

jojojustine / CS615C

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Files

main

Go to file

- Assignment3
- Assignment4
- Assignment5
- Assignment6
- Assignment7
- node\_quickstart
  - .DS\_Store
- images
  - .DS\_Store
  - CS615C Assignment 1 Jojo Justin...
  - CS615C Assignment 1 Jojo Justin...
  - CS615C Assignment 2 Jojo Justin...
  - CS615C Assignment 2 Jojo Justin...
  - README.md
  - Sample.docx
  - index.html

CS615C / Assignment7 /

Add file ...

jojojustine Assignment 7 changes 3f16b5a · 1 minute ago History

Name	Last commit message	Last commit date
..		
node_quickstart	Assignment 7 changes	1 minute ago
.DS_Store	Assignment 7 changes	1 minute ago

EXPLORER

ASSIGNMENT7

- node\_quickstart
  - node\_modules
  - connect.js
  - package-lock.json
  - package.json

node\_quickstart > JS connect.js > run

```
1 const { MongoClient } = require("mongodb");
2
3 // Replace the following with your Atlas connection string
4 const url = "mongodb+srv://Jojojustine:Jojo@1234@cluster0.ybwnxto.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0";
5
6 // Connect to your Atlas cluster
7 const client = new MongoClient(url);
8
9
10 async function run() {
11   try {
12     await client.connect();
13     console.log("Successfully connected to Atlas");
14   } catch (err) {
15     console.log(err.stack);
16   }
17   finally {
18     await client.close();
19   }
20 }
21
22 run().catch(console.dir);
```

Atlas Jojo's Org - ... Access Manager Billing All Clusters Get Help Jojo

CMTT Data Services Charts

Overview

DATABASE

Clusters

SERVICES

Atlas Search

Stream Processing

Triggers

Migration

Data Federation

SECURITY

Quickstart

Backup

Database Access

Network Access

Advanced

New On Atlas 3

Goto

## JOJO'S ORG - 2025-03-19 > CMTT

# Clusters

Find a database deployment...

Edit Config Create

Sample dataset successfully loaded. Access it in [Collections](#) or by connecting with the MongoDB Shell.

Cluster0 Connect View Monitoring Browse Collections ... FREE

Visualize Your Data

Build dashboards and charts, and embed them in your apps with MongoDB Charts.

Dismiss Explore Charts

R 0 W Last 21 minutes 100.0%

Connections 0 Last 26 minutes 3.0

In 0.00 B/s Out 0.00 B/s Last 21 minutes 271.98 KB/s

Data Size 134.66 MB / 512.00 MB (26%) Last 26 minutes 512.00 MB

VERSION	REGION	TYPE	BACKUPS	LINKED APP SERVICES	ATLAS SQL	ATLAS SEARCH
8.0.6	AWS / Ireland (eu-west-1)	Replica Set - 3 nodes	Inactive	None Linked	<a href="#">Connect</a>	<a href="#">Create Index</a>

+ Add Tag

```

EXPLORER
... Welcome JS connect.js U JS index.js U X
v ASSIGNMENT7
  node_quickstart
    > node_modules
    JS connect.js U
    JS index.js U
    {} package-lock.json U
    {} package.json U

node_quickstart > JS index.js > run
1 const { MongoClient } = require("mongodb");
2
3 // Replace the uri string with your connection string.
4 const uri = "mongodb+srv://Jojojustine:Jojo1234@cluster0.ybwnxto.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0";
5
6 const client = new MongoClient(uri);
7
8 async function run() {
9   try {
10     const database = client.db('sample_mflix');
11     const movies = database.collection('movies');
12     const newMovie = {
13       director: 'Jojo Justine', year: 2025,
14       director: 'ID - 24258470', genres: ['Tech', 'Sci-Fi', 'Adventure'], rating: 8.5;
15     };
16     // Task 1. Insertion (Avoid duplicate insertions)
17     const existing = await movies.findOne({ title: 'Jojo Justine' });
18     if (existing) {
19       console.log('Document already exists:', existing._id);
20     } else {
21       const insertResult = await movies.insertOne(newMovie);
22       console.log('Inserted document ID: ', insertResult.insertedId);
23     }
24   }
25 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
zsh - node_quickstart
poster: 'https://m.media-amazon.com/images/M/MV5BZmU0M2Y1OGU4ZjIxN100ZjBkLTg1MjgtOWIyNThiZWVhYjRlXkEyXkFqcGdeQXVyMTQxNzMzNDI@._V1_SY1000_SX677_AL_.jpg',
title: 'Back to the Future',
fulPlot: 'Marty McFly, a typical American teenager of the Eighties, is accidentally sent back to 1955 in a plutonium-powered DeLorean "time machine" invented by a slightly mad scientist. During his often hysterical, always amazing trip back in time, Marty must make certain his teenage parents-to-be meet and fall in love - so he can get back to the future.',
languages: [ 'English' ],
released: 1985-07-03T00:00:00.000Z,
directors: [ 'Robert Zemeckis' ],
writers: [ 'Robert Zemeckis', 'Bob Gale' ],
awards: {
  wins: 19,
  nominations: 24,
  text: 'Won 1 Oscar. Another 18 wins & 24 nominations.'
},
lastupdated: '2015-09-12 00:29:36.890000000',
year: 1985,
imdb: { rating: 8.5, votes: 636511, id: 88763 },
countries: [ 'USA' ],
type: 'movie',
num_mflix_comments: 0
}
jojojustine@MacBookAir node_quickstart %

```

Part 3

EXPLORER

ASSIGNMENT7

- node\_quickstart
  - node\_modules
  - connect.js
  - example-insert.js
  - example-query.js
  - example-query2.js
  - index.js
  - package-lock.json
  - package.json

```
node_quickstart > JS example-query.js > ...
1 //Part 2 of lab assignment 7, using
2 //code source: https://www.mongodb.com/docs/drivers/node/upcoming/quick-start/connect-to-mongodb/
3
4 //This code assumes database is running, and with connection uri = mongodb://localhost:27017
5 //This code also assumes that the following database, with the following collection, containing the following document exists:
6 //database: EmployeeDB
7 //collection: Employee
8 //Document: EmployeeID: 1 , EmployeeName: 'Martin'
9
10 //tested using: Node V18.15.0 and MongoDB V6.0.5
11
12
13 const { MongoClient } = require("mongodb");
14
15 // Use 127.0.0.1 instead of localhost to connect
16 const url = "mongodb://127.0.0.1:27017";
17 const client = new MongoClient(url);
18 async function run() {
19   try {
20     const database = client.db('EmployeeDB');
21     const employees = database.collection('Employee');
22
23     // Query for an Employee with name 'Martin'
24     const query = { EmployeeName: 'Martin' };
25
26   } catch (error) {
27     console.error('Error connecting to MongoDB:', error);
28   }
29 }
30 run().catch(console.error);
```

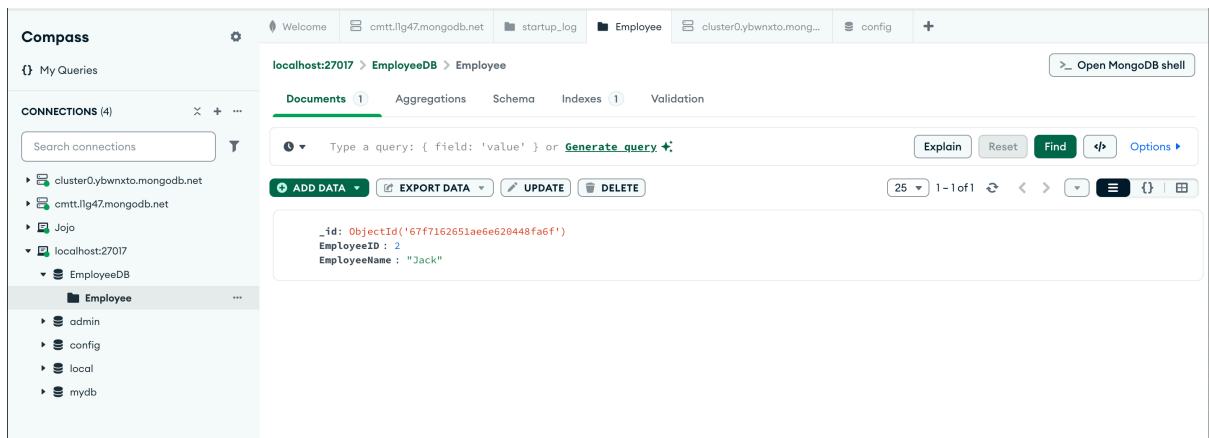
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

zsh - node\_quickstart

```
at Object.onceWrapper (node:events:622:26)
at Socket.emit (node:events:507:28)
at emitErrorNT (node:internal/streams/destroy:170:8)
at emitErrorCloseNT (node:internal/streams/destroy:129:3)
at process.processTicksAndRejections (node:internal/process/task_queues:90:21) {
  errorLabelSet: Set(1) { 'ResetPool' },
  beforeHandshake: false,
  [cause]: Error: connect ECONNREFUSED 127.0.0.1:27017
    at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1636:16) {
      errno: -61,
      code: 'ECONNREFUSED',
      syscall: 'connect',
      address: '127.0.0.1',
      port: 27017
    }
  }
}
```

jojojustine@MacBookAir node\_quickstart % node example-insert.js

A document was inserted with the \_id: 67f7162651ae6e20448fa6f



EXPLORER

...

JS connect.js U JS index.js U JS example-insert.js U JS example-query.js U X

ASSIGNMENT7

node\_quickstart > JS example-query.js > run

node\_modules

connect.js

example-insert.js

example-query.js

example-query2.js

index.js

package-lock.json

package.json

```
9
10 //tested using: Node V18.15.0 and MongoDB V6.0.5
11
12
13 const { MongoClient } = require("mongodb");
14
15 // Use 127.0.0.1 instead of localhost to connect
16 const url = "mongodb://127.0.0.1:27017";
17 const client = new MongoClient(url);
18 async function run() {
19   try {
20     const database = client.db('EmployeeDB');
21     const employees = database.collection('Employee');
22
23     // Query for an Employee with name 'Jack'
24     const query = { EmployeeName: 'Jack' };
25     const employee = await employees.findOne(query);
26     console.log(employee);
27   } finally {
28     // Ensures that the client will close when you finish/error
29     await client.close();
30   }
31 }
32
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

zsh - node\_quickstart

at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1636:16) {  
 errno: -61,  
 code: 'ECONNREFUSED',  
 syscall: 'connect',  
 address: '127.0.0.1',  
 port: 27017  
}  
}
jojojustine@MacBookAir node\_quickstart % node example-insert.js
A document was inserted with the \_id: 67f7162651ae6e620448fa6f
jojojustine@MacBookAir node\_quickstart % node example-query.js
null
jojojustine@MacBookAir node\_quickstart % node example-query.js
null
jojojustine@MacBookAir node\_quickstart % node example-query.js
{
 \_id: new ObjectId('67f7162651ae6e620448fa6f'),
 EmployeeID: 2,
 EmployeeName: 'Jack'
}

OUTLINE

TIMELINE

main\* 0 0 0 Ln 23, Col 43 Spaces: 2 UTF-8 CRLF ( ) JavaScript Go Live

EXPLORER

...

JS connect.js U JS index.js U JS example-insert.js U JS example-query.js U JS example-query2.js U X

ASSIGNMENT7

node\_quickstart > JS example-query2.js > ...

node\_modules

connect.js

example-insert.js

example-query.js

example-query2.js

index.js

package-lock.json

package.json

```
21 async function run() {
47
48
49 } finally {
50   // Ensures that the client will close when you finish/error
51   await client.close();
52 }
53 }
54 }
55 run().catch(console.dir);
56
57
58
59
60
61
62
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

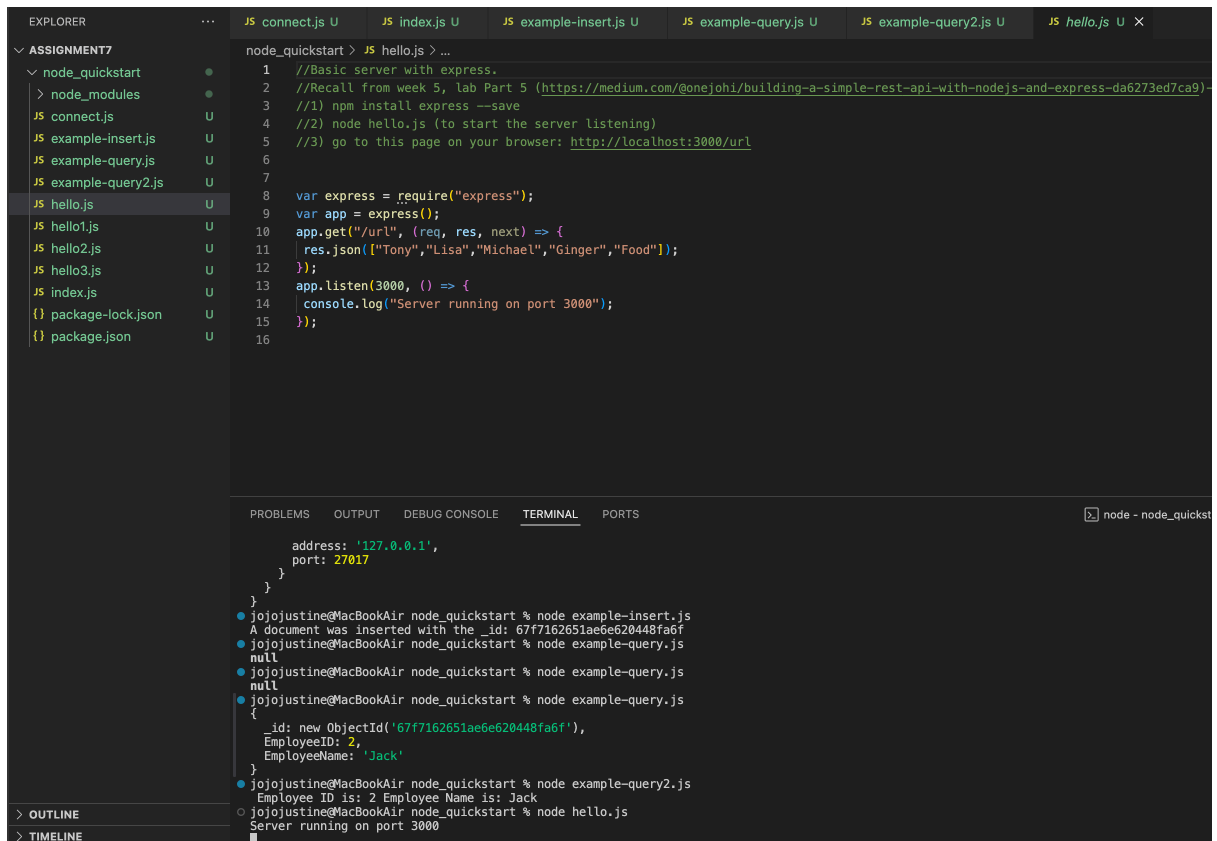
zsh - node\_quickstart

code: 'ECONNREFUSED',  
syscall: 'connect',  
address: '127.0.0.1',  
port: 27017  
}  
}
jojojustine@MacBookAir node\_quickstart % node example-insert.js
A document was inserted with the \_id: 67f7162651ae6e620448fa6f
jojojustine@MacBookAir node\_quickstart % node example-query.js
null
jojojustine@MacBookAir node\_quickstart % node example-query.js
null
jojojustine@MacBookAir node\_quickstart % node example-query.js
{
 \_id: new ObjectId('67f7162651ae6e620448fa6f'),
 EmployeeID: 2,
 EmployeeName: 'Jack'
}

OUTLINE

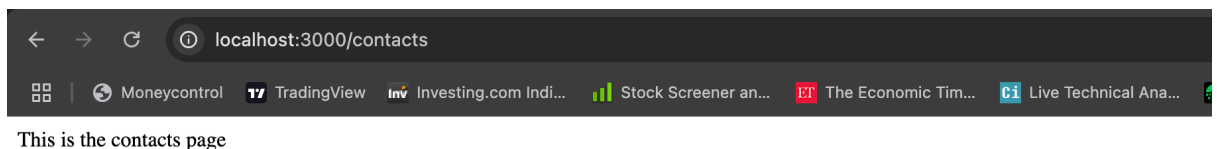
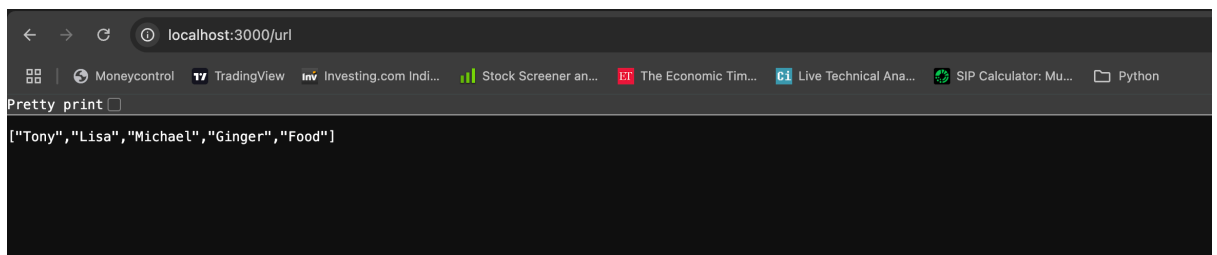
TIMELINE

jojojustine@MacBookAir node\_quickstart % node example-query2.js  
Employee ID is: 2 Employee Name is: Jack  
jojojustine@MacBookAir node\_quickstart %



```
node_quickstart > JS hello.js > ...
1 //Basic server with express.
2 //Recall from week 5, lab Part 5 (https://medium.com/@onejohi/building-a-simple-rest-api-with-nodejs-and-express-da6273ed7ca9)
3 //1) npm install express --save
4 //2) node hello.js (to start the server listening)
5 //3) go to this page on your browser: http://localhost:3000/url
6
7
8 var express = require("express");
9 var app = express();
10 app.get("/url", (req, res, next) => {
11   res.json(["Tony", "Lisa", "Michael", "Ginger", "Food"]);
12 });
13 app.listen(3000, () => {
14   console.log("Server running on port 3000");
15 });
16
```

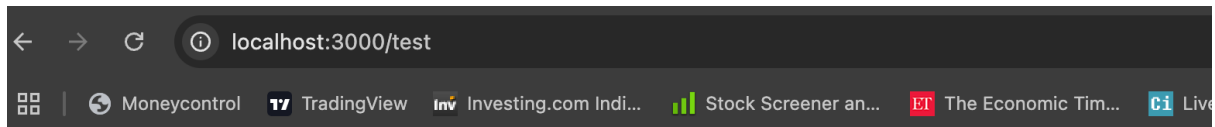
```
address: '127.0.0.1',
port: 27017
}
}
jojojustine@MacBookAir node_quickstart % node example-insert.js
A document was inserted with the _id: 67f7162651ae6e620448fa6f
jojojustine@MacBookAir node_quickstart % node example-query.js
null
jojojustine@MacBookAir node_quickstart % node example-query.js
null
jojojustine@MacBookAir node_quickstart % node example-query.js
{
  _id: new ObjectId('67f7162651ae6e620448fa6f'),
  EmployeeID: 2,
  EmployeeName: 'Jack'
}
jojojustine@MacBookAir node_quickstart % node example-query2.js
Employee ID is: 2 Employee Name is: Jack
jojojustine@MacBookAir node_quickstart % node hello.js
Server running on port 3000
```



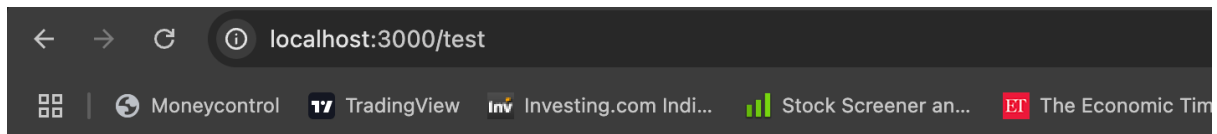
The screenshot shows the VS Code editor interface. The Explorer sidebar on the left displays a project named 'ASSIGNMENT7' with a file structure including 'node\_quickstart' and 'node\_modules'. The main editor area shows a REST client with a GET request to 'localhost:3000/test'. The request body is a JSON object with 'EmployeeID' set to 2 and 'EmployeeName' set to 'Jack'. The response is a JSON object with the same fields. The terminal at the bottom shows the output of the application, including the message 'Server running on port 3000' and the response body.

```
node_quickstart > JS hello2.js > app.get("/test") callback > run
21 app.get("/test", (req, res) => {
25   async function run() {
35     //res.send(employee); //display the full document on the browser [uncomment (and comment the next res.send) if you want to try this]
36     res.send("Employee ID is: " + employee.EmployeeID); //display the EmployeeID of this document to the browser
37   } finally {
38     // Ensures that the client will close when you finish/error
39     await client.close();
40   }
41 }
42
43 run().catch(console.dir);
44
45 //
46 //end route - app.get /test
47
48
49
50
51
52
53 app.listen(3000, () => {
54   console.log("Server running on port 3000");
55 });
56
```

```
jojojustine@MacBookAir node_quickstart % node example-query.js
{
  _id: new ObjectId('67f7162651ae6e620448fa6f'),
  EmployeeID: 2,
  EmployeeName: 'Jack'
}
jojojustine@MacBookAir node_quickstart % node example-query2.js
Employee ID is: 2 Employee Name is: Jack
jojojustine@MacBookAir node_quickstart % node hello.js
Server running on port 3000
^C
jojojustine@MacBookAir node_quickstart % node hello1.js
Server running on port 3000
^C
jojojustine@MacBookAir node_quickstart % node hello2.js
Server running on port 3000
{
  _id: new ObjectId('67f7162651ae6e620448fa6f'),
  EmployeeID: 2,
  EmployeeName: 'Jack'
}
```



Employee ID is: 2



Employee ID is: 2

