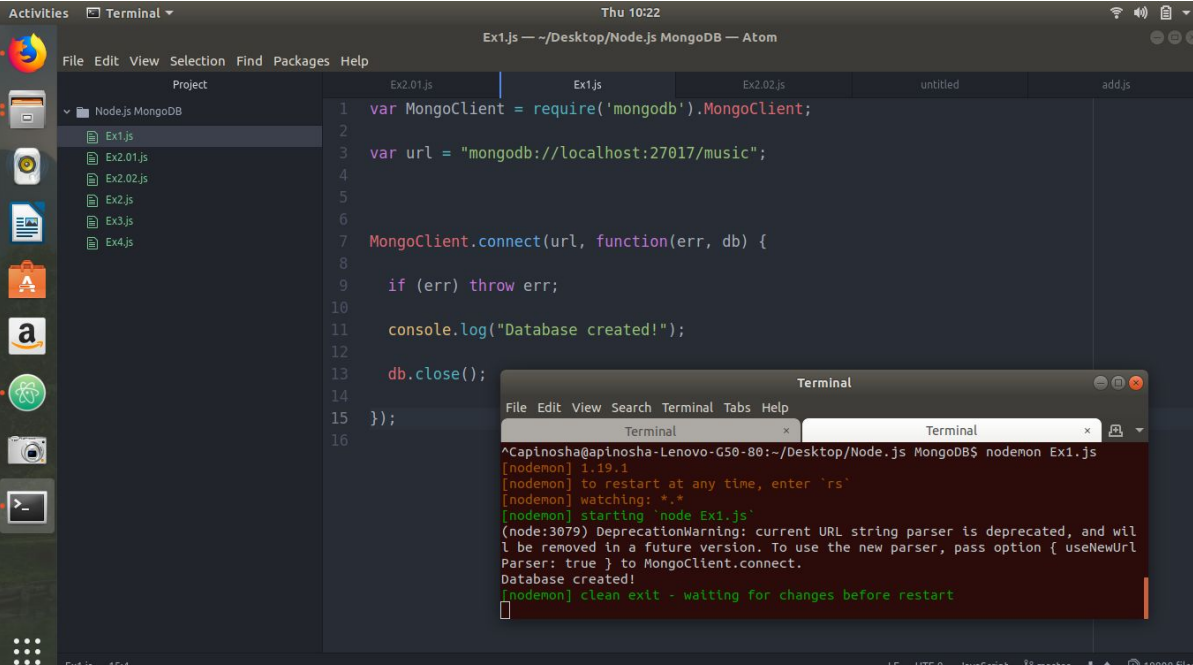


1) Create a Database called music.



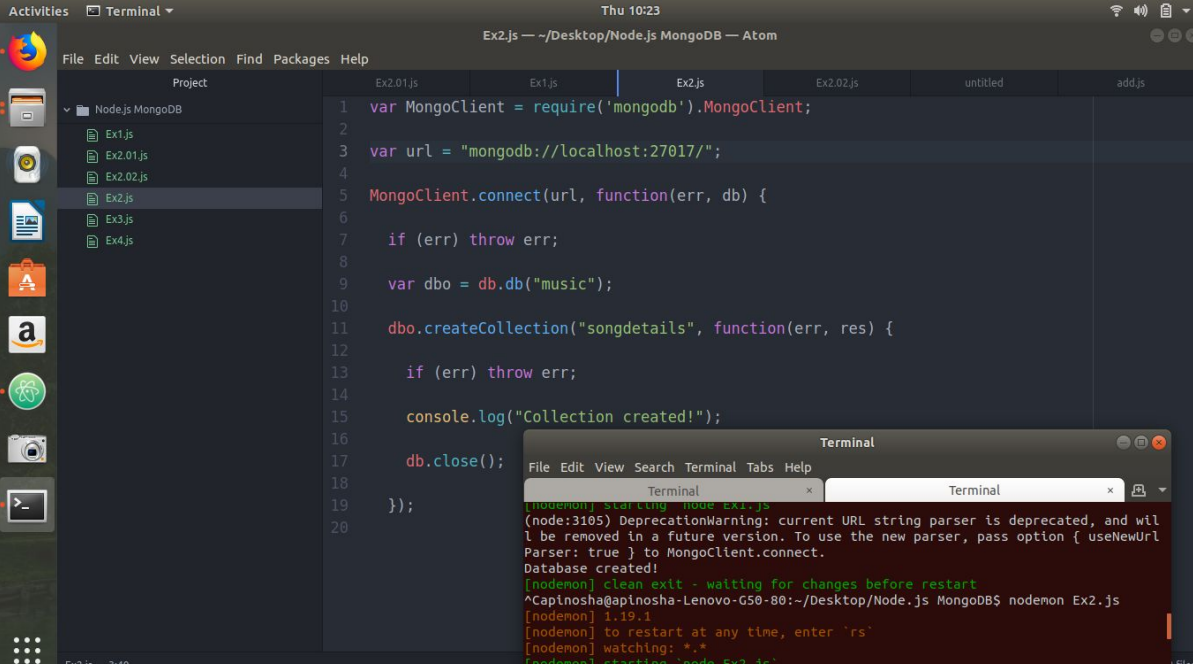
The screenshot shows the Atom editor with a project named 'Node.js MongoDB'. The file 'Ex1.js' is open, containing the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/music";
4
5
6
7 MongoClient.connect(url, function(err, db) {
8
9     if (err) throw err;
10
11     console.log("Database created!");
12
13     db.close();
14
15 });
16
```

A terminal window is open in the foreground, showing the command `nodemon Ex1.js` being executed. The terminal output includes:

```
^Capinosha@apinosha-Lenovo-G50-80:~/Desktop/Node.js MongoDB$ nodemon Ex1.js
[nodemon] 1.19.1
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching: *.*
[nodemon] starting 'node Ex1.js'
(node:3079) DeprecationWarning: current URL string parser is deprecated, and will
be removed in a future version. To use the new parser, pass option { useNewUrlParser: true } to MongoClient.connect.
Database created!
[nodemon] clean exit - waiting for changes before restart
```

2) Create a collection called songdetails.



The screenshot shows the Atom editor with the same project. The file 'Ex2.js' is open, containing the following JavaScript code:

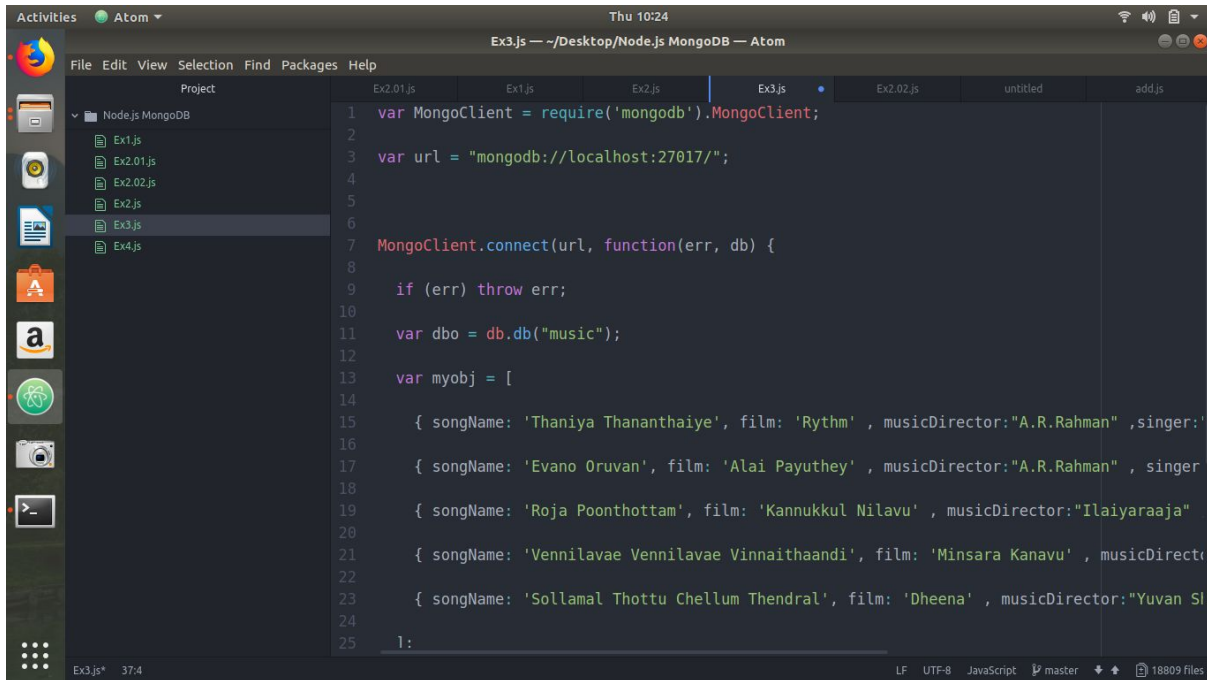
```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7     if (err) throw err;
8
9     var dbo = db.db("music");
10
11     dbo.createCollection("songdetails", function(err, res) {
12
13         if (err) throw err;
14
15         console.log("Collection created!");
16
17         db.close();
18
19     });
20
21 });
22
```

A terminal window is open in the foreground, showing the command `nodemon Ex2.js` being executed. The terminal output includes:

```
^Capinosha@apinosha-Lenovo-G50-80:~/Desktop/Node.js MongoDB$ nodemon Ex2.js
[nodemon] 1.19.1
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching: *.*
[nodemon] starting 'node Ex2.js'
(node:3105) DeprecationWarning: current URL string parser is deprecated, and will
be removed in a future version. To use the new parser, pass option { useNewUrlParser: true } to MongoClient.connect.
Database created!
[nodemon] clean exit - waiting for changes before restart
```

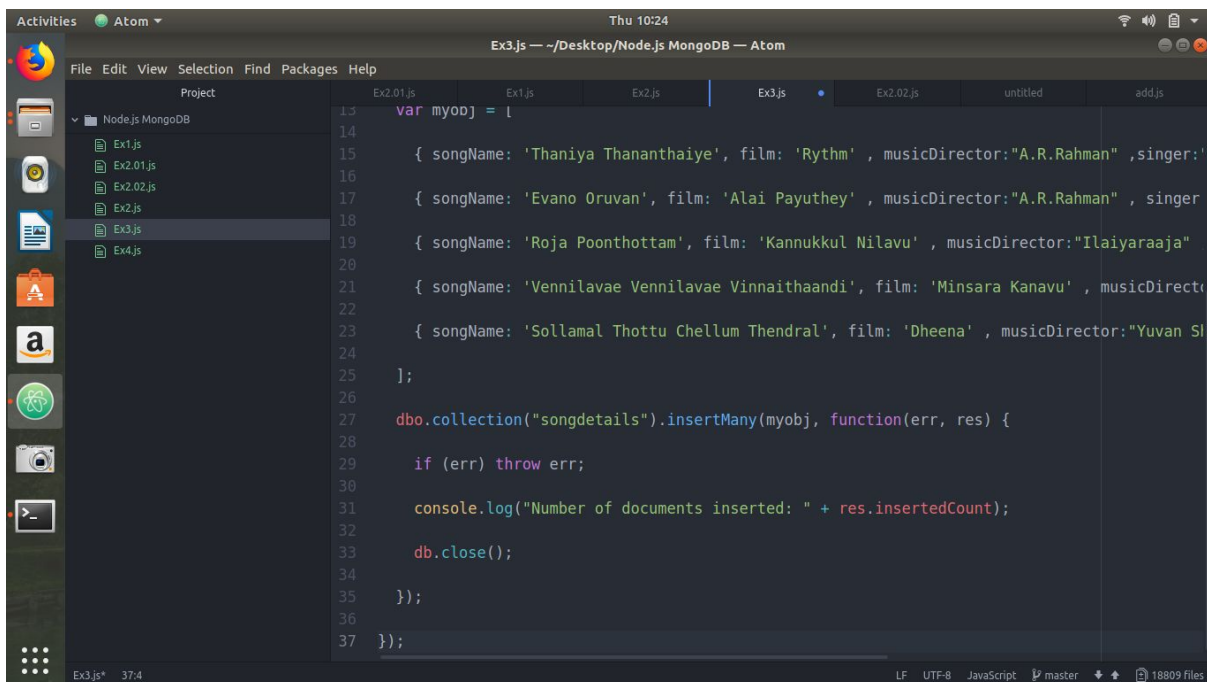
3) Create the above 5 song documents.

4) List all documents created.



This screenshot shows the Atom editor interface with the file 'Ex3.js' open. The left sidebar displays a project tree for 'Node.js MongoDB' containing files 'Ex1.js', 'Ex2.01.js', 'Ex2.02.js', 'Ex2.js', 'Ex3.js', and 'Ex4.js'. The main editor area shows the following JavaScript code:

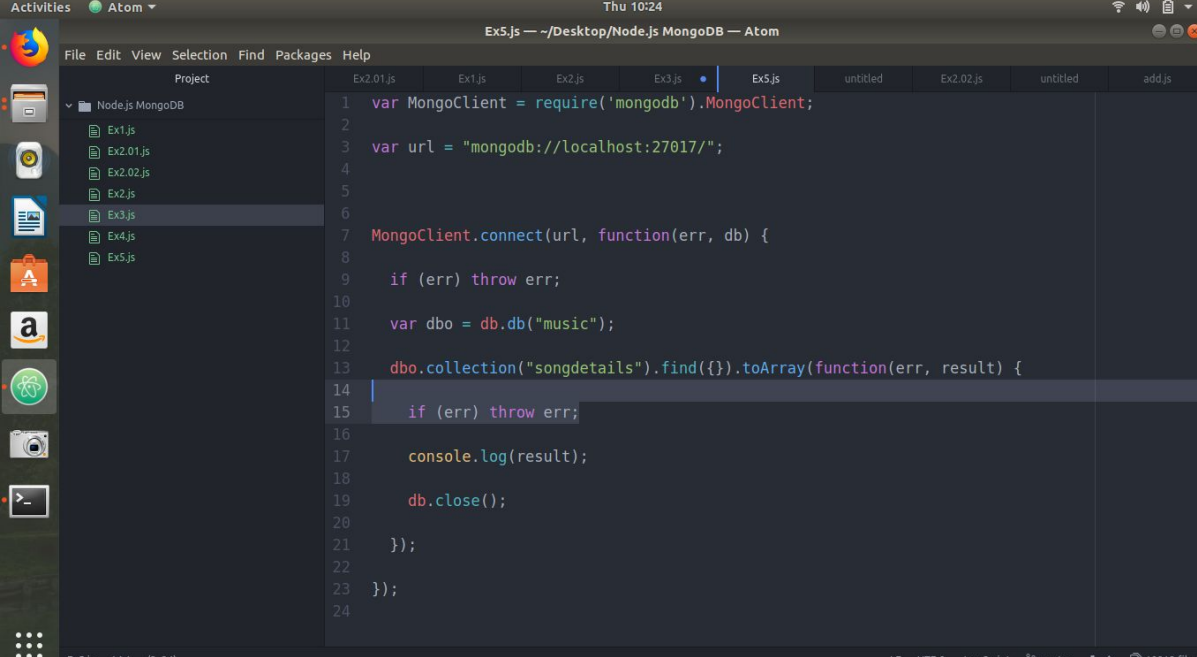
```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5
6
7 MongoClient.connect(url, function(err, db) {
8
9   if (err) throw err;
10
11   var dbo = db.db("music");
12
13   var myobj = [
14
15     { songName: 'Thaniya Thananthaiye', film: 'Rythm' , musicDirector:"A.R.Rahman" , singer:'
16
17     { songName: 'Evano Oruvan', film: 'Alai Payuthey' , musicDirector:"A.R.Rahman" , singer
18
19     { songName: 'Roja Poonthottam', film: 'Kannukkul Nilavu' , musicDirector:"Ilaiyaraaja"
20
21     { songName: 'Vennilavae Vennilavae Vinnaithaandi', film: 'Minsara Kanavu' , musicDirect
22
23     { songName: 'Sollamal Thottu Chellum Thendral', film: 'Dheena' , musicDirector:"Yuvan SI
24
25   ];
```



This screenshot shows the Atom editor interface with the file 'Ex3.js' open, displaying the continuation of the JavaScript code from the previous screenshot:

```
13 var myobj = [
14
15   { songName: 'Thaniya Thananthaiye', film: 'Rythm' , musicDirector:"A.R.Rahman" , singer:'
16
17   { songName: 'Evano Oruvan', film: 'Alai Payuthey' , musicDirector:"A.R.Rahman" , singer
18
19   { songName: 'Roja Poonthottam', film: 'Kannukkul Nilavu' , musicDirector:"Ilaiyaraaja"
20
21   { songName: 'Vennilavae Vennilavae Vinnaithaandi', film: 'Minsara Kanavu' , musicDirect
22
23   { songName: 'Sollamal Thottu Chellum Thendral', film: 'Dheena' , musicDirector:"Yuvan SI
24
25 ];
26
27 dbo.collection("songdetails").insertMany(myobj, function(err, res) {
28
29   if (err) throw err;
30
31   console.log("Number of documents inserted: " + res.insertedCount);
32
33   db.close();
34
35 });
36
37 };
```

5) List A.R.Rahman's songs.

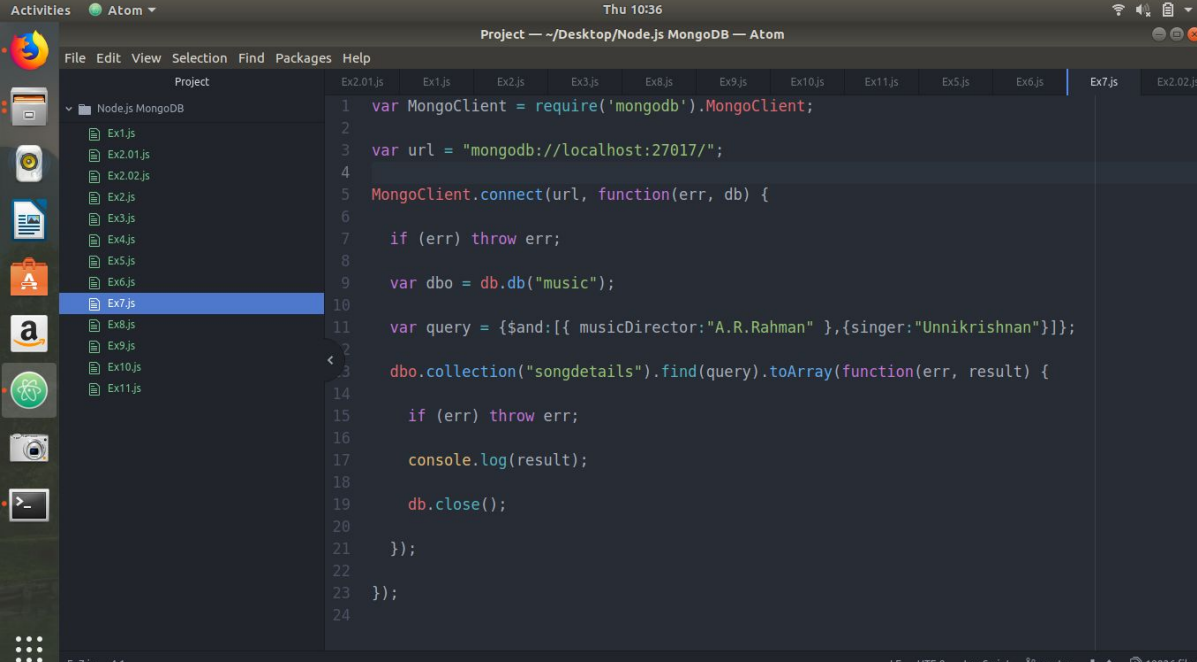


The screenshot shows the Atom editor interface. The left sidebar displays a project named 'Node.js MongoDB' with files Ex1.js through Ex5.js. The main editor window is titled 'Ex5.js' and contains the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5
6
7 MongoClient.connect(url, function(err, db) {
8
9   if (err) throw err;
10
11   var dbo = db.db("music");
12
13   dbo.collection("songdetails").find({}).toArray(function(err, result) {
14
15     if (err) throw err;
16
17     console.log(result);
18
19     db.close();
20
21   });
22
23 });
24
```

The status bar at the bottom indicates 'Ex5.js 14:1 (2, 24)' and '18812 files'.

6) List A.R.Rahman's songs sung by Unnikrishnan.

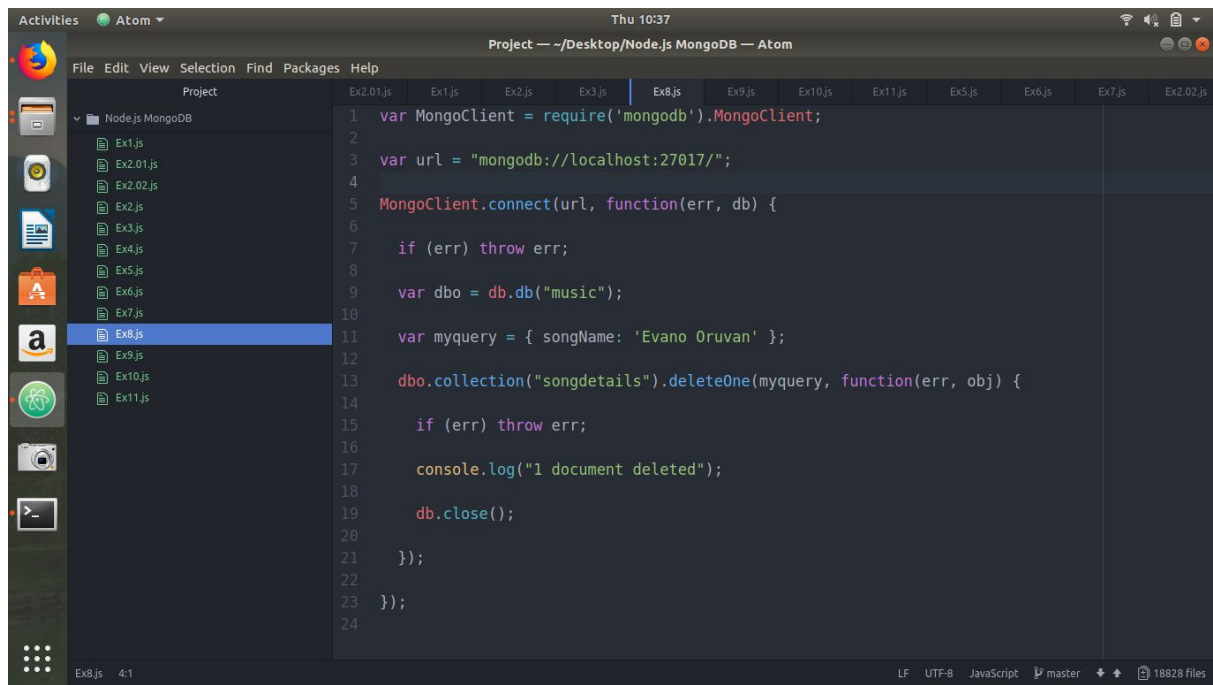


The screenshot shows the Atom editor interface. The left sidebar displays a project named 'Node.js MongoDB' with files Ex1.js through Ex11.js. The main editor window is titled 'Ex7.js' and contains the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7   if (err) throw err;
8
9   var dbo = db.db("music");
10
11   var query = {$and:[{ musicDirector:"A.R.Rahman" },{singer:"Unnikrishnan"}]};
12
13   dbo.collection("songdetails").find(query).toArray(function(err, result) {
14
15     if (err) throw err;
16
17     console.log(result);
18
19     db.close();
20
21   });
22
23 });
24
```

The status bar at the bottom indicates 'Ex7.js 4:1' and '18826 files'.

7) Delete the song which you don't like.

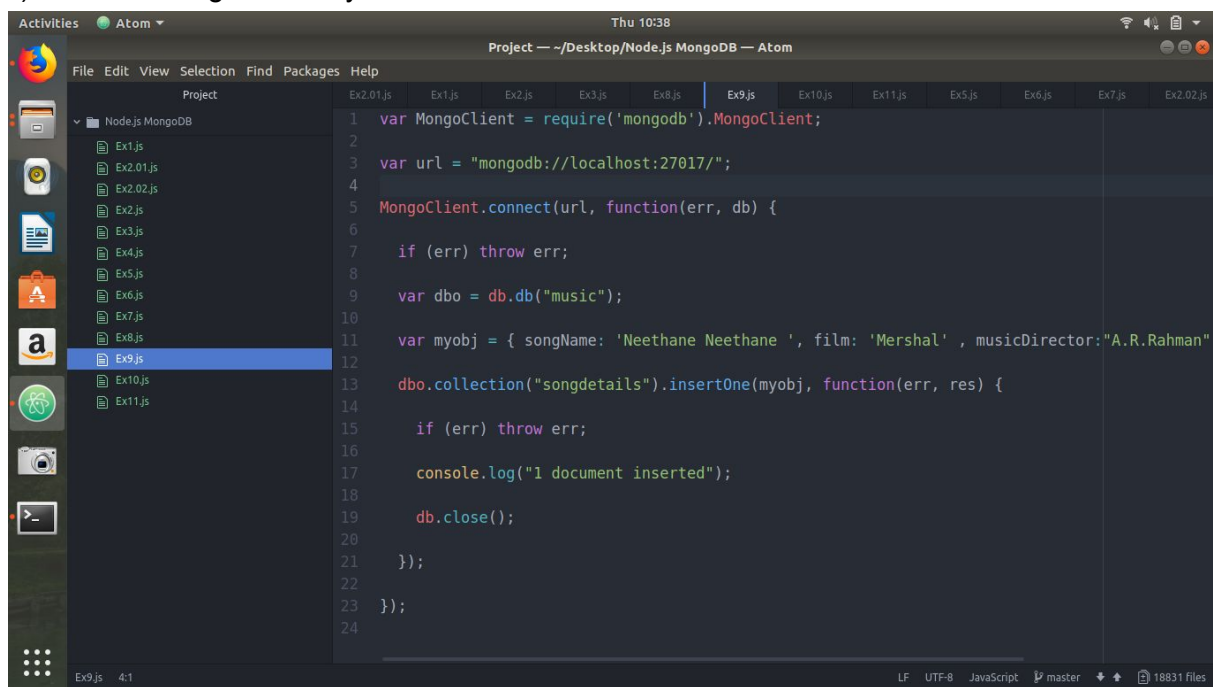


The screenshot shows the Atom editor interface. The left sidebar displays a project named 'Node.js MongoDB' with a file explorer showing files from Ex1.js to Ex11.js. The main editor window is open to 'Ex8.js' and contains the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7     if (err) throw err;
8
9     var dbo = db.db("music");
10
11     var myquery = { songName: 'Evano Oruvan' };
12
13     dbo.collection("songdetails").deleteOne(myquery, function(err, obj) {
14
15         if (err) throw err;
16
17         console.log("1 document deleted");
18
19         db.close();
20
21     });
22
23 });
24
```

The status bar at the bottom indicates the file is 'Ex8.js', line 4:1, using the 'LF' line ending, 'UTF-8' encoding, 'JavaScript' language, and 'master' branch, with 18828 files in the project.

8) Add new song which is your favourite.

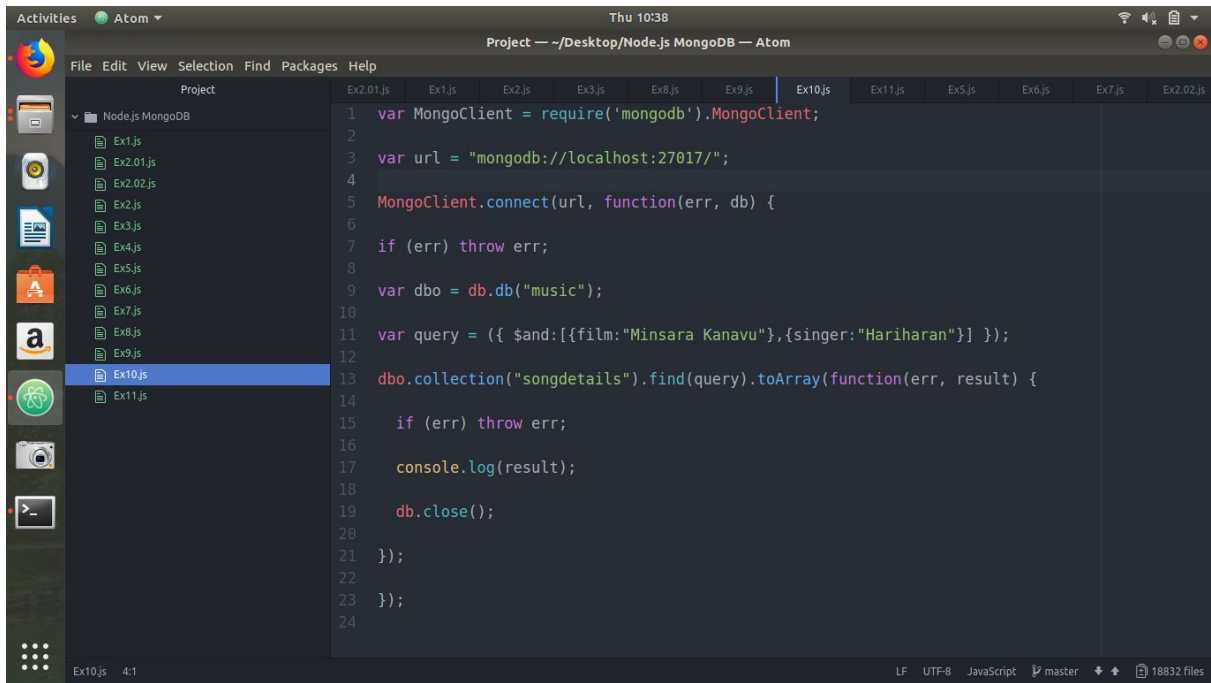


The screenshot shows the Atom editor interface. The left sidebar displays the same project 'Node.js MongoDB' with the file explorer. The main editor window is now open to 'Ex9.js' and contains the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7     if (err) throw err;
8
9     var dbo = db.db("music");
10
11     var myobj = { songName: 'Neethane Neethane ', film: 'Mershal' , musicDirector:"A.R.Rahman"
12
13     dbo.collection("songdetails").insertOne(myobj, function(err, res) {
14
15         if (err) throw err;
16
17         console.log("1 document inserted");
18
19         db.close();
20
21     });
22
23 });
24
```

The status bar at the bottom indicates the file is 'Ex9.js', line 4:1, using the 'LF' line ending, 'UTF-8' encoding, 'JavaScript' language, and 'master' branch, with 18831 files in the project.

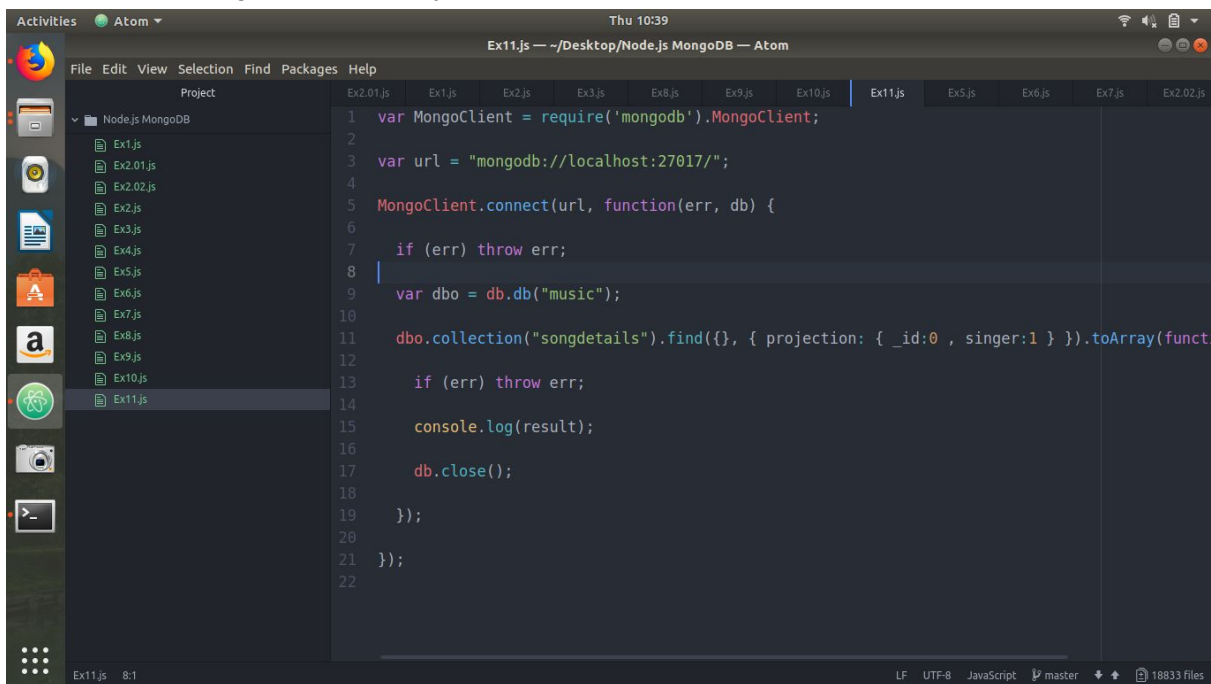
9) List Songs sung by Hariharan from Minsara kanavu film.



The screenshot shows the Atom editor interface with a project named "Node.js MongoDB". The file explorer on the left lists files from Ex1.js to Ex11.js, with Ex10.js selected. The main editor pane displays the code in Ex10.js, which connects to a MongoDB instance and queries a collection named "songdetails" for documents where the film is "Minsara Kanavu" and the singer is "Hariharan".

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7   if (err) throw err;
8
9   var dbo = db.db("music");
10
11  var query = ({ $and:[{film:"Minsara Kanavu"},{singer:"Hariharan"}] });
12
13  dbo.collection("songdetails").find(query).toArray(function(err, result) {
14
15    if (err) throw err;
16
17    console.log(result);
18
19    db.close();
20  });
21
22
23 });
24
```

10)List out the singers' names in your document.



The screenshot shows the Atom editor interface with the same project. The file explorer shows Ex11.js selected. The main editor pane displays the code in Ex11.js, which connects to the same MongoDB instance and queries the "songdetails" collection for documents where the singer is "Hariharan", projecting only the singer's name.

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 var url = "mongodb://localhost:27017/";
4
5 MongoClient.connect(url, function(err, db) {
6
7   if (err) throw err;
8
9   var dbo = db.db("music");
10
11  dbo.collection("songdetails").find({}, { projection: { _id:0 , singer:1 } }).toArray(function(err, result) {
12
13    if (err) throw err;
14
15    console.log(result);
16
17    db.close();
18  });
19
20
21 });
22
```