Learn The MERN Stack

youtube playlist

Table of Contents

1 Express & MongoDB Rest API

1.1 Preconditions

Create a new repository in gitHub. Clone your new empty repo to your local computer. Start a new npm project:

CLI=> ***npm init -y***

Create backend folder to store all the files that related to backend logic.

Create a main file server.js

1.2 Installing 3rd party libraries

CLI=> ***npm i express dotenv mongoose colors***

CLI=> ***npm i -D nodemon***

1.3 Change default run scripts

Open package.json file (it should be in root folder of your project)

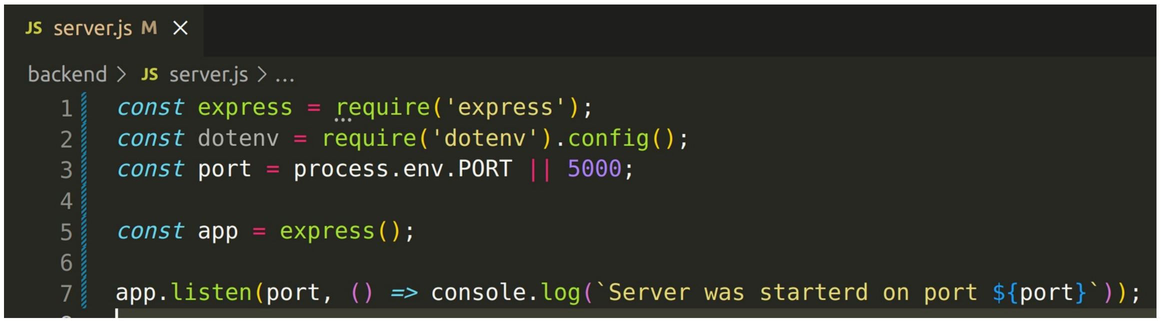
```
"scripts": {
   "start": "node backend/server.js",
   "server": "nodemon backend/server.js"
 },
```

Now you can run your server using next command:

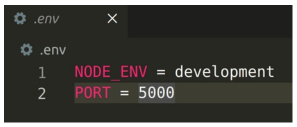
CLI=> ***npm run server***

1.4 Create a simple express app

```
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;
const app = express();
app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```



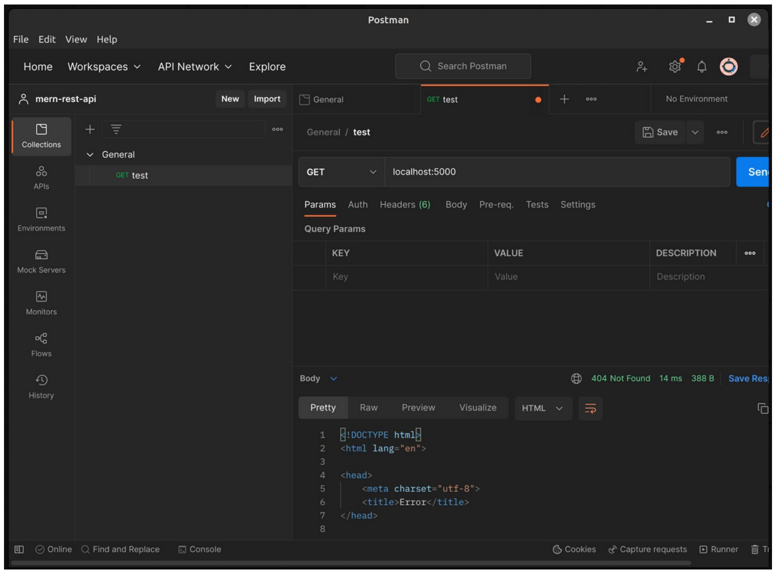
```
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```



```
NODE_ENV = development
PORT = 5000
```

1.5 Postman

Download Postman. Create a new collection and a first test request.

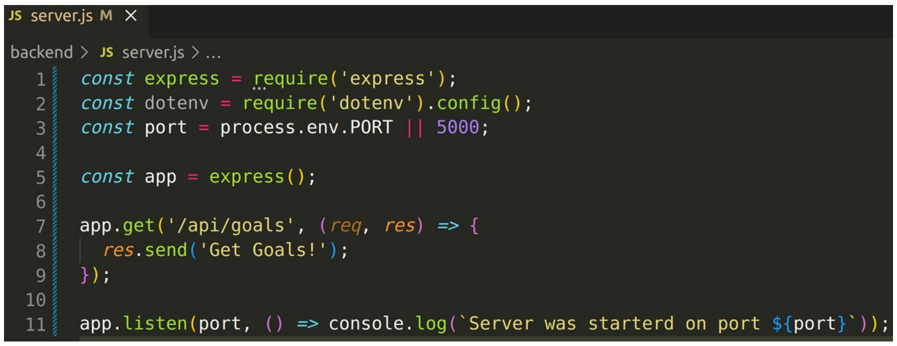

Postman
File Edit View Help
Home Workspaces API Network Explore
Search Postman
mern-rest-api New Import
Collections APIs Environments Mock Servers Monitors Flows History
General
GET test
No Environment
General / test
Save
GET
Params Auth Headers (6) Body Pre-req. Tests Settings
Query Params
KEY VALUE DESCRIPTION
Body
Pretty Raw Preview Visualize HTML
Online Find and Replace Console Cookies Capture requests Runner

```
localhost:5000
```

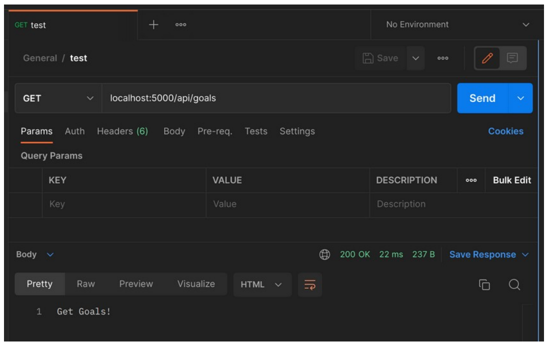
```
404 Not Found   14 ms   388 B
```

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <title>Error</title>
</head>
```

1.6 Create a simple test route with text response



```javascript
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.get('/api/goals', (req, res) => {
  res.send('Get Goals!');
});

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```

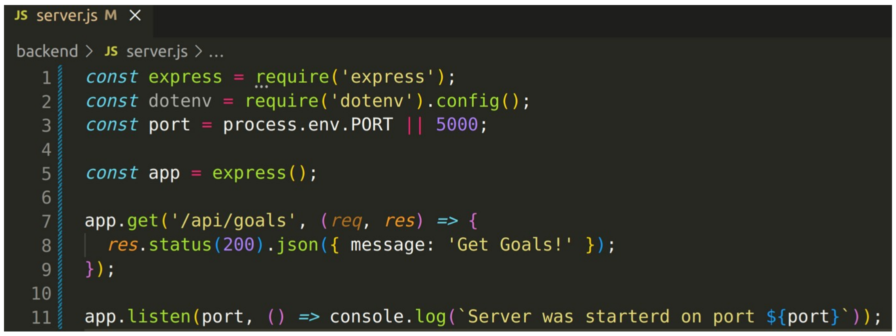

GET test
No Environment
General / test
Save
GET
Send
Params Auth Headers (6) Body Pre-req. Tests Settings Cookies
Query Params
KEY VALUE DESCRIPTION Bulk Edit
Body
Save Response
Pretty Raw Preview Visualize HTML

```
localhost:5000/api/goals
```

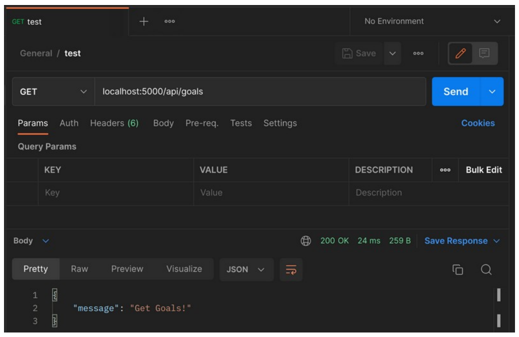
```
200 OK   22 ms   237 B
```

```
Get Goals!
```

1.7 Setup a response (as a JSON object) and a status code



```javascript
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.get('/api/goals', (req, res) => {
  res.status(200).json({ message: 'Get Goals!' });
});

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```


GET test
No Environment
General / test
Save
GET
Send
Params Auth Headers (6) Body Pre-req. Tests Settings Cookies
Query Params
KEY VALUE DESCRIPTION Bulk Edit
Body
Save Response
Pretty Raw Preview Visualize JSON

```
localhost:5000/api/goals
```

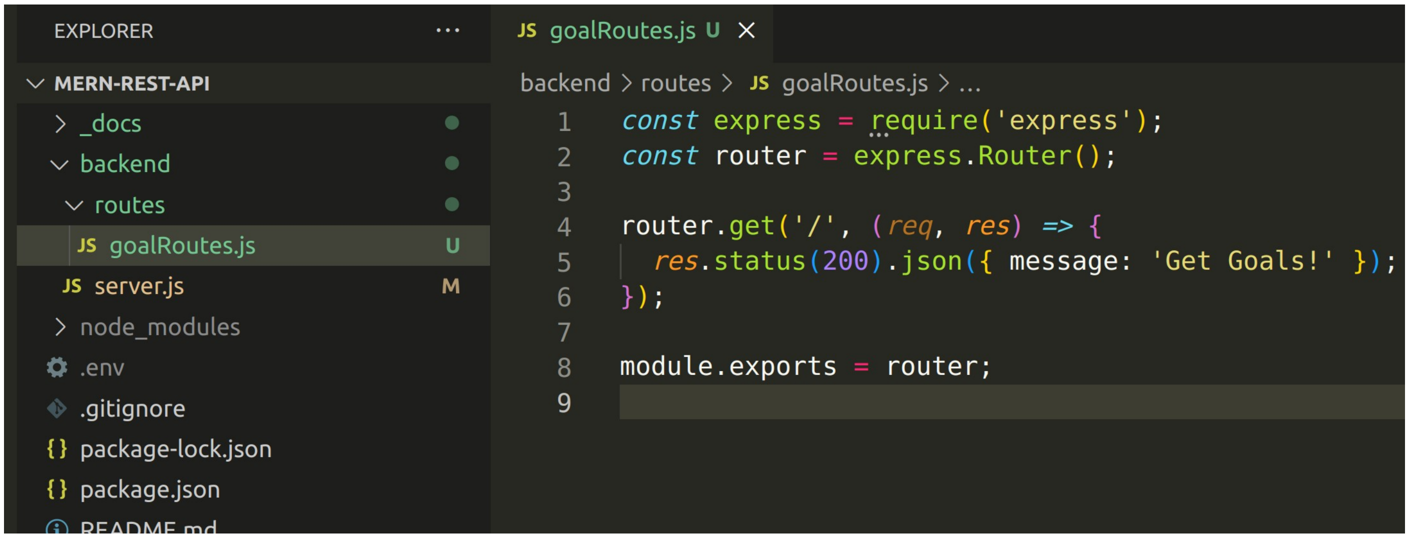
```
200 OK   24 ms   259 B
```

```
{
    "message": "Get Goals!"
}
```

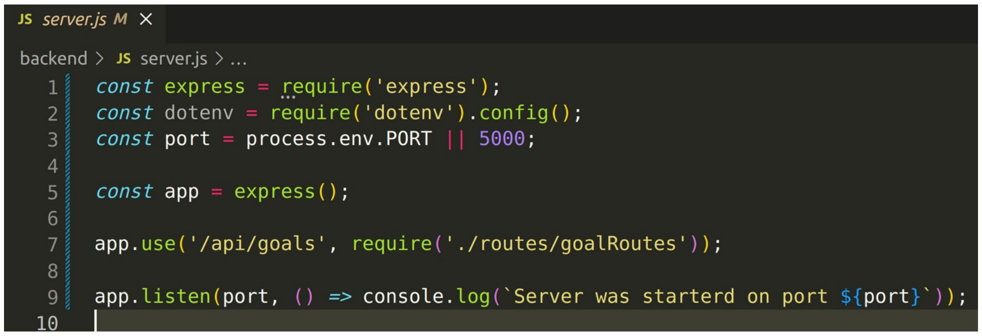
1.8 Split out all the routes into different files (entities)

Create a new folder backend/routes to store a few main routes (for each of main entities).

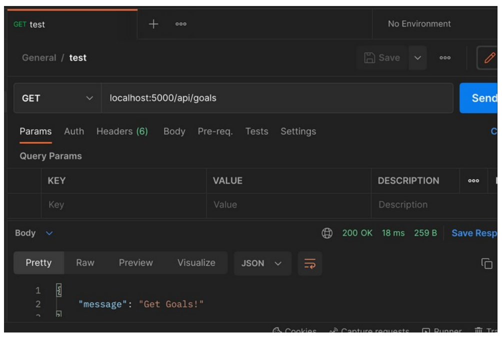
Create a first route file for Goal entity.



```javascript
const express = require('express');
const router = express.Router();

router.get('/', (req, res) => {
  res.status(200).json({ message: 'Get Goals!' });
});

module.exports = router;
```



```javascript
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.use('/api/goals', require('./routes/goalRoutes'));

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```

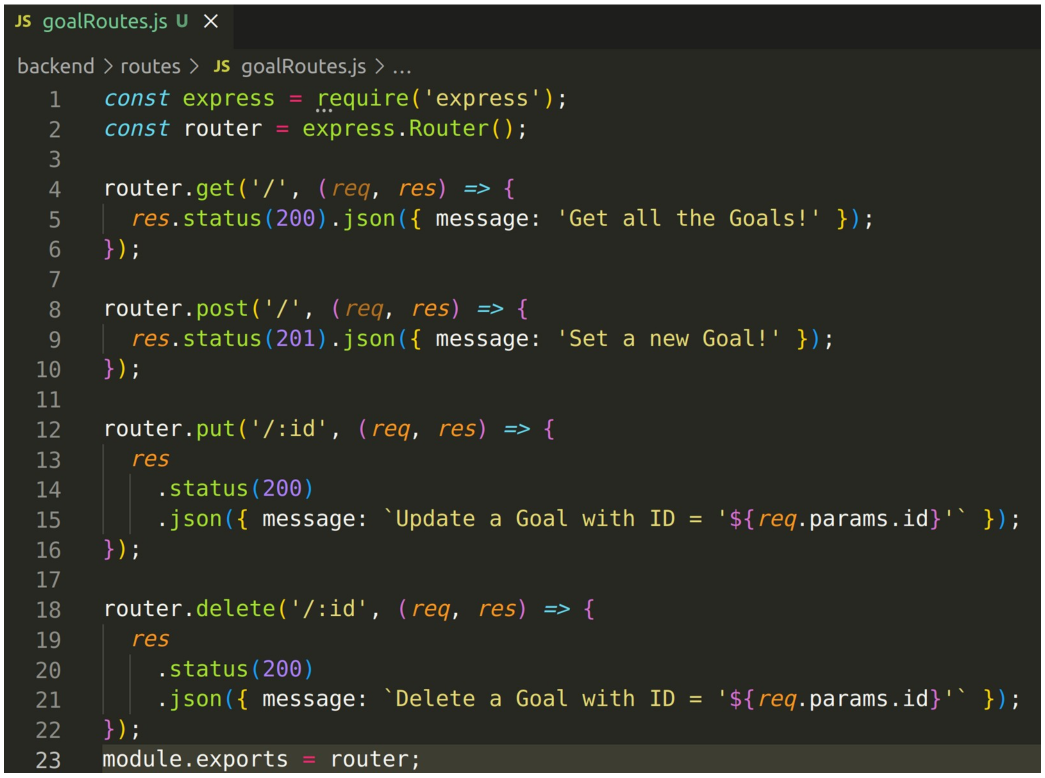


GET localhost:5000/api/goals

200 OK 18 ms 259 B

```
{
    "message": "Get Goals!"
}
```

1.9 Add different routes for Goal entity

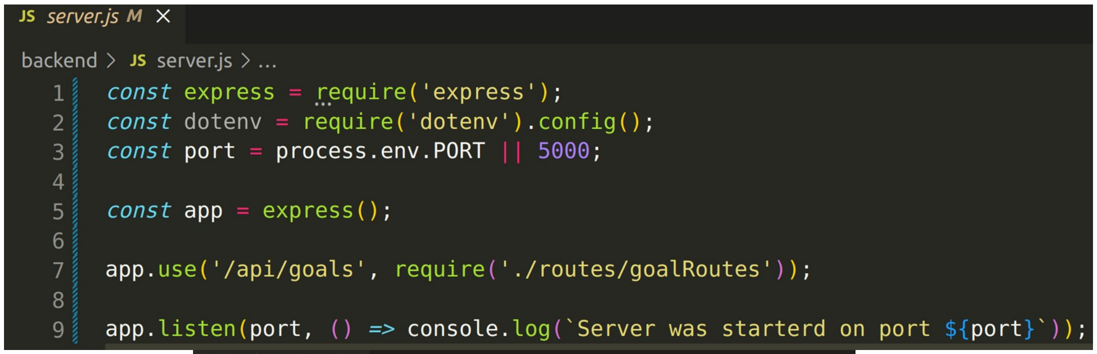

goalRoutes.js U

backend > routes > goalRoutes.js > ...

```
const express = require('express');
const router = express.Router();

router.get('/', (req, res) => {
  res.status(200).json({ message: 'Get all the Goals!' });
});

router.post('/', (req, res) => {
  res.status(201).json({ message: 'Set a new Goal!' });
});

router.put('/:id', (req, res) => {
  res
    .status(200)
    .json({ message: `Update a Goal with ID = '${req.params.id}'` });
});

router.delete('/:id', (req, res) => {
  res
    .status(200)
    .json({ message: `Delete a Goal with ID = '${req.params.id}'` });
});
module.exports = router;
```

1.10 Add Goal controller

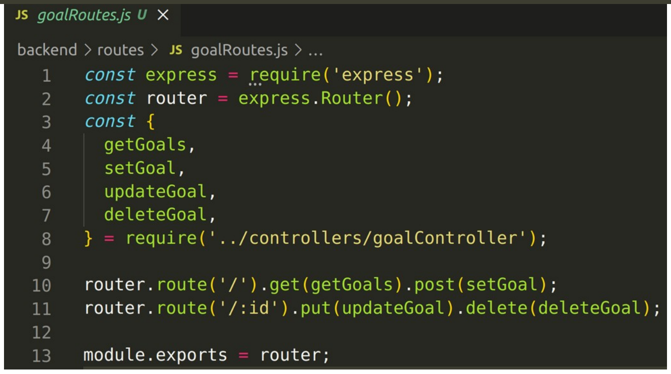
Create a new folder backend/controllers to store all the controllers. Create a new file backend/controllers/goalController.js



server.js M

backend > server.js > ...

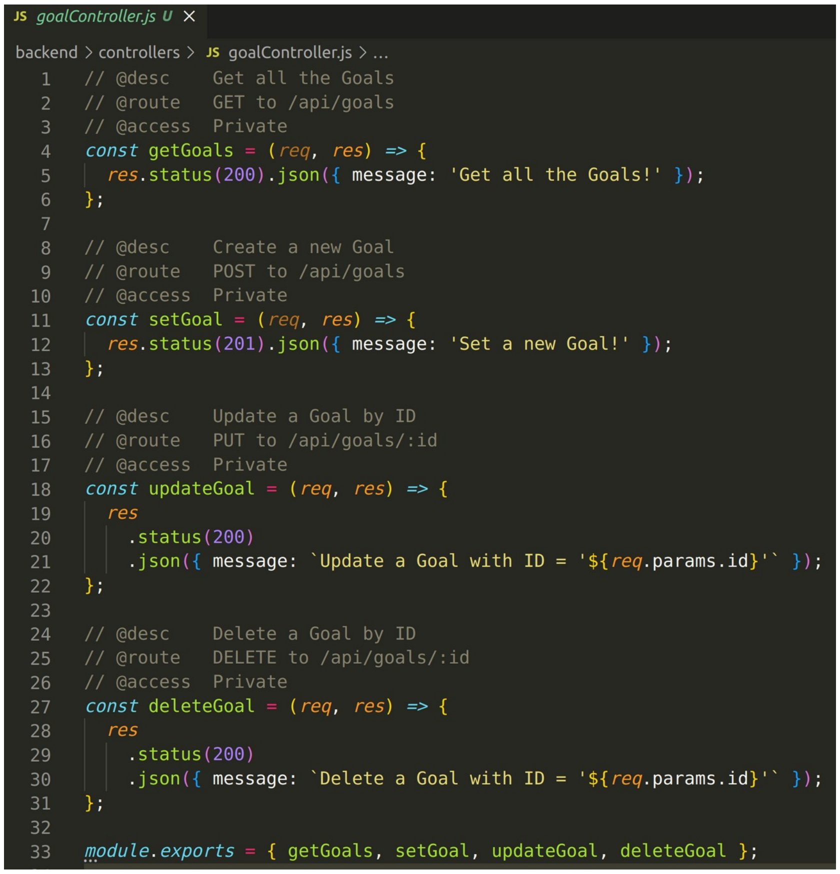
```javascript
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.use('/api/goals', require('./routes/goalRoutes'));

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```



goalRoutes.js U

backend > routes > goalRoutes.js > ...

```javascript
const express = require('express');
const router = express.Router();
const {
  getGoals,
  setGoal,
  updateGoal,
  deleteGoal,
} = require('../controllers/goalController');

router.route('/').get(getGoals).post(setGoal);
router.route('/:id').put(updateGoal).delete(deleteGoal);

module.exports = router;
```



goalController.js U

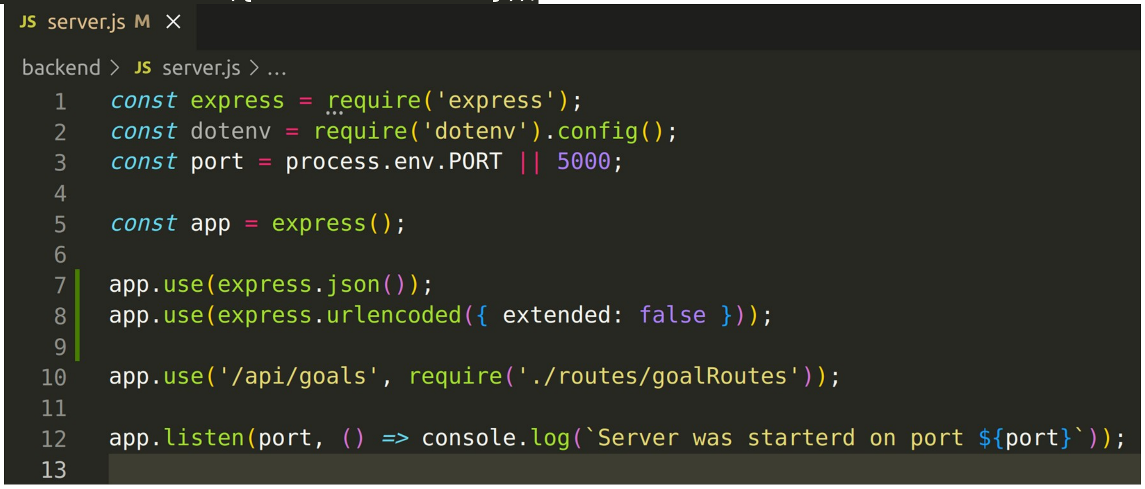
backend > controllers > goalController.js > ...

```javascript
// @desc    Get all the Goals
// @route   GET to /api/goals
// @access  Private
const getGoals = (req, res) => {
  res.status(200).json({ message: 'Get all the Goals!' });
};

// @desc    Create a new Goal
// @route   POST to /api/goals
// @access  Private
const setGoal = (req, res) => {
  res.status(201).json({ message: 'Set a new Goal!' });
};

// @desc    Update a Goal by ID
// @route   PUT to /api/goals/:id
// @access  Private
const updateGoal = (req, res) => {
  res
    .status(200)
    .json({ message: `Update a Goal with ID = '${req.params.id}'` });
};

// @desc    Delete a Goal by ID
// @route   DELETE to /api/goals/:id
// @access  Private
const deleteGoal = (req, res) => {
  res
    .status(200)
    .json({ message: `Delete a Goal with ID = '${req.params.id}'` });
};

module.exports = { getGoals, setGoal, updateGoal, deleteGoal };
```

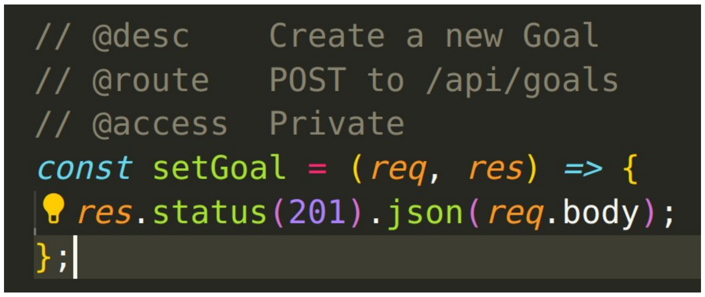
1.11 Work with data from request

Add these 2 lines to be able to work with data in requests

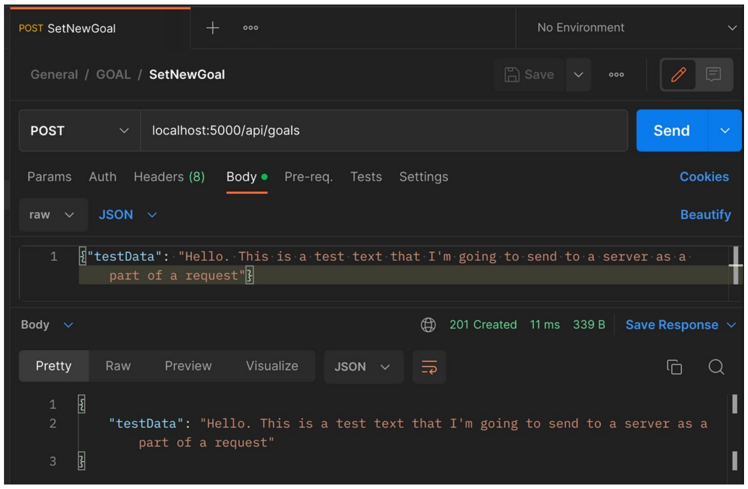
```
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
```



```
const express = require('express');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.use(express.json());
app.use(express.urlencoded({ extended: false }));

app.use('/api/goals', require('./routes/goalRoutes'));

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```



```
// @desc    Create a new Goal
// @route   POST to /api/goals
// @access  Private
const setGoal = (req, res) => {
  res.status(201).json(req.body);
};
```



POST localhost:5000/api/goals

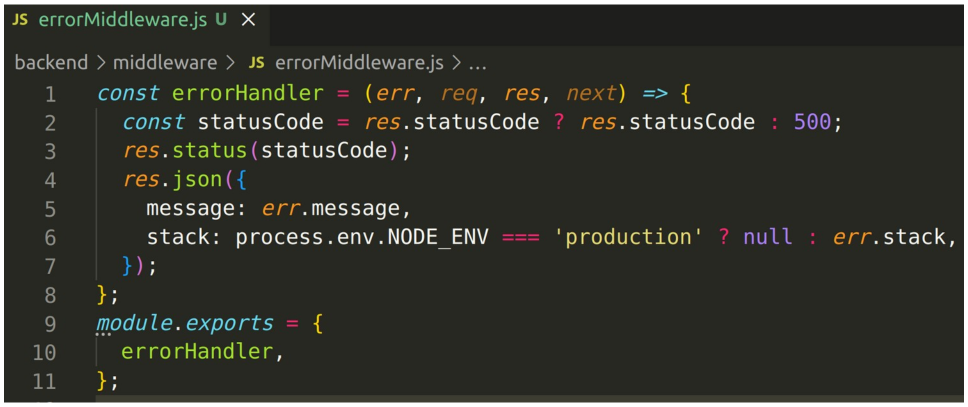
```
{"testData": "Hello. This is a test text that I'm going to send to a server as a part of a request"}
```

201 Created 11 ms 339 B

```
{
    "testData": "Hello. This is a test text that I'm going to send to a server as a part of a request"
}
```

1.12 Express error handler

Create a new folder backend/middleware to store all the middlewares. Create a new file backend/middleware/errorMiddleware.js

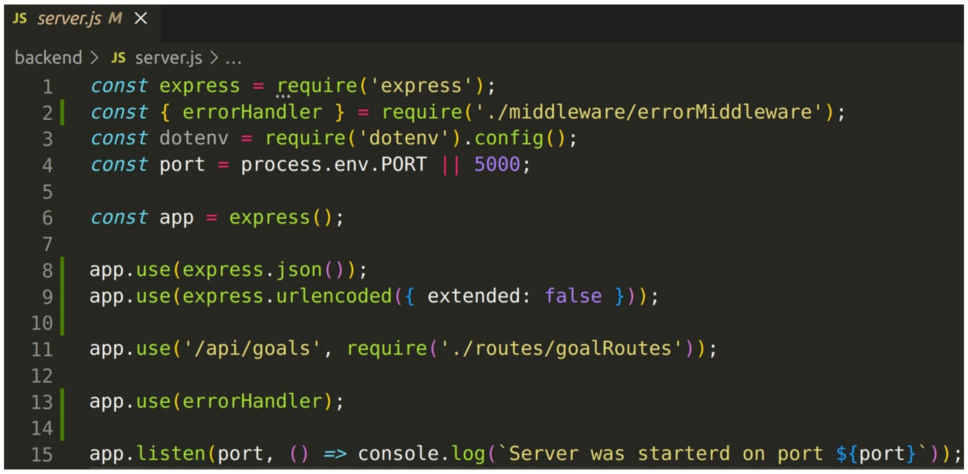


errorMiddleware.js U

backend > middleware > errorMiddleware.js > ...

```js
const errorHandler = (err, req, res, next) => {
  const statusCode = res.statusCode ? res.statusCode : 500;
  res.status(statusCode);
  res.json({
    message: err.message,
    stack: process.env.NODE_ENV === 'production' ? null : err.stack,
  });
};
module.exports = {
  errorHandler,
};
```

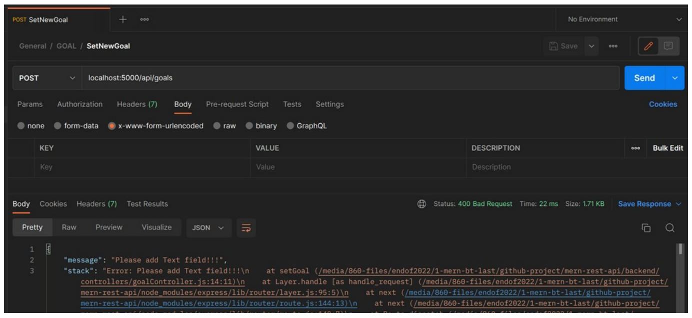
Import your first middleware into server.js and apply it



server.js M

backend > server.js > ...

```js
const express = require('express');
const { errorHandler } = require('./middleware/errorMiddleware');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

const app = express();

app.use(express.json());
app.use(express.urlencoded({ extended: false }));

app.use('/api/goals', require('./routes/goalRoutes'));

app.use(errorHandler);

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```

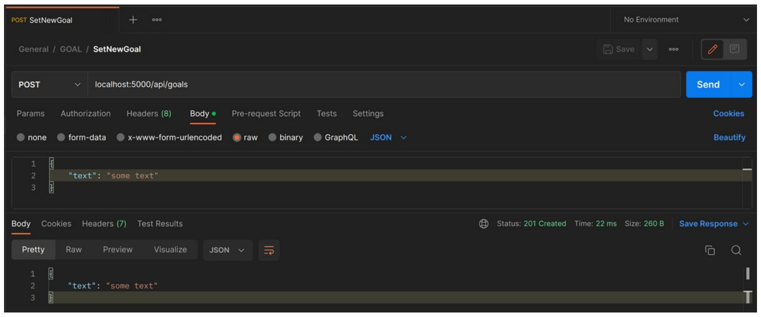


POST localhost:5000/api/goals

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 400 Bad Request Time: 22 ms Size: 1.71 KB

```json
{
    "message": "Please add Text field!!!",
    "stack": "Error: Please add Text field!!!\n    at setGoal (/media/860-files/endof2022/1-mern-bt-last/github-project/mern-rest-api/backend/
        controllers/goalController.js:14:11)\n    at Layer.handle [as handle_request] (/media/860-files/endof2022/1-mern-bt-last/github-project/
        mern-rest-api/node_modules/express/lib/router/layer.js:95:5)\n    at next (/media/860-files/endof2022/1-mern-bt-last/github-project/
        mern-rest-api/node_modules/express/lib/router/route.js:144:13)\n    at next (/media/860-files/endof2022/1-mern-bt-last/github-project/
```



POST localhost:5000/api/goals

```json
{
    "text": "some text"
}
```

Status: 201 Created Time: 22 ms Size: 260 B

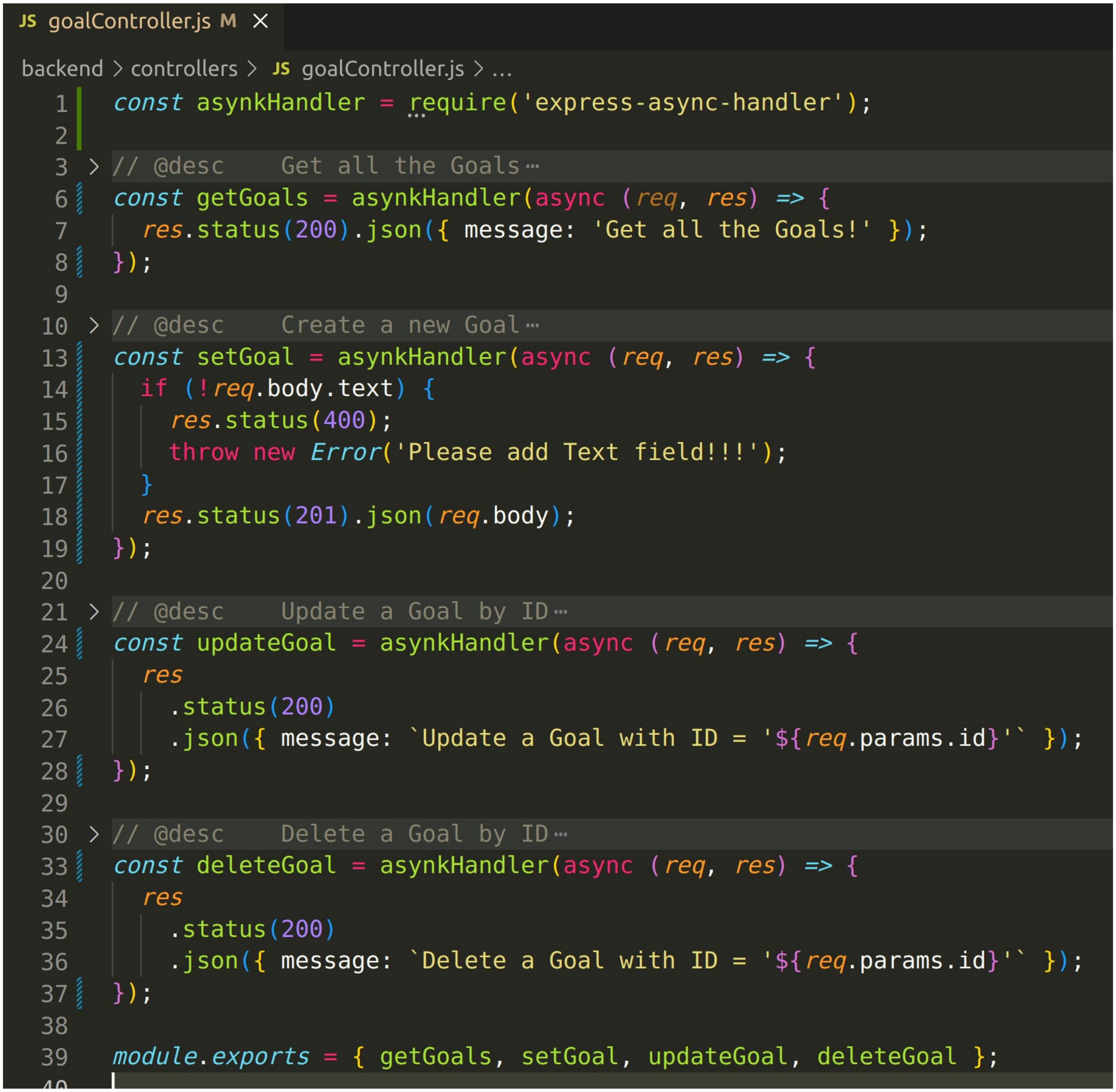
```json
{
    "text": "some text"
}
```

1.13 Using express-async-handler

Install express-async-handler to be able to work witn async/await requests (for mongoose/mongoDB) and try/catch blocks

CLI=> ***npm i express-async-handler***

Import in into controller and change each function



JS goalController.js M ×

backend > controllers > JS goalController.js > ...

```
const asynkHandler = require('express-async-handler');

// @desc    Get all the Goals⋯
const getGoals = asynkHandler(async (req, res) => {
  res.status(200).json({ message: 'Get all the Goals!' });
});

// @desc    Create a new Goal⋯
const setGoal = asynkHandler(async (req, res) => {
  if (!req.body.text) {
    res.status(400);
    throw new Error('Please add Text field!!!');
  }
  res.status(201).json(req.body);
});

// @desc    Update a Goal by ID⋯
const updateGoal = asynkHandler(async (req, res) => {
  res
    .status(200)
    .json({ message: `Update a Goal with ID = '${req.params.id}'` });
});

// @desc    Delete a Goal by ID⋯
const deleteGoal = asynkHandler(async (req, res) => {
  res
    .status(200)
    .json({ message: `Delete a Goal with ID = '${req.params.id}'` });
});

module.exports = { getGoals, setGoal, updateGoal, deleteGoal };
```

1.14 MongoDB

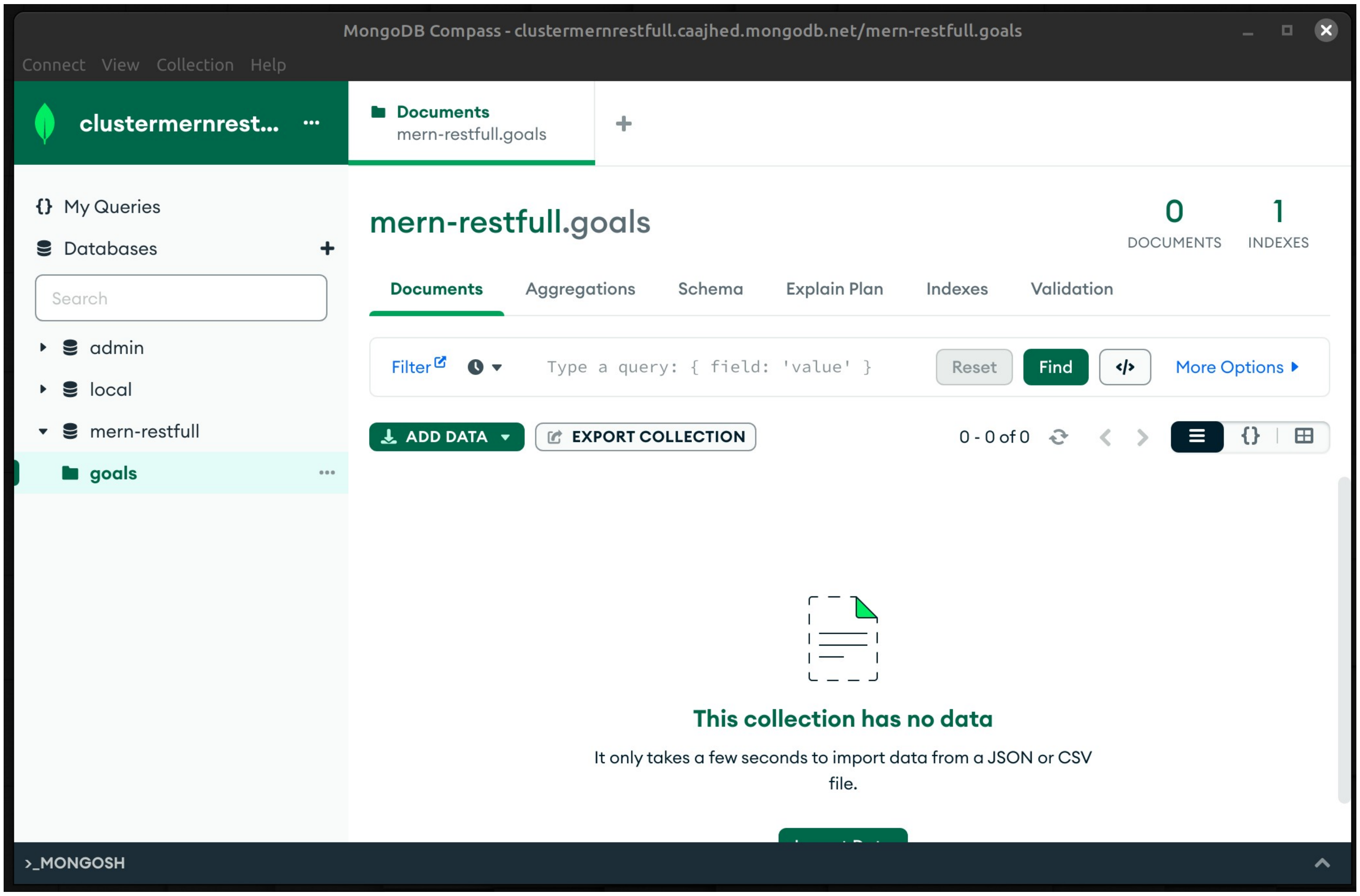
Log in into https://cloud.mongodb.com/ and create a new project.

Create a new DataBase.

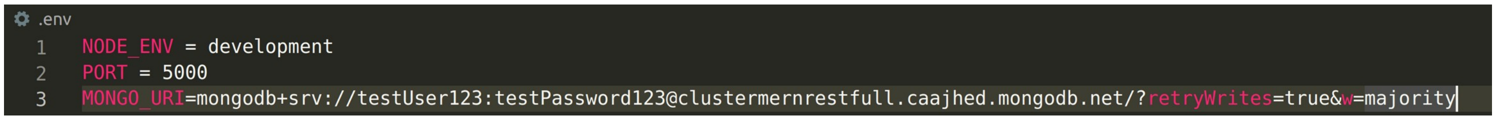
testUser123 and testPassword123.

Install compass. Create a new database. Create a new collection goals. Click connect to a database using compass.

Copy connection link. Open Compass and click connect. Paste connection link. Add password and change database name. Click connect.

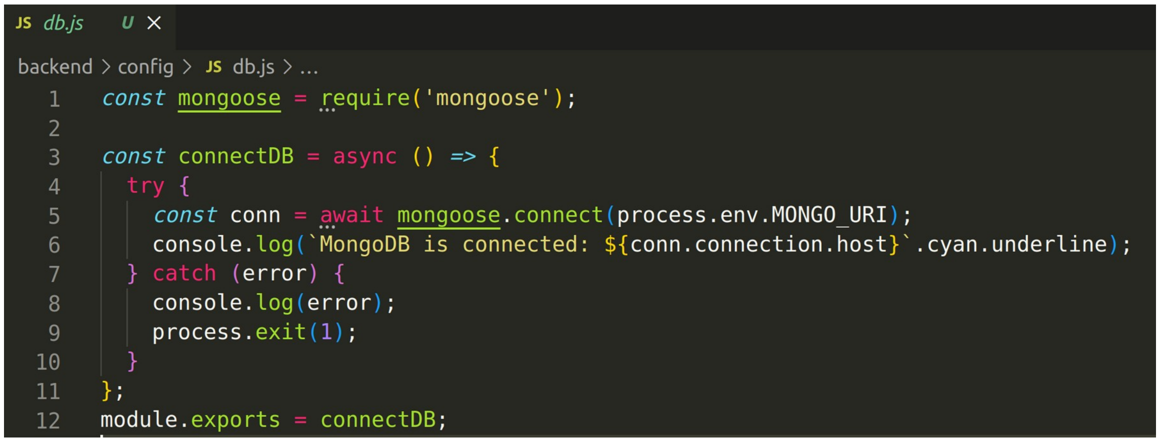

MongoDB Compass - clustermernrestfull.caajhed.mongodb.net/mern-restfull.goals
Connect View Collection Help
clustermernrest...
Documents
mern-restfull.goals
My Queries
Databases
Search
admin
local
mern-restfull
goals
mern-restfull.goals
0 DOCUMENTS
1 INDEXES
Documents Aggregations Schema Explain Plan Indexes Validation
Filter
Type a query: { field: 'value' }
Reset
Find
More Options
ADD DATA
EXPORT COLLECTION
0 - 0 of 0
This collection has no data
It only takes a few seconds to import data from a JSON or CSV file.
>_MONGOSH

In web version click connect to a database and choose connect your application. Copy connection link. Create a new ENV variable MONGO_URI.



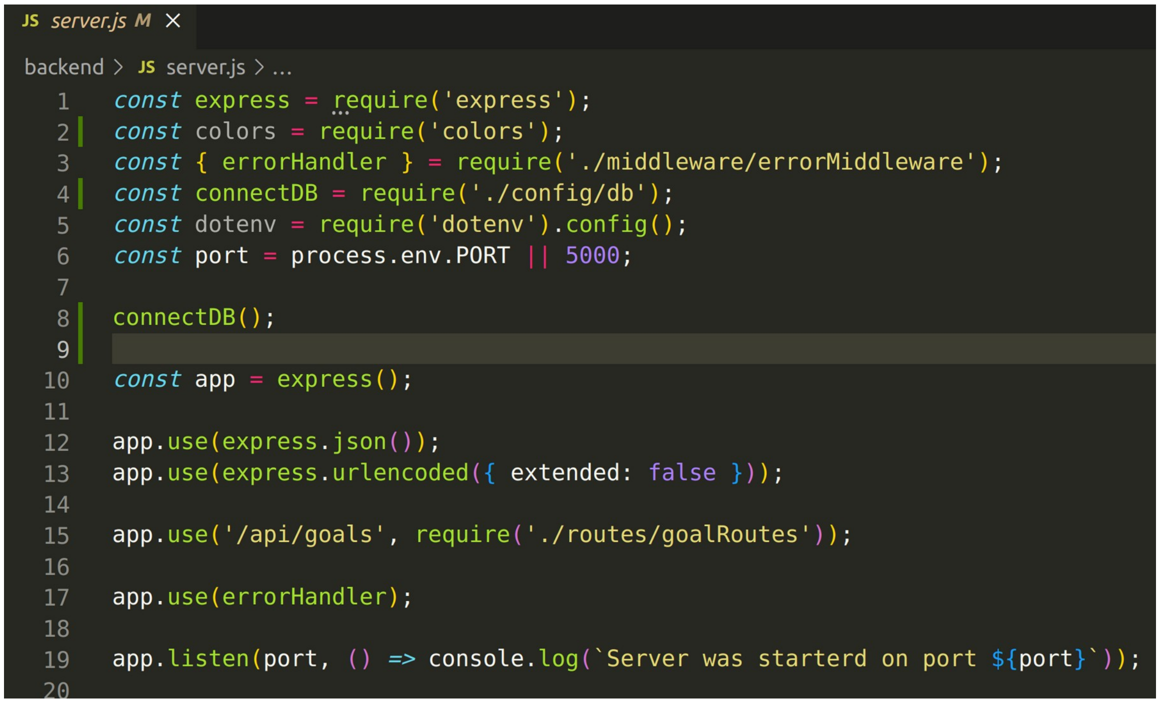
```
.env
NODE_ENV = development
PORT = 5000
MONGO_URI=mongodb+srv://testUser123:testPassword123@clustermernrestfull.caajhed.mongodb.net/?retryWrites=true&w=majority
```

Create a new folder backend/config. Create a new file backend/config/db.js

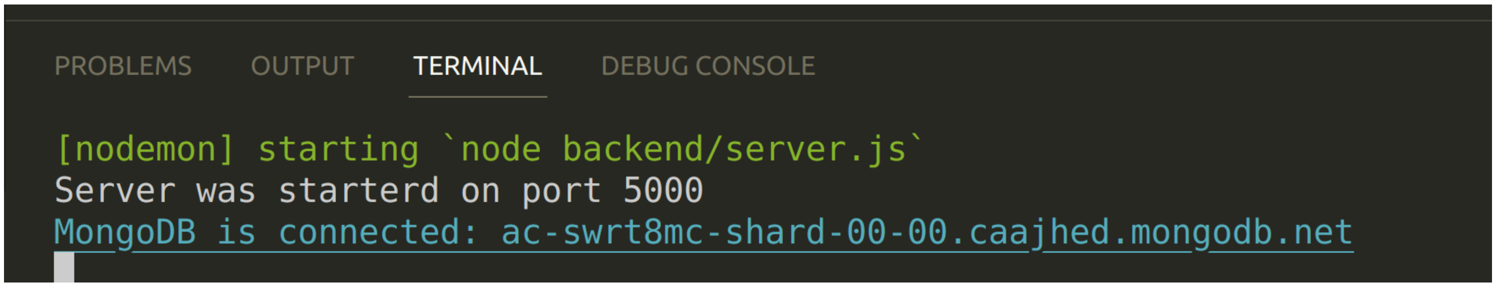


```javascript
const mongoose = require('mongoose');

const connectDB = async () => {
  try {
    const conn = await mongoose.connect(process.env.MONGO_URI);
    console.log(`MongoDB is connected: ${conn.connection.host}`.cyan.underline);
  } catch (error) {
    console.log(error);
    process.exit(1);
  }
};
module.exports = connectDB;
```

Import it into server.js and invoke it.



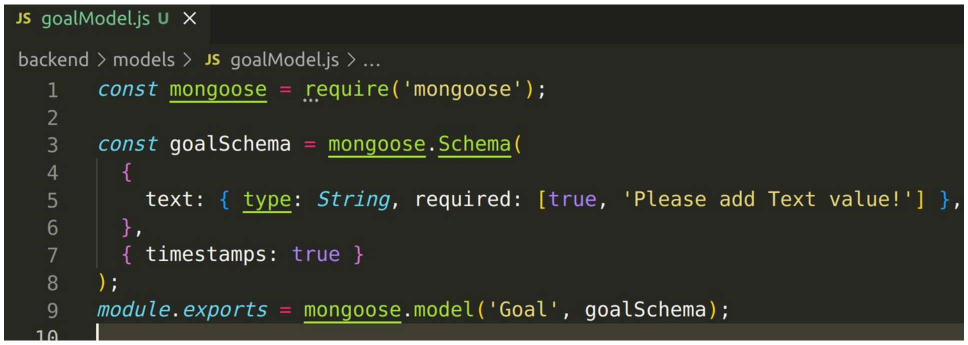
```javascript
const express = require('express');
const colors = require('colors');
const { errorHandler } = require('./middleware/errorMiddleware');
const connectDB = require('./config/db');
const dotenv = require('dotenv').config();
const port = process.env.PORT || 5000;

connectDB();

const app = express();

app.use(express.json());
app.use(express.urlencoded({ extended: false }));

app.use('/api/goals', require('./routes/goalRoutes'));

app.use(errorHandler);

app.listen(port, () => console.log(`Server was starterd on port ${port}`));
```



```
[nodemon] starting `node backend/server.js`
Server was starterd on port 5000
MongoDB is connected: ac-swrt8mc-shard-00-00.caajhed.mongodb.net
```

1.15 Create Goal model

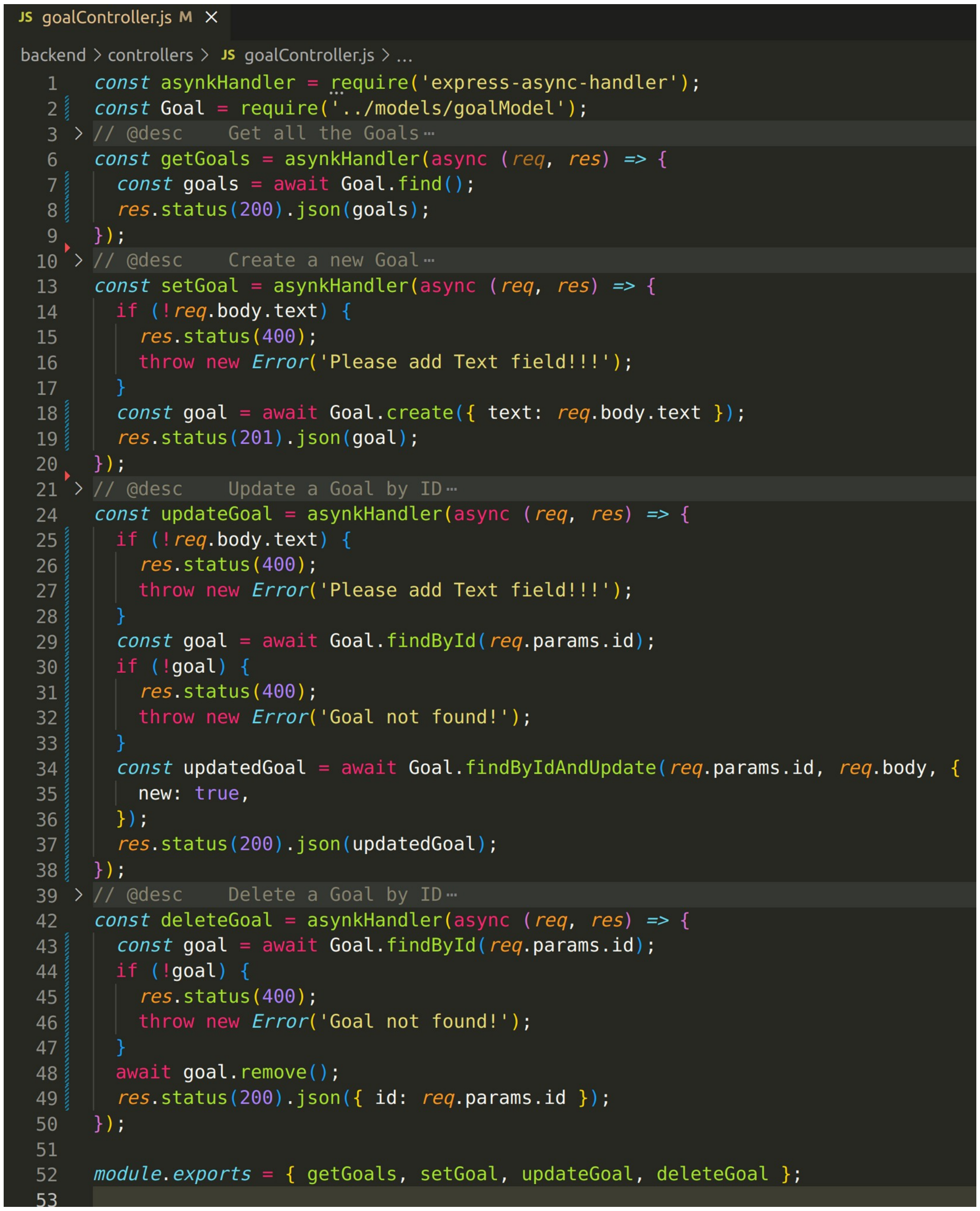
Create a new folder backend/models to store all the models. Create a new file backend/models/goalModel.js



goalModel.js U

backend > models > goalModel.js > ...

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

const goalSchema = mongoose.Schema(
  {
    text: { type: String, required: [true, 'Please add Text value!'] },
  },
  { timestamps: true }
);
module.exports = mongoose.model('Goal', goalSchema);
```



goalController.js M

backend > controllers > goalController.js > ...

```
const asynkHandler = require('express-async-handler');
const Goal = require('../models/goalModel');
// @desc    Get all the Goals…
const getGoals = asynkHandler(async (req, res) => {
  const goals = await Goal.find();
  res.status(200).json(goals);
});
// @desc    Create a new Goal…
const setGoal = asynkHandler(async (req, res) => {
  if (!req.body.text) {
    res.status(400);
    throw new Error('Please add Text field!!!');
  }
  const goal = await Goal.create({ text: req.body.text });
  res.status(201).json(goal);
});
// @desc    Update a Goal by ID…
const updateGoal = asynkHandler(async (req, res) => {
  if (!req.body.text) {
    res.status(400);
    throw new Error('Please add Text field!!!');
  }
  const goal = await Goal.findById(req.params.id);
  if (!goal) {
    res.status(400);
    throw new Error('Goal not found!');
  }
  const updatedGoal = await Goal.findByIdAndUpdate(req.params.id, req.body, {
    new: true,
  });
  res.status(200).json(updatedGoal);
});
// @desc    Delete a Goal by ID…
const deleteGoal = asynkHandler(async (req, res) => {
  const goal = await Goal.findById(req.params.id);
  if (!goal) {
    res.status(400);
    throw new Error('Goal not found!');
  }
  await goal.remove();
  res.status(200).json({ id: req.params.id });
});

module.exports = { getGoals, setGoal, updateGoal, deleteGoal };
```