**Materials Management System Explanation**

**Core Features**

**1. File & Folder Management**

* **Create Folders**: Create directory structures
* **Upload Files**: Support file uploads up to 10MB
* **Delete**: Remove files/folders with recursive deletion
* **List Contents**: View files/folders in a directory

**2. Hierarchical Structure**

Course/

  └── Assignments/

       ├── Week1/

       │    ├── assignment1.pdf

       │    └── notes.txt

       └── Week2/

            └── project.zip

**3. File Properties**

* Name
* Type (file/folder)
* Path
* File Size (for files only)
* MIME Type (for files only)
* Creation Date
* Parent Folder Reference
* Course Association

**4. Key API Endpoints**

// 1. Upload/Create Material

POST /api/materials

Body: {

  course: "courseId",

  name: "filename.pdf",

  type: "file", // or "folder"

  parentPath: "Assignments/Week1"

}

+ file attachment

// 2. List Materials

GET /api/materials/:courseId?parentPath=Assignments/Week1

// 3. Download File

GET /api/materials/:fileId

// 4. Delete Material

DELETE /api/materials/:id

// 5. Update Material

PATCH /api/materials/:id

**5. Safety Features**

1. **File Size Limits**:

const MAX\_FILE\_SIZE = 10 \* 1024 \* 1024; // 10MB limit

1. **Path Validation**:

// Prevents directory traversal attacks

const normalizedPath = path.normalize(userProvidedPath);

1. **Parent Folder Checks**:

// Ensures parent folders exist before creating new items

if (!parentFolder) {

  return error("Parent folder not found");

}

**6. Convenience Features**

1. **Human Readable File Sizes**:

// Converts 1048576 bytes to "1 MB"

materialSchema.virtual('humanSize')

1. **Automatic File Extensions**:

// Adds file extension if missing

if (!path.extname(name)) {

  name += fileExtension;

}

1. **File Streaming**:

// Streams large files instead of loading into memory

const stream = fs.createReadStream(filePath);

stream.pipe(response);

**7. Database Schema**

Material {

  name: String,          // "assignment1.pdf"

  type: String,          // "file" or "folder"

  path: String,          // "Assignments/Week1/assignment1.pdf"

  filePath: String,      // Actual filesystem path

  size: Number,          // File size in bytes

  mimeType: String,      // "application/pdf"

  course: ObjectId,      // Reference to course

  parentFolder: ObjectId // Reference to parent folder

  createdAt: Date       // Creation timestamp

}

**8. Error Handling**

* Missing required fields
* File size limits
* Non-existent folders
* Invalid file types
* File system errors
* Database errors

**9. File Organization**

* Files are stored in an [uploads](vscode-file://vscode-app/c:/Users/sadap/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) directory
* Maintains folder hierarchy matching the virtual structure
* Automatically creates necessary directories
* Cleans up files when deleted

This system provides a complete file management solution for courses, with proper security, error handling, and user-friendly features.