[1] User (Student/Tutor)

|

| Upload Request

v

[2] Frontend App (Browser)

|

| -> Request Pre-signed URL

v

[3] Backend Server (Node.js / Python)

|

| -> Generate S3 Pre-signed Upload URL

v

[4] Frontend receives Pre-signed URL

|

| -> Upload file directly to S3

v

[5] AWS S3 Bucket (Files Stored)

|

| <- Upload Success Callback (optional)

v

[6] Backend Receives Upload Metadata

|

| -> Store file metadata (owner ID, path, type, etc.) in DB

|

| -> [OPTIONAL] Push “Generate Thumbnail” task to Queue

v

[7] Background Worker / Thumbnail Processor

|

| -> Pull file from S3 using metadata

| -> Detect file type (image/video/docx/pdf)

| -> Extract first page/frame

| -> Convert to JPEG

| -> Compress image

| -> Encode as Base64 string

v

[8] Save Thumbnail (Base64) in Database

|

| (Linked to file's metadata via userID + fileID)

v

[9] Frontend Requests File List

|

| -> Backend queries DB for metadata + base64 thumbnails

v

[10] Frontend receives:

- File Names

- Base64 Thumbnails

- File Types (optional)

|

v

[11] Frontend Renders: Thumbnail + Filename Grid View

|

✅ User sees preview of their files

**🧠 Highlights:**

* Only **original file** is stored in S3.
* **Thumbnails** are small and fast to load, stored in DB.
* **Queue and Worker** can be added later (right now you can simulate with direct function call).
* This structure supports scalability + async later.