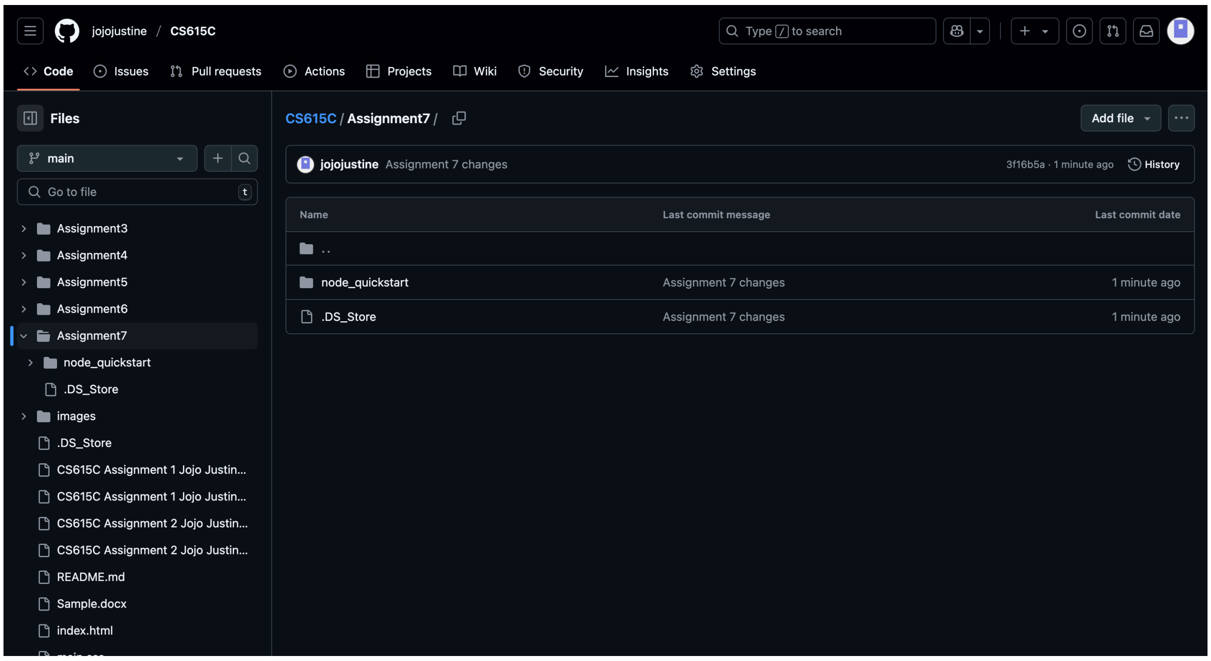
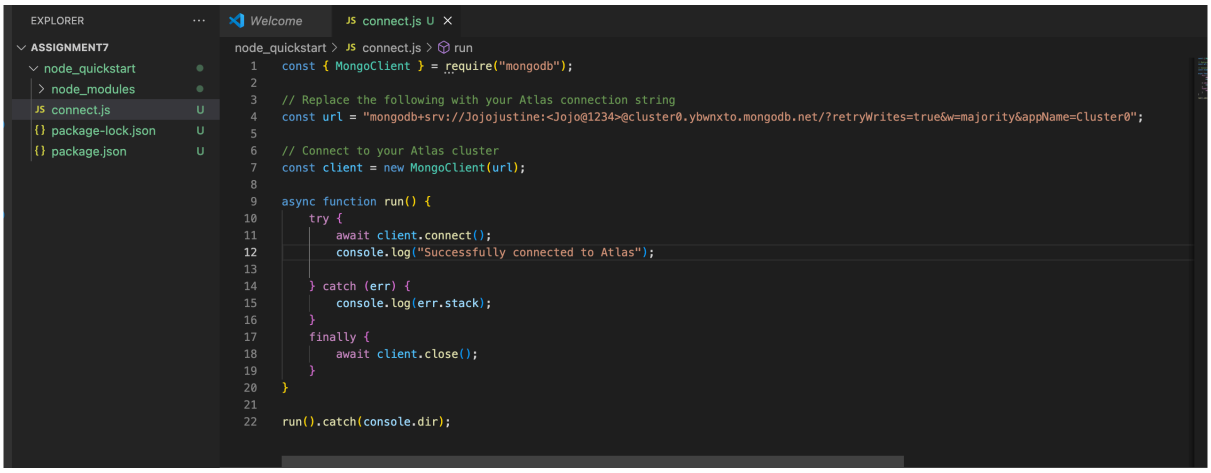

jojojustine / CS615C

CS615C / Assignment7 /

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

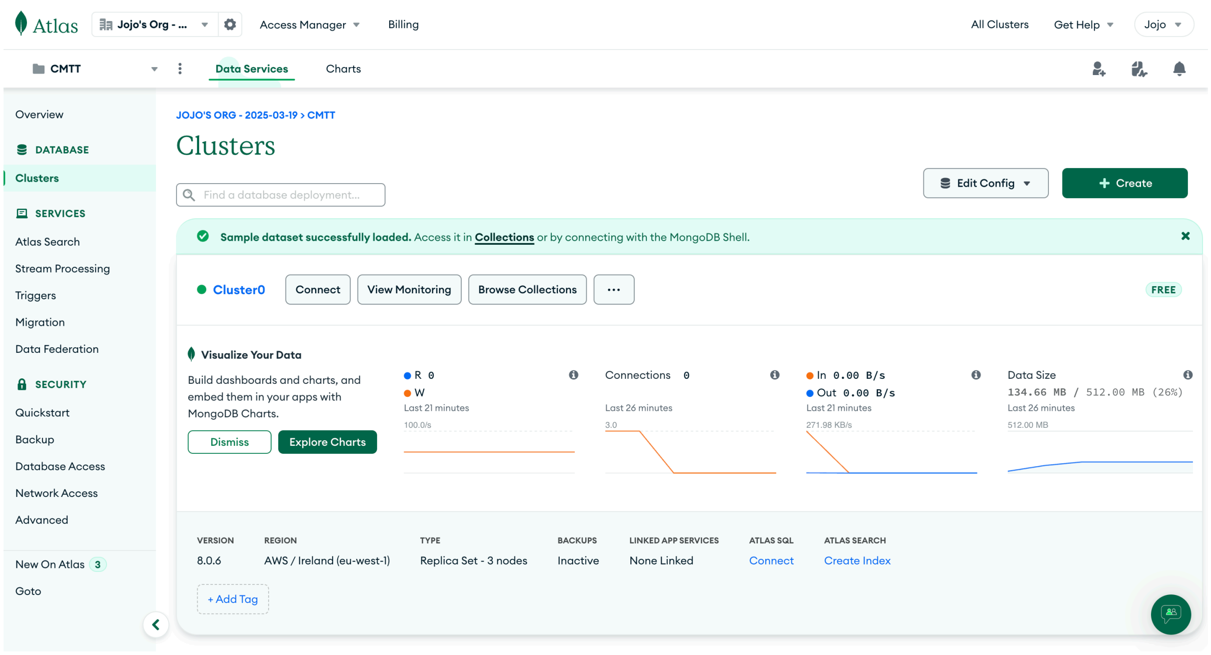
Files: Assignment3, Assignment4, Assignment5, Assignment6, Assignment7, node_quickstart, .DS_Store, images, .DS_Store, CS615C Assignment 1 Jojo Justin..., CS615C Assignment 1 Jojo Justin..., CS615C Assignment 2 Jojo Justin..., CS615C Assignment 2 Jojo Justin..., README.md, Sample.docx, index.html


EXPLORER

ASSIGNMENT7
- node_quickstart
  - node_modules
  - connect.js U
  - package-lock.json U
  - package.json U

node_quickstart > connect.js > run

```javascript
const { MongoClient } = require("mongodb");

// Replace the following with your Atlas connection string
const url = "mongodb+srv://Jojojustine:<Jojo@1234>@cluster0.ybwnxto.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0";

// Connect to your Atlas cluster
const client = new MongoClient(url);

async function run() {
    try {
        await client.connect();
        console.log("Successfully connected to Atlas");

    } catch (err) {
        console.log(err.stack);
    }
    finally {
        await client.close();
    }
}

run().catch(console.dir);
```



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

Clusters

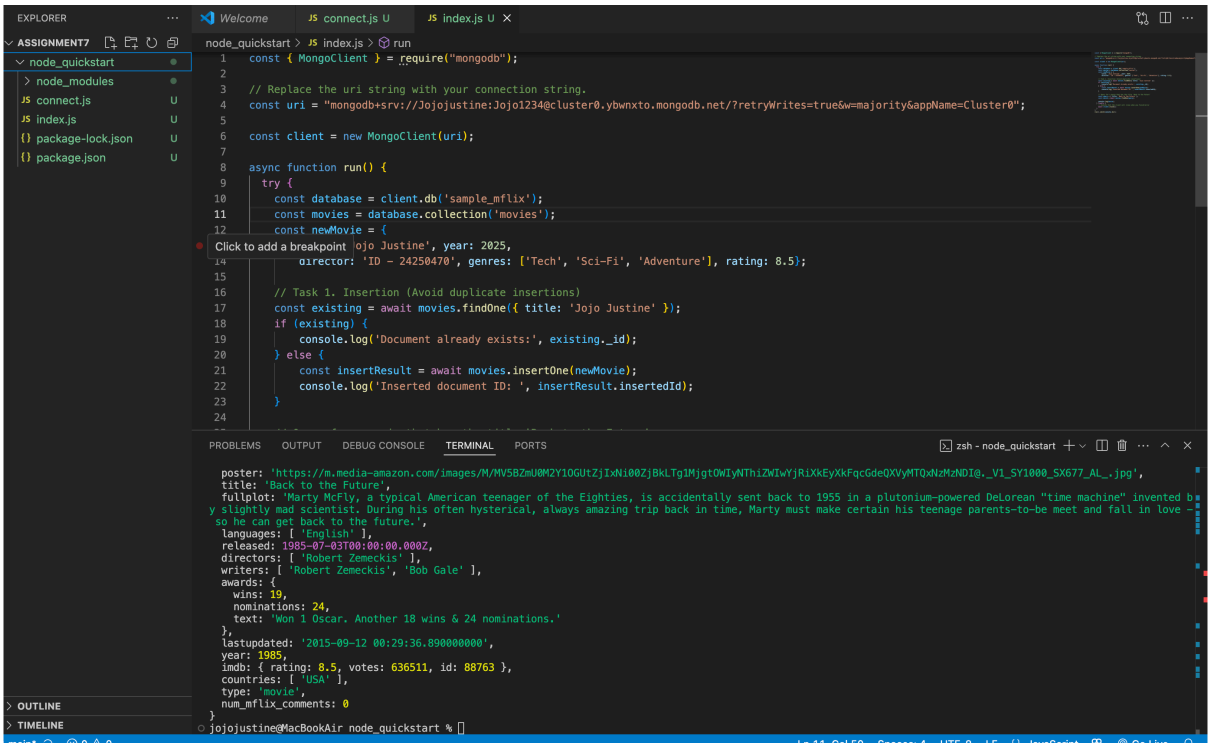
Sample dataset successfully loaded. Access it in Collections or by connecting with the MongoDB Shell.

Visualize Your Data

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

```
● R 0
● W
Last 21 minutes
100.0/s

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
```

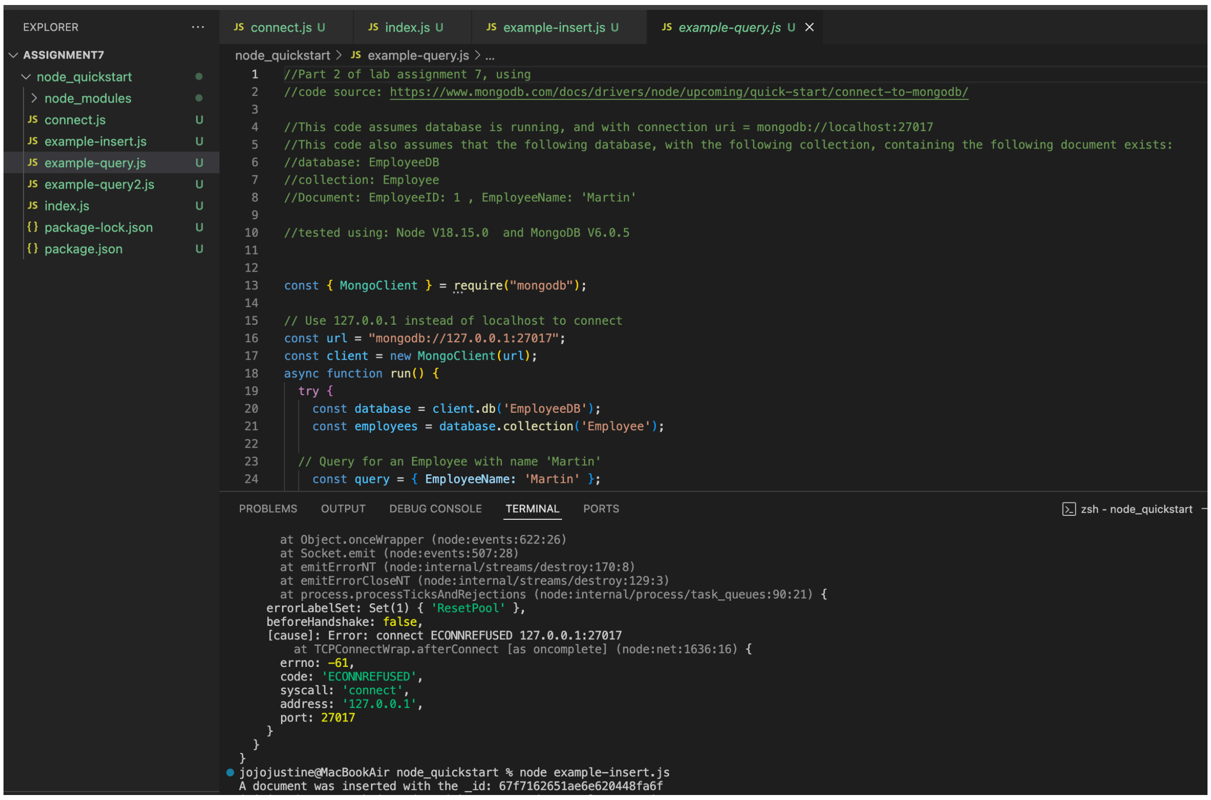
VERSION	REGION	TYPE	BACKUPS	LINKED APP SERVICES	ATLAS SQL	ATLAS SEARCH
8.0.6	AWS / Ireland (eu-west-1)	Replica Set - 3 nodes	Inactive	None Linked	Connect	Create Index



```
const { MongoClient } = require("mongodb");

// Replace the uri string with your connection string.
const uri = "mongodb+srv://Jojojustine:Jojo1234@cluster0.ybwnxto.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0";

const client = new MongoClient(uri);

async function run() {
  try {
    const database = client.db('sample_mflix');
    const movies = database.collection('movies');
    const newMovie = {
                          ojo Justine', year: 2025,
        director: 'ID - 24250470', genres: ['Tech', 'Sci-Fi', 'Adventure'], rating: 8.5};

    // Task 1. Insertion (Avoid duplicate insertions)
    const existing = await movies.findOne({ title: 'Jojo Justine' });
    if (existing) {
        console.log('Document already exists:', existing._id);
    } else {
        const insertResult = await movies.insertOne(newMovie);
        console.log('Inserted document ID: ', insertResult.insertedId);
    }

```

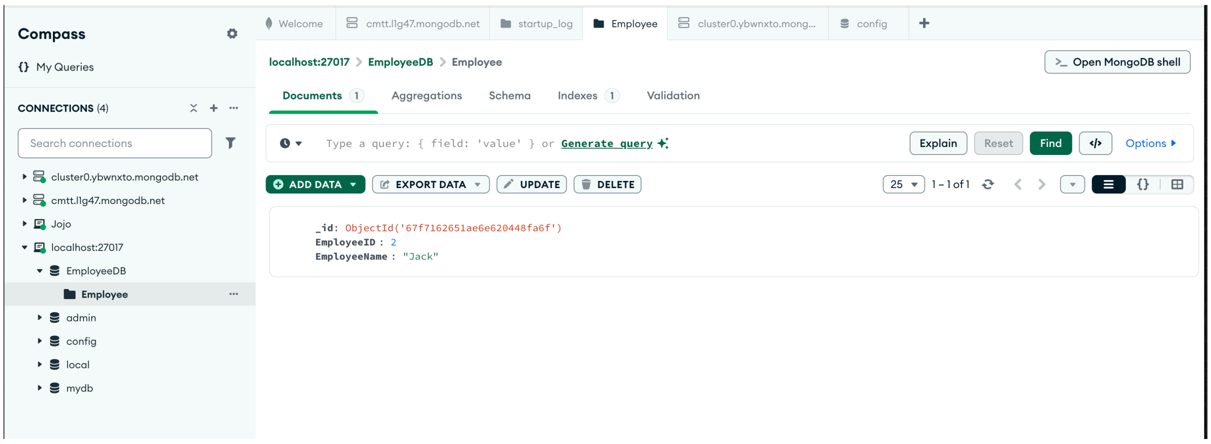
```
  poster: 'https://m.media-amazon.com/images/M/MV5BZmU0M2Y1OGUtZjIxNi00ZjBkLTg1MjgtOWIyNThiZWIwYjRiXkEyXkFqcGdeQXVyMTQxNzMzNDI@._V1_SY1000_SX677_AL_.jpg',
  title: 'Back to the Future',
  fullplot: 'Marty McFly, a typical American teenager of the Eighties, is accidentally sent back to 1955 in a plutonium-powered DeLorean "time machine" invented b
y 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

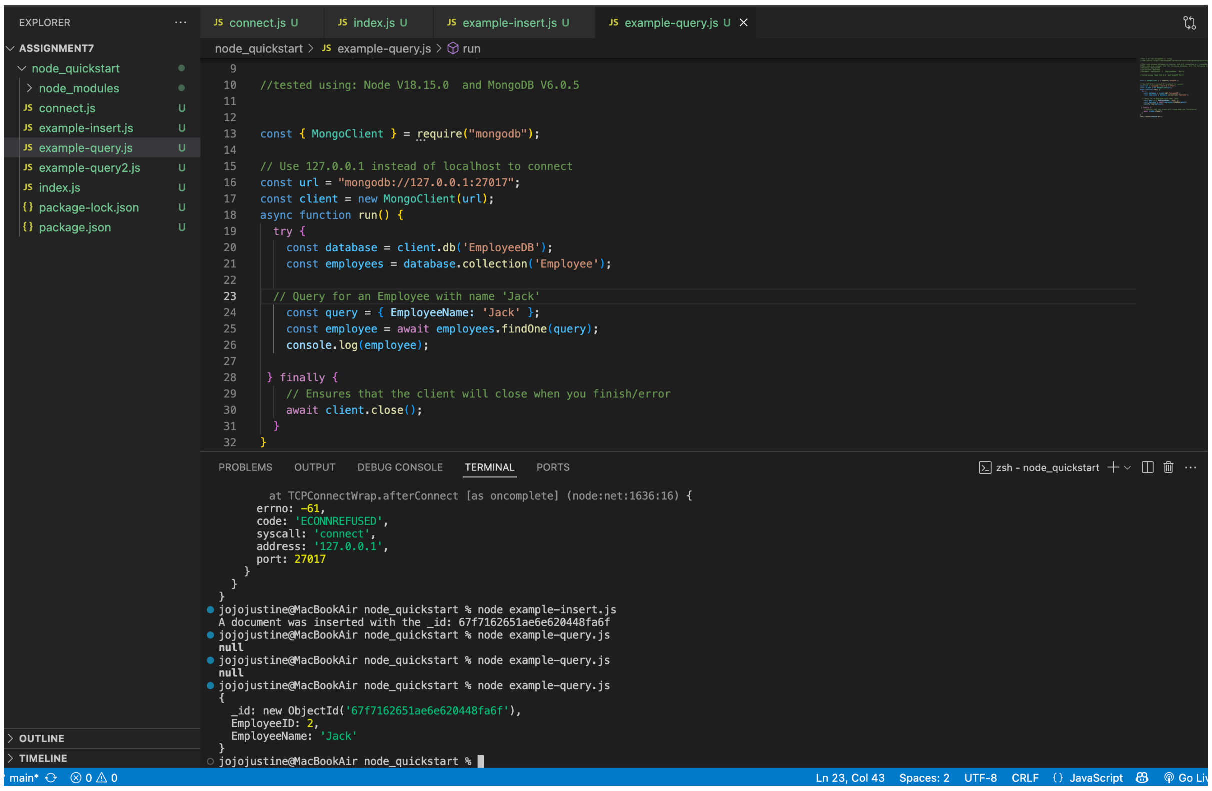


```
//Part 2 of lab assignment 7, using
//code source: https://www.mongodb.com/docs/drivers/node/upcoming/quick-start/connect-to-mongodb/

//This code assumes database is running, and with connection uri = mongodb://localhost:27017
//This code also assumes that the following database, with the following collection, containing the following document exists:
//database: EmployeeDB
//collection: Employee
//Document: EmployeeID: 1 , EmployeeName: 'Martin'

//tested using: Node V18.15.0  and MongoDB V6.0.5


const { MongoClient } = require("mongodb");

// Use 127.0.0.1 instead of localhost to connect
const url = "mongodb://127.0.0.1:27017";
const client = new MongoClient(url);
async function run() {
  try {
    const database = client.db('EmployeeDB');
    const employees = database.collection('Employee');

  // Query for an Employee with name 'Martin'
    const query = { EmployeeName: 'Martin' };
```

```
      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: 67f7162651ae6e620448fa6f
```

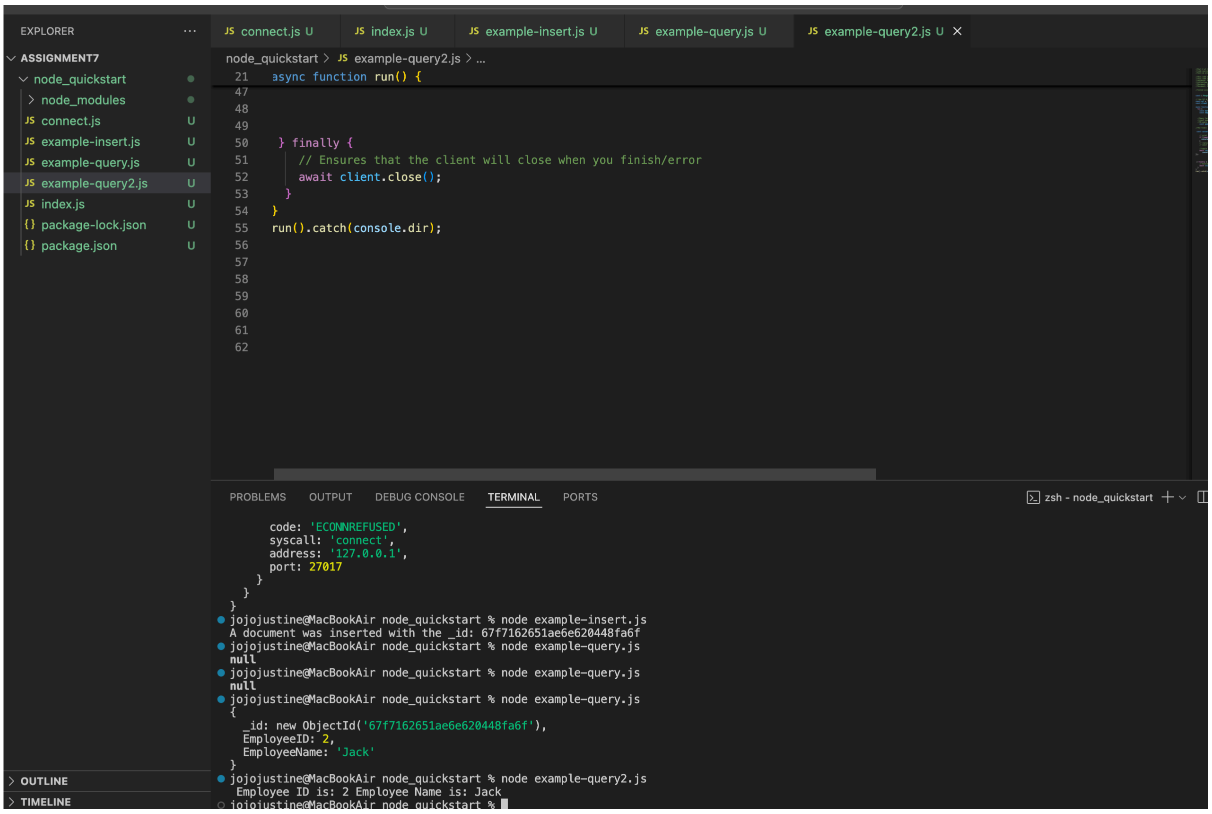


```
_id: ObjectId('67f7162651ae6e620448fa6f')
EmployeeID : 2
EmployeeName : "Jack"
```



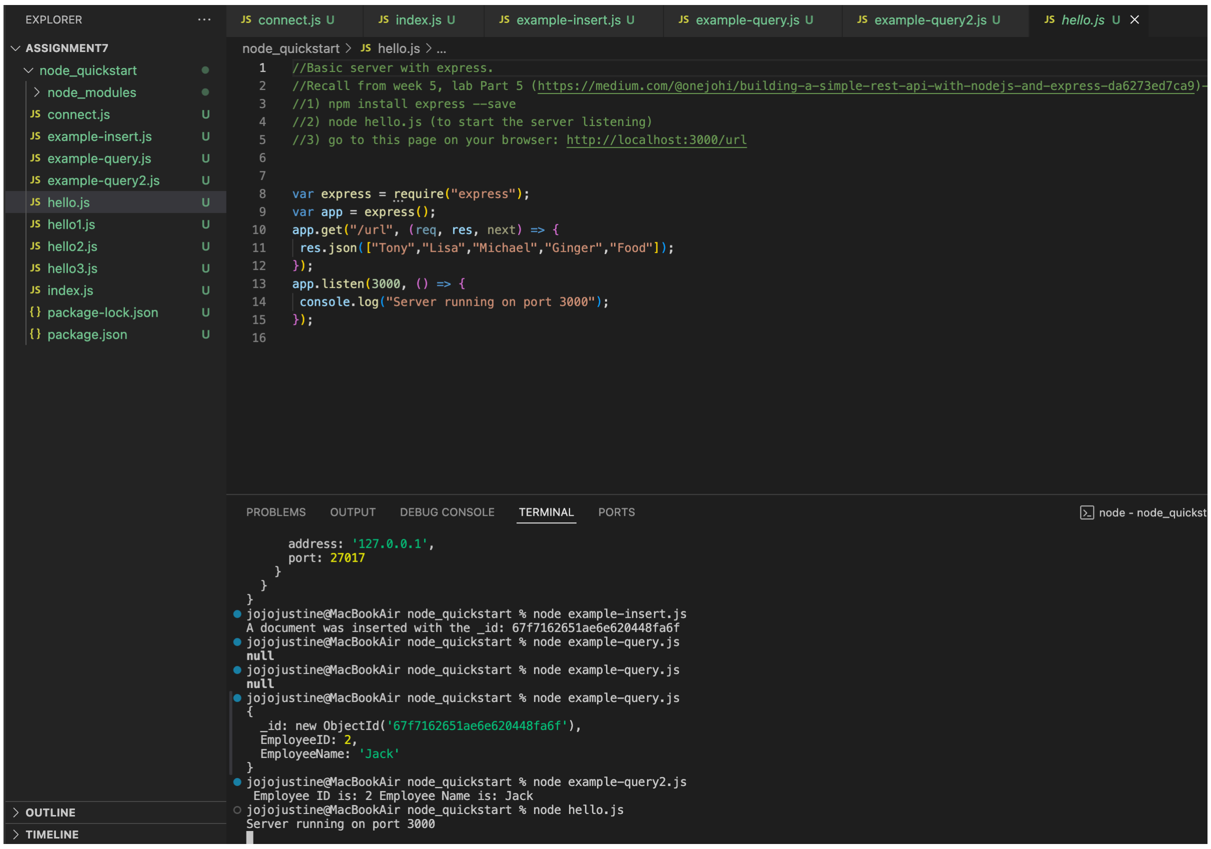
```javascript

//tested using: Node V18.15.0  and MongoDB V6.0.5


const { MongoClient } = require("mongodb");

// Use 127.0.0.1 instead of localhost to connect
const url = "mongodb://127.0.0.1:27017";
const client = new MongoClient(url);
async function run() {
  try {
    const database = client.db('EmployeeDB');
    const employees = database.collection('Employee');

  // Query for an Employee with name 'Jack'
    const query = { EmployeeName: 'Jack' };
    const employee = await employees.findOne(query);
    console.log(employee);

 } finally {
    // Ensures that the client will close when you finish/error
    await client.close();
  }
}
```

```
        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'
}
jojojustine@MacBookAir node_quickstart %
```



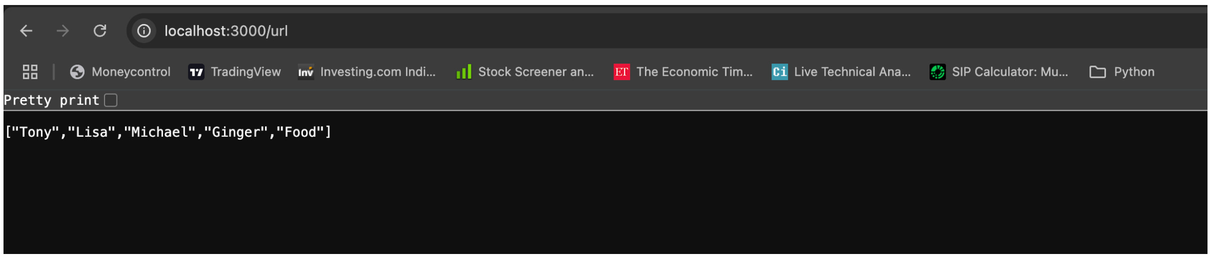
```javascript
async function run() {



  } finally {
    // Ensures that the client will close when you finish/error
    await client.close();
  }
}
run().catch(console.dir);
```

```
      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'
}
jojojustine@MacBookAir node_quickstart % node example-query2.js
 Employee ID is: 2 Employee Name is: Jack
jojojustine@MacBookAir node_quickstart %
```

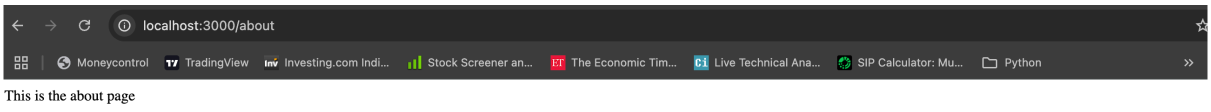


```javascript
//Basic server with express.
//Recall from week 5, lab Part 5 (https://medium.com/@onejohi/building-a-simple-rest-api-with-nodejs-and-express-da6273ed7ca9)-
//1) npm install express --save
//2) node hello.js (to start the server listening)
//3) go to this page on your browser: http://localhost:3000/url


var express = require("express");
var app = express();
app.get("/url", (req, res, next) => {
 res.json(["Tony","Lisa","Michael","Ginger","Food"]);
});
app.listen(3000, () => {
 console.log("Server running on port 3000");
});

```

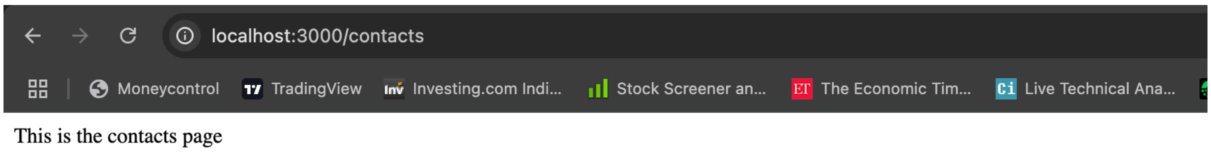
```
      address: '127.0.0.1',
      port: 27017
    }
  }
}
jojojustine@MacBookAir node_quickstart % node example-insert.js
A document was inserted with the _id: 67f7162651ae6e620448fa6f
jojojustine@MacBookAir node_quickstart % node example-query.js
null
jojojustine@MacBookAir node_quickstart % node 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
```



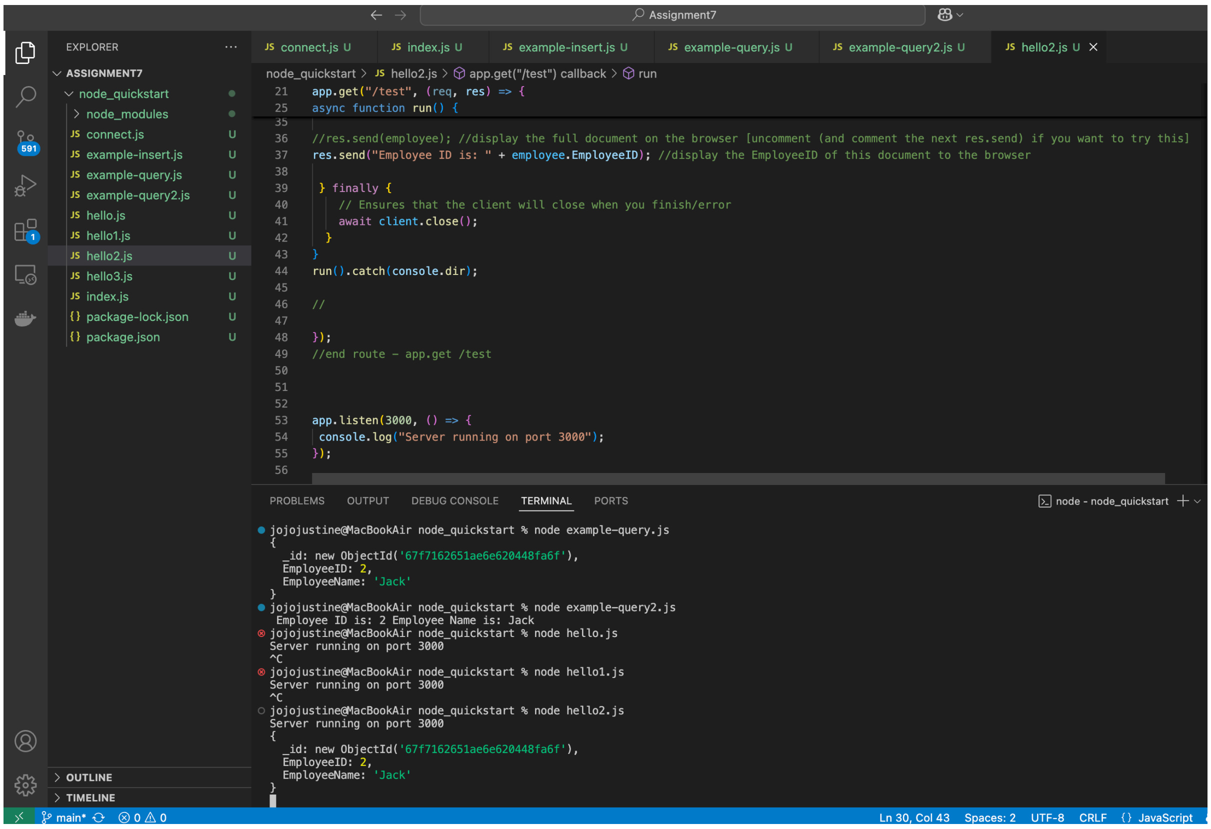
```
["Tony","Lisa","Michael","Ginger","Food"]
```



This is the about page

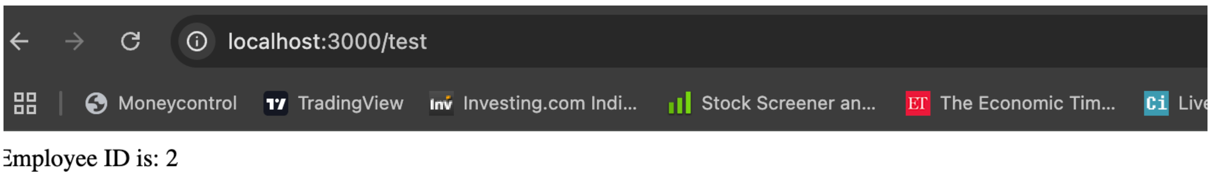


This is the contacts page



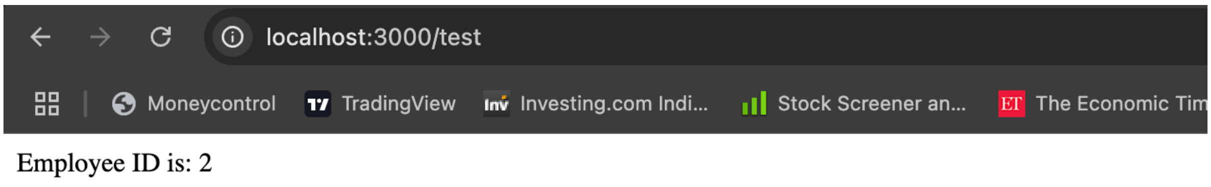
```
app.get("/test", (req, res) => {
async function run() {

//res.send(employee); //display the full document on the browser [uncomment (and comment the next res.send) if you want to try this]
res.send("Employee ID is: " + employee.EmployeeID); //display the EmployeeID of this document to the browser

 } finally {
   // Ensures that the client will close when you finish/error
   await client.close();
  }
}
run().catch(console.dir);

//

});
//end route - app.get /test



app.listen(3000, () => {
 console.log("Server running on port 3000");
});
```

```
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'
}
```

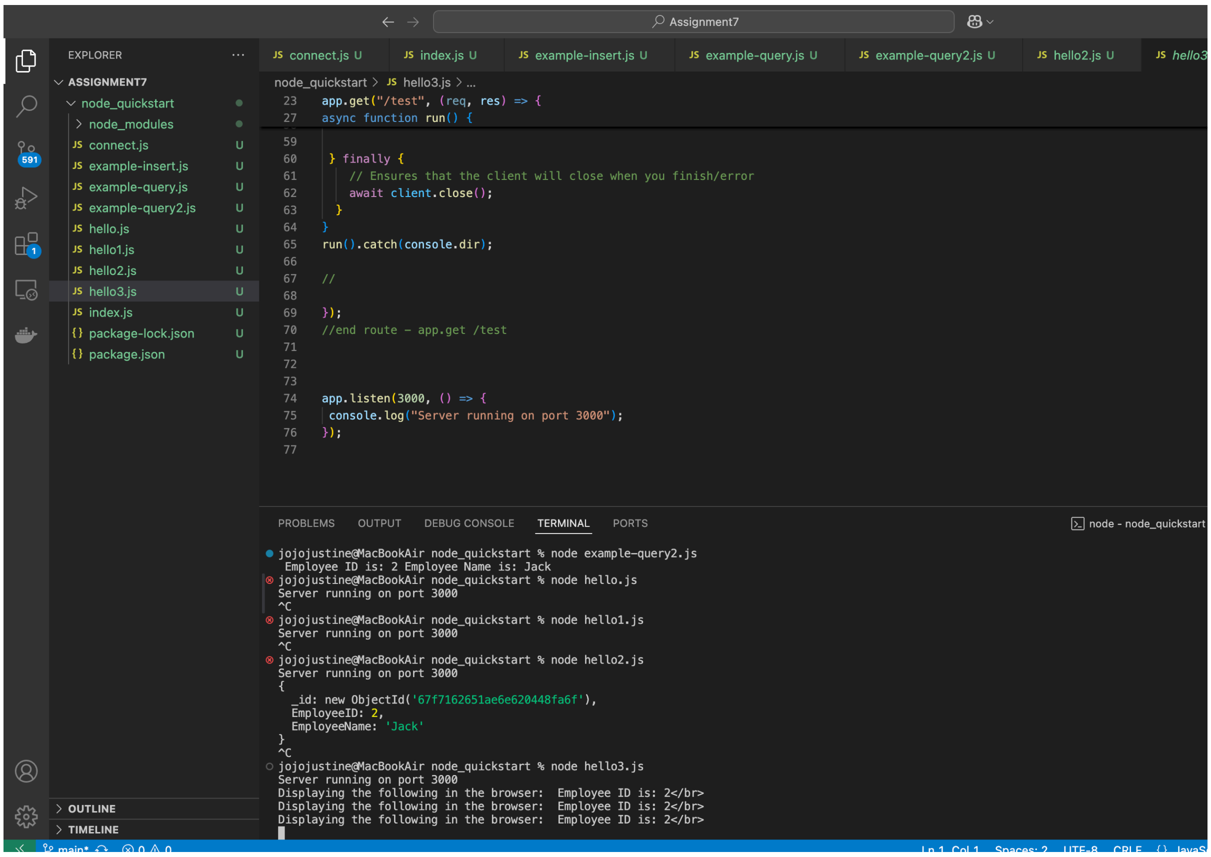


localhost:3000/test

Employee ID is: 2



Employee ID is: 2



```
app.get("/test", (req, res) => {
async function run() {

  } finally {
    // Ensures that the client will close when you finish/error
    await client.close();
  }
}
run().catch(console.dir);

//

});
//end route - app.get /test



app.listen(3000, () => {
  console.log("Server running on port 3000");
});
```

```
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'
}
^C
jojojustine@MacBookAir node_quickstart % node hello3.js
Server running on port 3000
Displaying the following in the browser:  Employee ID is: 2</br>
Displaying the following in the browser:  Employee ID is: 2</br>
Displaying the following in the browser:  Employee ID is: 2</br>
```