Using multer for File Uploads in Express.js

multer is a middleware for handling **multipart/form-data**, primarily used for file uploads in **Express.js**.

1. Install multer

```
npm install multer
```

2. Basic File Upload Example

Step 1: Import and Configure multer

```
const express = require('express');
const multer = require('multer');
const path = require('path');
const app = express();
// Configure multer storage
const storage = multer.diskStorage({
    destination: './uploads/', // Destination folder
    filename: (req, file, cb) => {
        cb(null, file.fieldname + '-' + Date.now() +
path.extname(file.originalname));
    }
});
// Initialize multer with storage
const upload = multer({ storage: storage });

app.use(express.static('uploads')); // Serve uploaded files
```

Step 2: Create Upload Route

```
app.post('/upload', upload.single('file'), (req, res) => {
    if (!req.file) {
        return res.status(400).json({ message: 'No file uploaded' });
    }
    res.json({ message: 'File uploaded successfully', file: req.file });
});
```

➢ **Example Form Request (Using Postman)**

- **Method:** POST

- **URL:** http://localhost:3000/upload
- **Body:** Select **form-data**, key = "file", and choose a file.

➢ **Example Response**

```
{
   "message": "File uploaded successfully",
   "file": {
      "fieldname": "file",
      "originalname": "image.png",
      "encoding": "7bit",
      "mimetype": "image/png",
      "destination": "./uploads/",
      "filename": "file-1700000000.png",
      "path": "uploads/file-1700000000.png",
      "size": 102400
   }
}
```

---

3. Uploading Multiple Files

Modify the upload middleware:

```
app.post('/uploads', upload.array('files', 5), (req, res) => {
   if (!req.files) {
      return res.status(400).json({ message: 'No files uploaded' });
   }
   res.json({ message: 'Files uploaded successfully', files: req.files });
});
```

➢ Allows uploading **up to 5 files** at once.

---

4. File Type & Size Validation

```
const fileFilter = (req, file, cb) => {
   const allowedTypes = /jpeg|jpg|png|gif/;
   const extname =
allowedTypes.test(path.extname(file.originalname).toLowerCase());
   const mimetype = allowedTypes.test(file.mimetype);
   if (extname && mimetype) {
      return cb(null, true);
   } else {
      return cb(new Error('Only images are allowed!'), false);
   }
};
const uploadWithValidation = multer({
    storage: storage,
    limits: { fileSize: 1 * 1024 * 1024 }, // Limit: 1MB
    fileFilter: fileFilter
});
```

Use in Route

```
app.post('/upload-image', uploadWithValidation.single('image'), (req, res) => {
    res.json({ message: 'Image uploaded successfully', file: req.file });
});
```

- **Limits file size to 1MB** and allows only images.

---

5. Storing Files in Memory (Buffer)

Instead of saving files to disk, store them in memory:

```
  const memoryUpload = multer({ storage:
  multer.memoryStorage() });
  app.post('/upload-memory', memoryUpload.single('file'), (req,
  res) => {
res.json({ message: 'File stored in memory', buffer:
  req.file.buffer });
  });
```

- Useful when processing files before saving (e.g., **resize images**).

---

6. Summary

Feature	Code Example
Install multer	npm install multer
Upload Single File	upload.single('file')
Upload Multiple Files	upload.array('files', 5)
Validate File Type	fileFilter function
Limit File Size	limits: { fileSize: 1MB }
Store in Memory (Buffer)	multer.memoryStorage()

Would you like an example with **image processing (Sharp)**?