# **Simple Security Overview — Secure File Sharing (Node)**

**Scope:** Lightweight Node.js app that lets users upload files, stores them **encrypted at rest**, and lets them download (decrypt) later.

## 1) What the app does

* **Upload:** Browser sends a file to /upload.
* **Store:** Server encrypts the file and saves the ciphertext in storage/ with a unique ID.
* **List:** /files returns a simple list (no sensitive crypto details).
* **Download:** /download/:id streams the file back **after decryption**.

## 2) Encryption at rest

* **Algorithm:** AES‑256‑GCM (gives confidentiality + integrity).
* **Per‑file IV:** A fresh random IV is generated for every file.
* **Integrity tag:** Each file gets an auth tag; if data is changed, decryption fails.

## 3) Key handling

* **One server secret:** A passphrase from environment (SECRET\_PASSPHRASE).
* **Derived key:** We derive a 256‑bit key with scrypt (slow KDF) and keep it **in memory only** at runtime.
* **Where secrets live:** In environment variables/secret store (never committed to Git).
* **Rotation (basic):** Change SECRET\_PASSPHRASE and redeploy; new uploads use the new key.
  + *(Optional later: envelope encryption for per‑file keys if we need easier rotations.)*

## 4) What we store

* **Ciphertext file** (storage/<uuid>.enc).
* **Metadata** (in index.json): id, original\_name, size\_bytes, created\_at, iv, tag, path.
* The public list **hides** iv, tag, and file path.

## 5) Upload rules

* **Max size:** 25 MB per file.
* **Allowed types:** .pdf, .png, .jpg, .jpeg, .txt, .doc, .docx, .xlsx, .zip, .csv.
* **Safety:** Reject path traversal and sanitize download filenames.