คณะศิลปศาสตร์และวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน (FLAS KPS KU)

ส่วนที่ 1: รายงานสรุปภาพรวมระบบ กลไกการทำงาน และสิ่งที่ต้องพัฒนาเพิ่ม

FLAS KPS KU E-OFFICE + (Loki HR & Circular Document Management System)

สถานะ: **v2.4 (Production Ready)** วันที่: **27 สิงหาคม 2569** เทคโนโลยี: **Next.js 14 + PostgreSQL 16** การควบคุม: **Docker + GitHub Actions CI/CD**

🚀 1. สรุปภาพรวมว่าระบบได้รับการอัปเดตอะไรบ้าง (What Was Updated)

🏛️ โครงสร้างองค์กรและสายบังคับบัญชา 4 ลำดับชั้น

- **Tier 1 (ผู้บริหารระดับคณะ):** คณบดี, รองคณบดีฝ่ายวิชาการ, ฝ่ายบริหาร ฯลฯ
- **Tier 2 (หัวหน้าภาควิชา & หัวหน้างาน):** หน.ภาควิชา CS/IT, วิทย์กายภาพ, ชีววิทยา, หน.สำนักงานเลขานุการ
- **Tier 3 (อาจารย์ประจำ):** อาจารย์ประจำสาขา CS, IT, เคมี ฯลฯ
- **Tier 4 (สายสนับสนุน):** เจ้าหน้าที่สารบรรณ, การเงินและพัสดุ, บริการการศึกษา

🛡️ ความปลอดภัยระดับองค์กร (Security Hardening & Zero-Trust)

- **ถอด Quick User Switcher:** ตัดเมนูสลับตัวตนจากหน้าจอ บังคับตรวจสอบสิทธิ์ผ่านระบบล็อกอินจริง
- **2-Factor Authentication (2FA):** ส่งรหัส OTP 6 หลักเข้าอีเมลจริง หมดอายุใน 120 วินาที
- **High-Entropy Passwords:** รหัสผ่านเริ่มต้นและที่สุ่มใหม่มีความปลอดภัยสูงตามมาตรฐานองค์กร
- **First-Login Enforcement:** บังคับเปลี่ยนรหัสผ่านทันทีเมื่อล็อกอินเข้าสู่ระบบครั้งแรก

🗄️ ฐานข้อมูล PostgreSQL 16 & Serverless Fallback

- ย้ายจาก SQLite เดิมสู่ **PostgreSQL 16** เต็มรูปแบบผ่าน Prisma Client ORM
- รองรับการซิงค์ข้อมูลคู่ขนานกับ **Supabase Cloud Database**
- มีระบบ **In-Memory & StorageService Resilient Fallback** ช่วยให้ระบบทำงานได้ 100% แม้รันบน Serverless (เช่น Vercel)

🐳 คอนเทนเนอร์ & CI/CD Pipeline อัตโนมัติ

- **Multi-stage Dockerfile:** สร้าง Image บน `node:20-alpine` (Standalone mode) รันด้วย Non-root
- **Docker Compose:** ควบคุม Container `postgres-db` และ `web-app` พร้อม Healthcheck
- **GitHub Actions CI/CD:** ตรวจสอบ Prisma Schema, Type/Lint, Build Test และ Docker Dry-run

⚙️ 2. กลไกการทำงานของระบบ (How the System Works)

1. ขั้นตอนการสร้างผู้ใช้งานและจ่ายสิทธิ์ (Admin User Onboarding Flow)

แอดมินกรอกชื่อ, อีเมล, ฝ่าย, ตำแหน่ง และเลือก Tier ➔ ระบบสุ่ม Employee ID (`EMP-XXXX`) และ Password ➔ บันทึกลง PostgreSQL ➔ ซิงค์ Supabase ➔ ส่งอีเมลแจ้งรหัสผ่านชั่วคราวเข้าอีเมลจริงของบุคลากรทันที

2. ขั้นตอนการยืนยันตัวตนและการเข้าใช้งาน (Authentication & 2FA Flow)

ผู้ใช้ระบุ ID และ Password ➔ ระบบส่งรหัส 2FA OTP 6 หลักเข้าอีเมลจริง (อายุ 120s) ➔ กรอก OTP ถูกต้อง ➔ หากเข้าใช้งานครั้งแรก ระบบจะบังคับเปลี่ยนรหัสผ่านใหม่ทันที ➔ เมื่อสำเร็จจะออก Session Cookie (อายุ 8 ชม.)

3. ขั้นตอนการสร้างและส่งจดหมายเวียน (Circular Document Routing Flow)

ผู้ใช้สร้างบันทึกข้อความ แนบไฟล์ และเลือก **Target Scope**: 🏛️ เวียนทั้งคณะ (`FACULTY`), 📂 ภายในภาควิชา (`DEPARTMENT`), ⬆️ ตามลำดับชั้นสั่งการ (`HIERARCHICAL`), หรือ 👤 เฉพาะบุคคล (`INDIVIDUAL`) พร้อมกำหนดระดับความสำคัญ (ปกติ/ด่วน/ด่วนที่สุด/ลับ)

4. การติดตามการเปิดอ่าน การตอบกลับ และยื่นใบลา (Read Tracking & Reply Flow)

เมื่อผู้รับกดเปิดดูเอกสาร ระบบจะทำการ **Auto Mark as Read** ประทับเวลา Real-time Timestamp ลงตาราง `ReadLog` ทันที (ผู้ส่งเช็กได้ว่าใครเปิดอ่านเมื่อไหร่) ➔ ผู้รับสามารถพิมพ์ตอบกลับ หรือยื่นคำขอลา (`isLeaveRequest: true`) พร้อมแนบไฟล์รายงานได้

5. การเชื่อมต่อปฏิทินนัดหมาย (Google Calendar & iCal Integration)

บนการ์ดเอกสารและหน้ารายละเอียด มีปุ่ม "📅 **Google**" สำหรับเปิด Event ใน Google Calendar ทันที และปุ่ม "📥 **iCal**" สำหรับโหลดไฟล์ `.ics` นำเข้านัดหมายใน Apple Calendar, Outlook หรือ Windows Calendar

🔍 3. สิ่งที่ยังขาด / สิ่งที่ต้องพัฒนาเพิ่มเติมในอนาคต (Gap Analysis & Roadmap)

รายการต่อไปนี้คือประเด็นทางวิศวกรรมซอฟต์แวร์ที่แนะนำสำหรับการพัฒนาในเฟสถัดไปเพื่อยกระดับสู่ระบบราชการอิเล็กทรอนิกส์เต็มรูปแบบ (Production Enterprise Grade)

🔒 1. การเข้ารหัสรหัสผ่าน (Password Hashing: Bcrypt / Argon2)

สถานะปัจจุบัน: รหัสผ่านยังถูกจัดเก็บเป็นข้อความแบบ Plaintext ในฐานข้อมูล
สิ่งที่ต้องทำเพิ่ม: ติดตั้งไลบรารี `bcryptjs` หรือ `argon2` ใน Prisma Middleware เพื่อทำการ Salt & Hash รหัสผ่านก่อนบันทึกลง PostgreSQL เพื่อความปลอดภัยตามมาตรฐาน PDPA

☁️ 2. ที่จัดเก็บไฟล์บนคลาวด์จริง (Cloud Object Storage: S3 / Cloudflare R2)

สถานะปัจจุบัน: ระบบใช้การจำลอง URL และชื่อไฟล์ใน Mock Storage
สิ่งที่ต้องทำเพิ่ม: เชื่อมต่อ AWS S3, Cloudflare R2 หรือ Supabase Storage เพื่อสร้าง Pre-signed URL สำหรับอัปโหลดและดาวน์โหลดไฟล์เอกสาร PDF/คำสั่งตัวจริงขนาดใหญ่

⏱️ 3. ระบบจำกัดอัตราการเรียกใช้งาน (Rate Limiting & Anti-Brute Force)

สถานะปัจจุบัน: ยังไม่มีการจำกัดจำนวน Request บนเส้นทาง Login และ OTP
สิ่งที่ต้องทำเพิ่ม: ติดตั้ง Redis หรือ Upstash Rate Limiter เพื่อจำกัดจำนวนการขอ OTP (เช่น ไม่เกิน 3 ครั้งใน 5 นาที) และป้องกันการเดารหัสผ่านแบบ Brute-force

✍️ 4. ลายมือชื่ออิเล็กทรอนิกส์ (Digital Signatures & PKI)

สถานะปัจจุบัน: ยังไม่มีระบบลงนามกำกับในเอกสารคำสั่ง
สิ่งที่ต้องทำเพิ่ม: พัฒนาระบบ e-Signature หรือ Digital Signature (ตาม พ.ร.บ. ว่าด้วยธุรกรรมทางอิเล็กทรอนิกส์) สำหรับให้คณบดีหรือหัวหน้าภาควิชาลงนามเกษียนหนังสือได้บนหน้าเว็บ

📜 5. บันทึกประวัติการกระทำของระบบ (System Audit Trail & Access Logs)

สถานะปัจจุบัน: มีเฉพาะตาราง `ReadLog` สำหรับบันทึกการเปิดอ่านเอกสาร
สิ่งที่ต้องทำเพิ่ม: เพิ่มตาราง `AuditLog` บันทึกกิจกรรมสำคัญ เช่น ใครสร้าง/แก้ไข/ลบเอกสารฉบับไหน พร้อม IP Address และ User-Agent สำหรับตรวจสอบย้อนหลัง

📲 6. ระบบแจ้งเตือนภายนอก (LINE Notify / Push Notification)

สถานะปัจจุบัน: แจ้งเตือนรหัส OTP และรหัสผ่านทางอีเมล (SMTP)
สิ่งที่ต้องทำเพิ่ม: เชื่อมต่อ LINE Official Account หรือ Web Push Notifications เพื่อแจ้งเตือนหนังสือเข้าใหม่ หรือเอกสาร "ด่วนที่สุด" ส่งตรงถึงสมาร์ทโฟนของบุคลากร

🎯 4. สรุปความพร้อมในการส่งมอบงานและนำเสนออาจารย์

✅ ความสมบูรณ์ของระบบในปัจจุบัน (Readiness Summary)

- ระบบสร้างขึ้นด้วย **Next.js 14 App Router** เชื่อมต่อฐานข้อมูล **PostgreSQL 16** และ **Supabase** ผ่าน **Prisma ORM**
- ผ่านการทดสอบ Build ระดับ Production (`npm run build`) ผ่าน 100% ครบทั้ง 14 หน้าและ API Routes ทั้งหมด
- มีระบบคอนเทนเนอร์ **Docker & Docker Compose** พร้อมรันได้ทันทีบน Server หรือเครื่องสาธิตด้วยคำสั่งเดียว: `docker compose up -d --build`
- มีระบบ **GitHub Actions CI/CD** คอยตรวจสอบความถูกต้องของโค้ดและทดสอบ Build อัตโนมัติทุกครั้งที่มีการอัปเดต

เอกสารรายงานฉบับนี้จัดทำขึ้นสำหรับการนำเสนอและส่งมอบโครงงานระบบ FLAS KPS KU E-OFFICE + คณะศิลปศาสตร์และวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน