File Uploader

File Uploader

Build a full-stack file upload and management system.

Overview

Attribute	Value
Level	Intern
Stack	Fullstack
Time Limit	4-5 hours
Difficulty	Easy-Medium

Problem Statement

Create a file upload service where users can upload files, view them, and manage uploads. Build both the API and React frontend.

Requirements

Backend Features

- [ ] Upload single file
- [ ] Upload multiple files
- [ ] List all uploads
- [ ] Get file details
- [ ] Download file
- [ ] Delete file
- [ ] File type validation

Frontend Features

- [ ] Drag & drop upload area
- [ ] File selection dialog
- [ ] Upload progress indicator
- [ ] File list with previews
- [ ] Download/Delete actions

API Specification

Upload File

```
POST /api/files
Content-Type: multipart/form-data
```

Request: Form data with file field

Response (201 Created):

```
{
  "id": "file_123",
  "originalName": "document.pdf",
  "filename": "1702293000000-document.pdf",
  "mimeType": "application/pdf",
  "size": 245678,
  "url": "/api/files/file_123/download",
  "uploadedAt": "2025-12-11T10:30:00Z"
}
```

Upload Multiple Files

```
POST /api/files/batch
Content-Type: multipart/form-data
```

Response:

```
{
  "files": [
    { "id": "file_123", "originalName": "doc1.pdf", ... },
    { "id": "file_124", "originalName": "doc2.pdf", ... }
  ],
  "count": 2
}
```

List All Files

```
GET /api/files
```

Response:

```
{
  "files": [
    {
      "id": "file_123",
      "originalName": "document.pdf",
      "mimeType": "application/pdf",
      "size": 245678,
      "uploadedAt": "2025-12-11T10:30:00Z"
    }
  ],
  "total": 15,
  "totalSize": "45.2 MB"
}
```

Get File Details

```
GET /api/files/:id
```

Download File

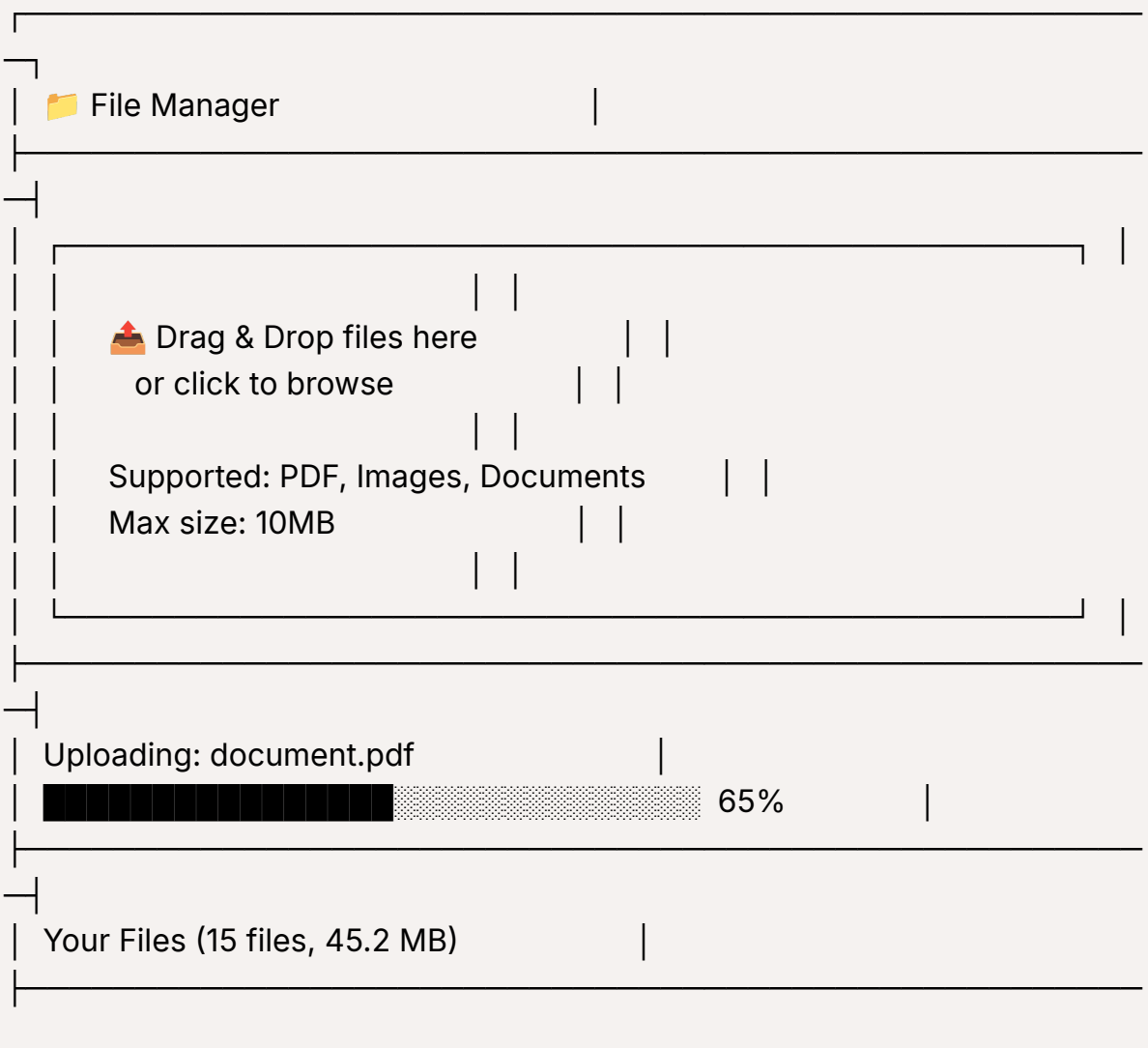
```
GET /api/files/:id/download
```

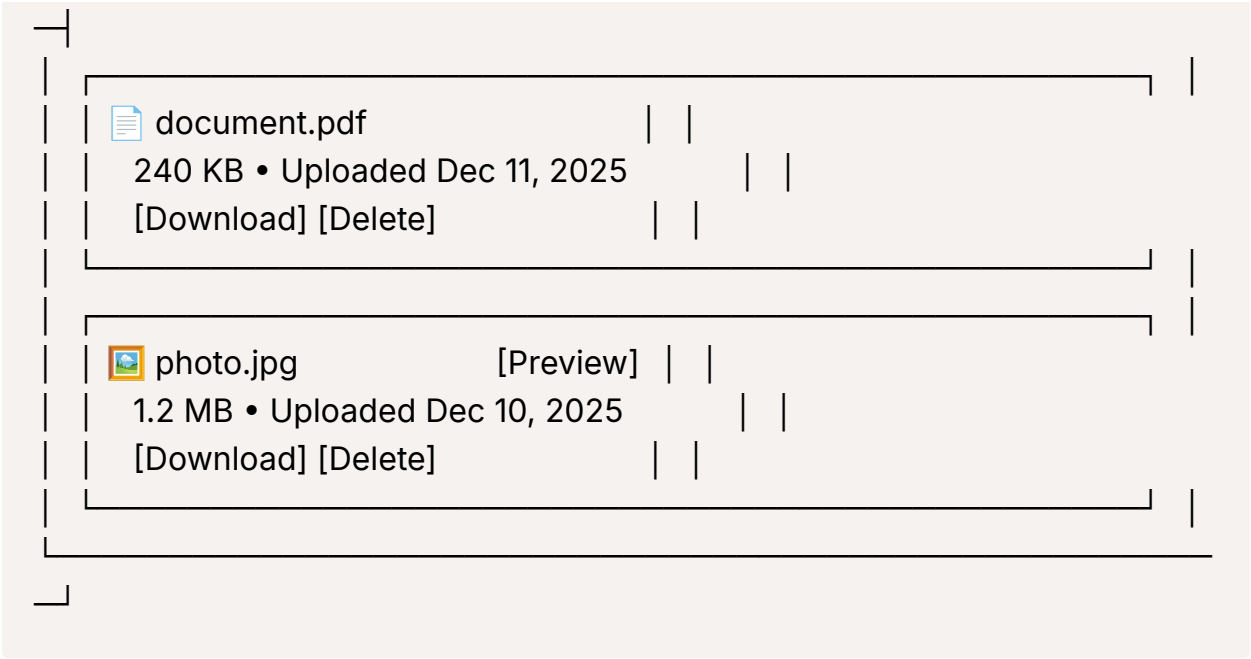
Returns file with appropriate Content-Type header.

Delete File

```
DELETE /api/files/:id
```

Wireframe





Getting Started

Backend

```
mkdir file-uploader && cd file-uploader
mkdir backend && cd backend
npm init -y
npm install express cors multer
mkdir uploads
```

Frontend

```
cd ..
npm create vite@latest frontend -- --template react
cd frontend
npm install axios
```

Multer Configuration

```
const multer = require('multer');
const path = require('path');

const storage = multer.diskStorage({
  destination: './uploads',
  filename: (req, file, cb) => {
    const uniqueName = `${Date.now()}-${file.originalname}`;
    cb(null, uniqueName);
  }
});

const fileFilter = (req, file, cb) => {
  const allowedTypes = [
    'image/jpeg',
    'image/png',
    'image/gif',
    'application/pdf',
    'text/plain'
  ];

  if (allowedTypes.includes(file.mimetype)) {
    cb(null, true);
  } else {
    cb(new Error('Invalid file type'), false);
  }
};

const upload = multer({
  storage,
  fileFilter,
  limits: { fileSize: 10 * 1024 * 1024 } // 10MB
});
```

Data Model

```
// File
{
  id: "file_123",
  originalName: "document.pdf",
  filename: "1702293000000-document.pdf",
  mimeType: "application/pdf",
  size: 245678,
  path: "./uploads/1702293000000-document.pdf",
  uploadedAt: Date
}
```

Frontend Upload

```
async function uploadFile(file, onProgress) {
  const formData = new FormData();
  formData.append('file', file);

  const response = await axios.post('/api/files', formData, {
    headers: { 'Content-Type': 'multipart/form-data' },
    onUploadProgress: (progressEvent) => {
      const percent = Math.round(
        (progressEvent.loaded * 100) / progressEvent.total
      );
      onProgress(percent);
    }
  });

  return response.data;
}
```

Evaluation Rubric

Criteria	Weight	Excellent (5)	Good (4)	Acceptable (3)
Backend API	30%	Full API + validation	Basic upload/list	Upload only
Frontend UI	30%	Drag/drop + progress	Basic form	Form only
Integration	20%	Smooth, error handling	Works	Basic
Code Quality	20%	Clean both sides	Organized	Works

Bonus Points

- [ ] Image thumbnail generation
- [ ] File preview in modal
- [ ] Folders/organization
- [ ] File sharing links

Do not share this problem publicly.