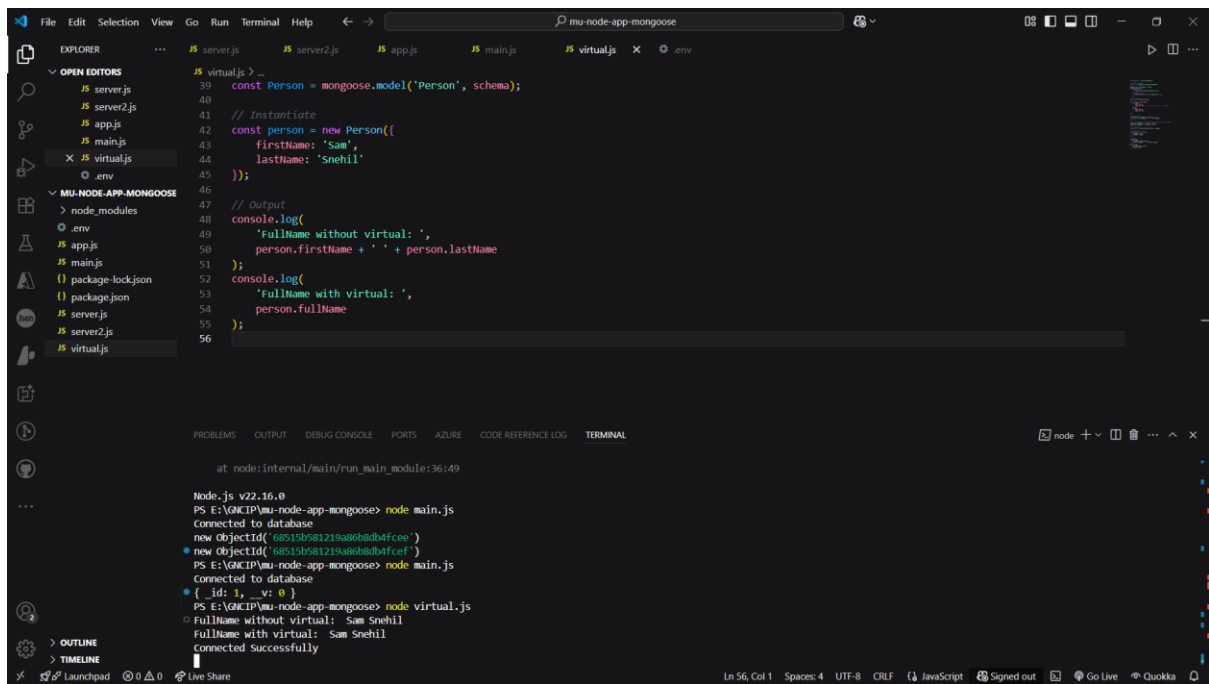
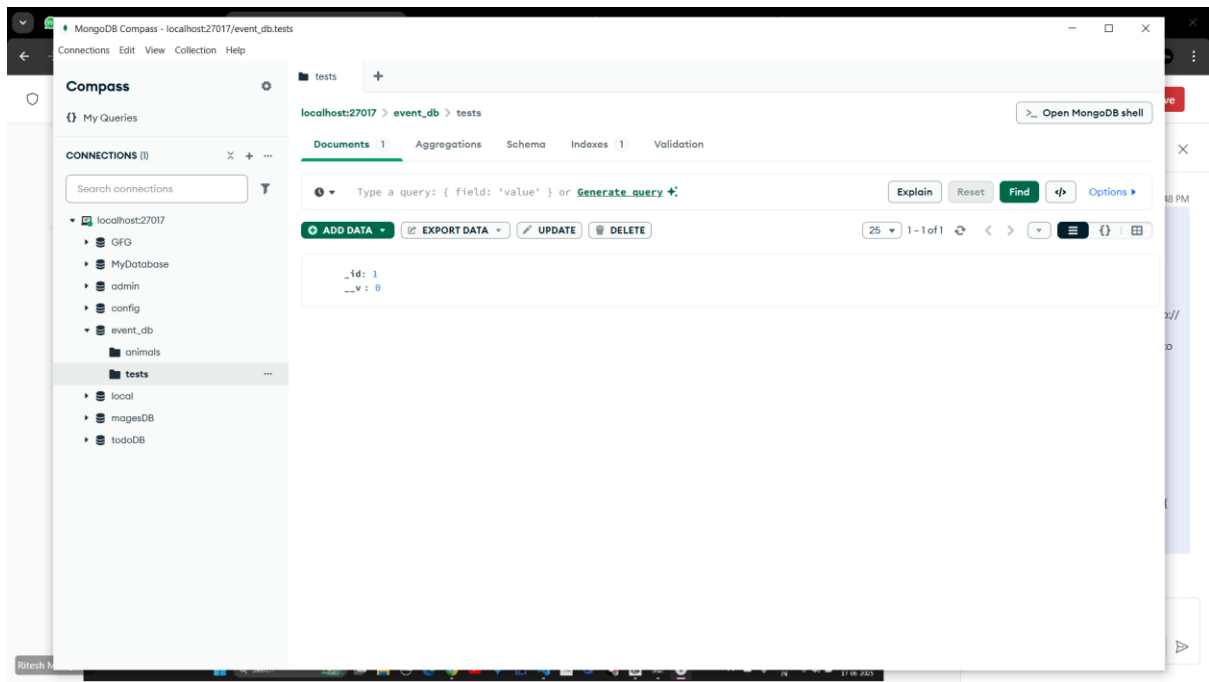
```
3 async function main() {
4   try {
5     await mongoose.connect('mongodb://localhost:27017/event_db');
6     console.log("Connected to database");
7
8     const schema = new mongoose.Schema({ _id: Number });
9     const Model = mongoose.model('Test', schema);
10
11     const doc = new Model({ _id: 1 });
12
13     const res = await doc.save();
14     console.log(res);
15
16     await mongoose.connection.close();
17   } catch (err) {
18     console.error("Error:", err);
19   }
20}
21
22main();
23
```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```
at Function.<.> (node:internal/modules/cjs/loader:1282:12)
at TracingChannel.traceSync (node:diagnostics_channel:322:14)
at wrapModuleLoad (node:internal/modules/cjs/loader:235:24)
at Function.<.> [as runMain] (node:internal/modules/run_main:171:5)
at node:internal/main/run_main_module:36:49

Node.js v22.16.0
PS E:\GNCIP\mu-node-app-mongoose> node main.js
Connected to database
new ObjectId('68515b581219a86b8db4fcee')
new ObjectId('68515b581219a86b8db4fcef')
PS E:\GNCIP\mu-node-app-mongoose> node main.js
Connected to database
{ _id: 1, __v: 0 }
PS E:\GNCIP\mu-node-app-mongoose>
```





```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
EXPLORER
  JS server.js
  JS server2.js
  JS app.js
  JS main.js
  X JS virtual.js
  .env
  MU-NODE-APP-MONGOOSE
    node_modules
    .env
    app.js
    main.js
    package-lock.json
    package.json
    server.js
    server2.js
    virtual.js
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
node + v ...
PS E:\GNCIP\mu-node-app-mongoose> node main.js
Connected to database
{ _id: 1, __v: 0 }
PS E:\GNCIP\mu-node-app-mongoose> node virtual.js
Fullname without virtual: Sam Snehil
Fullname with virtual: Sam Snehil
Connected Successfully
PS E:\GNCIP\mu-node-app-mongoose> node virtual.js
Fullname without virtual: Sam Snehil
Fullname with virtual: Sam Snehil
Connected Successfully
PS E:\GNCIP\mu-node-app-mongoose> node virtual.js
Sam is from NewDelhi
connected Successfully
Ln 44, Col 31 Spaces: 4 UTF-8 CRLF JavaScript Signed out Go Live Quokka
```

```
File Edit Selection View Go Run Terminal Help mu-node-app-mongoose
EXPLORER
  JS server.js
  JS server2.js
  JS app.js
  JS main.js
  JS virtual.js
  X JS main2.js
  .env
  MU-NODE-APP-MONGOOSE
    node_modules
    .env
    app.js
    main.js
    package-lock.json
    package.json
    server.js
    server2.js
    virtual.js
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
powershell + v ...
at Object.<js> (node:internal/modules/cjs/loader:1895:10)
at Module.load (node:internal/modules/cjs/loader:1465:32)
at Function._load (node:internal/modules/cjs/loader:1282:12)
at TracingChannel.traceSync (node:diagnostics_channel:322:14)
at wrapModuleload (node:internal/modules/cjs/loader:235:24)
at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:171:5)
at node:internal/main/run_main_module:36:49
Node.js v22.16.0
PS E:\GNCIP\mu-node-app-mongoose> node main2.js
connected to database
person.name (alias): Val
person.n (real field): Val
Saved document: { n: 'Val', _id: new ObjectId('6851614712d039876c7fd31'), __v: 0 }
PS E:\GNCIP\mu-node-app-mongoose>
Ln 37, Col 1 Spaces: 4 UTF-8 CRLF JavaScript Signed out Go Live Quokka
```

```
1 async function main() {  
2  
3     // Create a person using alias names  
4     const person = new Person({ first: 'Hello', last: 'World' });  
5  
6     // Show how alias and real field work  
7     console.log('Alias - first:', person.first); // Hello  
8     console.log('Alias - last:', person.last);   // World  
9     console.log('Real field - f:', person.f);    // Hello  
10    console.log('Real field - l:', person.l);     // World  
11  
12    // Save to database  
13    await person.save();  
14    console.log('Document saved:', person);  
15  
16    await mongoose.connection.close();  
17    catch (error) {  
18        console.error('Error:', error);  
19    }  
20 }  
21  
22 main();  
23
```

person.n (real field): Val  
Saved document: { n: 'Val', \_id: new ObjectId('68516147212d039876c7fd31'), \_\_v: 0 }  
PS E:\GNCIP\mu-node-app-mongoose> node main2.js  
Connected to database  
Alias - first: Hello  
Alias - last: World  
Real field - f: Hello  
Real field - l: World  
Document saved: {  
 f: 'Hello',  
 l: 'World',  
 \_id: new ObjectId('685161ae4405f4bcd5e6162a'),  
 \_\_v: 0  
}

```
1 const mongoose = require('mongoose');  
2  
3 class MyClass {  
4     myMethod() { return "Geeksforgeeks"; }  
5 }  
6  
7 const schema = new mongoose.Schema();  
8 schema.loadClass(MyClass);  
9  
10 console.log(schema.methods.myMethod());
```

PS E:\GNCIP\mu-node-app-mongoose> node main2.js  
Connected to database  
Alias - first: Hello  
Alias - last: World  
Real field - f: Hello  
Real field - l: World  
Document saved: {  
 f: 'Hello',  
 l: 'World',  
 \_id: new ObjectId('685161ae4405f4bcd5e6162a'),  
 \_\_v: 0  
}

PS E:\GNCIP\mu-node-app-mongoose> node ES6Classes.js  
Geeksforgeeks  
PS E:\GNCIP\mu-node-app-mongoose>

The screenshot shows the VS Code editor with the file `ES6Classes.js` open. The code defines a `MyClass` with a static method `myStatic()` that returns `"Geeksforgeeks"`. It also uses `mongoose.Schema` to create a schema and load the class. The terminal shows the output of running `node ES6Classes.js`, which prints the static method's result and a document object.

```
1 const mongoose = require('mongoose')
2
3 class MyClass {
4   static myStatic() { return "Geeksforgeeks"; }
5 }
6
7 const schema = new mongoose.Schema();
8 schema.loadClass(MyClass);
9
10 console.log(schema.statics.myStatic());
```

```
Alias - first: Hello
Alias - last: World
Real field - f: Hello
Real field - l: World
Document saved: {
  f: 'Hello',
  l: 'World',
  _id: new ObjectId('685161ae4405f4bcd566162a'),
  __v: 0
}
```

PS E:\GNCIP\mu-node-app-mongoose> node ES6Classes.js  
Geeksforgeeks  
PS E:\GNCIP\mu-node-app-mongoose> node ES6Classes.js  
Geeksforgeeks  
PS E:\GNCIP\mu-node-app-mongoose>

The screenshot shows the VS Code editor with the file `query.js` open. The code defines an `async function main()` that creates an `Animal` schema and model, inserts data, and queries the database. The terminal shows the output of running `node query.js`, which connects to the database and prints the query results.

```
3 async function main() {
10   const animalSchema = new mongoose.Schema(
19   );
20
21   const Animal = mongoose.model("Animal", animalSchema);
22
23   // Clear existing data and insert new animals
24   await Animal.deleteMany({});
25   await Animal.insertMany([
26     { name: 'bond', type: 'dog' },
27     { name: 'cevin', type: 'cat' }
28   ]);
29
30   // Use query helper by type
31   const cats = await Animal.find().byType('cat');
32   console.log('Cats:', cats);
33
34   await mongoose.connection.close();
35   } catch (err) {
36     console.error('Error:', err);
37   }
38 }
39
```

```
at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:171:5)
at node:internal/main/run_main_module:36:49

Node.js v22.16.0
PS E:\GNCIP\mu-node-app-mongoose> node query.js
Connected to database
Cats: [
  {
    _id: new ObjectId('6851639a0051f3898e9722f5'),
    name: 'cevin',
    type: 'cat',
    __v: 0
  }
]
```

PS E:\GNCIP\mu-node-app-mongoose>