

## Práctica Grupal: Servicios REST con NodeJS para interactuar con MongoDB

1. Iniciar MongoDB con el comando:

```
$ mongod
```

2. Conectarse a MongoDB por medio de una ventana de comandos, con el comando:

```
$ mongo
```

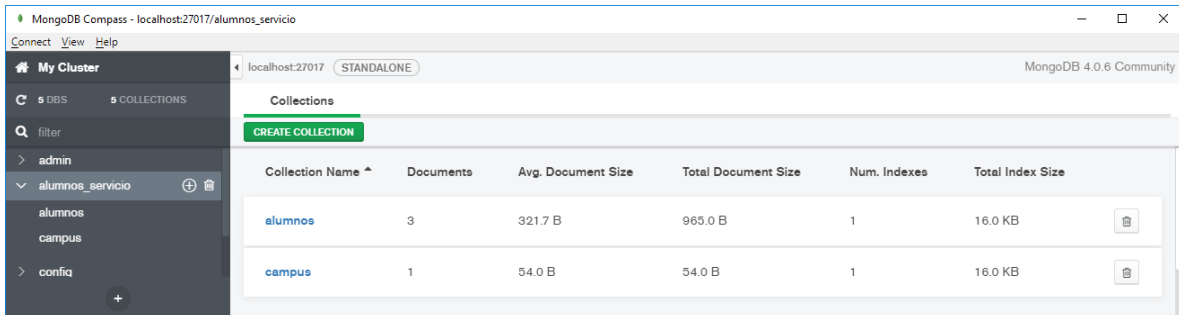
3. Crear la base de datos llamada “alumnos\_servicio” con el comando:

```
$ use alumnos_servicio;
```

4. Cargar el script proporcionado en clase con el comando:

```
$ load("[Directorio]/scriptalumnos.js");
```

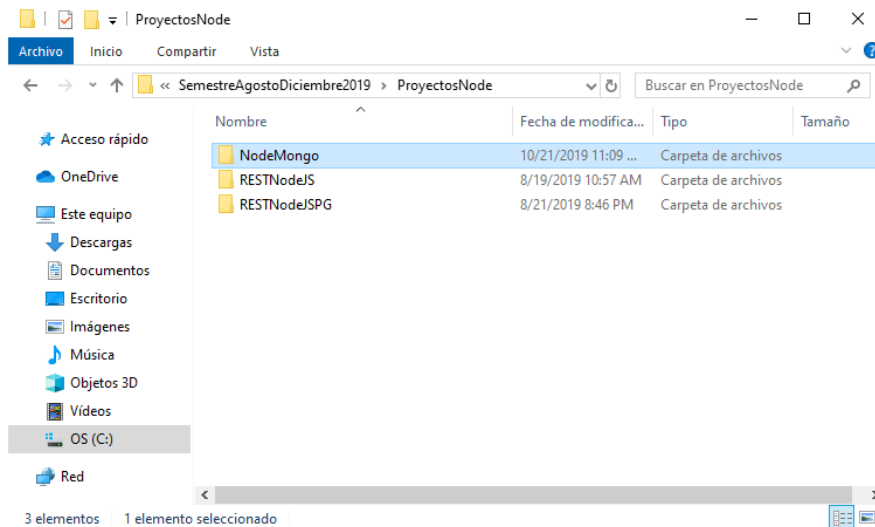
5. Validar la base de datos con Compass:



The screenshot shows the MongoDB Compass interface. The left sidebar displays the 'My Cluster' structure with 'alumnos\_servicio' selected. The main panel shows the 'Collections' tab with a table of collections:

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size
alumnos	3	321.7 B	965.0 B	1	16.0 KB
campus	1	54.0 B	54.0 B	1	16.0 KB

6. Crear una carpeta para el proyecto de NodeJS.



7. Abrir una ventana de comandos y ubicarse en la carpeta generada. Ejecutar el siguiente comando y especificar los datos que solicite:

```
$ npm init
```

8. Instalar express con el siguiente comando:

```
$ npm install express --save
```

9. Instalar body-parser con el siguiente comando:

```
$ npm install body-parser --save
```

10. Instalar el driver de MongoDB con el siguiente comando:

```
$ npm install mongodb --save
```

11. Crear un archivo llamado "studentCollectionRoutes.js" y colocar el siguiente contenido:

```
'use strict';

module.exports = function(app) {
  var studentCollection = require('./studentCollectionController');

  app.route('/students')
    .get(studentCollection.obtener_estudiantes)
```

```
.post(studentCollection.agregar_estudiante);

app.route('/students/:matricula')
  .get(studentCollection.obtener_estudiante);
};
```

12. Crear un archivo llamado “studentCollectionController.js” y colocar el siguiente contenido:

```
'use strict';

const MongoClient = require('mongodb').MongoClient;
const url = "mongodb://localhost:27017";
const dbName = 'alumnos_servicio';

exports.obtener_estudiantes = function(req, res) {
  MongoClient.connect(url, { useNewUrlParser: true }, function(err, mdbclient) {
    if (err){
      throw err;
    }

    const db = mdbclient.db(dbName);

    //Solamente obtenemos el nombre y la matricula
    db.collection("alumnos").find({}).project({nombre:1, matricula:1}).toArray(function(err, result) {
      if (err){
        throw err;
      }
      console.log("Resultados Obtenidos: " + result.length);
      mdbclient.close();
      res.end( JSON.stringify(result));
    });
  });
};

exports.obtener_estudiante = function(req, res) {
  MongoClient.connect(url, { useNewUrlParser: true }, function(err, mdbclient) {
    if (err){
      throw err;
    }

    const db = mdbclient.db(dbName);

    var matriculaAlumno = req.params.matricula;

    db.collection("alumnos").findOne({matricula:matriculaAlumno}, function(err, result) {
      if (err){
        throw err;
      }
    }
  });
};
```

```
        console.log("Consulta ejecutada...");
        mdbclient.close();
        res.end( JSON.stringify(result));
    });
};

exports.agregar_estudiante = function(req, res) {
    MongoClient.connect(url, { useNewUrlParser: true }, function(err, mdbclient) {
        if (err){
            throw err;
        }

        const db = mdbclient.db(dbName);

        var nuevoAlumno = req.body;

        db.collection("alumnos").insertOne(nuevoAlumno, function(err, res) {
            if (err){
                throw err;
            }
            console.log("Insert ejecutado...");
            mdbclient.close();
        });

        res.end();
    });
};
```

13. Crear un archivo llamado "server.js" y colocar el siguiente contenido:

```
var express = require('express'),
    app = express(),
    port = process.env.PORT || 8585,
    bodyParser = require('body-parser');

app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

var routes = require('./studentCollectionRoutes');
routes(app);

app.listen(port);

console.log('Servidor escuchando en puerto: ' + port);
```

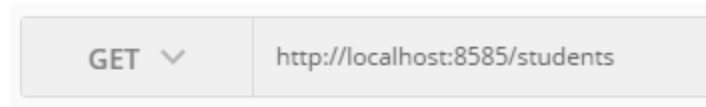
14. Ejecutar server.js con el siguiente commando:

\$ node server.js

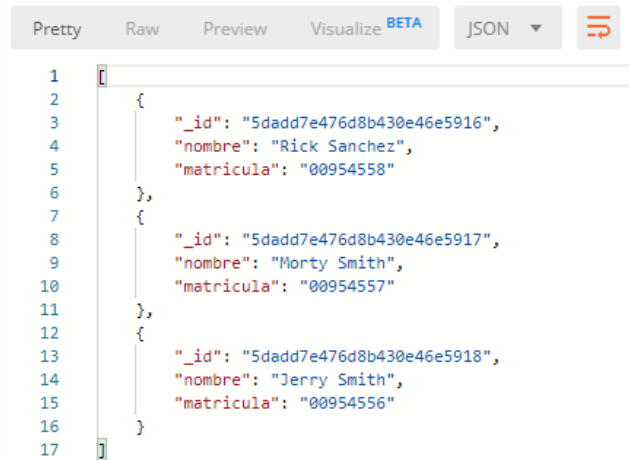
15. Utilice Postman para probar sus servicios:

a. GET Students:

i. Elija el método GET y utilice la URL <http://localhost:8585/students>:

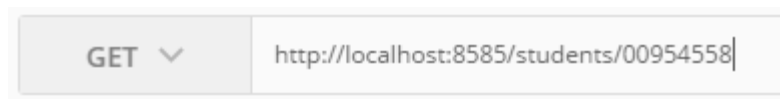


ii. Haga clic en SEND. Postman debe mostrar todos los registros de la colección "alumno". Valide la información mostrada utilizando Compass:



b. GET student por matrícula:

i. Elija el método GET y utilice la URL <http://localhost:8585/students/00954558>:



ii. Haga clic en SEND. Postman debe mostrar el registro que corresponda a esa matrícula de "alumno". Valide la información mostrada utilizando Compass:

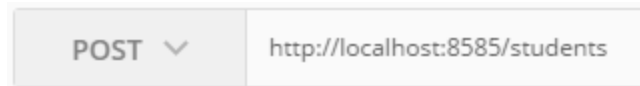
```

1  {
2    "_id": "5dadd7e476d8b430e46e5918",
3    "campus": "CEM",
4    "nombre": "Jerry Smith",
5    "matricula": "00954556",
6    "materias": [
7      {
8        "_id": "TC2025",
9        "nombre": "Programacion Avanzada"
10     }
11   ],
12   "foto": "https://pbs.twimg.com/profile_images/738078769920020481/xplW4r-Tr_400x400.jpg"
13 }

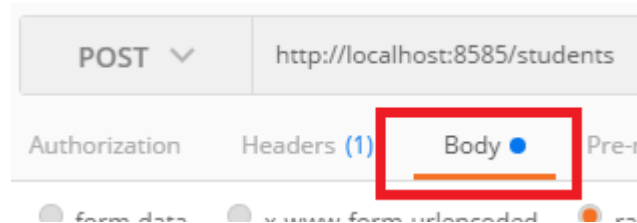
```

c. POST student:

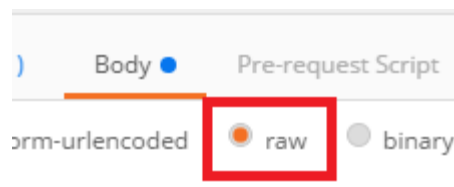
i. Elija el método POST y utilice la URL <http://localhost:8585/students>.



ii. Elija la pestaña "Body":



iii. Elija la opción "raw":



iv. Elija la opción "JSON (application/json)" de la lista:



v. Introduzca el siguiente objeto de JSON en el área de texto:

```
{ "campus": "CEM", "nombre": "Neo", "matricula": "00101010", "materias": [{ "_id": "TC3052", "nombre": "Laboratorio de Desarrollo de Aplicaciones Web" }, { "_id": "TC2026", "nombre": "Desarrollo de Aplicaciones Web" }, { "_id": "TC2025", "nombre": "Programacion Avanzada" } ] }
```

- d. Haga clic en SEND. Valide el alumno insertado en la colección “alumno” utilizando Compass:

```
> {
  _id: ObjectId("59f80193b02eff14b00c71a0")
  campus: "CEM"
  nombre: "Neo"
  matricula: "00101010"
  > materias: Array
```

- e. GET student por palabra clave:

- i. Generar un nuevo route en el archivo studentCollectionRoutes.js:

```
app.route('/students/busqueda/:palabraClave')
  .get(studentCollection.buscar_palabra_clave);
```

- ii. Implementar la función “buscar\_palabra\_clave” en el archivo studentCollectionController.js:

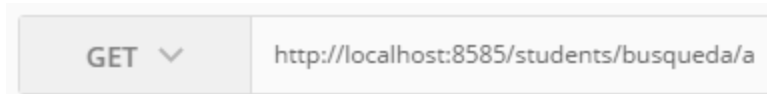
```
exports.buscar_palabra_clave = function(req, res) {
  MongoClient.connect(url, { useNewUrlParser: true }, function(err, mongodbclient) {
    if (err){
      throw err;
    }

    const db = mongodbclient.db(dbName);

    var palabraClave = req.params.palabraClave;

    db.collection("alumnos").find({nombre:new RegExp(palabraClave,'i')}}).toArray(function(err, result) {
      if (err){
        throw err;
      }
      console.log("Resultados obtenidos: " + result.length);
      mongodbclient.close();
      res.end( JSON.stringify(result));
    });
  });
};
```

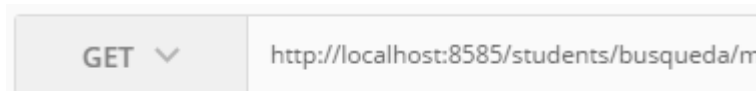
- iii. En Postman, elija el método GET y utilice la URL <http://localhost:8585/students/busqueda/a>:



En el log de NodeJS debe mostrarse:

**Resultados Obtenidos: 1**

- iv. En Postman, elija el método GET y utilice la URL `http://localhost:8585/students/busqueda/m:`



En el log de NodeJS debe mostrarse:

**Resultados Obtenidos: 2**

```
1  {
2    {
3      "_id": "5dadd7e476d8b430e46e5917",
4      "campus": "CEM",
5      "nombre": "Morty Smith",
6      "matricula": "00954557",
7      "materias": [
8        {
9          "_id": "TC3052",
10         "nombre": "Laboratorio de Desarrollo de Aplicaciones Web"
11       },
12       {
13         "_id": "TC2025",
14         "nombre": "Programacion Avanzada"
15       }
16     ],
17     "foto": "https://images.8tracks.com/cover/i/012/765/788/Eyepatchevilmorty-8791.jpg"
18   },
19   {
20     "_id": "5dadd7e476d8b430e46e5918",
21     "campus": "CEM",
22     "nombre": "Jerry Smith",
23     "matricula": "00954556",
24     "materias": [
25       {
26         "_id": "TC2025",
27         "nombre": "Programacion Avanzada"
28       }
29     ],
30     "foto": "https://pbs.twimg.com/profile_images/738078769920020481/xpw4r-Tr_400x400.jpg"
31   }
32 }
```



a. GET materias de un estudiante en específico (únicamente los nombres de las materias):

i. Generar un nuevo route en el archivo studentCollectionRoutes.js:

```
app.route('/students/materias/:matricula')
  .get(studentCollection.obtener_materias_estudiante);
```

ii. Implementar la función obtener\_materias\_estudiante” en el archivo studentCollectionController.js:

```
exports.obtener_materias_estudiante = function(req, res) {
  MongoClient.connect(url, { useNewUrlParser: true }, function(err, mdbclient) {
    if (err){
      throw err;
    }

    const db = mdbclient.db(dbName);

    var matriculaAlumno = req.params.matricula;

    //Solamente obtenemos el nombre de cada materia
    db.collection("alumnos").findOne({matricula:matriculaAlumno}, {projection:{_id:0, "materias.nombre":1}}, function(err, result) {
      if (err){
        throw err;
      }
      console.log("Consulta ejecutada...");
      mdbclient.close();
      res.end( JSON.stringify(result));
    });
  });
};
```

iii. En Postman, elija el método GET y utilice la URL <http://localhost:8585/students/materias/00954558>:

