

Exercise (Instructions): Uploading Files

Objectives and Outcomes

In this exercise you will leverage the Multer module to set up your Express server to accept file uploads. At the end of this exercise you will be able to:

- Configure the Multer module to enable file uploads
- Use the Multer module in your Express application to enable file uploads to a designated folder

Installing Multer

- At the prompt in your conFusionServer project, type the following to install Multer:

```
1  npm install multer@1.3.1 --save
```

Setting up File Uploading

- Add a new Express router named *uploadRouter.js* in the routes folder and add the following code to it:

```

1  const express = require('express');
2  const bodyParser = require('body-parser');
3  const authenticate = require('../authenticate');
4  const multer = require('multer');
5
6  const storage = multer.diskStorage({
7    destination: (req, file, cb) => {
8      cb(null, 'public/images');
9    },
10
11    filename: (req, file, cb) => {
12      cb(null, file.originalname)
13    }
14  });
15
16  const imageFileFilter = (req, file, cb) => {
17    if(!file.originalname.match(/\.(jpg|jpeg|png|gif)$/)) {
18      return cb(new Error('You can upload only image files!'), false);
19    }
20    cb(null, true);
21  };
22
23  const upload = multer({ storage: storage, fileFilter: imageFileFilter });
24
25  const uploadRouter = express.Router();
26
27  uploadRouter.use(bodyParser.json());
28
29  uploadRouter.route('/')
30    .get(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {
31      res.statusCode = 403;
32      res.end('GET operation not supported on /imageUpload');
33    })
34    .post(authenticate.verifyUser, authenticate.verifyAdmin, upload.single
35      ('imageFile'), (req, res) => {
36      res.statusCode = 200;
37      res.setHeader('Content-Type', 'application/json');
38      res.json(req.file);
39    })
40    .put(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {
41      res.statusCode = 403;
42      res.end('PUT operation not supported on /imageUpload');
43    })
44    .delete(authenticate.verifyUser, authenticate.verifyAdmin, (req, res, next) => {
45      res.statusCode = 403;
46      res.end('DELETE operation not supported on /imageUpload');
47    });
48  module.exports = uploadRouter;

```

- Then update app.js to import the uploadRouter and configure the new route as follows:

```

1  . . .
2
3  const uploadRouter = require('../routes/uploadRouter');
4
5
6  . . .
7
8  app.use('/imageUpload', uploadRouter);
9
10 . . .

```

- Save all the changes and test your server.
- Do a Git commit with the message "File Upload".

Conclusions

In this exercise you learnt to leverage the Multer module to configure your Express server to enable file uploads.