CDI-APP-03-00

แบบฟอร์มประเมินมาตรฐานการรักษาความมั่นคงปลอดภัยทางไซเบอร์

เพื่อประกอบการพิจารณาของ คณะกรรมการบริหารการพัฒนาระบบสารสนเทศและแอปพลิเคชัน

1. **ชื่อ-สกุล ผู้ขอใช้บริการ**.....

   เลขหมายโทรศัพท์ภายใน.....

   โทรศัพท์มือถือ.....

   อีเมล.....

2. **หน่วยงานที่สังกัด**.....

   ฝ่าย/งาน/สาขาวิชา.....

3. **ชื่อระบบที่ต้องการพัฒนา**

   .....

4. **การใช้พื้นที่เครื่องแม่ข่ายสำหรับระบบที่ต้องการพัฒนา**

   [ ] ใช้พื้นที่เครื่องแม่ข่ายของศูนย์นวัตกรรมดิจิทัล

   [ ] ใช้พื้นที่เครื่องแม่ข่ายที่ให้บริการอื่น ระบุรายละเอียด.....

   .....

   .....

5. รายการประเมินตนเองมาตรฐานการรักษาความมั่นคงปลอดภัยทางไซเบอร์

หมวดที่ 1 รายการประเมินสำหรับผู้พัฒนาระบบสารสนเทศและแอปพลิเคชัน

ลำดับ	หัวข้อ	หน่วยงานประเมินตนเอง		ความเห็นของศูนย์นวัตกรรมดิจิทัล (ยอมรับได้/ต้องปรับปรุง พร้อมระบุเหตุผล)
		ทำได้	ทำไม่ได้ (ระบุแนวทางทดแทน)	
1	การพิสูจน์ตัวตนและการควบคุมการเข้าถึง (Authentication & Access Control) มาตรฐานที่อ้างอิง : OWASP 2021 (A01 และ A07) และ Zero Trust			
1.1	ใช้มาตรฐาน OAuth 2.0 หรือ OpenID Connect สำหรับการพิสูจน์ตัวตน เช่น การเข้าสู่ระบบด้วยบัญชี Google, Facebook หรือ ThaID เพื่อลดความเสี่ยงจากการจัดการรหัสผ่านโดยตรงและอำนวยความสะดวกให้ผู้ใช้			
1.2	พัฒนาระบบการยืนยันตัวตนหลายขั้นตอน (Multi-Factor Authentication: MFA) เช่น การใช้รหัสผ่านร่วมกับการยืนยันผ่านโทรศัพท์มือถือ (SMS OTP) ซึ่งจะช่วยลดความเสี่ยงได้ถึง 99.9% แม้รหัสผ่านจะถูกขโมย (อ้างอิงจาก NIST)			
1.3	กำหนดนโยบายรหัสผ่านที่รัดกุม โดยมีความยาวขั้นต่ำ 12 ตัวอักษร และประกอบด้วยตัวอักษรใหญ่-เล็ก ตัวเลข และสัญลักษณ์พิเศษเพื่อเพิ่มความปลอดภัยจากการโจมตีด้วยวิธี Brute Force Attack			
1.4	พัฒนาระบบการจัดการเซสชัน (Session Management) ของผู้ใช้ให้ปลอดภัย เช่น การเข้ารหัส Session ID และการกำหนดเวลาหมดอายุของเซสชันเพื่อป้องกันการเข้าถึงระบบโดยไม่ได้รับอนุญาต			

	1.4.1 มีการเข้ารหัส Session ID			
	1.4.2 มีการกำหนด Session Timeout เช่น ตั้งค่าหมดอายุภายใน 30 นาทีเมื่อไม่ใช้งาน			
	1.4.3 มีการสร้าง Session ID แบบสุ่มที่คาดเดาไม่ได้			
	1.4.4 มีการส่ง Session ID ผ่านช่องทางที่เข้ารหัสลับ			
2	การเข้ารหัสข้อมูล (Data Encryption) มาตรฐานที่อ้างอิง : Cybersecurity Framework (NIST) และ Zero Trust			
2.1	เข้ารหัสข้อมูลที่สำคัญหรือข้อมูลที่จัดเก็บเพื่อให้สอดคล้องกับพระราชบัญญัติคุ้มครองข้อมูลส่วนบุคคล (PDPA) (https://www.ratchakitcha.soc.go.th/DATA/PDF/2562/A/069/T_0052.PDF) มาตรา 37 และ 40 โดยใช้มาตรฐานเทียบเท่าหรือดีกว่า เช่น ข้อมูลทั่วไป : ใช้มาตรฐานขั้นต่ำ AES-128 หรือ RSA-2048 ข้อมูลอ่อนไหว (Sensitive Data) : ใช้มาตรฐาน AES-256 หรือ ECC-256			
2.2	ใช้โปรโตคอล TLS 1.2 ขึ้นไปเพื่อเข้ารหัสข้อมูลทุกครั้งที่มีการสื่อสารผ่านเครือข่ายเพื่อป้องกันการดักจับหรือแก้ไขข้อมูลโดยผู้ไม่หวังดี (Man-in-the-Middle Attacks) และสอดคล้องกับข้อกำหนดทางกฎหมายตามพระราชบัญญัติว่าด้วยการกระทำความผิดเกี่ยวกับคอมพิวเตอร์ พ.ศ. 2560 มาตรา 5 และปิดการใช้งานโปรโตคอล TLS ที่ต่ำกว่า 1.2 ลงไปทั้งหมด			
3	การป้องกันภัยคุกคาม (Threat Protection) มาตรฐานที่อ้างอิง : OWASP 2021 (A03 และ A05) และ Cybersecurity Framework			

3.1	ป้องกันการโจมตีที่ใช้ช่องโหว่ของฐานข้อมูล (SQL Injection) เช่น ใช้ Prepared Statements, Stored Procedures หรือการเรียกใช้ข้อมูลผ่าน API			
3.2	พัฒนาระบบ Input Validation เช่น ตรวจสอบชนิด/รูปแบบ/ช่วงของข้อมูล เพื่อการตรวจสอบข้อมูลที่ผู้ใช้ป้อนเข้าสู่ระบบว่าถูกต้องตามรูปแบบที่คาดหวังหรือไม่ ก่อนที่จะนำข้อมูลนั้นไปประมวลผลต่อ ช่วยลดความเสี่ยงจากการโจมตีหลายรูปแบบ			
3.3	พัฒนาระบบ Output Validation เช่น การ Encoding Output, จำกัดข้อมูลที่ส่งออกเท่าที่จำเป็น			
3.4	พัฒนาระบบ Encoding เช่น เข้ารหัสข้อมูลที่รับส่งบนเครือข่ายโดยใช้ TLS 1.2 ขึ้นไปสำหรับรับส่งข้อมูลทั่วไป ใช้ AES-256-GCM กับข้อมูลที่เป็นความลับสูง			
3.5	ป้องกันการโจมตีที่แทรกโค้ดที่เป็นอันตรายลงในเว็บไซต์ (Cross-Site Scripting : XSS) เพื่อป้องกันการขโมย Session ของผู้ใช้			
3.6	ใช้งาน HTTPOnly Cookie flag บน Framework/Server			
3.7	พัฒนาระบบ Sanitization เช่น ใช้ Whitelisting แทน Blacklisting, ตรวจสอบอักขระพิเศษ (<>'"&), กำหนดความยาวสูงสุด (Max Length), ตรวจสอบว่าเป็นตัวเลขจริง, กำหนดช่วงค่า (Min-Max), ใช้ HTML Encoding ตรวจสอบว่าไม่มี JavaScript Code ปน, ตรวจสอบ MIME Type จริง (ไม่ใช่นามสกุลไฟล์), เปลี่ยนชื่อไฟล์ใหม่ (secure_filename), จำกัดขนาดไฟล์, หลีกเลี่ยง String Concatenation ใน SQL, แยก Command และ Arguments, ตรวจสอบ Input ก่อนส่งไปยัง Shell			

3.8	ป้องกันการโจมตีที่ใช้การปลอมแปลงคำขอจากผู้ใช้ (Cross-site request forgery : CSRF) โดยใช้ Token สำหรับยืนยันคำขอทุกครั้ง			
3.9	ใช้ Unique Token เช่น Cryptographic Random Token กำหนดอายุของ Token ระยะสั้น หรือใช้เครื่องมือ เช่น JWT (JSON Web Tokens), Redis			
3.10	ติดตั้งและใช้งานระบบ Captcha			
3.11	ตรวจสอบ Referrer เช่น กำหนดโดเมนที่อนุญาต, ตรวจสอบ Request ทุกรายการที่ไม่ใช่ GET หรือใช้ HTTS เท่านั้น			
3.12	ควบคุมการแสดงข้อความแจ้งเตือนและข้อผิดพลาด เช่น ไม่แสดงรายละเอียดระบบในข้อผิดพลาด เช่น Database Schema, Server Paths, ใช้ข้อความทั่วไปสำหรับผู้ใช้ แต่บันทึกรายละเอียดใน Log, ใช้ HTTP Status Code ที่ถูกต้อง เช่น 404, 500 เป็นต้น			
3.13	ไม่ใช้ Autocomplete ในฟอร์มที่สำคัญ			
3.14	หลีกเลี่ยงการใช้ URL ที่คาดเดาได้ง่าย			
4	การควบคุมการพัฒนาและทดสอบ (Development & Testing Controls) มาตรฐานที่อ้างอิง : OWASP 2021 (A04 และ A06)			
4.1	ใช้ Secure Software Development Life Cycle (SSDLC) ซึ่งเป็นกระบวนการพัฒนาซอฟต์แวร์ที่มีการพิจารณาความปลอดภัยในทุกขั้นตอน			
4.2	ทำการตรวจสอบโค้ดเพื่อหาช่องโหว่ด้านความปลอดภัย (Secure Code Review)			
4.3	ใช้เครื่องมือ Static Application Security Testing (SAST) เพื่อตรวจหาช่องโหว่			

5	การจัดการ Dependencies และ Components มาตรฐานที่อ้างอิง : OWASP 2021 (A06 และ A08)			
5.1	ตรวจสอบ Dependencies เพื่อหาช่องโหว่ โดยใช้เครื่องมือที่ใช้ในการตรวจสอบและจัดการส่วนประกอบของซอฟต์แวร์ หรือ Software Composition Analysis (SCA) Tools เช่น Dependency-Check			
5.2	จัดทำบัญชีรายการ Dependencies			
5.3	อัปเดต Libraries และ Components ทุก 3 เดือนหรือเมื่อพบช่องโหว่ร้ายแรง (CVSS >=7 (High และ Critical) อ้างอิงจาก CVSS v3.1)			

หมวดที่ 2 แบบฟอร์มตรวจสอบการพัฒนา API ตามมาตรฐานเพื่อการพัฒนาระบบสารสนเทศอย่างปลอดภัย

(กรณี<u>ไม่ได้</u>พัฒนาระบบด้วยวิธีการ API ไม่ต้องประเมินตนเองในหมวดที่ 2)

ลำดับ	หัวข้อ	หน่วยงานประเมินตนเอง		ความเห็นของศูนย์นวัตกรรมดิจิทัล (ยอมรับได้/ต้องปรับปรุง พร้อมระบุเหตุผล)
		ทำได้	ทำไม่ได้ (ระบุแนวทางทดแทน)	
1	การพิสูจน์ตัวตนและการอนุญาต (Authentication & Authorization) มาตรฐานที่อ้างอิง : OWASP 2021 (A01, A07), Zero Trust			
1.2	รองรับ OAuth 2.0 และ OpenID Connect			

1.3	ใช้ API Keys สำหรับ Service-to-Service Authentication และต้องไม่เก็บ API Keys ไว้ในโค้ดหรือ Repository โดยตรง แต่ใช้ระบบจัดการความลับ (Secret Management) เช่น HashiCorp Vault, Bitwarden, Keycloak หรือ CyberArk Conjur			
1.4	กำหนด Rate Limiting เพื่อป้องกันการใช้งานเกินขีดจำกัด			
1.5	มีระบบ API Gateway สำหรับการจัดการการเข้าถึง			
2	การจัดการข้อมูล (Data Handling) มาตรฐานที่อ้างอิง : OWASP 2021 (A03), Cybersecurity Framework			
2.1	ตรวจสอบและกรองข้อมูล Input ทั้งหมด โดยนำ Library ในการพัฒนาระบบของ Framework ที่ใช้ หรือติดตั้ง Component มาใช้งานเพื่อป้องกัน SQL Injection และ Cross Site Scripting (XSS)			
2.2	เข้ารหัสข้อมูลที่ส่งผ่าน API ด้วย HTTPS			
2.3	จำกัดขนาดของ Request และ Response			
2.4	กำหนดรูปแบบข้อมูลที่ชัดเจนด้วย Schema Validation			
2.5	ป้องกันการรั่วไหลของข้อมูลในข้อความแจ้งข้อผิดพลาด ห้ามแสดงข้อมูลระบบ เช่น รายละเอียดข้อผิดพลาด (Stack Trace), โครงสร้างฐานข้อมูล (Database Schema) หรือชื่อเครื่องเซิร์ฟเวอร์ (Server Hostname)			
3	การป้องกันการโจมตี (Security Controls) มาตรฐานที่อ้างอิง : OWASP 2021 (A02, A05), Zero Trust			
3.1	ใช้ Security Headers ที่เหมาะสม เช่น CORS หรือ CSP			

3.2	ป้องกัน API-specific Attacks เช่น Mass Assignment โดยการใช้ Allowlist สำหรับฟิลด์ที่อนุญาตให้ปรับปรุงลงใน Request Body			
3.3	มีระบบป้องกัน Brute Force Attacks			
3.4	ตรวจสอบและป้องกัน API-specific Injection Attacks เช่น ใช้ Parameterized Queries สำหรับการ Query ฐานข้อมูล หรือตรวจสอบ Input ด้วย Regular Expression เพื่อป้องกัน Command Injection			
3.5	มีระบบป้องกัน Man-in-the-Middle Attacks เช่น ต้องใช้งาน HTTPS พร้อมรองรับ TLS 1.2 เป็นอย่างต่ำ, ใช้ Certificate Pinning สำหรับ Mobile API			
4	การตรวจสอบและเฝ้าระวัง (Monitoring & Logging) มาตรฐานที่อ้างอิง : Cybersecurity Framework, Zero Trust			
4.1	บันทึก Log การเรียกใช้ API ทั้งหมด			
4.2	ติดตามและแจ้งเตือนการใช้งานที่ผิดปกติ โดยใช้ SIEM (Security Information and Event Management) เช่น OSSIM (Open Source Security Information Management), ELK Stack (Elasticsearch Logstash Kibana), Wazuh หรือกำหนด Threshold สำหรับแจ้งเตือน เช่น เรียก API เกิน 100 ครั้ง/นาที, เรียก API Endpoint เดียวกันเกิน 500 ครั้ง/วินาที			
4.3	ใช้ Unique Request IDs สำหรับการติดตามการใช้งาน			
4.4	มีระบบวิเคราะห์พฤติกรรมการใช้งาน API			
4.5	จัดเก็บ Metrics สำหรับการวิเคราะห์ประสิทธิภาพและความปลอดภัย เช่น Latency, Error Rate, Request Count หรือ API Usage by Endpoint			

5	การจัดการเวอร์ชัน (API Versioning) มาตรฐานที่อ้างอิง : OWASP 2021 (A04)			
5.1	กำหนดนโยบายการจัดการเวอร์ชันที่ชัดเจน			
	5.1.1 กำหนดรูปแบบการใช้ Version เช่น /api/v1/resource			
	5.1.2 มีแผนการยกเลิกการใช้งาน API เวอร์ชันเก่า			
	5.1.3 กำหนดระยะเวลารองรับเวอร์ชันเก่าล่วงหน้าเมื่อทราบว่าจะสิ้นสุดการใช้บริการ			
	5.1.4 แจ้งเตือนผู้ใช้งานล่วงหน้า 3 เดือน ก่อนยกเลิกเวอร์ชันเก่า			
5.2	รักษาความเข้ากันได้ย้อนหลัง (Backward Compatibility)			
5.3	จัดทำเอกสาร API ที่ครบถ้วนสำหรับทุกเวอร์ชัน			
6	การทดสอบ API (API Testing) มาตรฐานที่อ้างอิง : OWASP 2021 (A10)			
6.1	ทำการทดสอบความปลอดภัยของ API อย่างสม่ำเสมอ			
6.2	ใช้เครื่องมือทดสอบอัตโนมัติสำหรับ API Security เช่น OWASP ZAP, Postman หรือ Burp Suite			
6.3	ทดสอบการทำงานของ Rate Limiting และการป้องกันการโจมตี			
6.4	ทดสอบประสิทธิภาพและความทนทานของ API			
6.5	จำลองสถานการณ์การโจมตีเพื่อทดสอบการป้องกัน เช่น ส่ง Malicious Payload เพื่อทดสอบ SQL Injection ทดสอบ DDoS ด้วยเครื่องมือ เช่น JMeter หรือ Locust			

หมวดที่ 3 รายการประเมินรายการตรวจสอบเครื่องแม่ข่าย

(กรณีขอใช้พื้นที่เครื่องแม่ข่ายของศูนย์นวัตกรรมดิจิทัลไม่ต้องประเมินตนเองในหมวดที่ 3)

ลำดับ	หัวข้อ	หน่วยงานประเมินตนเอง		ความเห็นศูนย์นวัตกรรมดิจิทัล (ยอมรับได้/ต้องปรับปรุง พร้อมระบุเหตุผล)
		ทำได้	ทำไม่ได้ (ระบุแนวทางทดแทน)	
1	การพิสูจน์ตัวตนและการควบคุมการเข้าถึง (Authentication & Access Control) มาตรฐานที่อ้างอิง : OWASP 2021 (A01, A07), Zero Trust			
1.1	การควบคุมการเข้าถึงโดยใช้งาน IP Address Allowlist (รายการ IP Address ที่ได้รับอนุญาต) เพื่อป้องกันการเข้าถึงจากบุคคลที่ไม่ได้รับอนุญาต และลดความเสี่ยงจากการโจมตีจากแหล่งที่ไม่รู้จัก			
	1.1.1 กำหนดรายการ IP Address หรือ URL หรือเครือข่ายที่อนุญาตให้เข้าถึงเครื่องแม่ข่าย			
	1.1.2 ไม่อนุญาตการเข้าถึงเครื่องแม่ข่ายจาก IP Address สาธารณะ			
	1.1.3 ทบทวนและปรับปรุงรายการ IP Address Allowlist อย่างน้อยปีละครั้ง			
	1.1.4 ทดสอบการบล็อก IP Address นอก Allowlist อย่างน้อยปีละครั้ง			
1.2	กำหนดสิทธิการเข้าถึงโฟลเดอร์ระบบจัดการภายใน โดยให้สิทธิการเข้าถึงเฉพาะผู้ที่มีหน้าที่เกี่ยวข้องเท่านั้นโดยตั้งค่าการอนุญาตแบบ Least Privilege			

1.3	ตรวจสอบและปรับแต่งสิทธิ์ไฟล์ (Read/Write/Execute) ให้เหมาะสม โดยระบุสิทธิ์น้อยที่สุดเท่าที่จำเป็น (Principle of Least Privilege) ตัวอย่างเช่น ไฟล์ config (740), ไฟล์ log (640), scripts (750), ข้อมูลที่ละเอียดอ่อน (600) และเนื้อหาเว็บสาธารณะ (644)			
1.4	ตรวจสอบและทบทวนสิทธิ์อย่างน้อยปีละครั้ง			
1.5	ติดตั้งและกำหนดค่า Firewall สำหรับระบบฐานข้อมูล เพื่อป้องกันการเข้าถึงฐานข้อมูลที่ไม่พึงประสงค์			
1.6	จำกัดการเข้าถึงฐานข้อมูลเฉพาะเครื่องแม่ข่าย หรือโปรแกรมที่เกี่ยวข้องเพื่อความปลอดภัยของข้อมูล (บางฐานข้อมูลอาจสามารถทำได้เพียงอย่างใดอย่างหนึ่ง)			
1.7	กำหนดสิทธิการใช้งานฐานข้อมูลตามบทบาทและหน้าที่เพื่อป้องกันการเข้าถึงข้อมูลที่ไม่จำเป็น			
2	การเข้ารหัสข้อมูล (Data Encryption) มาตรฐานที่อ้างอิง : Cybersecurity Framework (NIST), Zero Trust			
2.1	จัดการระบบกุญแจเข้ารหัส (Key Management) ให้มั่นใจว่าข้อมูลที่ถูกเข้ารหัสจะสามารถถอดรหัสได้เฉพาะผู้ที่มีสิทธิ์เท่านั้นเพื่อป้องกันการเข้าถึงข้อมูลที่ไม่ได้รับอนุญาต			
	2.1.1 การเลือกและใช้อัลกอริทึมเข้ารหัสที่เหมาะสม (Appropriate Encryption Algorithm Selection and Usage) โดยใช้อัลกอริทึมตามมาตรฐาน NIST SP 800-57 เช่น AES-256 (Advanced Encryption Standard) สำหรับข้อมูลสำคัญ			

	และ TLS 1.2 ขึ้นไป (Transport Layer Security) สำหรับการสื่อสาร เลือกใช้ตามประเภทและความสำคัญของข้อมูล			
	2.1.2 การจัดการกุญแจตลอดวงจรชีวิต (Key Management Lifecycle) มีกระบวนการสร้าง จัดเก็บ หมุนเวียน และทำลายกุญแจอย่างปลอดภัย ใช้ HSM (Hardware Security Module) / TPM (Trusted Platform Module) / KMS (Key Management System) สำหรับจัดเก็บ ไม่ใช้ Config File หรือ Database เก็บกุญแจโดยตรง ไม่เก็บเป็น Plaintext แม้บนเครือข่ายภายใน			
	2.1.3 การควบคุมการเข้าถึงและการตรวจสอบ (Access Control and Audit) ใช้ RBAC (Role-Based Access Control) และ ACL (Access Control List) เพื่อจำกัดสิทธิ์การเข้าถึง นำหลักการ Least Privilege มาใช้ และบันทึกทุกกิจกรรมที่เกี่ยวข้องกับกุญแจเข้ารหัส (Key Access Logging) เพื่อตรวจสอบย้อนกลับ			
	2.1.4 การตรวจสอบและเฝ้าระวังระบบเข้ารหัส (Encryption System Monitoring and Surveillance) มีการตรวจสอบระบบอย่างสม่ำเสมอ มีระบบแจ้งเตือนเมื่อมีความพยายามเข้าถึงที่ผิดปกติ และทดสอบประสิทธิภาพของการเข้ารหัสเป็นระยะ ใช้หลักการ Continuous Monitoring ตามแนวคิด Zero Trust			

	2.1.5 มีแผนรับมือเหตุการณ์และการกู้คืนกุญแจเมื่อเกิดการละเมิดความปลอดภัย มีกระบวนการกู้คืนกุญแจเข้ารหัสที่ชัดเจน (Key Recovery Procedures) และทดสอบอย่างน้อยปีละครั้ง รวมถึงมีขั้นตอนการจัดการเมื่อมีการเปลี่ยนแปลงบุคลากรที่เกี่ยวข้อง (Personnel Change Management)			
2.2	กำหนดค่าและดูแล TLS Certificates ซึ่งเป็นเทคโนโลยีที่ช่วยให้การสื่อสารระหว่างเครื่องแม่ข่ายและผู้ใช้งานมีความปลอดภัยเพื่อป้องกันการดักจับข้อมูล			
	2.2.1 ใช้ TLS เวอร์ชันตั้งแต่ 1.2 ขึ้นไป ปิดการใช้งาน SSL ทุกเวอร์ชัน และปิดการใช้งาน TLS ที่ต่ำกว่า 1.2 ลงไปทั้งหมด			
	2.2.2 เลือก Certificate จาก CA ที่น่าเชื่อถือ			
3	การป้องกันภัยคุกคาม (Threat Protection) มาตรฐานที่อ้างอิง : OWASP 2021 (A03, A05), Cybersecurity Framework			
3.1	กำหนดค่าเครื่องแม่ข่ายให้อยู่หลัง Web Application Firewall (WAF) ของมหาวิทยาลัยเพื่อป้องกันการโจมตีที่อาจเกิดขึ้นผ่านแอปพลิเคชันเว็บ โดย WAF จะต้องสามารถบล็อกการโจมตี เช่น SQL Injection, Cross-Site Scripting (XSS) และ Cross-Site Request Forgery (CSRF) ได้ และต้องปรับปรุงกฎของ WAF ให้มีประสิทธิภาพเสมอ			

3.2	ติดตั้งระบบป้องกัน Distributed Denial of Service (DDoS) ซึ่งเป็นการโจมตีที่ทำให้ระบบไม่สามารถให้บริการได้ (หากเครื่องแม่ข่ายอยู่ภายใต้การดูแลของศูนย์นวัตกรรมดิจิทัลจะมีการติดตั้ง Firewall และ WAF)			
3.3	ตั้งค่า HTTP Response Headers ให้สามารถช่วยป้องกันช่องโหว่ด้านความปลอดภัย			
	3.3.1 X-Frame-Options : ตั้งค่าเป็น DENY เพื่อป้องกัน Clickjacking			
	3.3.2 X-XSS-Protection : ตั้งค่าเป็น 0 หรือใช้ CSP แทน			
	3.3.3 X-Content-Type-Options : ตั้งค่าเป็น nosniff เพื่อป้องกัน MIME type sniffing			
	3.3.4 Referrer-Policy : ตั้งค่าเป็น strict-origin-when-cross-origin เพื่อควบคุมข้อมูล referrer			
	3.3.5 Content-Type : ตั้งค่าให้ถูกต้อง เช่น text/html; charset=UTF-8 สำหรับหน้า HTML			
	3.3.6 Strict-Transport-Security : ใช้ max-age=63072000; includeSubDomains; preload เพื่อบังคับให้ใช้ HTTPS			

	3.3.7 Content-Security-Policy : กำหนดแหล่งที่มาของเนื้อหาที่อนุญาตให้โหลดได้เพื่อป้องกัน XSS			
	3.3.8 Access-Control-Allow-Origin : ระบุ origin ที่เฉพาะเจาะจงแทนการใช้ *			
	3.3.9 Cross-Origin-Opener-Policy : ตั้งค่าเป็น same-origin เพื่อแยก browsing context			
	3.3.10 Cross-Origin-Embedder-Policy : ตั้งค่าเป็น require-corp เพื่อควบคุมการโหลดทรัพยากรข้าม origin			
	3.3.11 Cross-Origin-Resource-Policy : ตั้งค่าเป็น same-site เพื่อจำกัดการโหลดทรัพยากรเฉพาะไซต์เดียวกัน			
	3.3.12 Permissions-Policy : ปิดการใช้งานฟีเจอร์ของเบราว์เซอร์ที่ไม่จำเป็น เช่น geolocation=(), camera=(), microphone=()			
	3.3.13 ลบ Expect-CT Header ออกจากโค้ดที่มีอยู่ทั้งหมด : เพื่อลดความเสี่ยงด้านความปลอดภัยจากการตั้งค่าที่ล้าสมัย หรือขัดแย้งกับระบบปัจจุบัน และลดการใช้ทรัพยากรที่ไม่จำเป็นทำให้ประหยัดแบนด์วิดท์			
	3.3.14 ลบ Server Header : เพื่อไม่เปิดเผยข้อมูลของเซิร์ฟเวอร์			
	3.3.15 ลบ X-Powered-By Header : เพื่อไม่เปิดเผยข้อมูลเทคโนโลยีที่ใช้			

	3.3.16 ลบ X-AspNet-Version Header : เพื่อไม่เปิดเผยข้อมูลเวอร์ชัน .NET			
	3.3.17 ลบ X-AspNetMvc-Version Header : เพื่อไม่เปิดเผยข้อมูลเวอร์ชัน .NET MVC			
	3.3.18 X-DNS-Prefetch-Control : ตั้งค่าเป็น Off ถ้าไม่ต้องการให้มีการโหลด DNS ล่วงหน้า			
	3.3.19 ปิดการใช้ Public-Key-Pins (HPKP) : เพื่อลดช่องโหว่ของการโจมตีแบบ MITM			
3.4	ปิดการแสดงเวอร์ชัน Server-Side Script Engine เพื่อไม่ให้ผู้ไม่ประสงค์ดีทราบข้อมูลที่เป็นช่องโหว่ของระบบ เช่น ถ้าใช้งาน Apache เป็น Web Server ต้องกำหนดค่าที่ไ ฟล์ httpd.conf หรือ apache2.conf โดยการเพิ่มบรรทัด ServerSignature Off ServerTokens Prod			
3.5	ควบคุมการแสดงข้อผิดพลาดของ Server เพื่อป้องกันไม่ให้ข้อมูลสำคัญรั่วไหลออกไป โดยการตั้งค่าให้แสดงข้อความทั่วไปแทนข้อความแสดงผลจากระบบ เช่น ใช้ข้อความว่า "เกิดข้อผิดพลาด" แทน "Error 500 : Database Connection Failed"			
4	การตรวจสอบและบันทึก (Logging & Monitoring) มาตรฐานที่อ้างอิง : Zero Trust, Cybersecurity Framework			

4.1	ตั้งค่าการบันทึกเหตุการณ์ด้านความปลอดภัย โดยเปิดใช้งาน system logging ให้บันทึกการเข้าสู่ระบบทั้งสำเร็จและล้มเหลว, การใช้สิทธิ์ root/admin และการเปลี่ยนแปลงไฟล์สำคัญ			
4.2	ติดตั้งเครื่องมือป้องกันพื้นฐาน เช่น fail2ban (Linux) หรือ Windows Firewall with Advanced Security ตั้งค่าให้บล็อกการพยายามเข้าถึงที่ผิดปกติโดยอัตโนมัติ			
4.3	ทบทวนและวิเคราะห์ Log อย่างสม่ำเสมอ โดยใช้เครื่องมือพื้นฐาน เช่น grep (Linux) หรือ Event Viewer (Windows) เพื่อค้นหารูปแบบที่ผิดปกติเป็นประจำ			
4.4	ตั้งค่า Log Rotation เพื่อป้องกันพื้นที่เก็บข้อมูล Log เต็ม และสำรองข้อมูล Log ไปยัง External Storage			
4.5	จำกัดการเข้าถึงไฟล์ Log ให้เฉพาะผู้ดูแลระบบที่ได้รับอนุญาตเท่านั้น เพื่อป้องกันการแก้ไขหรือลบ Log โดยไม่ได้รับอนุญาต			
5	การควบคุมการพัฒนาและทดสอบ (Development & Testing Controls) มาตรฐานที่อ้างอิง : OWASP 2021 (A04, A06)			
5.1	ปิดเครื่องแม่ข่ายหรือเครื่องแม่ข่ายเสมือนที่ไม่ได้ใช้งานเพื่อลดความเสี่ยงในการถูกโจมตี			
5.2	จัดการการทดสอบเจาะระบบ (Penetration Testing) เพื่อหาช่องโหว่ของระบบและแก้ไขก่อนที่จะถูกโจมตี			

	โดยการทดสอบเจาะระบบอย่างน้อยปีละครั้ง เมื่อมีการ Upgrade/Patch ต้องทดสอบทุกครั้ง รายงานผลและแก้ไขช่องโหว่ตามนโยบายที่กำหนดไว้			
5.3	ลบโปรแกรมและไฟล์ที่ไม่ได้ใช้เพื่อลดความเสี่ยงในการถูกโจมตี			
5.4	เปลี่ยนค่าเริ่มต้นของเครื่องแม่ข่ายที่อาจเป็นช่องโหว่ เช่น รหัสผ่านเริ่มต้น และบัญชีผู้ใช้เริ่มต้นที่มากับเครื่องแม่ข่ายหรือตัวระบบที่ติดตั้ง			
5.5	ลบบัญชีผู้ใช้ที่ไม่ได้ใช้งานเพื่อป้องกันการเข้าถึงที่ไม่พึงประสงค์			
6	การจัดการ Dependencies และ Components มาตรฐานที่อ้างอิง : OWASP 2021 (A06, A08)			
6.1	อัปเดต Web Server Software เป็นเวอร์ชันปัจจุบันเพื่อความปลอดภัยและประสิทธิภาพที่ดีขึ้น			
6.2	ควบคุมการอนุมัติการใช้ Plugins หรือ Third-party Components ซึ่งเป็นส่วนประกอบจากภายนอกเพื่อลดความเสี่ยงจากการถูกโจมตีผ่านช่องโหว่ของส่วนประกอบเหล่านั้น			
7	การสำรองและกู้คืนข้อมูล (Data Backup & Recovery) มาตรฐานที่อ้างอิง : Cybersecurity Framework (NIST), Zero Trust			
7.1	จัดทำนโยบายและขั้นตอนการสำรองข้อมูลที่ชัดเจน เพื่อให้สามารถกู้คืนข้อมูลเมื่อเกิดปัญหาได้อย่างรวดเร็ว เช่น ทำ Full Backup อย่างน้อยสัปดาห์ละครั้ง, ทำ Incremental Backup ทุกวัน			
7.2	ติดตั้งระบบหรือกำหนดให้โปรแกรมสำรองข้อมูลตรวจสอบการสำรองข้อมูลเพื่อให้มั่นใจว่าข้อมูลถูกสำรองอย่างถูกต้อง			

7.3	จัดเก็บข้อมูลสำรองในสถานที่ที่ปลอดภัย เพื่อป้องกันการสูญหายของข้อมูลจากเหตุการณ์ไม่คาดคิดหรือจากภัยธรรมชาติ เช่น Off-site หรือ Air gap			
7.4	ทดสอบการกู้คืนข้อมูลเพื่อให้มั่นใจว่าข้อมูลสามารถกู้คืนได้จริง อย่างน้อยปีละครั้ง			
7.5	กำหนด Recovery Time Objective (RTO) คือ ระยะเวลาสูงสุดที่องค์กรสามารถยอมรับได้ในการกู้คืนระบบ แอปพลิเคชัน หรือบริการให้กลับมาใช้งานได้หลังจากเกิดเหตุขัดข้อง ก่อนที่จะส่งผลกระทบหรือเกิดความสูญเสียที่ไม่สามารถยอมรับได้ และ Recovery Point Objective (RPO) คือ ช่วงเวลาสูงสุดของข้อมูลที่สามารถสูญหายได้เมื่อต้องกู้คืนระบบหลังจากเกิดเหตุขัดข้องโดยกำหนดจากความถี่ของการสำรองข้อมูล (Backup) เพื่อให้สามารถวางแผนการกู้คืนได้อย่างมีประสิทธิภาพ			
7.6	ลบ/ทำลายสื่อบันทึกข้อมูลที่ไม่ใช้งานแล้วตามหลักการมาตรฐาน			
7.7	กำหนดระยะเวลาการเก็บข้อมูลสำรองเพื่อให้สามารถจัดการข้อมูลได้อย่างมีประสิทธิภาพ			

หมายเหตุ มาตราฐานต่างๆ อาจมีการเปลี่ยนแปลงเพื่อให้เป็นไปตามประกาศคณะกรรมการการรักษาความมั่นคงปลอดภัยไซเบอร์

5.ความเห็นของผู้บังคับบัญชาหน่วยงาน

.....

.....

ลงนาม .....

(.....)

ตำแหน่ง.....

ลงวันที่.....

6.ความเห็นในการพิจารณาให้หน่วยงานพัฒนาระบบเองหรือจ้างหน่วยงานภายนอก ตามเกณฑ์ด้านความมั่นคงปลอดภัยไซเบอร์ของมหาวิทยาลัย

6.1.ด้านการพัฒนาระบบสารสนเทศและแอปพลิเคชัน

.....

.....

.....

6.2.ด้านการพัฒนา API ตามมาตรฐานเพื่อการพัฒนาระบบสารสนเทศอย่างปลอดภัย

.....

.....

.....

ลงนาม.....

(.....)

ตำแหน่งหัวหน้าฝ่ายแอปพลิเคชันโซลูชัน

ลงวันที่.....

6.3.ด้านการตรวจสอบเครื่องแม่ข่าย

.....

.....

.....

ลงนาม.....

(.....)

หัวหน้าฝ่ายโครงสร้างพื้นฐานดิจิทัล

ลงวันที่.....

6.4.ความเห็นของผู้อำนวยการศูนย์นวัตกรรมดิจิทัล

[ ] เห็นชอบ เนื่องจาก.....

.....

[ ] ไม่เห็นชอบ เนื่องจาก.....

.....

[ ] ความเห็นอื่น ๆ .....

.....

ลงนาม.....

(.....)

ตำแหน่ง ผู้อำนวยการศูนย์นวัตกรรมดิจิทัล

ลงวันที่.....