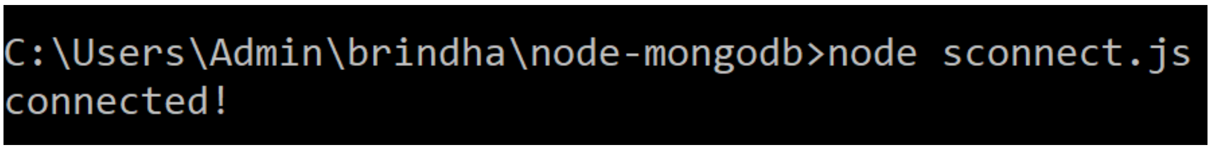
Mongo dB connection through mongoose library

```
const mongoose=require('mongoose');

mongoose.set('strictQuery', true);

 mongoose.connect('mongodb://127.0.0.1:27017/database',
{useNewUrlParser: true, useUnifiedTopology: true});

 const db=mongoose.connection;

 db.on('error', function (err) { throw err });

 db.once('open', function callback() {

    console.log('connected!');

    db.close();

});
```

Output

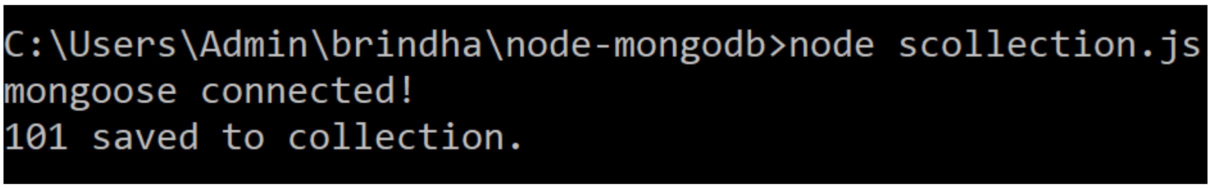


```
C:\Users\Admin\brindha\node-mongodb>node sconnect.js
connected!
```

Mongo dB connection through mongoose library- Creation of slotd1 creation

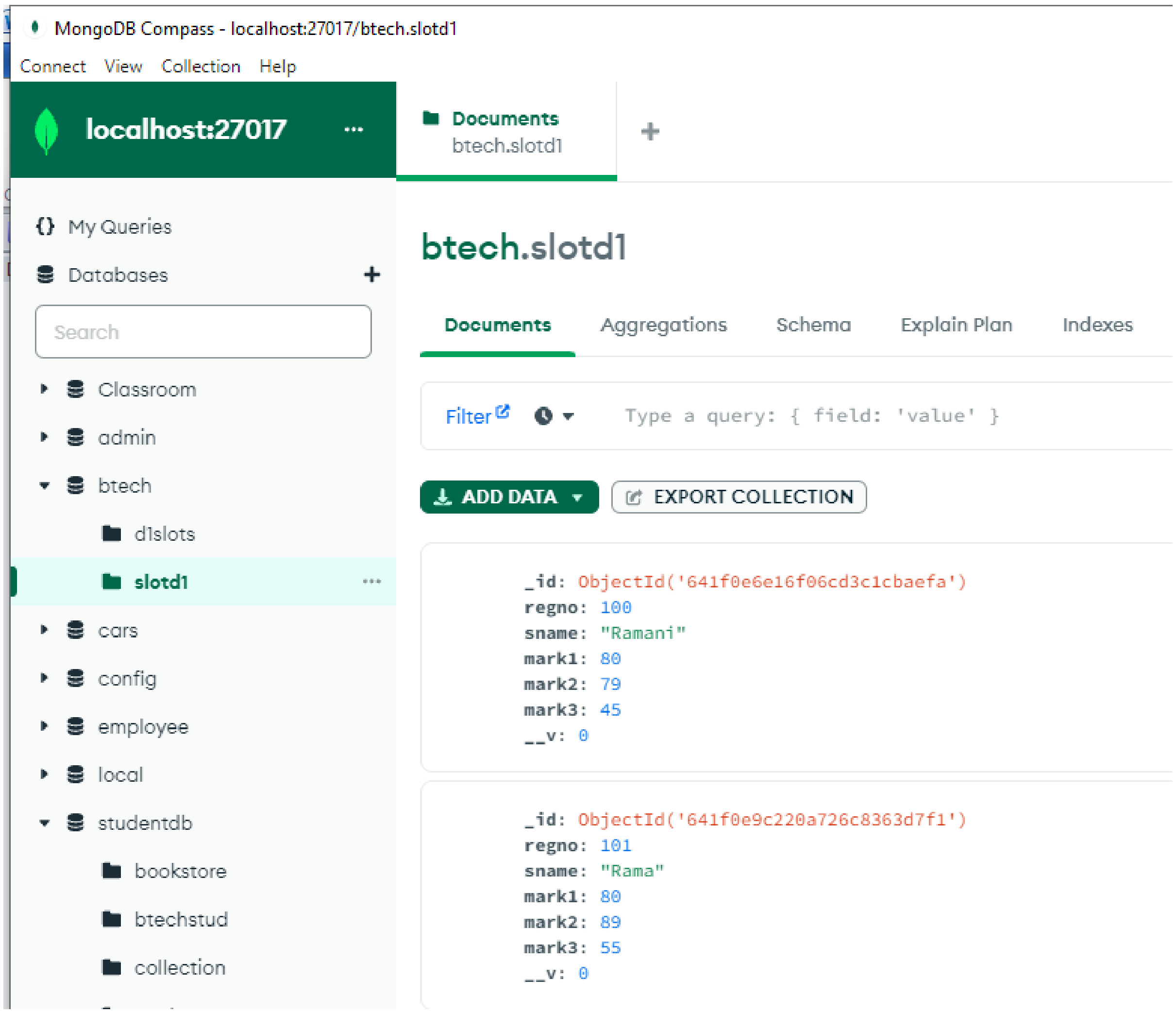
```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
const Schema=mongoose.Schema;
   const slot=new Schema({
    _id:mongoose.ObjectId,
    regno:{ type: Number, required: true },
    sname:{ type: String, required: true },
    mark1:{ type: Number, required: true },
    mark2: { type: Number, required: true },
         mark3:{ type: Number, required: true },
},{collection:'slotd1'});
var slotd1=mongoose.model("slotd1",slot);
var sd1=new slotd1({
  _id:new mongoose.Types.ObjectId(),
  regno:101,
  sname:"Rama",
  mark1:80,
  mark2:89,
  mark3:55
});
db.on('error', function (err) { throw err });
db.once('open', function() {
   console.log('mongoose connected!');
    sd1.save(function (err, data) {
     if (err){
         console.log(err);
         db.close();
        }
     else{
         console.log(data.regno + " saved to collection.");
         db.close();
        }
   });
  });
```

OUTPUT



```
C:\Users\Admin\brindha\node-mongodb>node scollection.js
mongoose connected!
101 saved to collection.
```

Check the output either through shell – mongosh or through compass

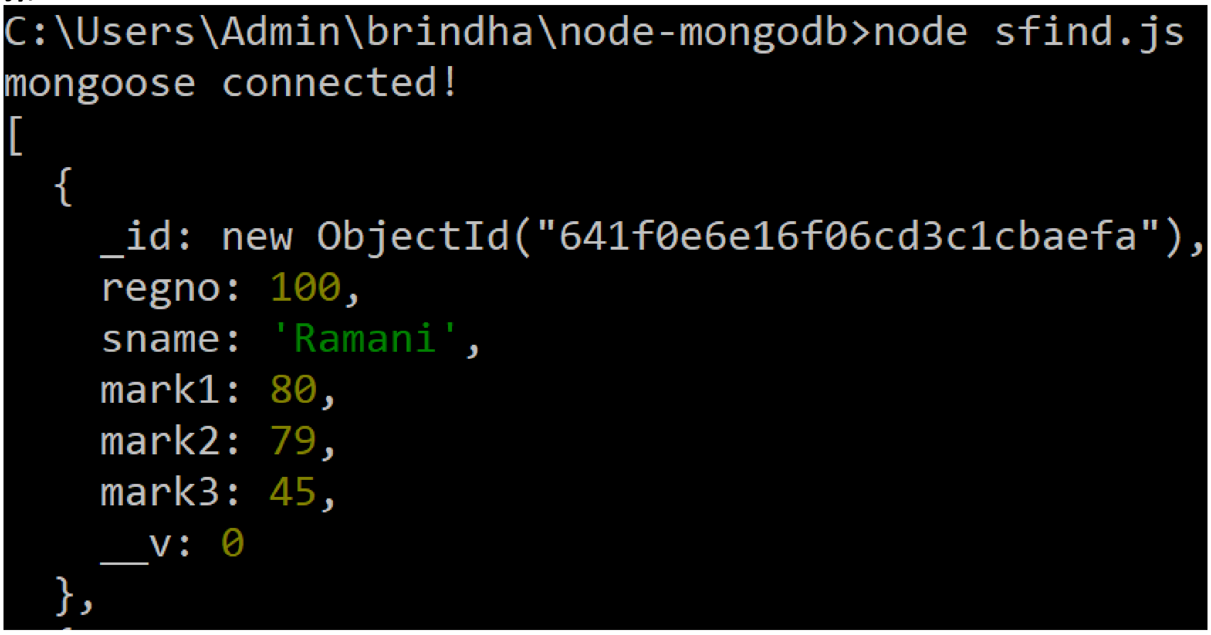


```
_id: ObjectId('641f0e6e16f06cd3c1cbaefa')
regno: 100
sname: "Ramani"
mark1: 80
mark2: 79
mark3: 45
__v: 0
```

```
_id: ObjectId('641f0e9c220a726c8363d7f1')
regno: 101
sname: "Rama"
mark1: 80
mark2: 89
mark3: 55
__v: 0
```

Mongo dB connection through mongoose library- find the document in slotd1 collection

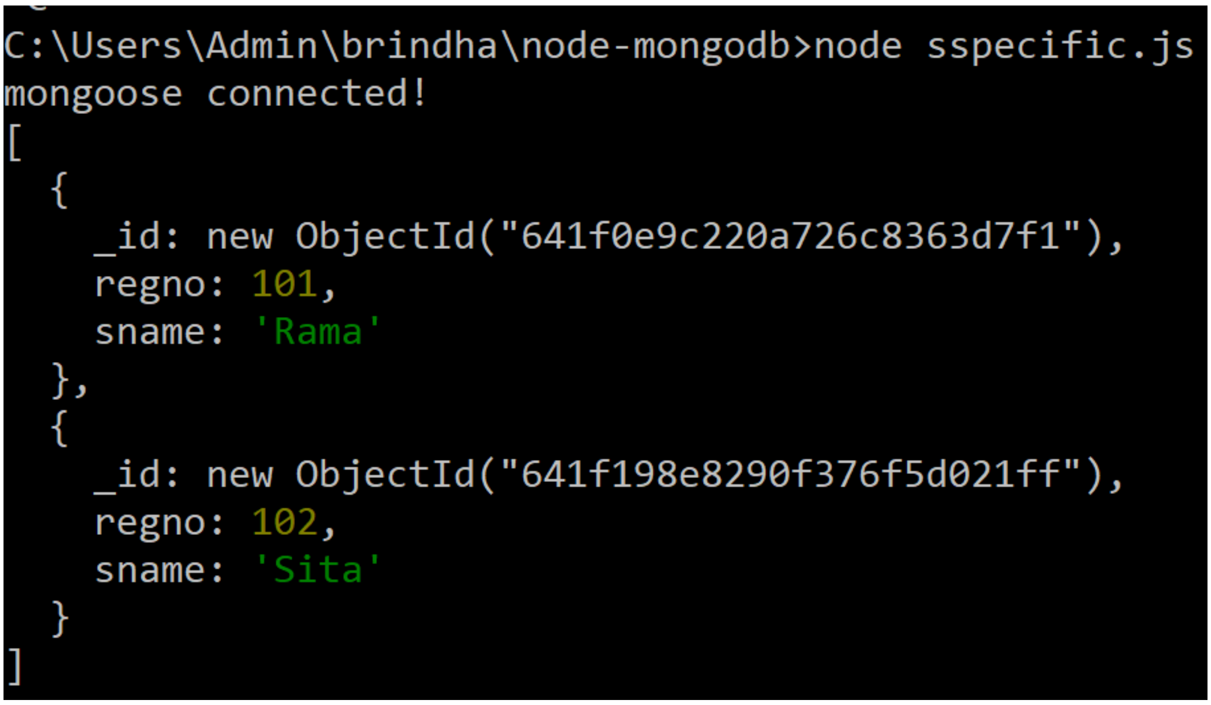
```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
db.on('error', function (err) { throw err });
db.once('open', function() {
  console.log('mongoose connected!');
  const Schema=mongoose.Schema;
  const slot=new Schema({
   _id:mongoose.ObjectId,
     regno:{ type: Number, required: true },
   sname:{ type: String, required: true },
   mark1:{ type: Number, required: true },
   mark2: { type: Number, required: true },
         mark3:{ type: Number, required: true },
},{collection:'slotd1'});
var slotd1=mongoose.model("slotd1",slot);
slotd1.find({},(err,data)=>{
   if(err){console.log(err)}
   else{ console.log(data)}
});

});
```



```
C:\Users\Admin\brindha\node-mongodb>node sfind.js
mongoose connected!
[
  {
    _id: new ObjectId("641f0e6e16f06cd3c1cbaefa"),
    regno: 100,
    sname: 'Ramani',
    mark1: 80,
    mark2: 79,
    mark3: 45,
    __v: 0
  },
```

Mongo DB – Connection and find the specific document in collection

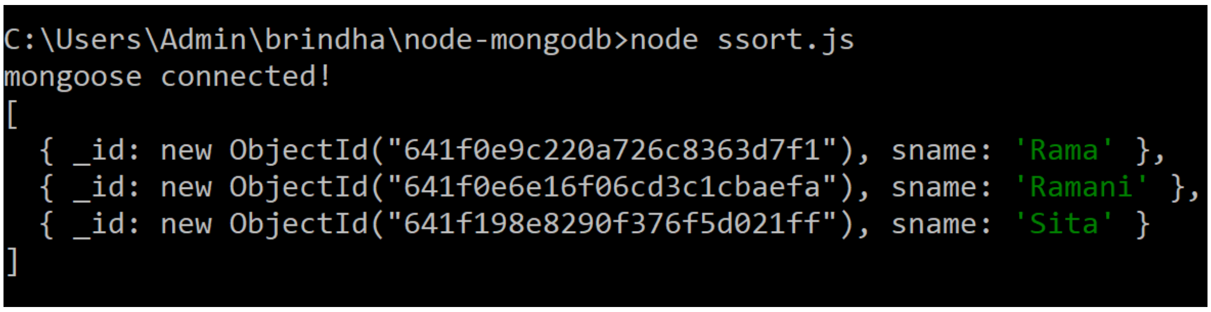
```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
db.on('error', function (err) { throw err });
db.once('open', function() {
  console.log('mongoose connected!');
  const Schema=mongoose.Schema;
  const slot=new Schema({
   _id:mongoose.ObjectId,
      regno:{ type: Number, required: true },
   sname:{ type: String, required: true },
   mark1:{ type: Number, required: true },
   mark2: { type: Number, required: true },
        mark3:{ type: Number, required: true },
},{collection:'slotd1'});
var slotd1=mongoose.model("slotd1",slot);
slotd1.find({mark2:89},'regno sname',(err,data)=>{
   if(err){console.log(err)}
   else{ console.log(data)}
});
 });
```



```
C:\Users\Admin\brindha\node-mongodb>node sspecific.js
mongoose connected!
[
  {
    _id: new ObjectId("641f0e9c220a726c8363d7f1"),
    regno: 101,
    sname: 'Rama'
  },
  {
    _id: new ObjectId("641f198e8290f376f5d021ff"),
    regno: 102,
    sname: 'Sita'
  }
]
```

Mongo DB – Connection and sort the documents in collection

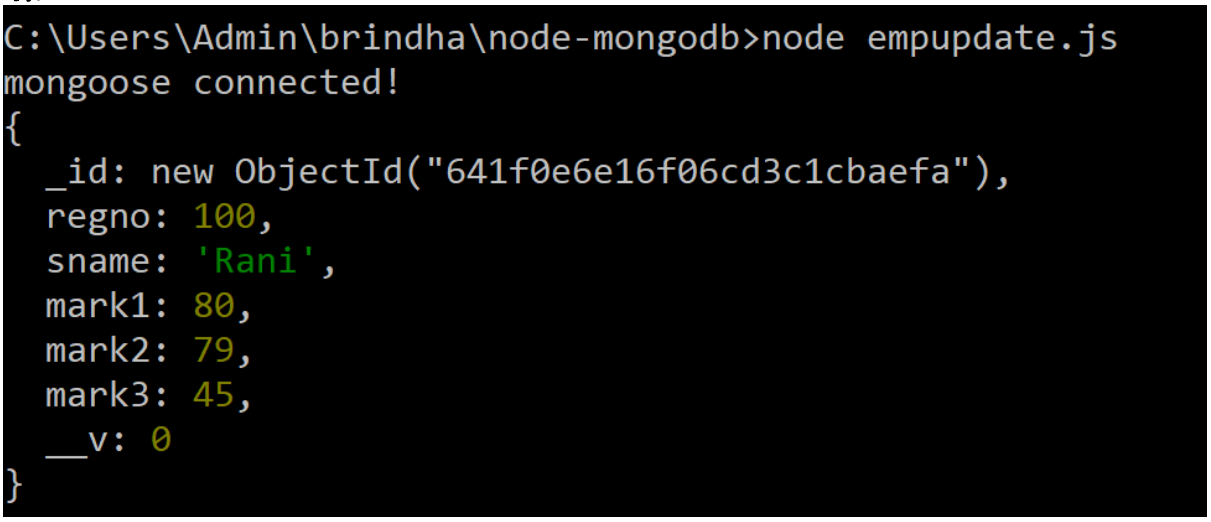
```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
db.on('error', function (err) { throw err });
db.once('open', function() {
  console.log('mongoose connected!');
  const Schema=mongoose.Schema;
  const slot=new Schema({
   _id:mongoose.ObjectId,
   regno:{ type: Number, required: true },
   sname:{ type: String, required: true },
   mark1:{ type: Number, required: true },
   mark2: { type: Number, required: true },
        mark3:{ type: Number, required: true },
},{collection:'slotd1'});

var slotd1=mongoose.model("slotd1",slot);

slotd1.find({},'sname',(err,data)=>{
   if(err){console.log(err)}
   else{ console.log(data)}
}).sort({sname:1});

});
```



```
C:\Users\Admin\brindha\node-mongodb>node ssort.js
mongoose connected!
[
  { _id: new ObjectId("641f0e9c220a726c8363d7f1"), sname: 'Rama' },
  { _id: new ObjectId("641f0e6e16f06cd3c1cbaefa"), sname: 'Ramani' },
  { _id: new ObjectId("641f198e8290f376f5d021ff"), sname: 'Sita' }
]
```

Mongo DB – Connection and update the column in collection

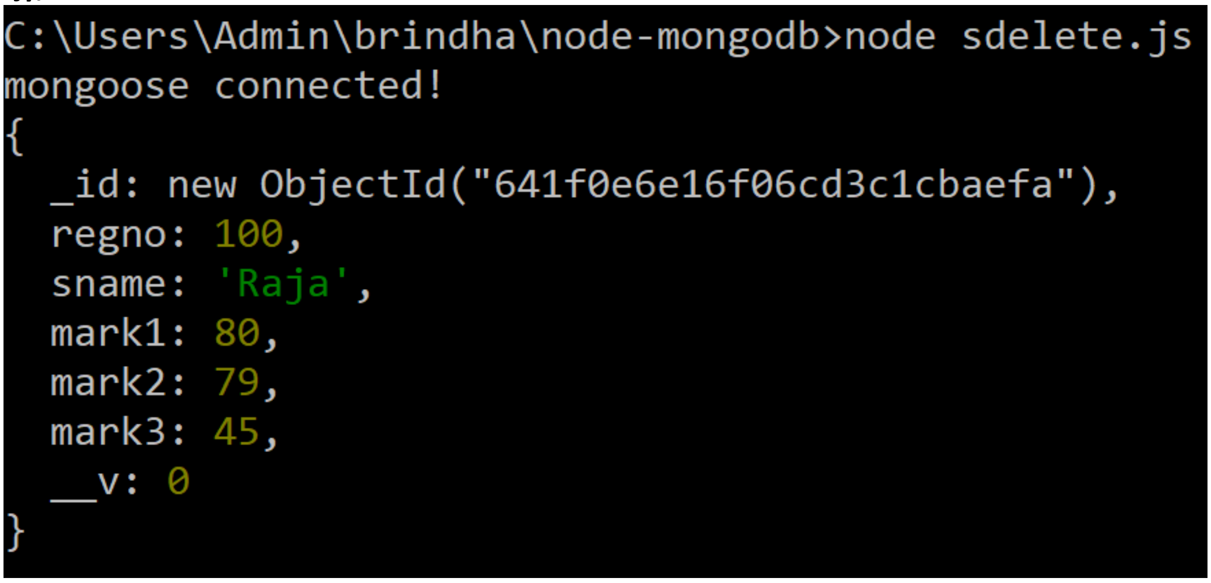
```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
db.on('error', function (err) { throw err });
db.once('open', function() {
  console.log('mongoose connected!');
  const Schema=mongoose.Schema;
  const slot=new Schema({
   _id:mongoose.ObjectId,
   regno:{ type: Number, required: true },
   sname:{ type: String, required: true },
   mark1:{ type: Number, required: true },
   mark2: { type: Number, required: true },
        mark3:{ type: Number, required: true },
},{collection:'slotd1'});
var slotd1=mongoose.model("slotd1",slot);
const query = { sname:'Rani' };
slotd1.findOneAndUpdate(query, { $set: { sname:'Raja' }},(err,data)=>{
   if(err){console.log(err)}
   else{
        console.log(data);}
});
 });
```



```
C:\Users\Admin\brindha\node-mongodb>node empupdate.js
mongoose connected!
{
  _id: new ObjectId("641f0e6e16f06cd3c1cbaefa"),
  regno: 100,
  sname: 'Rani',
  mark1: 80,
  mark2: 79,
  mark3: 45,
  __v: 0
}
```

Mongo DB – Connection and Delete the document in collection

```
const mongoose=require('mongoose');
mongoose.set('strictQuery', true);
mongoose.connect('mongodb://127.0.0.1:27017/btech', {useNewUrlParser:
true, useUnifiedTopology: true});
const db=mongoose.connection;
db.on('error', function (err) { throw err });
db.once('open', function() {
  console.log('mongoose connected!');
  const Schema=mongoose.Schema;
  const slot=new Schema({
   _id:mongoose.ObjectId,
   regno:{ type: Number, required: true },
   sname:{ type: String, required: true },
   mark1:{ type: Number, required: true },
   mark2: { type: Number, required: true },
        mark3:{ type: Number, required: true },
},{collection:'slotd1'});
var slotd1=mongoose.model("slotd1",slot);
const query = { sname:'Raja' };
slotd1.findOneAndDelete(query, (err,docs)=>{
   if(err){console.log(err)}
   else{
         console.log(docs);}
});
 });
```



```
C:\Users\Admin\brindha\node-mongodb>node sdelete.js
mongoose connected!
{
  _id: new ObjectId("641f0e6e16f06cd3c1cbaefa"),
  regno: 100,
  sname: 'Raja',
  mark1: 80,
  mark2: 79,
  mark3: 45,
  __v: 0
}
```

To Build a REST API with Express and Mongoose

Install the following packages
Create a folder which you want to store application.
npm install express
npm install ejs
npm install mongoose
npm install body-parser

Procedure to insert data into mongoDB using mongoose and node js

Step 1 – Create Node Express js App
Step 2 – Install express ejs body-parser mongoose dependencies
Step 3 – Connect App to MongoDB
Step 4 – Create Model
Step 5 – Create Route
Step 6 – Create HTML Form
Step 7 – Import Modules in App.js
Step 8 – Start App Server

Application package

To get the required module

```
const express = require("express");
const app = express();
const mongoose=require("mongoose");
mongoose.set('strictQuery', true);
const bodyParser = require("body-parser");
const ejs=require("ejs");
// defining schema
var Schema = mongoose.Schema;
// connection url and mongodb connection
var url="mongodb://127.0.0.1:27017/btech";
mongoose.connect(url, {useNewUrlParser:true});
app.set("view engine", "ejs");
app.use(bodyParser.urlencoded({extended:true}));
// Schema creation
var mySchema = new mongoose.Schema({
regno:Number,
sname :String,
mark1:Number,
mark2:Number,
mark3:Number});
// get the data through the form
var d1slot=mongoose.model("d1slot", mySchema);
app.get('/', (req, res)=> {
        res.render('index');
});
app.post('/',  (req, res)=>{
var info={
regno:req.body.regno,
sname:req.body.sname,
mark1:req.body.mark1,
mark2:req.body.mark2,
mark3:req.body.mark3
};
```

Saving the form data in collection

```
var me = new d1slot (info);
me.save(function (err) {
if (err) {
```

```
console.log("error occured");
}
else
{
  console.log("done");
}
});
res.send("Done!")
});
app.listen(8000, ()=> console.log("listening on 8000"));
```

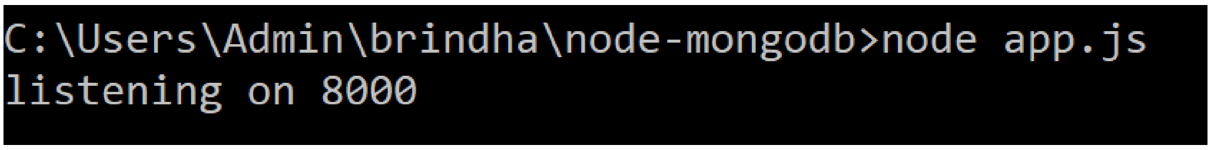
Save this file as app.js

```
<html>
<head> <title> mongoose </title></head>
<body>
<form action ="/" method="post">
<div>
<label> Regno: </label>
<input type="number" name="regno" placeholder="Enter register no"/>
</div>
<div>
<label>Student Name:</label>
<input type="text" name="sname" placeholder="Enter Student Name"/>
</div> <br/>
<div>
<label>DBMS:</label>
<input type="number" name="mark1" placeholder="Enter DBMS mark"/>
</div> <br/>
<div>
<label>Web Tech:</label>
<input type="number" name="mark2" placeholder="Enter Web Tech
mark"/>
</div> <br/>
<div>
<label>Software Engineering:</label>
<input type="number" name="mark3" placeholder="Enter SE mark"/>
</div> <br/>
<div>
<input type="submit" value="submit"/>
</div> <br/>
```

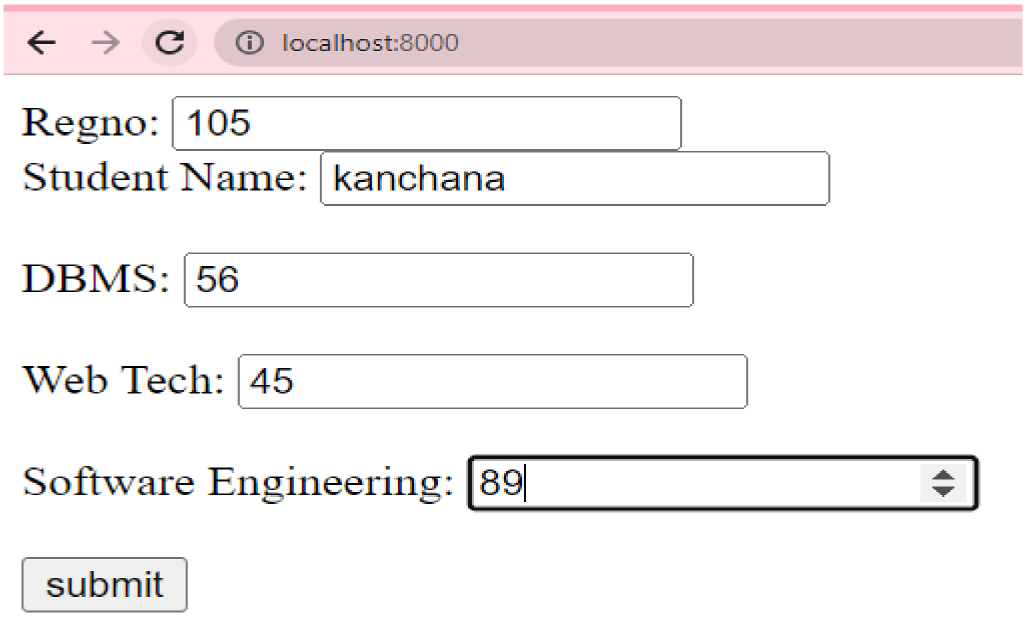
```
</form>
</body>
</html>
```

Save this file index.ejs in views folder

Run this application in console
node app.js

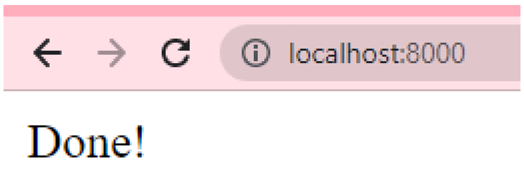


```
C:\Users\Admin\brindha\node-mongodb>node app.js
listening on 8000
```



localhost:8000

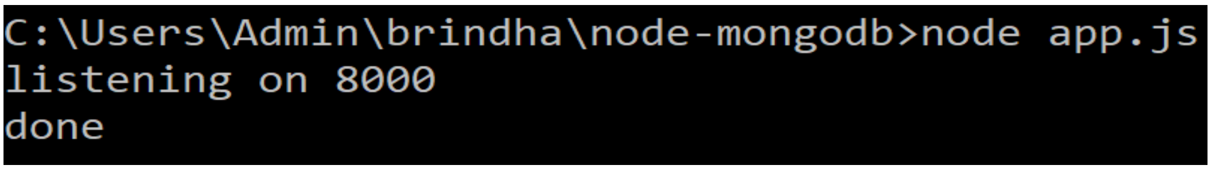
Regno: 105
Student Name: kanchana
DBMS: 56
Web Tech: 45
Software Engineering: 89



localhost:8000

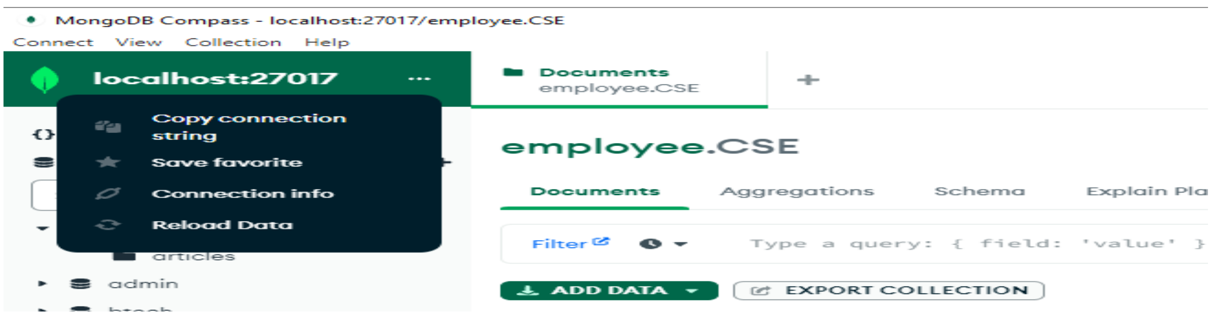
Done!

Check it with console



```
C:\Users\Admin\brindha\node-mongodb>node app.js
listening on 8000
done
```

Check whether form is updated or not using compass… Click reload data in compass



Node.js Fetch data from MongoDB Using Mongoose (without any constraint)

Step 1 – Create Node Express js App
Step 2 – Install express ejs body-parser mongoose dependencies
Step 3 – Connect App to MongoDB
Step 4 – Create Model
Step 5 – Create Routes
Step 6 – Create HTML Table and Display List
Step 7 – Import Modules in App.js
Step 8 – Start App Server

```
// Accessing required modules
var express = require("express"),
app = express(),
bodyparser = require("body-parser"),
mongoose = require("mongoose");
// connecting mongodb
mongoose.set('strictQuery', true);
mongoose.connect("mongodb://127.0.0.1:27017/btech",
{useNewUrlParser: true});
app.use(bodyparser.urlencoded({ extended: true }));
app.set("view engine", "ejs");
var mySchema = new mongoose.Schema({
regno:Number,
sname :String,
mark1:Number,
mark2:Number,
mark3:Number});
var d1slot=mongoose.model("d1slot", mySchema);
app.get("/", function (req, res) {
res.render("index",{ details: null })
})
app.get("/getdetails", function (req, res) {

d1slot.find({}, function (err, allDetails) {
   if (err) {
      console.log(err);
   } else {
      res.render("index", { details: allDetails })
   }
})
})
app.listen(8000, "localhost", function () {
console.log("server has started");
})
```

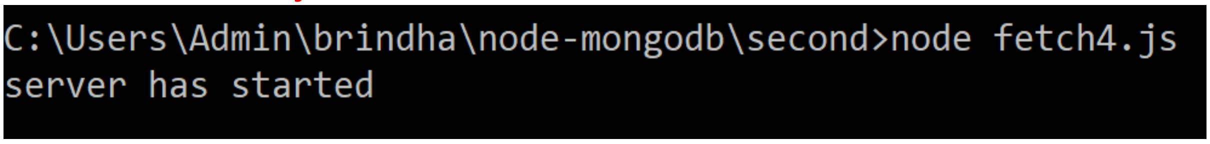
Save this file fetch4.js

```
<div>
<a href="/getdetails">Get Details</a>
</div>
<hr>
<% if(details!=null) { %>
<table border="4px" bordercolor="red">
<tr>
   <th>username</th>
   <th>password </th>
    <th>DBMS</th>
    <th>WebTech</th>
    <th>SE</th>
   </tr>
<% details.forEach(function(item){ %>
<tr>
   <td><%= item.regno%></td>
   <td><%= item.sname %></td>
    <td><%= item.mark1%></td>
   <td><%= item.mark2 %></td>
    <td><%= item.mark3%></td>
```

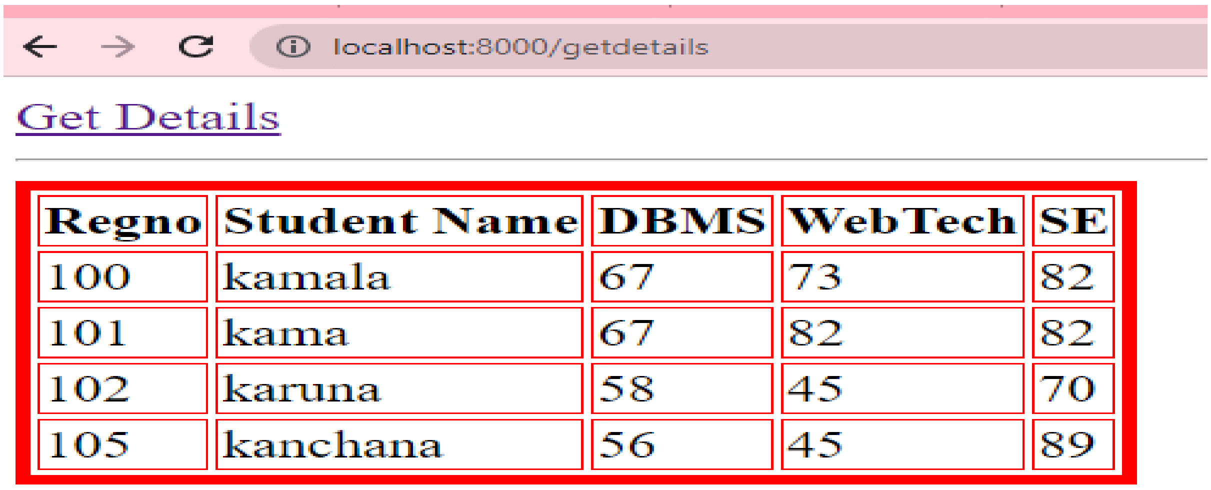
```
        </tr>
    <% }) %>
    </table>
    <% } %>
```

Save this file as index.ejs in view folder

- Run the application
- node fetch4.js



```
C:\Users\Admin\brindha\node-mongodb\second>node fetch4.js
server has started
```



Get Details

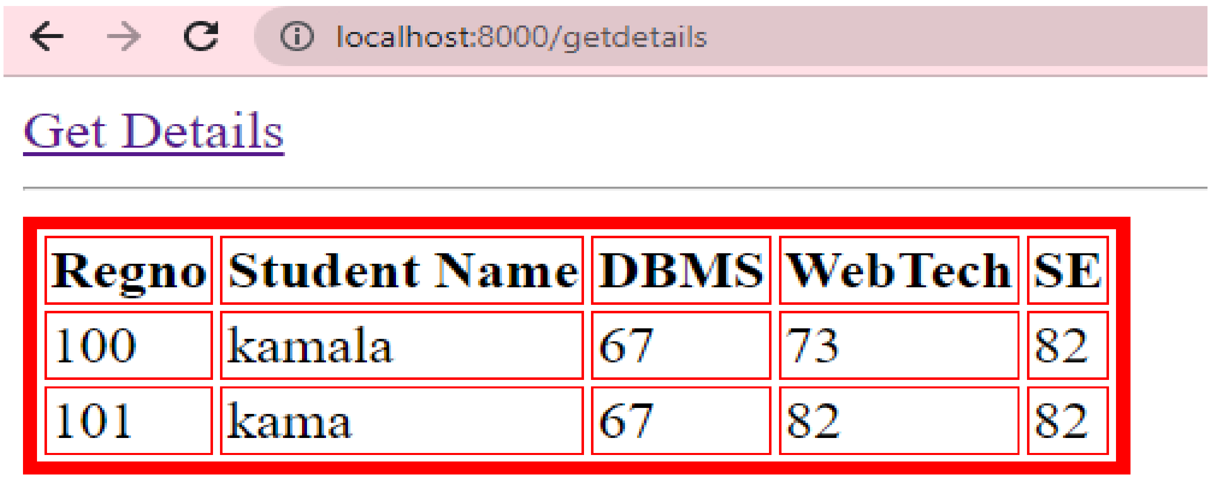
Regno	Student Name	DBMS	WebTech	SE
100	kamala	67	73	82
101	kama	67	82	82
102	karuna	58	45	70
105	kanchana	56	45	89

Node.js Fetch data from MongoDB Using Mongoose (based on constraint)

- Same code as in fetch4.js, except the find query
- Retrieve all the student details whose webtech mark greater than 45

```
d1slot.find({mark2: { $gt: 45 } }, function (err, allDetails) {
  if (err) {
     console.log(err);
  } else {
     res.render("index", { details: allDetails })
  }
})
```

- // update the above lines in fetch4.js


localhost:8000/getdetails

Get Details

Regno	Student Name	DBMS	WebTech	SE
100	kamala	67	73	82
101	kama	67	82	82