Functional Requirement Document (FRD)

**Project:** File Management API

**Version:** 1.0

**Date:** 2025-06-04

**Author:** Functional Analyst Team

### **1. Document Purpose**

เอกสารนี้จัดทำขึ้นเพื่อระบุความต้องการทางฟังก์ชันของระบบ API สำหรับให้บริการการแสดงรายการไฟล์ (List Files) และดาวน์โหลดไฟล์ (Download File) ซึ่งใช้สำหรับระบบ backend-to-backend ภายในองค์กรเท่านั้น

### **2. Scope**

#### **2.1 In Scope**

* RESTful API สำหรับเรียกรายการไฟล์ (List Files)
* RESTful API สำหรับดาวน์โหลดไฟล์ (Download File)
* ใช้งานในลักษณะ Server-to-Server (Internal backend communication)

#### **2.2 Out of Scope**

* UI สำหรับผู้ใช้งาน
* การอัปโหลดไฟล์

### **3. Functional Requirements**

#### **3.1 List Files API**

* ระบบต้องสามารถให้บริการ endpoint /files ด้วย method GET เพื่อแสดงรายการไฟล์ทั้งหมดที่มีอยู่
* ระบบต้องสามารถรองรับ query parameters เช่น page, size, sort ได้
* ต้องรองรับการ filter เช่น fileType, createdAfter, createdBefore
* ต้องแสดงข้อมูลไฟล์: id, filename, size, contentType, createdAt
* ต้องตรวจสอบ JWT Token ที่ส่งมาใน HTTP Header เพื่อยืนยันสิทธิ์
* ทุกการเรียก /files ต้องบันทึก audit log

**3.2 Download File API**

* ระบบต้องสามารถให้บริการ endpoint /files/{id}/download ด้วย method GET
* ระบบต้องตรวจสอบว่า JWT Token ที่ได้รับมีสิทธิ์ดาวน์โหลดไฟล์
* ถ้าไม่พบไฟล์ ให้ตอบกลับ HTTP 404
* ถ้าผู้ใช้ไม่มีสิทธิ์ ให้ตอบกลับ HTTP 403
* ต้อง log ข้อมูลการดาวน์โหลด เช่น fileId, userId, timestamp, result
* ไฟล์จะถูกส่งกลับในรูปแบบ binary stream (Content-Disposition: attachment)

**4. Security Requirements**

* ต้องมีการตรวจสอบ JWT Token ใน Authorization Header
* ต้องรองรับเฉพาะ HTTPS (TLS 1.2 ขึ้นไป)
* ต้องมีการจำกัด IP ให้เฉพาะ internal subnet ที่อนุญาตเท่านั้น
* ต้องมี Rate Limiting ต่อ client
* JWT ต้องมี scope: file.read, file.download

**5. Performance Requirements**

* คำขอ /files ต้องตอบกลับภายใน 1 วินาที
* รองรับ Concurrent Requests ได้อย่างน้อย 500 requests
* ไฟล์ขนาดไม่เกิน 50MB ต้องดาวน์โหลดได้ภายใน 5 วินาที

**6. Logging & Audit Trail**

| **Event** | **Fields** |
| --- | --- |
| List Files | timestamp, userId, action: LIST, filters, resultCount |
| Download File | timestamp, userId, fileId, action: DOWNLOAD, result: success/fail, IP address |

* Audit logs ต้องถูกส่งไปยังระบบ Central Logging และเก็บไว้อย่างน้อย 180 วัน

**7. API Summary**

#### **7.1 List Files**

* Method: GET
* Endpoint: /files
* Query Parameters (optional): page, size, sort, fileType, createdAfter, createdBefore
* Response:

| **[**  **{**  **"id": "uuid",**  **"filename": "example.pdf",**  **"size": 102400,**  **"contentType": "application/pdf",**  **"createdAt": "2025-05-01T12:00:00Z"**  **}**  **]** |
| --- |

#### **7.2 Download File**

* Method: GET
* Endpoint: /files/{id}/download
* Headers: Authorization: Bearer <token>
* Response: Binary file stream with Content-Disposition: attachment

**8. Use Case Flows**

### **8.1 Use Case: แสดงรายการไฟล์ (List Files)**

**Actor:**

* ระบบผู้ร้องขอ (Consumer System)

**Trigger:**

* ผู้ใช้ต้องการเรียกดูรายการไฟล์ที่สามารถดาวน์โหลดได้

**Main Flow:**

1. ผู้ใช้ระบบฝั่ง consumer เรียก API GET /files พร้อม JWT token
2. ระบบ provider ตรวจสอบความถูกต้องของ token
3. ระบบดึงข้อมูลรายการไฟล์จากฐานข้อมูลหรือ storage
4. ระบบส่ง response กลับเป็น JSON รายการไฟล์

**Alternative Flow:**

* 8.1.1: หาก token ไม่ถูกต้อง ระบบส่งกลับ HTTP 401 Unauthorized
* 8.1.2: หากระบบไม่สามารถดึงข้อมูลได้ ระบบส่งกลับ HTTP 500 Internal Server Error

### **8.2 Use Case: ดาวน์โหลดไฟล์ (Download File)**

**Actor:**

* ระบบผู้ร้องขอ (Consumer System)

**Trigger:**

* ผู้ใช้ต้องการดาวน์โหลดไฟล์ตาม ID ที่เลือก

**Main Flow:**

1. ผู้ใช้ระบบฝั่ง consumer เรียก API GET /files/{id}/download พร้อม JWT token
2. ระบบ provider ตรวจสอบความถูกต้องของ token
3. ระบบค้นหาไฟล์จาก ID ที่ร้องขอ
4. ระบบส่งไฟล์เป็น binary stream พร้อม HTTP header ที่เหมาะสม

**Alternative Flow:**

* 8.2.1: หากไม่พบไฟล์ ระบบส่งกลับ HTTP 404 Not Found
* 8.2.2: หาก token ไม่ถูกต้อง ระบบส่งกลับ HTTP 401 Unauthorized

### **8.3 Use Case: ดาวน์โหลดไฟล์แบบมีรหัสผ่าน (Download File with Password Protection)**

**Actor:**

* ระบบผู้ร้องขอ (Consumer System)

**Trigger:**

* ผู้ใช้ต้องการให้ระบบ provider สร้างไฟล์ที่มีการป้องกันด้วยรหัสผ่านก่อนดาวน์โหลด

**Main Flow:**

1. ผู้ใช้ระบบฝั่ง consumer เรียก API POST /files/{id}/download/encrypted พร้อม JWT token และระบุรหัสผ่านใน request body
2. ระบบ provider ตรวจสอบความถูกต้องของ token
3. ระบบตรวจสอบความแข็งแรงของรหัสผ่านที่ให้มา (เช่น ความยาวขั้นต่ำ หรือ pattern)
4. ระบบค้นหาไฟล์ที่เกี่ยวข้อง
5. ระบบเข้ารหัสไฟล์ (PDF หรือ ZIP) ด้วยรหัสผ่าน
6. ระบบส่งไฟล์ที่ถูกเข้ารหัสกลับไปยัง consumer

**Alternative Flow:**

* 8.3.1: หากรหัสผ่านไม่ผ่านเกณฑ์ ระบบส่ง HTTP 400 Bad Request
* 8.3.2: หากไม่พบไฟล์ ระบบส่ง HTTP 404 Not Found
* 8.3.3: หาก token ไม่ถูกต้อง ระบบส่ง HTTP 401 Unauthorized
* 8.3.4: หากการเข้ารหัสล้มเหลว ระบบส่ง HTTP 500 Internal Server Error

**Diagram:**

| //goto <https://sequencediagram.org> for preview  title ดาวน์โหลดไฟล์พร้อมการตั้งรหัสผ่าน  participant "Consumer System" as C  participant "API Gateway" as API  participant "Auth Service" as Auth  participant "File Service" as FS  participant "Encryption Module" as Enc  C->API: POST /files/{id}/download/encrypted\nพร้อม token และรหัสผ่าน  activate API  API->Auth: ตรวจสอบ token  activate Auth  Auth-->API: ผลการตรวจสอบ token  deactivate Auth  alt token ถูกต้อง  API->FS: ดึงไฟล์ตาม ID  activate FS  FS-->API: ส่งไฟล์กลับ  deactivate FS  API->Enc: เข้ารหัสไฟล์ด้วยรหัสผ่าน  activate Enc  Enc-->API: ส่งไฟล์ที่เข้ารหัสแล้ว  deactivate Enc  API-->C: ส่งไฟล์ที่เข้ารหัสแล้ว  else token ไม่ถูกต้อง  API-->C: HTTP 401 Unauthorized  end  deactivate API |
| --- |

**9. Non-Functional Requirements**

* รองรับไฟล์ขนาดใหญ่ (streaming)
* ระบบต้องพร้อมใช้งาน 99.9%
* ต้องมี health-check endpoint สำหรับใช้กับระบบ monitoring
* ต้องสามารถ scale ได้แนวนอน (horizontal scaling)

### **10. Assumptions & Constraints**

* ระบบภายนอกที่เรียก API นี้ต้องมีการลงทะเบียนและรับ JWT จาก Auth Service
* การเข้าถึงไฟล์ต้องตรวจสอบสิทธิ์ระดับ role หรือ scope