Learn The MERN Stack - Express & MongoDB Rest API

YOUTUBE

express
dotenv
mongoose
colors
express-async-handler

nodemon

Table of Contents

1. Project initial setup

1.1 Create an empty gitHub repository for the project

1.2 Clone the repository to the local computer

CLI=> **git clone https://github.com/ILopatenko/mern-app.git**

1.3 Initialize a new NPM project

CLI=> **npm init -y**

1.4 Project structure

All the files that belongs to backend should be stored inside ./backend folder

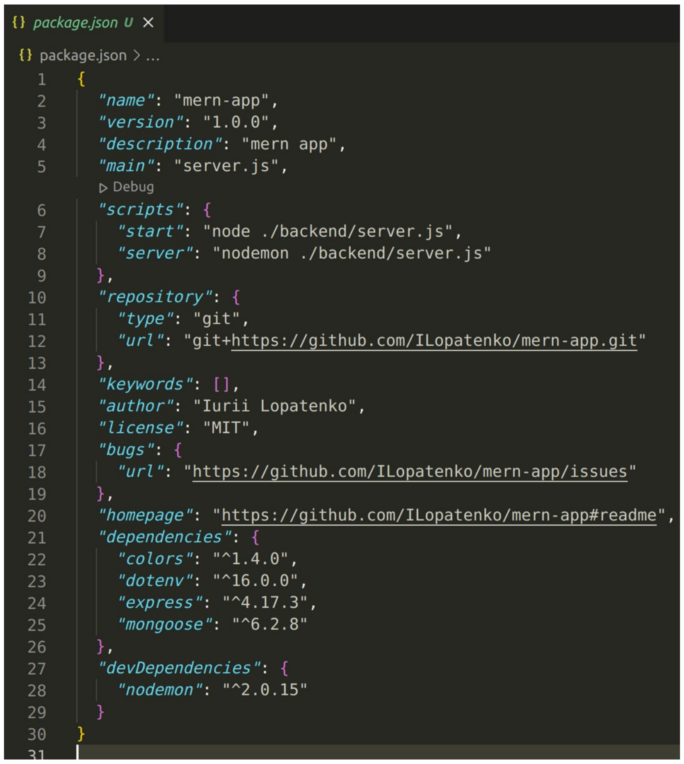
Main backend file – ./backend/server.js

1.5 Install additional packages

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

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

1.6 Change the start npm script

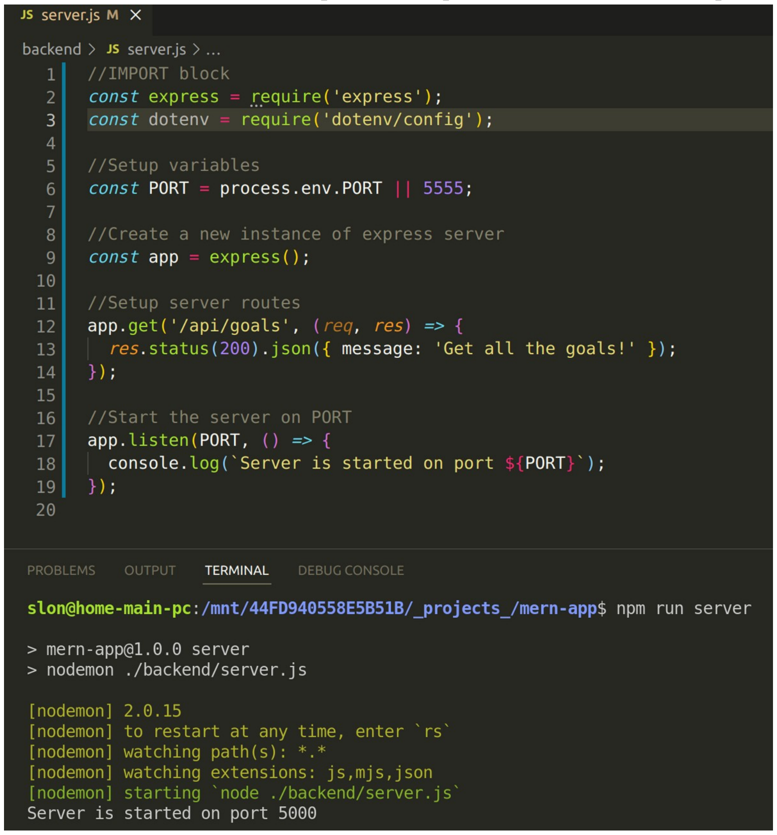


```json
{
  "name": "mern-app",
  "version": "1.0.0",
  "description": "mern app",
  "main": "server.js",
  "scripts": {
    "start": "node ./backend/server.js",
    "server": "nodemon ./backend/server.js"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/ILopatenko/mern-app.git"
  },
  "keywords": [],
  "author": "Iurii Lopatenko",
  "license": "MIT",
  "bugs": {
    "url": "https://github.com/ILopatenko/mern-app/issues"
  },
  "homepage": "https://github.com/ILopatenko/mern-app#readme",
  "dependencies": {
    "colors": "^1.4.0",
    "dotenv": "^16.0.0",
    "express": "^4.17.3",
    "mongoose": "^6.2.8"
  },
  "devDependencies": {
    "nodemon": "^2.0.15"
  }
}
```

Now I can use **npm run start** to start my server with node.js and **npm run server** to start my server with nodemon

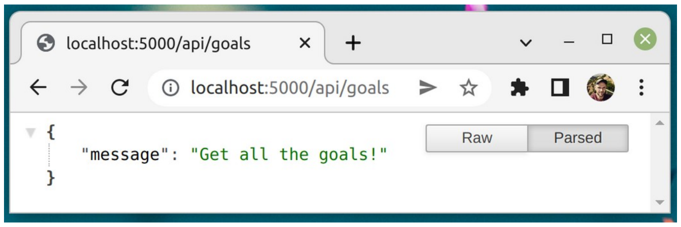
2. Backend

2.1 Create a simple express server (and .env file)



```javascript
//IMPORT block
const express = require('express');
const dotenv = require('dotenv/config');

//Setup variables
const PORT = process.env.PORT || 5555;

//Create a new instance of express server
const app = express();

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

//Start the server on PORT
app.listen(PORT, () => {
  console.log(`Server is started on port ${PORT}`);
});
```

```
slon@home-main-pc:/mnt/44FD940558E5B51B/_projects_/mern-app$ npm run server

> mern-app@1.0.0 server
> nodemon ./backend/server.js

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node ./backend/server.js`
Server is started on port 5000
```

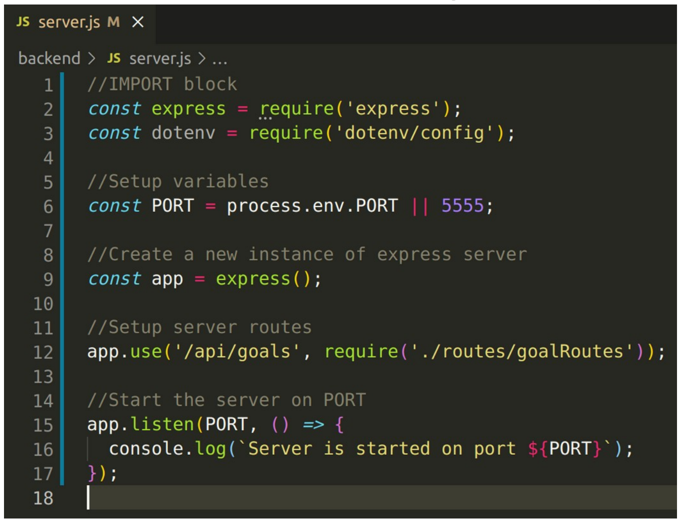


```
{
  "message": "Get all the goals!"
}
```

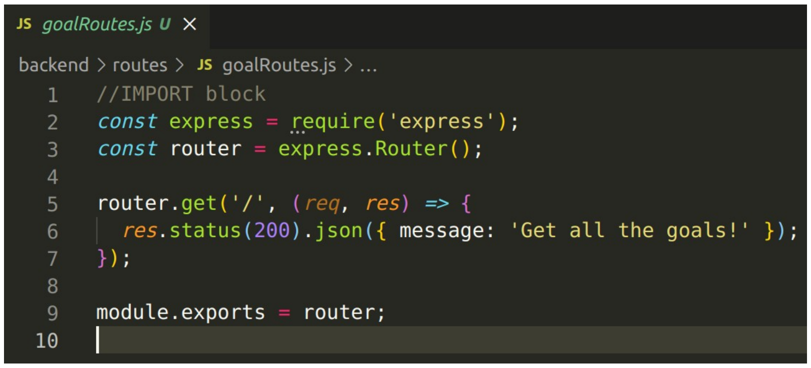
2.2 Split out all the logic into separate endpoints/routes

Create a new folder ./backend/routes to store/work with different routes.

Create a new file ./backend/routes/goalRoutes.js to store/work with all the routes that belong to GOAL entity.

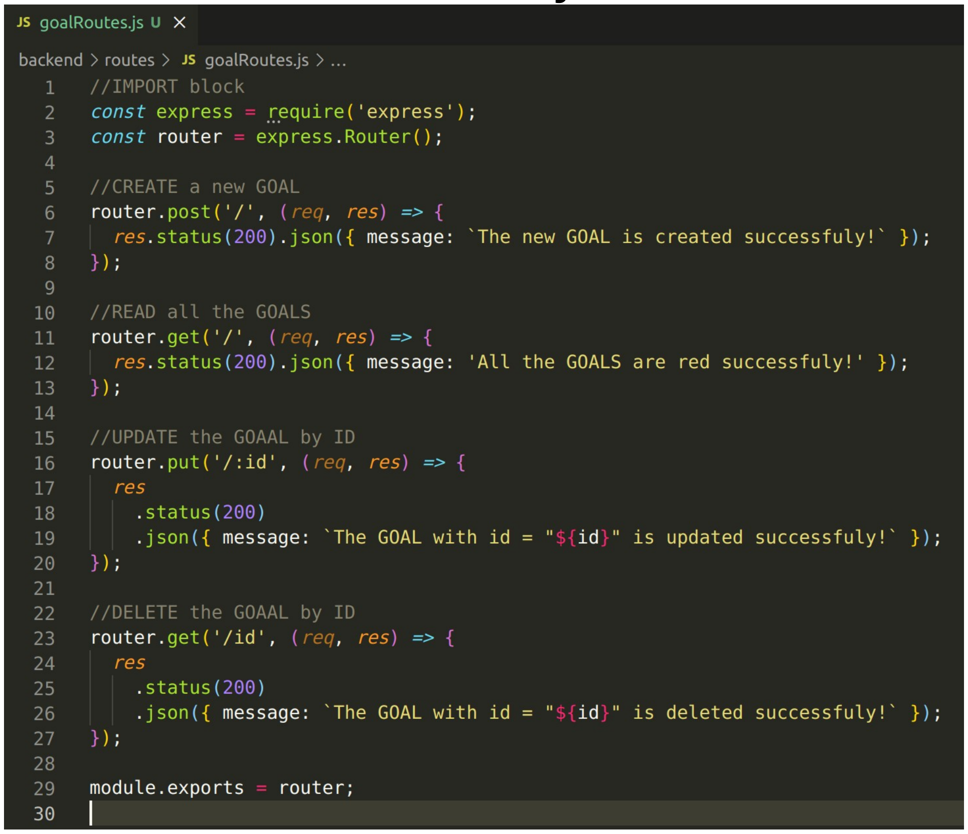


```javascript
//IMPORT block
const express = require('express');
const dotenv = require('dotenv/config');

//Setup variables
const PORT = process.env.PORT || 5555;

//Create a new instance of express server
const app = express();

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

//Start the server on PORT
app.listen(PORT, () => {
  console.log(`Server is started on port ${PORT}`);
});
```



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

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

module.exports = router;
```

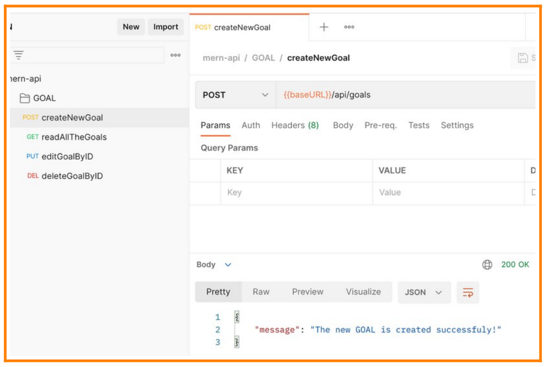
2.3 Add additional routes for GOAL entity


goalRoutes.js U

backend > routes > goalRoutes.js > ...

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

//CREATE a new GOAL
router.post('/', (req, res) => {
  res.status(200).json({ message: `The new GOAL is created successfuly!` });
});

//READ all the GOALS
router.get('/', (req, res) => {
  res.status(200).json({ message: 'All the GOALS are red successfuly!' });
});

//UPDATE the GOAAL by ID
router.put('/:id', (req, res) => {
  res
    .status(200)
    .json({ message: `The GOAL with id = "${id}" is updated successfuly!` });
});

//DELETE the GOAAL by ID
router.get('/id', (req, res) => {
  res
    .status(200)
    .json({ message: `The GOAL with id = "${id}" is deleted successfuly!` });
});

module.exports = router;
```

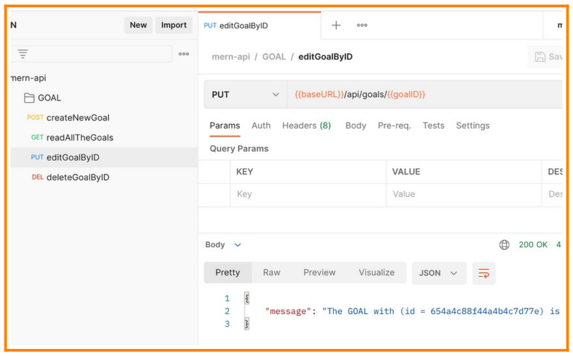
2.4 Create first requests in Postman


mern-api / GOAL / createNewGoal

POST {{baseURL}}/api/goals

Body 200 OK

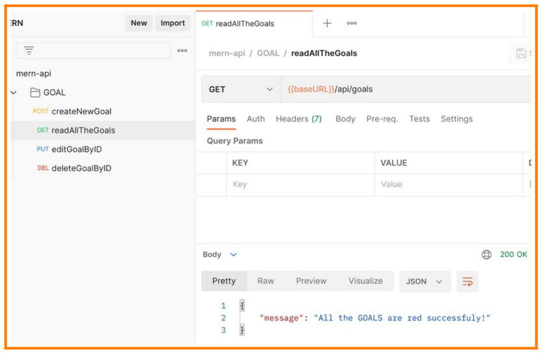
```
{
    "message": "The new GOAL is created successfuly!"
}
```


mern-api / GOAL / editGoalByID

PUT {{baseURL}}/api/goals/{{goalID}}

Body 200 OK

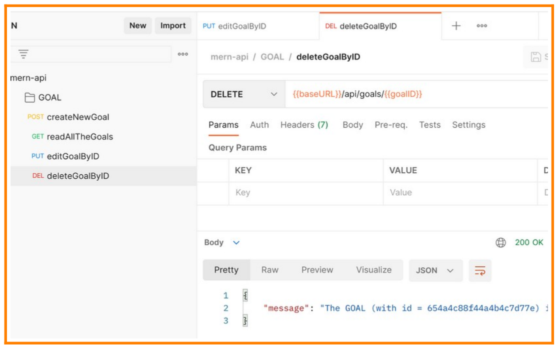
```
{
    "message": "The GOAL with (id = 654a4c88f44a4b4c7d77e) is
}
```


mern-api / GOAL / readAllTheGoals

GET {{baseURL}}/api/goals

Body 200 OK

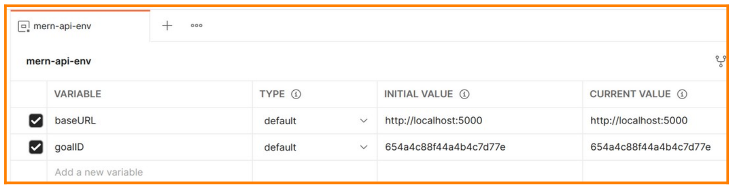
```
{
    "message": "All the GOALS are red successfuly!"
}
```


mern-api / GOAL / deleteGoalByID

DELETE {{baseURL}}/api/goals/{{goalID}}

Body 200 OK

```
{
    "message": "The GOAL (with id = 654a4c88f44a4b4c7d77e) i
}
```

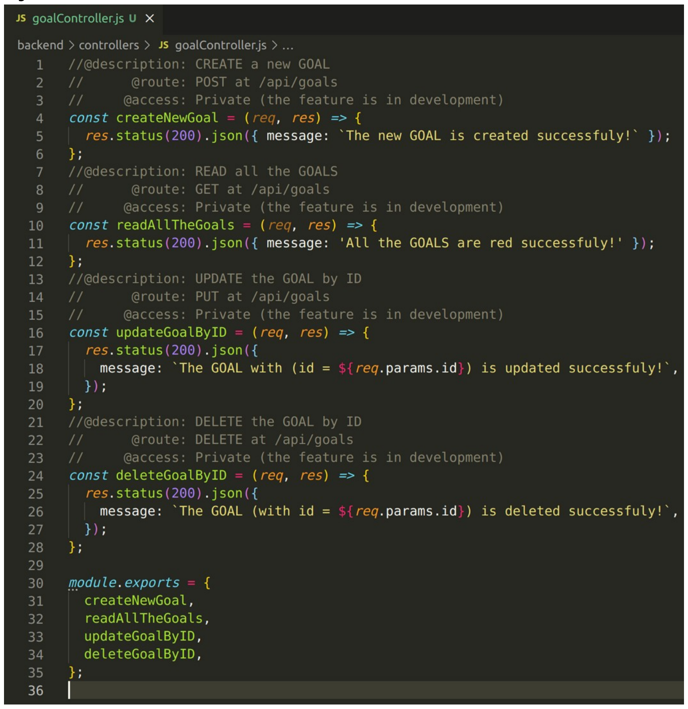

mern-api-env

	VARIABLE	TYPE	INITIAL VALUE	CURRENT VALUE
✓	baseURL	default	http://localhost:5000	http://localhost:5000
✓	goalID	default	654a4c88f44a4b4c7d77e	654a4c88f44a4b4c7d77e
	Add a new variable			

2.4 Create the first controller (for GOAL entity)

Create a new folder ./backend/controllers to store/work with different controllers.

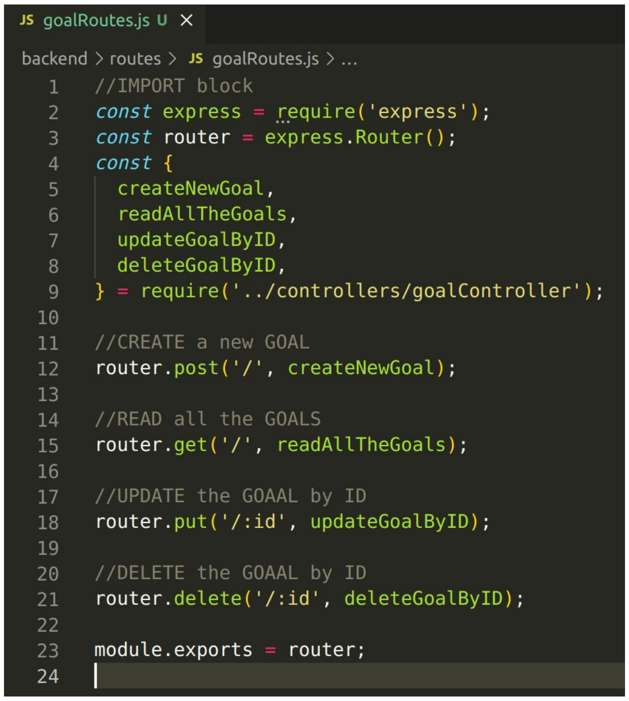
Create a new file ./backend/controllers/goalController.js to store/work with all the routes that belong to GOAL entity.



goalController.js U ×

backend > controllers > goalController.js > ...

```javascript
//@description: CREATE a new GOAL
//      @route: POST at /api/goals
//     @access: Private (the feature is in development)
const createNewGoal = (req, res) => {
  res.status(200).json({ message: `The new GOAL is created successfuly!` });
};
//@description: READ all the GOALS
//      @route: GET at /api/goals
//     @access: Private (the feature is in development)
const readAllTheGoals = (req, res) => {
  res.status(200).json({ message: 'All the GOALS are red successfuly!' });
};
//@description: UPDATE the GOAL by ID
//      @route: PUT at /api/goals
//     @access: Private (the feature is in development)
const updateGoalByID = (req, res) => {
  res.status(200).json({
    message: `The GOAL with (id = ${req.params.id}) is updated successfuly!`,
  });
};
//@description: DELETE the GOAL by ID
//      @route: DELETE at /api/goals
//     @access: Private (the feature is in development)
const deleteGoalByID = (req, res) => {
  res.status(200).json({
    message: `The GOAL (with id = ${req.params.id}) is deleted successfuly!`,
  });
};

module.exports = {
  createNewGoal,
  readAllTheGoals,
  updateGoalByID,
  deleteGoalByID,
};
```

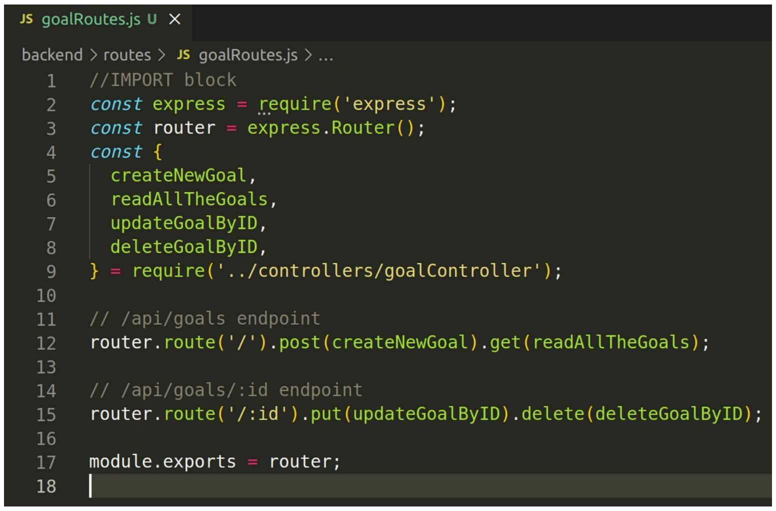


goalRoutes.js U ×

backend > routes > goalRoutes.js > ...

```javascript
//IMPORT block
const express = require('express');
const router = express.Router();
const {
  createNewGoal,
  readAllTheGoals,
  updateGoalByID,
  deleteGoalByID,
} = require('../controllers/goalController');

//CREATE a new GOAL
router.post('/', createNewGoal);

//READ all the GOALS
router.get('/', readAllTheGoals);

//UPDATE the GOAAL by ID
router.put('/:id', updateGoalByID);

//DELETE the GOAAL by ID
router.delete('/:id', deleteGoalByID);

module.exports = router;
```

And final update for goalRoutes.js

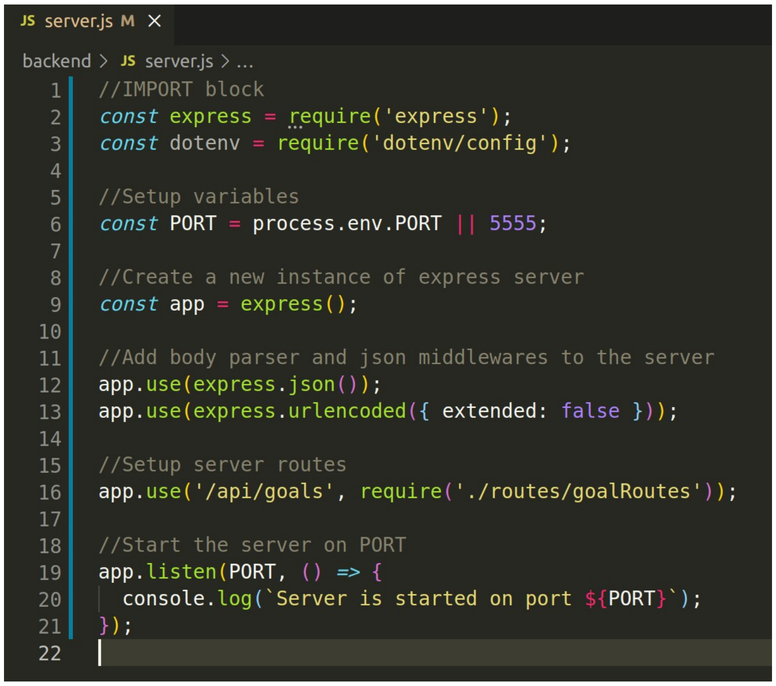


goalRoutes.js U ×

backend > routes > goalRoutes.js > ...

```javascript
//IMPORT block
const express = require('express');
const router = express.Router();
const {
  createNewGoal,
  readAllTheGoals,
  updateGoalByID,
  deleteGoalByID,
} = require('../controllers/goalController');

// /api/goals endpoint
router.route('/').post(createNewGoal).get(readAllTheGoals);

// /api/goals/:id endpoint
router.route('/:id').put(updateGoalByID).delete(deleteGoalByID);

module.exports = router;
```

2.5 Add additional functionality to be able to work with body of request/response and json data


JS server.js M ×

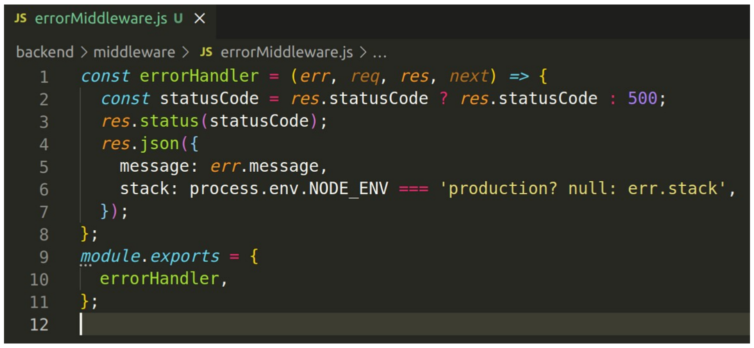
backend > JS server.js > ...

```javascript
//IMPORT block
const express = require('express');
const dotenv = require('dotenv/config');

//Setup variables
const PORT = process.env.PORT || 5555;

//Create a new instance of express server
const app = express();

//Add body parser and json middlewares to the server
app.use(express.json());
app.use(express.urlencoded({ extended: false }));

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

//Start the server on PORT
app.listen(PORT, () => {
  console.log(`Server is started on port ${PORT}`);
});

```

2.5 Add custom error handler

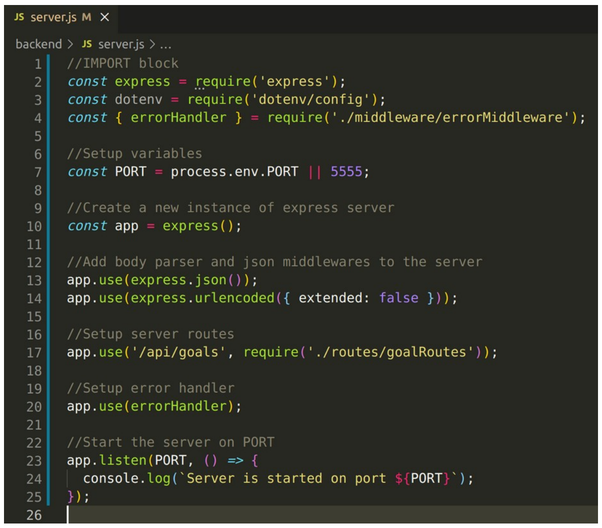
Create a new folder ./backend/middleware to store/work with different middlewares.

Create a new file ./backend/middleware/errorMiddleware.js to store/work with express error handler


JS errorMiddleware.js U ×

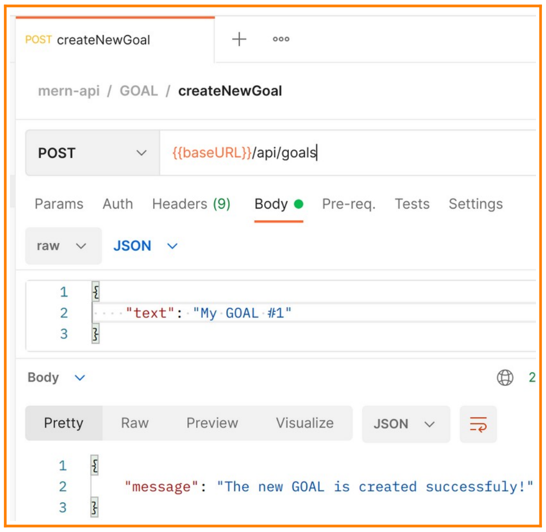
backend > middleware > JS errorMiddleware.js > ...

```javascript
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,
};

```


JS server.js M ×

backend > JS server.js > ...

```javascript
//IMPORT block
const express = require('express');
const dotenv = require('dotenv/config');
const { errorHandler } = require('./middleware/errorMiddleware');

//Setup variables
const PORT = process.env.PORT || 5555;

//Create a new instance of express server
const app = express();

//Add body parser and json middlewares to the server
app.use(express.json());
app.use(express.urlencoded({ extended: false }));

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

//Setup error handler
app.use(errorHandler);

//Start the server on PORT
app.listen(PORT, () => {
  console.log(`Server is started on port ${PORT}`);
});

```


POST createNewGoal

mern-api / GOAL / **createNewGoal**

POST {{baseURL}}/api/goals

Params Auth Headers (9) Body ● Pre-req. Tests Settings

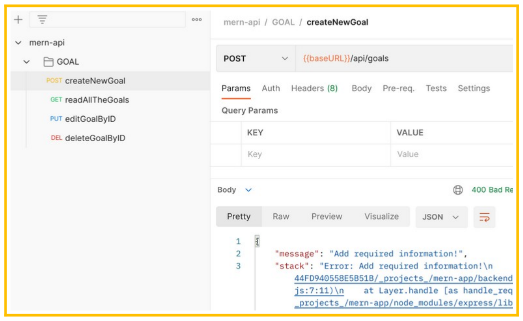
raw JSON

```json
{
    "text": "My GOAL #1"
}
```

Body 2

Pretty Raw Preview Visualize JSON

```json
{
    "message": "The new GOAL is created successfuly!"
}
```



mern-api / GOAL / **createNewGoal**

POST {{baseURL}}/api/goals

Params Auth Headers (8) Body Pre-req. Tests Settings

Query Params

KEY	VALUE
Key	Value

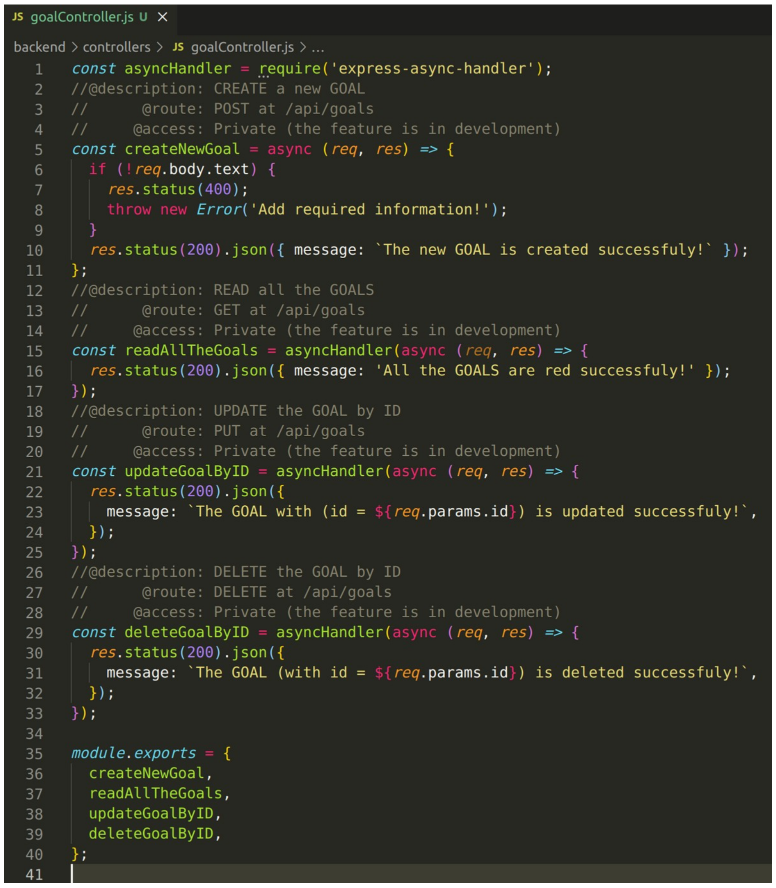
Body 400 Bad Re

Pretty Raw Preview Visualize JSON

```
{
    "message": "Add required information!",
    "stack": "Error: Add required information!\n
        44FD940558E5B51B/_projects_/mern-app/backend
        js:7:11)\n    at Layer.handle [as handle_req
        _projects_/mern-app/node_modules/express/lib
```

2.6 Add async/await syntax to our requests

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



```
const asyncHandler = require('express-async-handler');
//@description: CREATE a new GOAL
//      @route: POST at /api/goals
//     @access: Private (the feature is in development)
const createNewGoal = async (req, res) => {
  if (!req.body.text) {
    res.status(400);
    throw new Error('Add required information!');
  }
  res.status(200).json({ message: `The new GOAL is created successfuly!` });
};
//@description: READ all the GOALS
//      @route: GET at /api/goals
//     @access: Private (the feature is in development)
const readAllTheGoals = asyncHandler(async (req, res) => {
  res.status(200).json({ message: 'All the GOALS are red successfuly!' });
});
//@description: UPDATE the GOAL by ID
//      @route: PUT at /api/goals
//     @access: Private (the feature is in development)
const updateGoalByID = asyncHandler(async (req, res) => {
  res.status(200).json({
    message: `The GOAL with (id = ${req.params.id}) is updated successfuly!`,
  });
});
//@description: DELETE the GOAL by ID
//      @route: DELETE at /api/goals
//     @access: Private (the feature is in development)
const deleteGoalByID = asyncHandler(async (req, res) => {
  res.status(200).json({
    message: `The GOAL (with id = ${req.params.id}) is deleted successfuly!`,
  });
});

module.exports = {
  createNewGoal,
  readAllTheGoals,
  updateGoalByID,
  deleteGoalByID,
};

```

3. Frontend