



# (38) NodeJS and MongoDB Tutorial #4 - Uploading Files (Single & Multiple) - YouTube - Notes Export

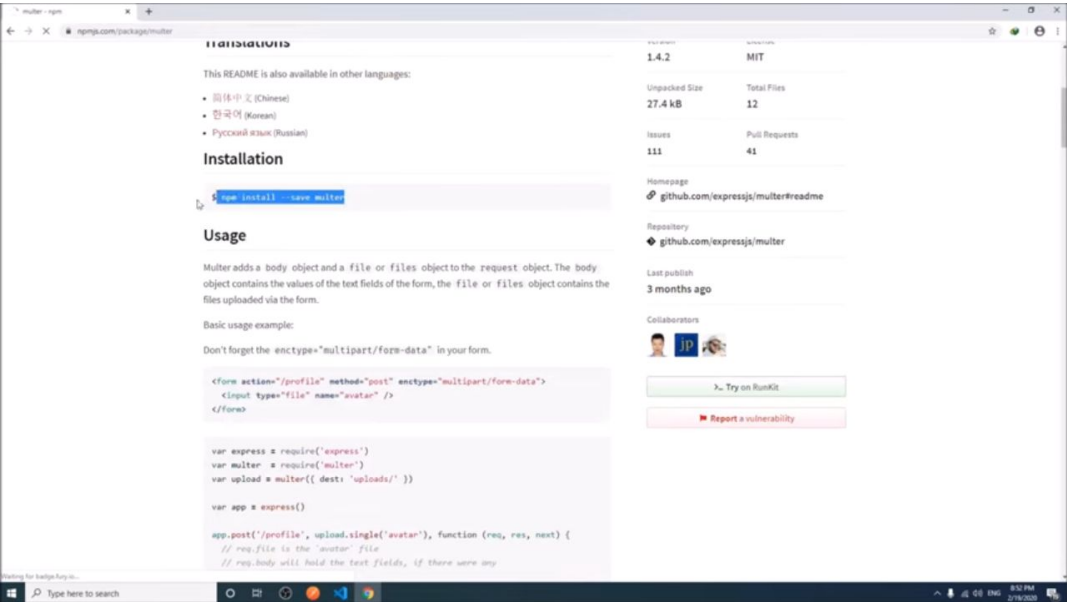
Generated on April 13, 2023

## Summary

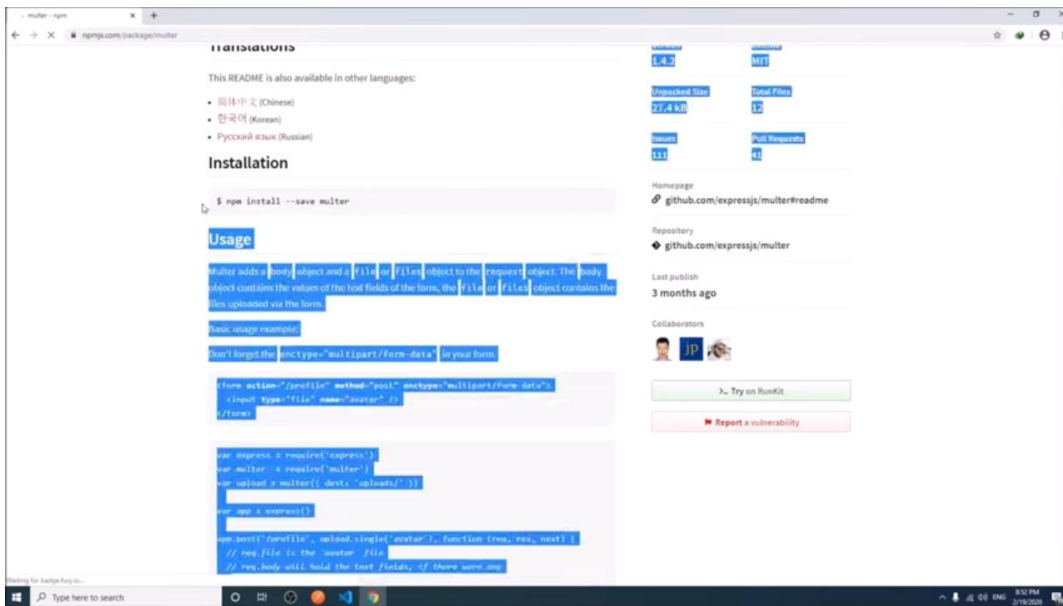
Notes	Screenshots
2	15

add feild to model

1:58



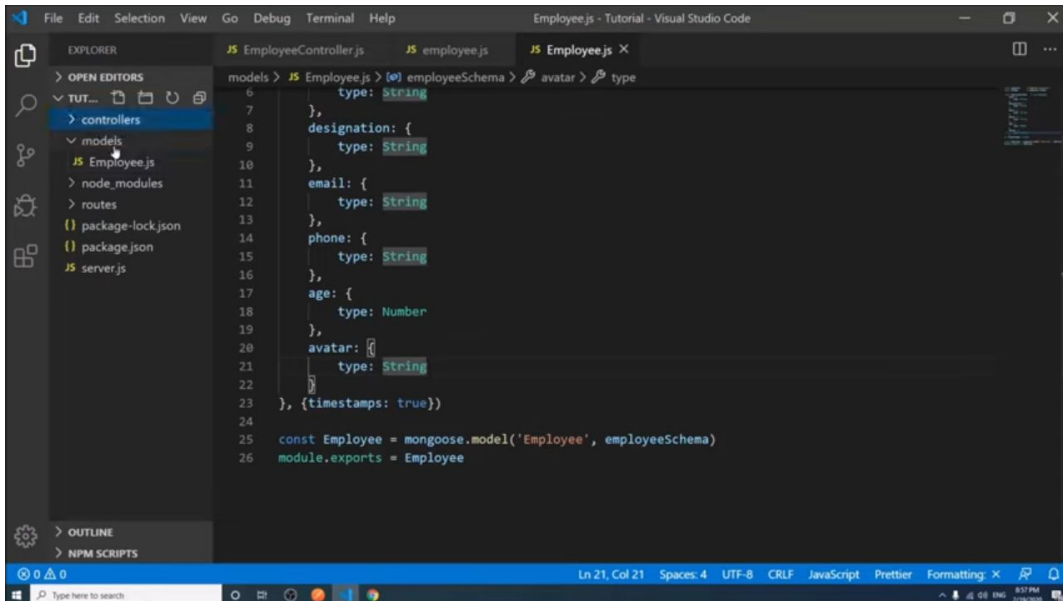
2:14



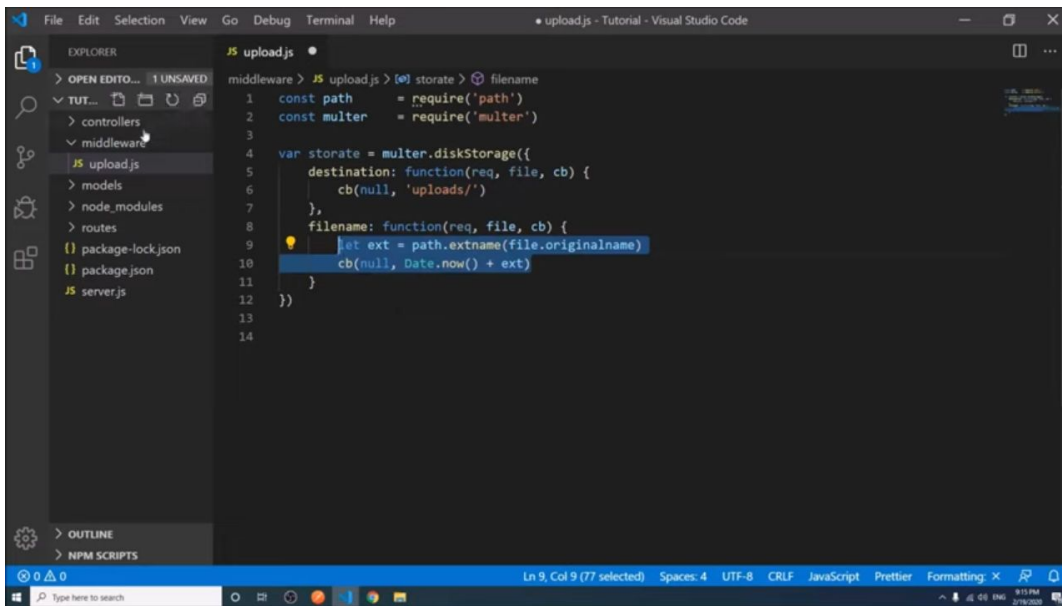
▶ 2:14

install multer

▶ 2:17

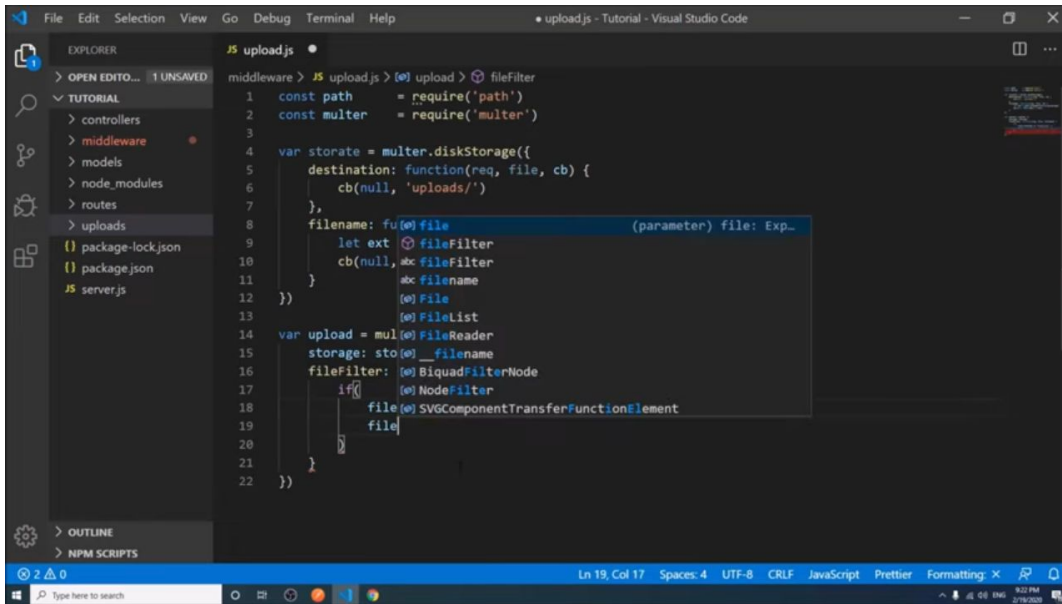


▶ 2:40



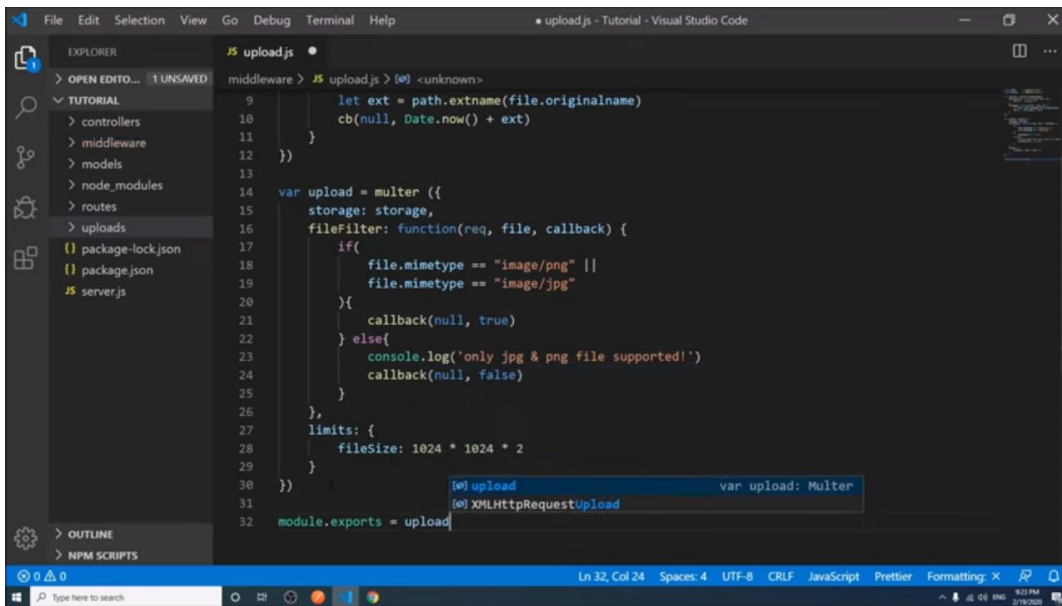
```
1 const path = require('path')
2 const multer = require('multer')
3
4 var storate = multer.diskStorage({
5   destination: function(req, file, cb) {
6     cb(null, 'uploads/')
7   },
8   filename: function(req, file, cb) {
9     let ext = path.extname(file.originalname)
10    cb(null, Date.now() + ext)
11  }
12 })
```

▶ 4:25



```
1 const path = require('path')
2 const multer = require('multer')
3
4 var storate = multer.diskStorage({
5   destination: function(req, file, cb) {
6     cb(null, 'uploads/')
7   },
8   filename: function(req, file, cb) {
9     let ext = path.extname(file.originalname)
10    cb(null, Date.now() + ext)
11  }
12 })
13
14 var upload = multer({
15   storage: storate,
16   fileFilter: function(req, file, cb) {
17     if (!file.mimetype.match(/\/(image|video)/)) {
18       cb(false, file, 'Not allowed')
19       return
20     }
21     cb(true, file)
22   }
23 })
```

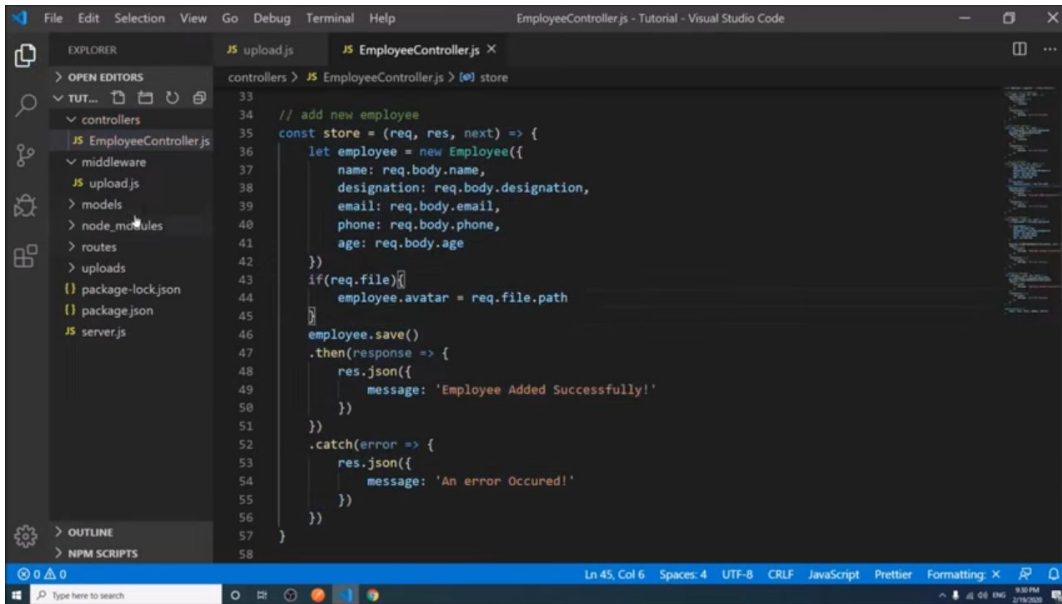
▶ 5:16



This screenshot shows the Visual Studio Code editor with a project named 'uploadjs - Tutorial'. The Explorer sidebar on the left shows a file tree with folders for 'controllers', 'middleware', 'models', 'node\_modules', 'routes', and 'uploads'. The 'uploads' folder is selected, showing files like 'package-lock.json', 'package.json', and 'server.js'. The main editor window displays the 'upload.js' file, which is currently empty. The status bar at the bottom indicates 'Ln 32, Col 24', 'Spaces: 4', 'UTF-8', 'CRLF', 'JavaScript', 'Prettier', and 'Formatting: X'.

```
9      let ext = path.extname(file.originalname)
10      cb(null, Date.now() + ext)
11    }
12  })
13
14  var upload = multer ({
15    storage: storage,
16    filefilter: function(req, file, callback) {
17      if(
18        file.mimetype == "image/png" ||
19        file.mimetype == "image/jpg"
20      ){
21        callback(null, true)
22      } else{
23        console.log('only jpg & png file supported!')
24        callback(null, false)
25      }
26    },
27    limits: {
28      fileSize: 1024 * 1024 * 2
29    }
30  })
31
32  module.exports = upload
```

▶ 6:26



This screenshot shows the Visual Studio Code editor with a project named 'EmployeeController.js - Tutorial'. The Explorer sidebar on the left shows a file tree with folders for 'controllers', 'middleware', 'models', 'node\_modules', 'routes', and 'uploads'. The 'controllers' folder is selected, showing files like 'EmployeeController.js', 'package-lock.json', 'package.json', and 'server.js'. The main editor window displays the 'EmployeeController.js' file, which contains code for adding a new employee. The status bar at the bottom indicates 'Ln 45, Col 6', 'Spaces: 4', 'UTF-8', 'CRLF', 'JavaScript', 'Prettier', and 'Formatting: X'.

```
33
34 // add new employee
35 const store = (req, res, next) => {
36   let employee = new Employee({
37     name: req.body.name,
38     designation: req.body.designation,
39     email: req.body.email,
40     phone: req.body.phone,
41     age: req.body.age
42   })
43   if(req.file){
44     employee.avatar = req.file.path
45   }
46   employee.save()
47   .then(response => {
48     res.json({
49       message: 'Employee Added Successfully!'
50     })
51   })
52   .catch(error => {
53     res.json({
54       message: 'An error Occured!'
55     })
56   })
57 }
58
```

▶ 6:59

VS Code interface showing the initial setup of the application. The Explorer sidebar shows the project structure with folders for controllers, middleware, routes, and uploads. The main editor shows the routes.js file with routes for show, store, update, and delete. The status bar at the bottom indicates 'Ln 8, Col 1 (47 selected)'.

7:01

VS Code interface showing the updated application. The Explorer sidebar shows the project structure with folders for controllers, middleware, routes, and uploads. The main editor shows the routes.js file with routes for show, store, update, and delete, including a new route for upload. The status bar at the bottom indicates 'Ln 9, Col 44'.

7:29

The screenshot shows the Visual Studio Code editor with the file explorer on the left. The file explorer shows a project structure with folders like 'controllers', 'middleware', 'upload.js', 'models', 'Employee.js', 'node\_modules', 'routes', 'employee.js', 'uploads', and files like 'package-lock.json', 'package.json', and 'server.js'. The 'server.js' file is open in the editor. The code in 'server.js' is as follows:

```
11 db.once('open', (err) => {
12   console.log(err)
13 })
14
15 db.once('open', () => {
16   console.log('Database Connection Established!')
17 })
18
19 const app = express()
20
21 app.use(morgan('dev'))
22 app.use(bodyParser.urlencoded({extended: true}))
23 app.use(bodyParser.json())
24 app.use('/uploads', express.static())
25 const PORT = process.env.PORT || 3000
26
27 app.listen(PORT, () => {
28   console.log('Server is running on port ${PORT}')
29 })
30
31 app.use('/api/employee', EmployeeRoute)
32
33
34
35
36
37
```

The code has a syntax error on line 24: `express.static()`. The error message in the bottom right corner says: `var e.static: typeof ...`. The status bar at the bottom shows 'Ln 24, Col 29', 'Spaces: 4', 'UTF-8', 'CRLF', 'JavaScript', 'Prettier', and 'Formatting: X'.

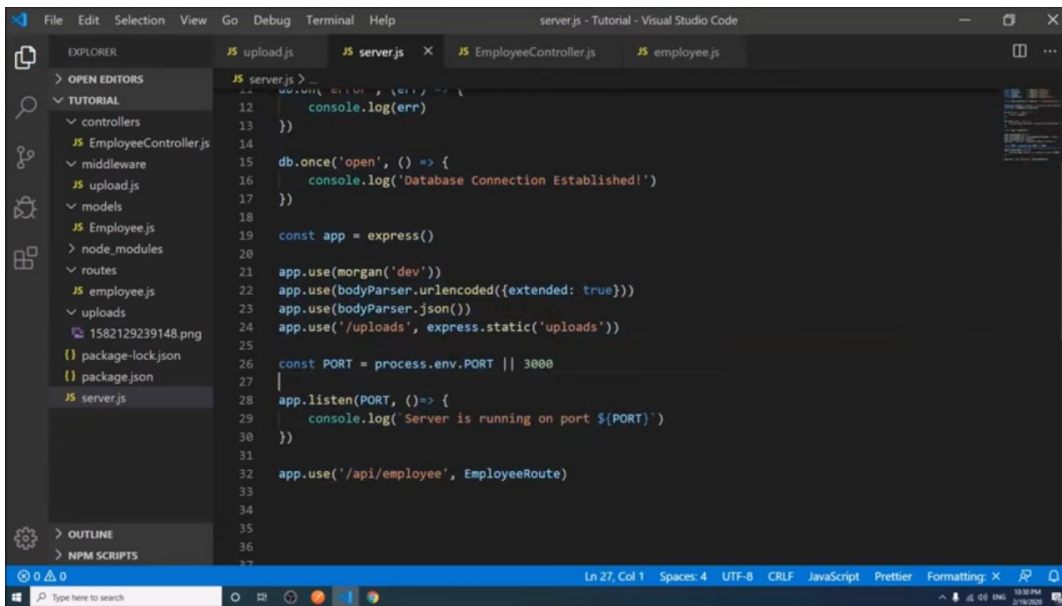
▶ 8:15

The screenshot shows the Visual Studio Code editor with the file explorer on the left. The file explorer shows a project structure with folders like 'controllers', 'middleware', 'upload.js', 'models', 'Employee.js', 'node\_modules', 'routes', 'employee.js', 'uploads', and files like 'package-lock.json', 'package.json', and 'server.js'. The 'server.js' file is open in the editor. The code in 'server.js' is as follows:

```
11 db.once('open', (err) => {
12   console.log(err)
13 })
14
15 db.once('open', () => {
16   console.log('Database Connection Established!')
17 })
18
19 const app = express()
20
21 app.use(morgan('dev'))
22 app.use(bodyParser.urlencoded({extended: true}))
23 app.use(bodyParser.json())
24 app.use('/uploads', express.static())
25 const PORT = process.env.PORT || 3000
26
27 app.listen(PORT, () => {
28   console.log('Server is running on port ${PORT}')
29 })
30
31 app.use('/api/employee', EmployeeRoute)
32
33
34
35
36
37
```

The code is now correct. The status bar at the bottom shows 'Ln 24, Col 37', 'Spaces: 4', 'UTF-8', 'CRLF', 'JavaScript', 'Prettier', and 'Formatting: X'.

▶ 8:30

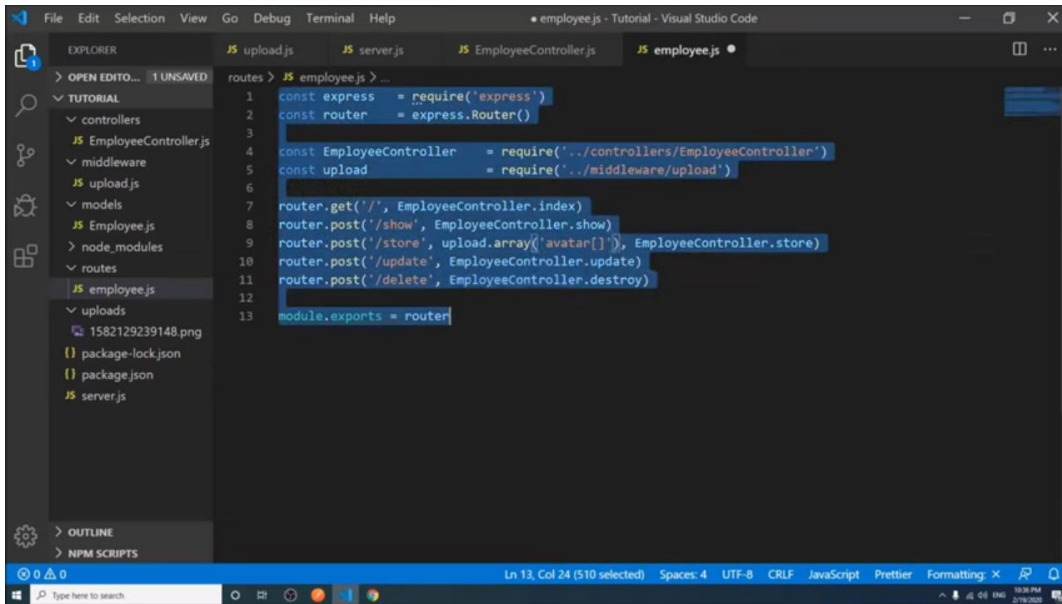


This screenshot shows the Visual Studio Code editor with the 'server.js' file open. The Explorer sidebar on the left shows a project structure with folders for 'controllers', 'middleware', 'models', 'routes', and 'uploads'. The main editor area displays the following JavaScript code:

```
11 db.once('open', (err) => {
12   console.log(err)
13 })
14
15 db.once('open', () => {
16   console.log('Database Connection Established!')
17 })
18
19 const app = express()
20
21 app.use(morgan('dev'))
22 app.use(bodyParser.urlencoded({extended: true}))
23 app.use(bodyParser.json())
24 app.use('/uploads', express.static('uploads'))
25
26 const PORT = process.env.PORT || 3000
27
28 app.listen(PORT, () => {
29   console.log(`Server is running on port ${PORT}`)
30 })
31
32 app.use('/api/employee', EmployeeRoute)
```

The status bar at the bottom indicates the cursor is at line 27, column 1, with 4 spaces, UTF-8 encoding, and CRLF line endings.

▶ 8:33



This screenshot shows the Visual Studio Code editor with the 'employee.js' file open. The Explorer sidebar on the left shows the same project structure as the previous screenshot. The main editor area displays the following JavaScript code:

```
1 const express = require('express')
2 const router = express.Router()
3
4 const EmployeeController = require('../controllers/EmployeeController')
5 const upload = require('../middleware/upload')
6
7 router.get('/', EmployeeController.index)
8 router.post('/show', EmployeeController.show)
9 router.post('/store', upload.array('avatar[]'), EmployeeController.store)
10 router.post('/update', EmployeeController.update)
11 router.post('/delete', EmployeeController.destroy)
12
13 module.exports = router
```

The status bar at the bottom indicates the cursor is at line 13, column 24 (510 selected), with 4 spaces, UTF-8 encoding, and CRLF line endings.

▶ 9:07



This screenshot shows the Visual Studio Code editor with the `EmployeeController.js` file open. The code defines a `store` function that creates a new `Employee` object and saves it. A code completion suggestion box is visible over the `path` variable on line 48, showing options like `path`, `Path2D`, and `SVGClipPathElement`. The Explorer sidebar on the left shows the project structure, including `controllers`, `middleware`, `upload.js`, `models`, `Employee.js`, `node_modules`, `routes`, `employee.js`, `uploads`, `1582129239148.png`, `package-lock.json`, `package.json`, and `server.js`. The status bar at the bottom indicates the cursor is at line 48, column 20.

```
35 const store = (req, res, next) => {
36   let employee = new Employee({
37     name: req.body.name,
38     designation: req.body.designation,
39     email: req.body.email,
40     phone: req.body.phone,
41     age: req.body.age
42   })
43   if(req.files) {
44     let path = ''
45     req.files.forEach(function(files, index, arr){
46       path = path + files.path + ','
47     })
48     path = path
49     // let path: string
50   }
51   employee.save()
52   .then(response => {
53     res.json({
54       message: 'Employee Added Successfully!'
55     })
56   })
57   .catch(error => {
58     res.json({
59       message: 'An error Occured!'
60     })
61   })
62 }
```

▶ 10:01

This screenshot shows the same Visual Studio Code editor with `EmployeeController.js`. The code has been updated to use `path.substring(0, path.lastIndexOf(','))` to extract the file path from the comma-separated string. The Explorer sidebar and status bar are the same as in the previous screenshot. The status bar now indicates the cursor is at line 48, column 9, with 47 characters selected.

```
35 const store = (req, res, next) => {
36   let employee = new Employee({
37     name: req.body.name,
38     designation: req.body.designation,
39     email: req.body.email,
40     phone: req.body.phone,
41     age: req.body.age
42   })
43   if(req.files) {
44     let path = ''
45     req.files.forEach(function(files, index, arr){
46       path = path + files.path + ','
47     })
48     path = path.substring(0, path.lastIndexOf(','))
49     employee.avatar = path
50   }
51   employee.save()
52   .then(response => {
53     res.json({
54       message: 'Employee Added Successfully!'
55     })
56   })
57   .catch(error => {
58     res.json({
59       message: 'An error Occured!'
60     })
61   })
62 }
```

▶ 10:39