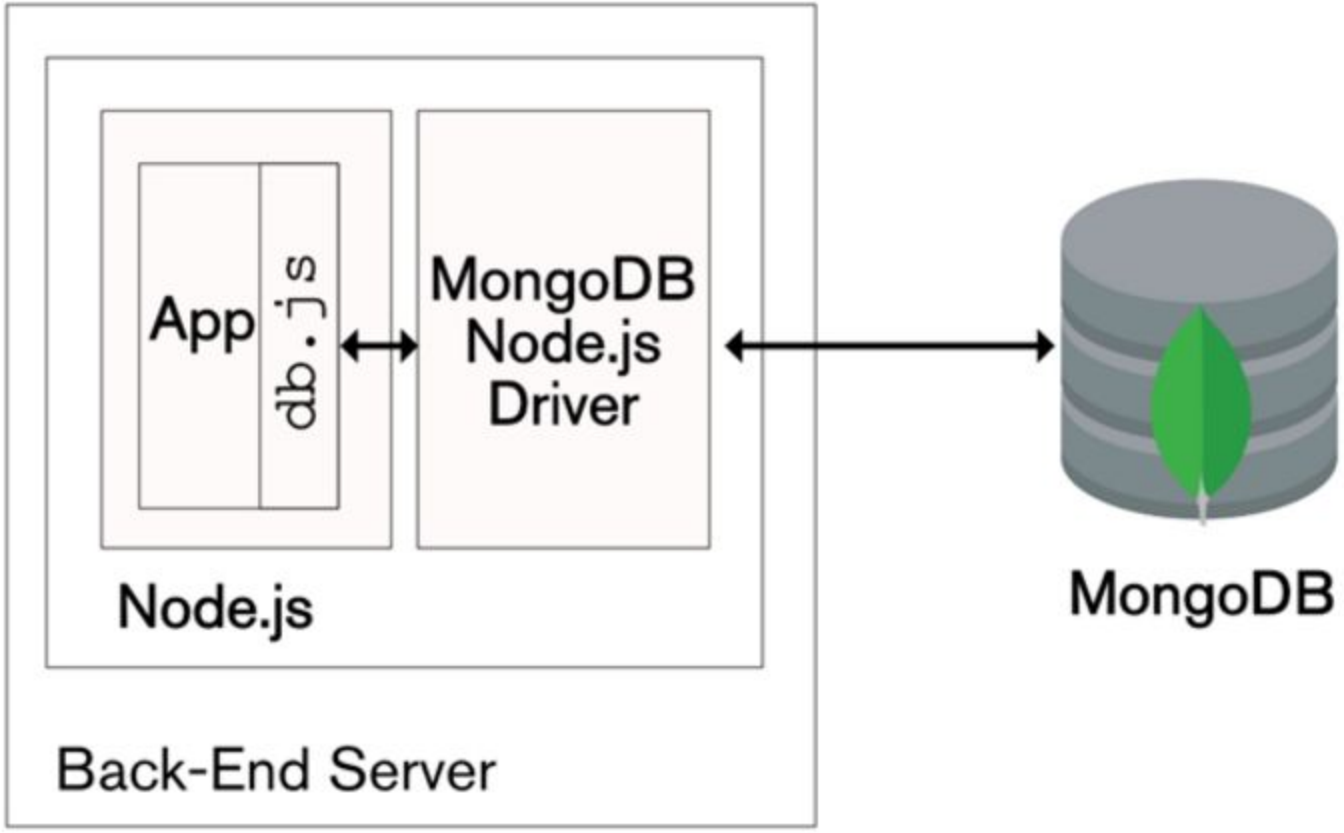
Access MongoDB in Node.js

Introduction to MongoDB

- MongoDB is a No SQL database. It is an open-source, cross-platform, document-oriented database written in C++.
- Mongo DB is developed and supported by a company named 10gen.
- Main purpose to build MongoDB:
  - Scalability
  - Performance
  - High Availability
  - Scaling from single server deployments to large, complex multi-site architectures.

Architecture



Mongodb connection

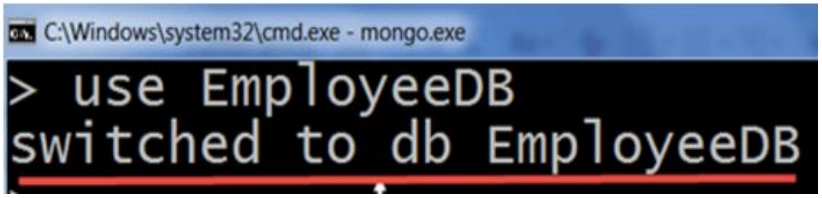
```
var MongoClient = require('mongodb').MongoClient;
// Connect to the db
MongoClient.connect("mongodb://127.0.0.1/mydb", function(err, db) {
  if(!err) {
    console.log("We are connected");
});
```

- The prototype has a single property – db which stores the database connection; it's initialised to null in the constructor.
- The connect() method returns the database reference if the specified database is already exists, otherwise it creates a new database.
- The MongoDB driver is asynchronous (the function returns without waiting for the requested operation to complete)
- The basic interaction model from the application should be:
  - Connect to the database
  - Perform all of the required database actions for the current request
  - Disconnect from the database

Create database

- There is no create database command in MongoDB. Actually, MongoDB do not provide any command to create database.
- Don't need to mention what you want to create, it will be automatically created at the time you save the value into the defined collection.
- If there is no existing database, the following command is used to create a new database.

```
use DATABASE_NAME
```



- To **check the database list**, use the command

```
>show dbs
```

MongoDB Collections

- In MongoDB, db.createCollection(name, options) is used to create collection. But usually you no need to create collection. MongoDB creates collection automatically when you insert some documents.

```
var MongoClient = require('mongodb').MongoClient;
// Connect to the db
MongoClient.connect("mongodb://127.0.0.1/mydb", function(err, db) {
  if(!err) {
    console.log("We are connected");
// create collection 'stud'
    var doc1 = ({usn:"cs-01",name:"bio"});
    db.collection('stud').insert(doc1);
    db.close()
    }
});
```

- To **check the created collection**

```
>show collections
```

- To drop collection

```
> use mydb
switched to db mydb
> show collections
mydb
stud
system.indexes
> db.mydb.drop()
true
> show collections
stud
system.indexes
```

Node.js script to Insert and Display monogDB documents

```
var MongoClient = require('mongodb').MongoClient;

MongoClient.connect('mongodb://127.0.0.1:27017/mydb', function(err, db) {
    if(err) throw err;

    var collection = db.collection('employee');
    collection.insert({empid:551,empname:"civil"}, function(err, docs) {
        collection.count(function(err, count) {
            console.log("count = %s", count);
        });
    });

    // Locate all the entries using find
    collection.find().toArray(function(err, results) {
        console.dir(results);
        // Let's close the db
        db.close();
    });
});
```

Node.js script to Insert and Count monogDB documents

```
db.collection('employee', function (err, collection) {
    collection.insert({ empid: 1, empname: 'Steve' });
    collection.insert({ empid: 2, empname: 'Bill' });
    collection.insert({ empid: 3, empname: 'James' });

    db.collection('employee').count(function (err, count) {
        if (err) throw err;

        console.log('Total Rows: ' + count);
    });
  });
```

- In the above example, db.collection() method creates or gets the reference of the specified collection. Collection is similar to table in relational database. We created a collection called **employee** in the above example and insert three documents (rows) in it. After that, we display the count of total documents stored in the collection.

Node.js script to Update/Delete monogDB documents

```
// To Update a Single Document
db.collection('employee').updateOne({"empname":"newemp"},{$set:{"empname":"oldemp"}});
// To Update a multiple Document
//db.collection('employee').update({"empname":"ele"},{$set:{"empname":"newemp"}});
db.collection('employee').deleteOne({"empname":"xxx"});
db.collection('employee').remove({empname:'ele'})
// counting number of records
```

Node.js script to Search monogDB documents

```
var myEmployee = db.collection('employee').find( { empid : { $gt:4 }});
//db.employee.find({empid:{$gt:4}});  mongodb shell command
console.log("greater than 4");
myEmployee.each(function(err,doc)
{
    console.log(doc)
});

var myEmp = db.collection('employee').find( { empname: 'oldemp'});
console.log("old employees");
myEmp.each(function(err,doc)
{
    console.log(doc)
});
```

Building a Simple CRUD web Application with Express and MongoDB

index.html

```html
<html>
<body>
<form action="process_get" method="get">
Employee ID: <input type="text" name="empid">  <br>
Employee Name: <input type="text" name="empname">
<input type="submit" value="Submit">
</form>
<a href="/about">about</a>
<a href="/">welcome</a>
<a href="/display">display</a>
<a href="search.html">search</a>
</body>
</html>
```

Employee ID: cse-01
Employee Name: HODCSE Submit

about welcome display search

Empoyee ID: cse-01

Employee name: HODCSE

Display using Embedded js

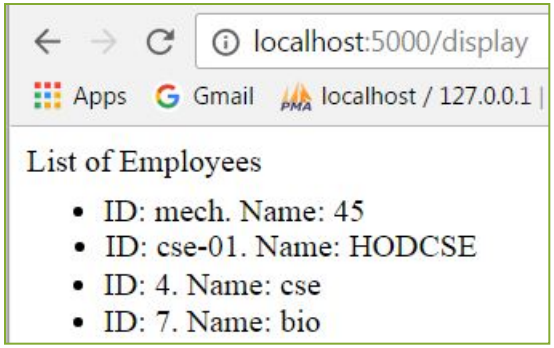
```
List of Employees
  <ul class="employees">
  <% for(var i=0; i<employees.length; i++) {%>
    <li class="employees">
      <span><%= " ID: " +employees[i].empid+"."%></span>
      <span><%=" Name: " + employees[i].empname%></span>
    </li>
  <% } %>
</ul>
```

Create 'disp.ejs' in 'views ' folder

disp.ejs

```
//------------------------
 db.collection('employee').find().toArray(function(err , i){
        if (err) return console.log(err)

        res.render('disp.ejs',{employees: i})
     })
//---------------------
```



Display in JSON format

```
//-----DISPLAY IN JSON FORMAT  -------------------------
db.collection('employee').find({}).toArray(function(err, docs) {
    if (err) {
      console.log("Failed to get data.");
    } else
    {
        res.status(200).json(docs);
    }
  });
```

localhost:5000/display

Apps Gmail localhost / 127.0.0.1 | msritcseonline Help Desk

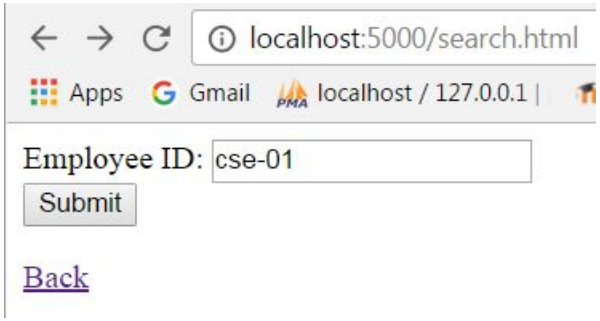

```
[{"_id":"58e4c81e6ad37813fc8e395c","empid":"hinew","empname":"new"},
{"_id":"58e4c84f6ad37813fc8e395d","empid":"mech","empname":"45"},
{"_id":"58e5cf3262f79c041428ed0e","hello":"doc1"},
{"_id":"58e75cfddb543b154c749b8b","empid":"cse-04","empname":"name4"}]
```

Search module

search.html

```html
<html>
<body>
<form action="search" method="get">
 Employee ID: <input type="text" name="empid">  <br>
 <input type="submit" value="Submit">
</form>
 <a href="/index.html">Back</a>

</body>
</html>
```



Search and display the result in JSON format

[{"_id":"58e756de466f5c136c99c2f4","empid":"cse-01","empname":"HODCSE"}]