

การพัฒนาบริการยืนยันตัวตนกลางแบบไร้สถานะด้วยมาตรฐาน OpenID Connect

เพื่อการบูรณาการเว็บแอปพลิเคชันในสถาบันอุดมศึกษา

กิตติศักดิ์ แก้วเนียม*

สำนักวิทยบริการ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตปัตตานี

เลขที่ 181 ถนนเจริญประดิษฐ์ ตำบลรูสะมิแล อำเภอเมือง จังหวัดปัตตานี 94000

Development of a Stateless Centralized Authentication Service with OpenID Connect for Web Application Integration in Higher Education

Kittisak Kaewneam*

Office of Academic Resources, Prince of Songkla University, Pattani Campus

181 Charoen Pradit Road, Rusamilae Subdistrict, Mueang District, Pattani Province 94000

E-mail: kittisak.k@psu.ac.th

DOI: 10.66692/pulinet.13.2.2672

► รับบทความ 25 กุมภาพันธ์ 2569 ► แก้ไขบทความ 16 มิถุนายน 2569 ► ตอรับบทความ 19 มิถุนายน 2569

บทคัดย่อ

การวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนา SSO-Authen ซึ่งเป็นบริการยืนยันตัวตนกลางแบบไร้สถานะ (Stateless) โดยประยุกต์ใช้มาตรฐาน OpenID Connect (OIDC) เพื่อแก้ปัญหาการยืนยันตัวตนที่กระจัดกระจายในสถาบันอุดมศึกษา โดยมุ่งออกแบบสถาปัตยกรรมที่ยืดหยุ่น สามารถรองรับผู้ให้บริการยืนยันตัวตนได้หลายราย (Multi-Provider) และบูรณาการกับแอปพลิเคชันที่หลากหลาย (Multi-Client) ผ่าน JSON Web Token (JWT) กระบวนการวิจัยดำเนินการตามกรอบ SDLC โดยออกแบบ SSO-Authen ให้เป็น Authentication Gateway ที่มีระบบจัดการแอปพลิเคชันและกลไกแปลงข้อมูล (Claim Mapping) เพื่อสร้างมาตรฐานข้อมูลผู้ใช้ การประเมินผลใช้วิธีเปรียบเทียบเชิงปริมาณกับเกณฑ์มาตรฐานเดิม (Baseline) และการประเมินความซับซ้อนเชิงโครงสร้างซอฟต์แวร์ (Software Complexity Metrics) โดยทดสอบการบูรณาการจริงกับเว็บแอปพลิเคชัน 3 ประเภทที่มีสถาปัตยกรรมแตกต่างกัน (PHP, WordPress, และ React/Node.js) ผลการวิจัยพบว่า บริการ SSO-Authen สามารถลดระยะเวลาในการพัฒนางานโดยเฉลี่ย 78% และลดความซับซ้อนทางวิศวกรรมซอฟต์แวร์ฝั่งไคลเอนต์ลงกว่า 80% ในทุกดัชนีชี้วัด ซึ่งเป็นผลจากประสิทธิภาพของกลไก Claim Mapping นอกจากนี้ ผลการประเมินประสิทธิภาพนักพัฒนา (N = 8) พบว่า ส่วนใหญ่มีความพึงพอใจในระดับดีมากในทุกด้าน ผลลัพธ์ที่ได้สามารถนำไปประยุกต์ใช้เป็น Authentication Gateway กลางสำหรับทั้งองค์กร เพื่อช่วยรวมศูนย์การจัดการยืนยันตัวตน ลดภาระของนักพัฒนา เพิ่มความปลอดภัย และสร้างประสบการณ์ Single Sign-On ที่มีประสิทธิภาพ

คำสำคัญ

บริการยืนยันตัวตน, มาตรฐานโอเพนไอดีคอนเนกต์, การบูรณาการเว็บแอปพลิเคชัน, ซิงเกิลไซน์ออน

Abstract

This research aims to develop SSO-Authen, a stateless centralized authentication service applying the OpenID Connect (OIDC) standard to solve fragmented authentication in higher education. The objective is to design a flexible architecture capable of supporting multiple identity providers (Multi-Provider) and integrating with diverse applications (Multi-Client) via JSON Web Tokens (JWT). The research was conducted using the SDLC framework, designing SSO-Authen as an Authentication Gateway featuring an Admin Panel and a Claim Mapping mechanism to standardize user data. Quantitative evaluation was performed by comparing results against the organizational baseline and evaluating structural software complexity metrics through practical integration testing with three distinct application architectures (PHP, WordPress, and React/Node.js). The findings indicate that SSO-Authen reduces development time by an average of 78% and client-side software complexity by over 80% across all metrics, primarily attributed to the standardized Claim Mapping mechanism. Furthermore, a qualitative developer experience assessment (N = 8) revealed that the majority of developers were highly satisfied in all aspects. The results can be implemented as a central Authentication Gateway for the entire organization to centralize identity management, reduce developer overhead, enhance security, and provide an effective Single Sign-On experience.

Keywords

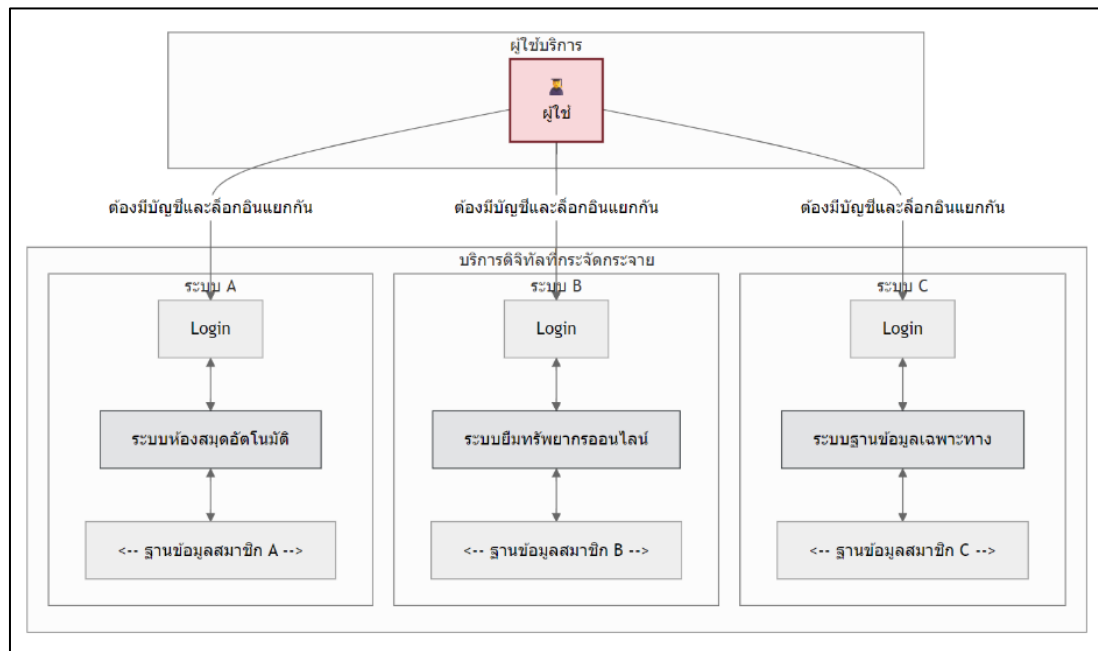
Authentication Service, OpenID Connect Standard, Web Application Integration, Single Sign-On (SSO)

บทนำ (Introduction)

ปัจจุบันสถาบันอุดมศึกษาในประเทศไทยต่างมีนโยบายมุ่งสู่การเป็นมหาวิทยาลัยดิจิทัล (Digital University) โดยการนำเทคโนโลยีและนวัตกรรมมาประยุกต์ใช้เพื่อเพิ่มประสิทธิภาพการให้บริการ ซึ่งทิศทางการดังกล่าวสอดคล้องกับแผนพัฒนารัฐบาลดิจิทัลของประเทศไทย พ.ศ. 2566-2570 ที่มุ่งเน้นยุทธศาสตร์การยกระดับบริการดิจิทัลให้มีความเชื่อมโยงและเป็นหนึ่งเดียว (สำนักงานพัฒนารัฐบาลดิจิทัล, 2566) โดยหนึ่งในกลไกสำคัญ คือ การกำหนดมาตรฐานกลางในการยืนยันตัวตนเพื่อลดความซ้ำซ้อนและสร้างมาตรฐานความปลอดภัยให้สอดคล้องกับเกณฑ์วิธีปฏิบัติทางเทคนิคของระบบสารสนเทศภาครัฐ (สำนักงานพัฒนารัฐบาลดิจิทัล, 2562) เพื่อขับเคลื่อนยุทธศาสตร์ดังกล่าวให้เกิดผลสัมฤทธิ์ในทางปฏิบัติ สถาบันอุดมศึกษาจึงมีความจำเป็นต้องพัฒนาระบบยืนยันตัวตนแบบลงชื่อเข้าใช้ครั้งเดียว (Single Sign-On: SSO) เพื่อเป็นโครงสร้างพื้นฐานดิจิทัลที่ช่วยให้บุคลากรและนักศึกษาสามารถเข้าถึงบริการต่าง ๆ ของสถาบันได้อย่างไร้รอยต่อ อย่างไรก็ตาม แม้ปัจจุบันสถาบันอุดมศึกษาหลายแห่งได้มีการพัฒนาระบบยืนยันตัวตนกลางที่ใช้มาตรฐาน OpenID Connect (OIDC) ขึ้นมาให้บริการแล้ว แต่ในกระบวนการนำไปประยุกต์ใช้จริงยังคงพบช่องว่างที่สำคัญ ส่งผลให้บริการดิจิทัลจำนวนมากยังไม่ได้ถูกบูรณาการเข้ากับระบบดังกล่าวอย่างสมบูรณ์

เมื่อพิจารณาในระดับหน่วยงานย่อย เช่น คณะ หรือสำนักวิทยบริการ ซึ่งต้องพัฒนาและดูแลระบบสารสนเทศเฉพาะทางจำนวนมาก ช่องว่างนี้ยิ่งปรากฏชัดเจนขึ้น บริการที่สำคัญต่อการเรียนการสอนและการวิจัย อาทิ ระบบห้องสมุดอัตโนมัติ ระบบยืมทรัพยากรออนไลน์ หรือระบบฐานข้อมูลเฉพาะทาง ต่างมีระบบสมาชิกและการเข้าสู่ระบบเป็นของตนเอง (ภาพที่ 1) การขาดมาตรฐานกลางในการเชื่อมต่อได้สร้างผลกระทบเชิงลบในหลายมิติ ทั้งต่อผู้ใช้บริการที่ประสบความยุ่งยากในการจดจำบัญชีผู้ใช้จำนวนมาก นำไปสู่ปัญหาการลืมรหัสผ่านและขั้นตอนการสมัครสมาชิกที่ซ้ำซ้อน และต่อผู้ดูแลระบบ

สารสนเทศที่ต้องเผชิญกับการจัดการบัญชีผู้ใช้ที่ซ้ำซ้อน ความเสี่ยงด้านความปลอดภัยจากการมีฐานข้อมูลรหัสผ่านหลายแห่ง และความสิ้นเปลืองทรัพยากรจากการต้องพัฒนาระบบยืนยันตัวตนใหม่สำหรับทุกแอปพลิเคชัน



ภาพที่ 1 แอปพลิเคชันมักมีฐานข้อมูลผู้ใช้แยกกันอย่างกระจาย

แนวโน้มการนำเทคโนโลยีดิจิทัลมาใช้เพื่อเพิ่มประสิทธิภาพสอดคล้องกับทิศทางการพัฒนาหน่วยงานห้องสมุดมหาวิทยาลัยในประเทศไทย ที่มุ่งเน้นการยกระดับบริการผ่านระบบสารสนเทศ (กุลประวีณ์ พรหมอารีย์, 2568; ปิยะบุตร ปัญญาดี และ อภิรติกาณ์ เวชพร, 2567; วิไล จันทรแก้ว และคณะ, 2567) ทว่าระบบส่วนใหญ่ยังคงเผชิญปัญหาการขาดกลไกการยืนยันตัวตนที่เป็นมาตรฐานกลาง ส่งผลให้เกิดภาวะข้อมูลแยกส่วน (Data Silo) ยิ่งไปกว่านั้น ความหลากหลายของเทคโนโลยีในปัจจุบัน ทั้งแอปพลิเคชันแบบดั้งเดิม (Traditional Web App) และแอปพลิเคชันสมัยใหม่ (Single-Page Application) ทำให้ระบบ SSO เดิมที่อาศัยการแชร์สถานะผ่านคุกกี้ (Session Cookie) ไม่สามารถตอบโจทย์ความยืดหยุ่นและการทำงานร่วมกันระหว่างเทคโนโลยีที่แตกต่างกันได้อย่างมีประสิทธิภาพ

นอกจากนี้ แม้จะมีผู้ให้บริการยืนยันตัวตนสาธารณะ (Public Identity Providers) เช่น Google Authentication หรือ ThaiID แต่บริการเหล่านี้มีข้อจำกัดสำคัญ คือ ทำหน้าที่เพียงยืนยันตัวตนบุคคลในระดับสากลเท่านั้น โดยไม่สามารถระบุข้อมูลบริบทเฉพาะทางของสถาบัน อาทิ รหัสนักศึกษา สังกัดหน่วยงาน หรือสถานภาพบุคลากร ซึ่งเป็นข้อมูลจำเป็นสำหรับการกำหนดสิทธิ์การใช้งาน (Authorization) ในระบบงานเฉพาะทาง งานวิจัยนี้จึงมุ่งเน้นการพัฒนา SSO-Authen ให้ทำหน้าที่เป็น "ตัวกลางแลกเปลี่ยนอัตลักษณ์" (Identity Broker) ที่ไม่ได้ทำหน้าที่เป็นผู้ให้บริการยืนยันตัวตน (IdP) เอง แต่ทำหน้าที่ประสานงานและจัดการข้อมูลผลลัพธ์ (Claim Mapping) จากผู้ให้บริการหลักของมหาวิทยาลัย เพื่อส่งมอบข้อมูลผู้ใช้ที่ครบถ้วนและเป็นมาตรฐานเดียวกันให้กับแอปพลิเคชันลูกข่าย

บทความนี้ นำเสนอการออกแบบและพัฒนา SSO-Authen ซึ่งเป็นบริการกลางสำหรับยืนยันตัวตน (Centralized Authentication Service) ที่ทำหน้าที่เป็นประตูผ่านทาง (Gateway) เชื่อมต่อเว็บแอปพลิเคชันกับผู้ให้บริการยืนยันตัวตน โดยใช้สถาปัตยกรรมแบบไร้สถานะ (Stateless) และการส่งมอบข้อมูลผ่าน JSON Web Token (JWT) เพื่อรองรับการบูรณาการกับเว็บแอปพลิเคชันทุกรูปแบบได้อย่างแท้จริง อันจะนำไปสู่การสร้างโครงสร้างพื้นฐานด้านการยืนยันตัวตนที่ยั่งยืน ช่วยทลายกำแพงข้อมูลระหว่างระบบ และต่อยอดสู่การพัฒนานวัตกรรมบริการดิจิทัลที่มีประสิทธิภาพเพื่อผลักดันองค์กรสู่การเป็นมหาวิทยาลัยดิจิทัลอย่างเต็มศักยภาพ

วัตถุประสงค์ (Objective)

1. เพื่อพัฒนาระบบการยืนยันตัวตนสำหรับบริการดิจิทัล โดยการสร้าง SSO-Authen ให้เป็นบริการกลางแบบไร้สถานะ (Stateless Centralized Authentication Service) ที่เชื่อมต่อกับระบบ Single Sign-On ด้วยมาตรฐาน OpenID Connect
2. เพื่อออกแบบสถาปัตยกรรมของบริการให้มีความยืดหยุ่น สามารถรองรับผู้ให้บริการยืนยันตัวตนได้หลายราย (Multi-Provider) และกำหนดกลไกการบูรณาการที่ชัดเจนผ่าน JSON Web Token (JWT) เพื่อให้สามารถรองรับการเชื่อมต่อจากเว็บแอปพลิเคชันที่สร้างด้วยเทคโนโลยีที่แตกต่างกันได้
3. เพื่อสร้างต้นแบบและแนวปฏิบัติที่ดี (Best Practice) สำหรับการบูรณาการระบบสารสนเทศใหม่เข้ากับบริการยืนยันตัวตนกลาง อันจะนำไปสู่การสร้างโครงสร้างพื้นฐานด้านการยืนยันตัวตนที่ยั่งยืนสำหรับทั้งองค์กร

แนวคิดและทฤษฎีที่เกี่ยวข้อง (Related Concepts and Theories)

หัวใจสำคัญของบริการนี้ คือ การประยุกต์ใช้มาตรฐาน OpenID Connect (OIDC) 1.0 (Sakimura et al., 2023) ซึ่งเป็นชั้นข้อมูลสำหรับการยืนยันตัวตน (Identity Layer) ที่ทำงานอยู่บนโปรโตคอล OAuth 2.0 (Gupta, 2024) โดยได้เลือกใช้กระบวนการแลกเปลี่ยนรหัสอนุญาต (Authorization Code Flow) ซึ่งเป็นกลไกที่แยกส่วนการทำงานระหว่างการปฏิสัมพันธ์ของผู้ใช้ผ่านเบราว์เซอร์ (User-Agent) และการประมวลผลของแอปพลิเคชันลูกข่ายบนเครื่องแม่ข่ายส่วนหลัง (Client Application) ออกจากกันอย่างสิ้นเชิง ทำให้ Access Token และ ID Token ถูกส่งผ่านช่องทางสื่อสารส่วนหลัง (Back-Channel) ที่ปลอดภัย สถาปัตยกรรมของ SSO-Authen ได้รับการออกแบบโดยยึดหลักการแยกหน้าที่ความรับผิดชอบอย่างชัดเจนระหว่าง Authentication และ Authorization

- Authentication (การยืนยันตัวตน) SSO-Authen จะมอบหมายหน้าที่นี้ให้กับผู้ให้บริการยืนยันตัวตน (OpenID Provider - OP) ทั้งหมด ไม่ว่าจะเป็นจากหน่วยงานที่ดูแลระบบไอทีของมหาวิทยาลัย หรือผู้ให้บริการทั่วไปจากภายนอก กระบวนการตรวจสอบชื่อผู้ใช้และรหัสผ่าน หรือวิธีการยืนยันตัวตนขั้นสูงอื่น ๆ เช่น การยืนยันตัวตนโดยไม่ใช้รหัสผ่าน (Password-Less Authentication) ดังที่ ปิติ สุวรรณคม และ อุทาน บุณยศักดิ์ศรี (2566) ได้นำเสนอไว้ จะเกิดขึ้นที่ฝั่งของผู้ให้บริการเท่านั้น โดย SSO-Authen จะทำหน้าที่เป็นระบบผู้รับรองอัตลักษณ์ (Relying Party: RP) ซึ่งมีหน้าที่ในการประสานงานกระบวนการสื่อสารตามมาตรฐาน OIDC เพื่อให้ได้มาซึ่งข้อมูลยืนยันตัวตนในรูปแบบโทเค็น (ID Token) ที่มีความถูกต้องแม่นยำและเชื่อถือได้จากผู้ให้บริการต้นทาง

- Authorization (การให้สิทธิ์) หลังจากที่ Authentication สำเร็จ SSO-Authen จะส่งต่อข้อมูลผู้ใช้ที่ผ่านการตรวจสอบแล้วให้กับเว็บแอปพลิเคชันเป้าหมาย (Client Application) ผ่านการสื่อสารไปยังจุดเชื่อมต่อสำหรับการให้บริการข้อมูล (API Endpoint) ที่กำหนดไว้ ซึ่งทำหน้าที่เป็นข้อตกลงในการเชื่อมต่อโปรแกรม (API Contract) ที่มีความเป็นสากล ทำให้เว็บแอปพลิเคชันซึ่งเป็นผู้รับผิดชอบในการจัดการข้อมูลผู้ใช้ สามารถตรวจสอบฐานข้อมูลของตนเองเพื่อกำหนดสิทธิ์ (Role) และอนุญาตให้ผู้ใช้เข้าถึงทรัพยากรต่าง ๆ ได้อย่างอิสระ จากนั้น จะส่งมอบสถานะการล็อกอินสุดท้ายกลับไปยังผู้ใช้ในรูปแบบของ JSON Web Token (JWT) ซึ่งเป็นการแยกขาดการทำงานออกจากกันอย่างสมบูรณ์

เพื่อให้บริการสามารถบำรุงรักษาได้ง่าย การพัฒนาจึงตั้งอยู่บนหลักการออกแบบเชิงวัตถุ (Object-Oriented Programming - OOP) และใช้ Composer สำหรับการบริหารจัดการส่วนประกอบซอฟต์แวร์ที่จำเป็น (Dependencies) โดยมีการตั้งค่า Autoloading ตามมาตรฐาน PSR-4 เพื่อให้สามารถเรียกใช้คลาสต่าง ๆ ได้โดยอัตโนมัติ

วิธีดำเนินการวิจัย (Methodology)

การพัฒนาบริการกลาง SSO-Authen ได้ดำเนินงานวิจัยโดยมีระเบียบวิธีวิจัยที่สอดคล้องกับวัตถุประสงค์ทั้ง 3 ข้อ ดังนี้

1. วัตถุประสงค์ข้อที่ 1 การพัฒนาบริการกลางแบบไร้สถานะด้วยมาตรฐาน OpenID Connect ใช้วงจรการพัฒนา ระบบ (Software Development Life Cycle - SDLC) เป็นกรอบการทำงานหลัก เพื่อให้กระบวนการสร้างบริการเป็นไป อย่างมีมาตรฐาน

2. วัตถุประสงค์ข้อที่ 2 การออกแบบสถาปัตยกรรมที่ยืดหยุ่น ใช้ระเบียบวิธี การออกแบบสถาปัตยกรรมซอฟต์แวร์ (Software Architecture Design) โดยมุ่งเน้นการออกแบบสถาปัตยกรรมบริการกลาง (Centralized Service) ให้เป็น Authentication Gateway ที่มีความยืดหยุ่นสูง รองรับ Multi-Provider และมีกลไกการจัดทำข้อมูลผู้ใช้ให้เป็นบรรทัดฐาน (Data Normalization) ซึ่งรายละเอียดของสถาปัตยกรรมนี้จะอธิบายในส่วน ระยะเวลาที่ 2 ออกแบบสถาปัตยกรรม (System Design) ต่อไป

3. วัตถุประสงค์ข้อที่ 3 การสร้างต้นแบบและแนวปฏิบัติที่ดี ใช้ระเบียบวิธี การทดสอบการบูรณาการเชิงปฏิบัติ (Practical Integration Testing) โดยนำต้นแบบบริการ (Prototype) ที่พัฒนาขึ้น ไปทดสอบใช้งานจริงกับกลุ่มตัวอย่างเว็บ แอปพลิเคชัน 3 ประเภทที่มีสถาปัตยกรรมแตกต่างกัน เพื่อประเมินผลเชิงปริมาณ (เช่น การเปรียบเทียบระยะเวลา และการ วิเคราะห์ดัชนีความซับซ้อนทางวิศวกรรมซอฟต์แวร์) และเชิงคุณภาพ (เช่น การประเมินประสบการณ์นักพัฒนาตาม มาตรฐานสากล) ซึ่งรายละเอียดของกระบวนการทดสอบนี้จะอธิบายในส่วน ระยะเวลาที่ 4 ทดสอบระบบ (System Testing) ต่อไป

ขั้นตอนการพัฒนา

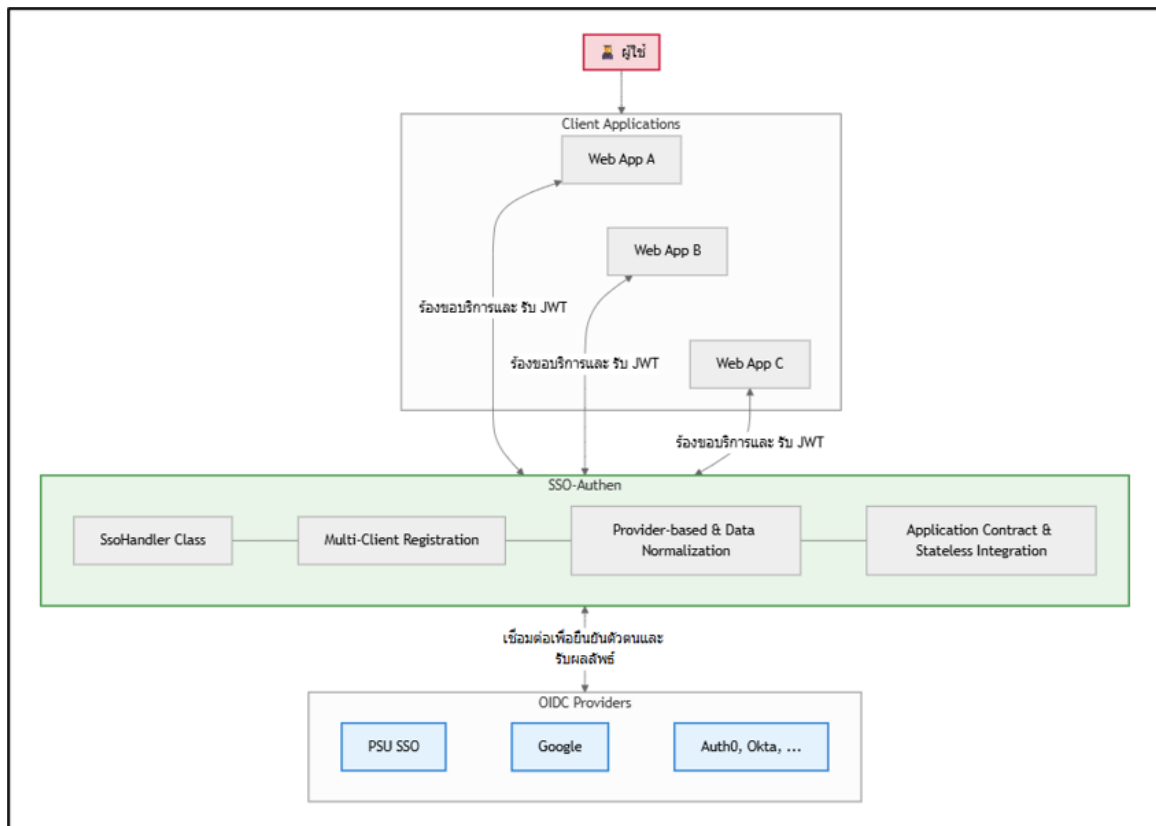
การพัฒนาถูกแบ่งออกเป็น 4 ระยะตามกรอบ SDLC ดังนี้

ระยะที่ 1 วิเคราะห์ความต้องการ (Requirement Analysis)

เริ่มต้นจากการวิเคราะห์สภาพปัญหาในระดับองค์กร ที่มีการให้บริการดิจิทัลบนหลากหลายแพลตฟอร์ม ทั้งระบบที่ พัฒนาขึ้นเอง (In-House Application) และระบบที่จัดหาจากภายนอก เช่น ระบบห้องสมุดอัตโนมัติ แต่ละระบบต่างมีกลไก การยืนยันตัวตนและฐานข้อมูลสมาชิกเป็นของตนเองอย่างกระจัดกระจาย จากการวิเคราะห์ พบว่า สภาพปัญหานี้ก่อให้เกิด ผลกระทบเชิงลบหลายประการ ได้แก่ ประสบการณ์ผู้ใช้ที่ไม่ต่อเนื่อง ความเสี่ยงด้านความปลอดภัยจากการมีฐานข้อมูล รหัสผ่านหลายแห่ง และต้นทุนที่สูงทั้งในด้านเวลาและทรัพยากรในการต้องพัฒนาระบบสมาชิกใหม่ซ้ำซ้อนสำหรับทุกโปรเจกต์ จึงสรุปความต้องการเป็นการสร้างบริการกลางที่เป็นมาตรฐาน สามารถเชื่อมต่อกับ SSO ของมหาวิทยาลัย และให้บริการ ยืนยันตัวตนกับทุกแอปพลิเคชันได้

ระยะที่ 2 ออกแบบสถาปัตยกรรม (System Design)

สถาปัตยกรรมของบริการ SSO-Authen ถูกออกแบบให้เป็นบริการกลาง (Centralized Service) ที่มีองค์ประกอบ หลัก 5 ส่วน (ภาพที่ 2) ดังนี้



ภาพที่ 2 องค์ประกอบของ SSO-Authn

1. Core Component หัวใจหลัก คือ คลาส SSOHandler ซึ่งทำหน้าที่เป็นตัวกลางในการสื่อสารระหว่าง OIDC Provider และ Client Application โดยจัดการขั้นตอนที่ซับซ้อนทั้งหมดของกระบวนการ OIDC ไม่ว่าจะเป็นการสร้างลิงก์สำหรับส่งผู้ใช้ไปยังยืนยันตัวตน การป้องกันการโจมตีแบบ Cross-Site Request Forgery (CSRF) การแลกเปลี่ยน Authorization Code เป็น ID Token ผ่านช่องทางหลังบ้านที่ปลอดภัย การตรวจสอบความถูกต้องของลายเซ็นดิจิทัล (Signature) บน ID Token และขั้นตอนสุดท้าย คือ การสร้างและลงลายเซ็นบน JSON Web Token (JWT) เพื่อส่งมอบให้กับ Client Application

2. Multi-Client Registration เพื่อให้บริการกลางสามารถรองรับแอปพลิเคชันที่หลากหลาย จึงได้ออกแบบและพัฒนาระบบจัดการแอปพลิเคชันผ่านหน้าเว็บ (Admin Panel) สำหรับลงทะเบียน Client Application โดยที่แต่ละแอปพลิเคชันจะถูกกำหนด Client_Id และข้อมูลที่จำเป็นอื่น ๆ เช่น Redirect_Uri และ Api_Secret_Key ทำให้สามารถเพิ่มหรือแก้ไขแอปพลิเคชันที่ต้องการเชื่อมต่อได้อย่างง่ายดาย

3. Provider-Based Configuration & Data Normalization เพื่อให้บริการกลางสามารถรองรับผู้ให้บริการยืนยันตัวตน (Provider) ที่หลากหลาย จึงได้แยกส่วนการตั้งค่าผู้ให้บริการออกจากตรรกะหลักโดยสิ้นเชิง โดยการสร้างไคลเรททอรี Config/Providers สำหรับเก็บไฟล์ตั้งค่าเฉพาะของแต่ละผู้ให้บริการ (เช่น psu.php, cmu.php, google.php, okta.php ฯลฯ) แต่ละไฟล์จะบรรจุข้อมูลที่จำเป็นทั้งหมด ตั้งแต่ Endpoints, Credentials, Scopes ไปจนถึงกระบวนการ Data Normalization กลไกที่ถูกออกแบบขึ้นเพื่อแก้ไขปัญหาความแตกต่างของข้อมูล (Claims) ที่ได้รับจากผู้ให้บริการยืนยันตัวตนแต่ละราย โดยจะทำหน้าที่แปลงข้อมูลผู้ใช้ที่ได้รับให้เป็นโครงสร้างข้อมูลผู้ใช้ที่เป็นมาตรฐานเดียวกันเสมือนดังภาพที่ 3 ทำให้แอปพลิเคชันสามารถนำไปใช้งานได้โดยสะดวกโดยไม่ต้องเขียนเงื่อนไขเพื่อจัดการความแตกต่างของผู้ให้บริการ

```
// Raw claims from PSU
{
  "psu_id": "123456",
  "display_name_th": "นายทดสอบ ระบบ",
  "email": "test@psu.ac.th"
}

// Normalized claims
{
  "id": "123456",
  "name": "นายทดสอบ ระบบ",
  "email": "test@psu.ac.th"
}
```

ภาพที่ 3 ตัวอย่างฟิลด์ข้อมูล (Claims) ที่ถูกแปลงให้เป็นมาตรฐานก่อนนำไปใช้งาน

4. Application Contract & Stateless Integration สถาปัตยกรรมได้กำหนด Application Contract ที่ชัดเจนผ่านการสื่อสารด้วย API Endpoint ซึ่งแอปพลิเคชันจะเป็นผู้สร้างขึ้นตามรูปแบบที่กำหนดไว้ โดยหลังจากที่ SSO-Authen ยืนยันตัวตนผู้ใช้กับ OIDC Provider สำเร็จ ก็จะเรียกไปยัง Endpoint ดังกล่าวพร้อมกับส่งข้อมูลผู้ใช้ที่ผ่านการ Normalize แล้ว (ภาพที่ 4) เพื่อให้แอปพลิเคชันจัดการข้อมูล (เช่น ค้นหาผู้ใช้และกำหนดสิทธิ์) ในฐานข้อมูลของตนเอง และส่งข้อมูลผู้ใช้ในระบบกลับมา กลไกสุดท้ายในการส่งมอบสถานะการล็อกอิน คือ JSON Web Token (JWT) ที่บริการกลางจะสร้างขึ้นโดยบรรจุข้อมูลที่ได้รับจาก API และลงลายเซ็นดิจิทัลก่อนจะส่งกลับไปที่แอปพลิเคชัน การออกแบบนี้ทำให้บริการกลางเป็นอิสระจากตรรกะการจัดการสิทธิ์และเทคโนโลยีของ Client Application อย่างสมบูรณ์

```
"normalizedUser": {
  "id": "user123",
  "username": "test.user",
  "name": "Test User",
  "firstName": "Test",
  "lastName": "User",
  "email": "test.u@example.com",
  "department": "IT Department"
},
"ssoUserInfo": {
  "sub": "user123",
  "preferred_username": "test.user",
  "name": "Test User",
  "email": "test.u@example.com",
  "...": "..."
}
```

ภาพที่ 4 ตัวอย่างข้อมูลผู้ใช้ที่ส่งผ่าน API ไปยังแอปพลิเคชันที่ใช้บริการ SSO-Authen

5. กลไกความปลอดภัยและการจัดการวงจรชีวิตโทเคน (Token Security & Lifecycle Management) เพื่อรักษาความมั่นคงปลอดภัยสารสนเทศและป้องกันความเสี่ยงจากการเข้าถึงข้อมูลโดยไม่ได้รับอนุญาต บริการ SSO-Authen ได้กำหนดมาตรฐานความปลอดภัยเชิงลึก โดยการลงลายเซ็นดิจิทัล (Digital Signature) ด้วยอัลกอริทึม RS256 (Asymmetric Encryption) ซึ่งใช้คู่กุญแจรหัสผ่านแบบอสมมาตรเป็นกลไกหลักในการรักษาความถูกต้องครบถ้วนของข้อมูล (Data Integrity) เพื่อป้องกันการแอบแก้ไขข้อมูลระหว่างทางและการพิสูจน์ที่มาของข้อมูล (Authenticity) เพื่อให้แอปพลิเคชัน

โคลเอนต์มั่นใจได้ว่าโทเค็นถูกออกโดยบริการ SSO-Authen ที่เชื่อถือได้เท่านั้น สำหรับการบริหารจัดการความปลอดภัยได้กำหนดกลไกไว้ 2 ระดับ ดังนี้

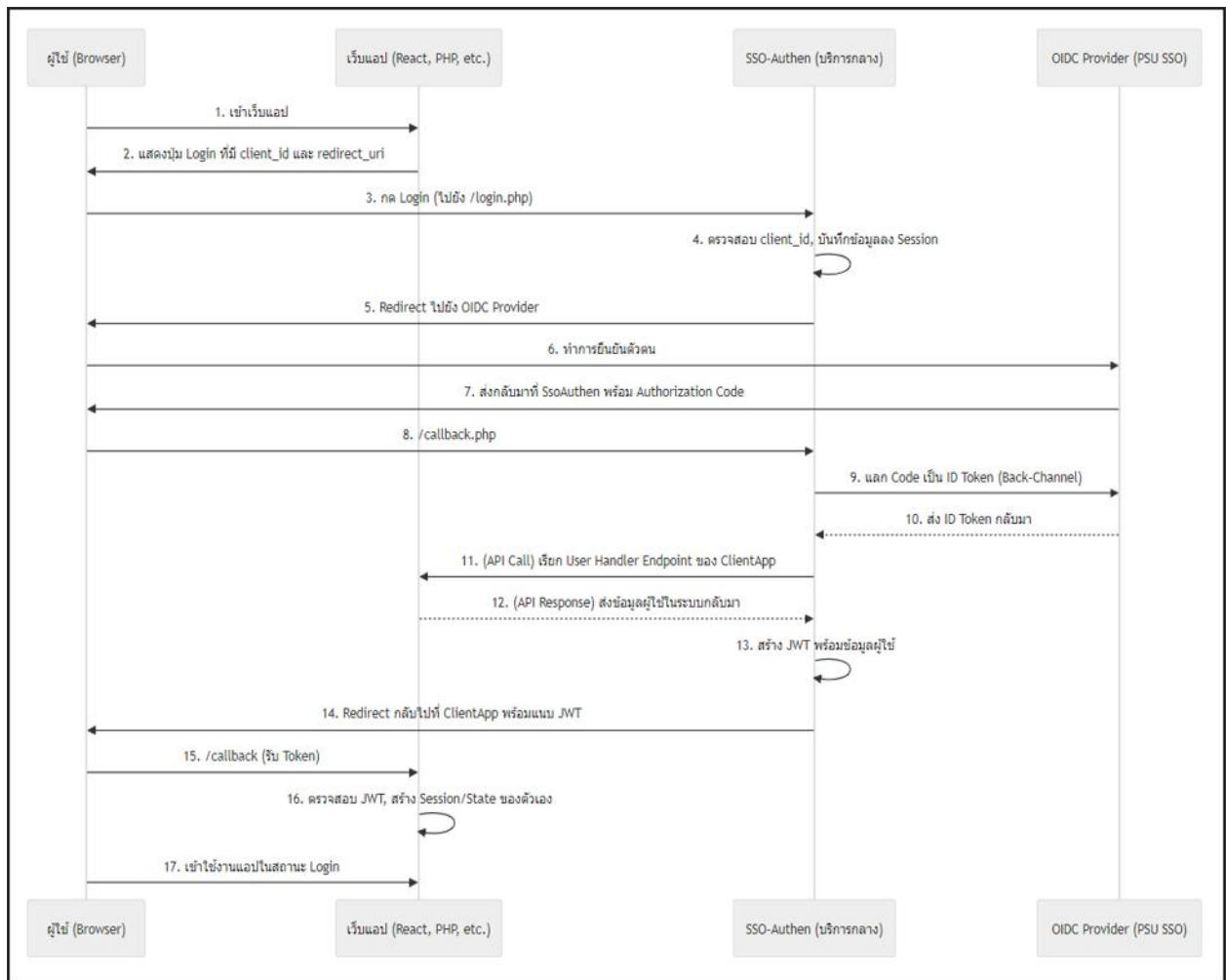
1) การรักษาความปลอดภัยระดับสถานะ (State Security) โดยกำหนดระยะเวลาหมดอายุของโทเค็น (Expiration Time: Exp) ให้มีอายุการใช้งานที่เหมาะสม (Short-Lived Token) โดยเริ่มต้นที่ 1 ชั่วโมง เพื่อลดโอกาสในการนำโทเค็นที่อาจถูกดักจับไปใช้งานซ้ำ (Replay Attack) และเป็นการจำกัดขอบเขตความเสียหายหากโทเค็นหลุดรอดไปสู่บุคคลภายนอก

2) การรักษาความปลอดภัยระดับการจัดเก็บและรับส่ง (Storage & Transport Security) มีการบังคับใช้โปรโตคอล HTTPS ตลอดทั้งกระบวนการเพื่อป้องกันการดักจับข้อมูล (Man-in-the-Middle Attack) พร้อมทั้งแนะนำให้นักพัฒนาแอปพลิเคชันจัดเก็บโทเค็นในรูปแบบคุกกี้ประเภท HttpOnly (HttpOnly Cookie) ซึ่งเป็นกลไกสำคัญในการป้องกันการขโมยโทเค็นผ่านการฉีดสคริปต์อันตรายหรือการโจมตีประเภท Cross-Site Scripting (XSS) ทำให้ระบบมีความปลอดภัยสูงแม้จะทำงานบนสถาปัตยกรรมแบบไร้สถานะ (Stateless) ก็ตาม

ระยะที่ 3 พัฒนาระบบ (System Development)

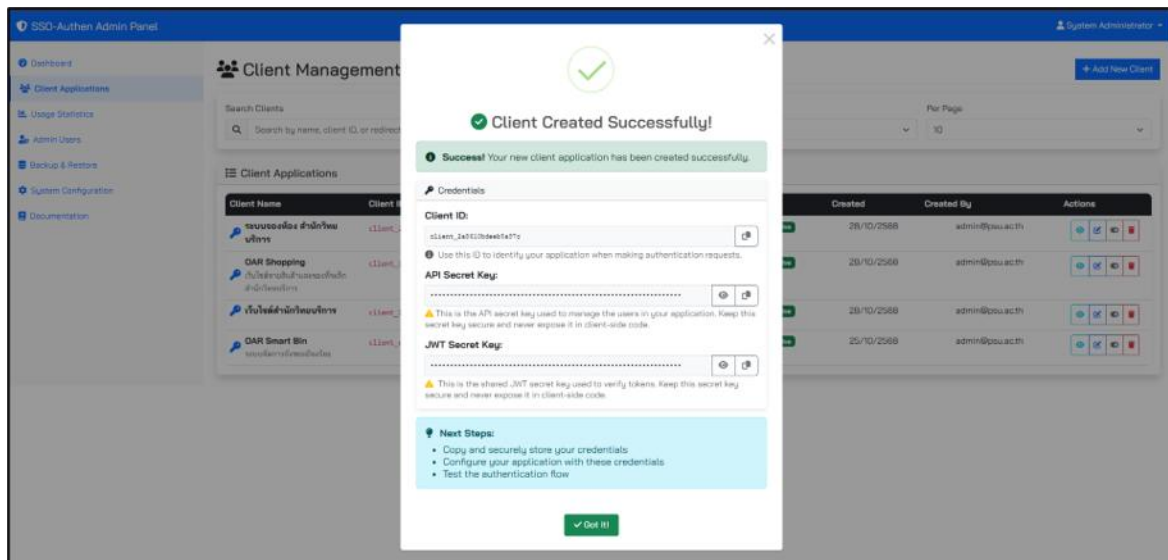
ผู้พัฒนาได้ดำเนินการพัฒนาคลาส SSOHandler และส่วนประกอบอื่น ๆ ตามที่ออกแบบไว้โดยใช้ภาษา PHP และจัดการ Dependencies ทั้งหมดผ่าน Composer และเพื่อเร่งรัดกระบวนการพัฒนาพร้อมรับประกันการทำงานที่สอดคล้องกับมาตรฐาน OIDC จึงได้มีการเลือกใช้ไลบรารี OpenID-Connect-PHP (Jett, 2024) ซึ่งเป็น Open-Source ที่เป็นที่ยอมรับในการจัดการโปรโตคอล OpenID Connect เข้ามาเป็นเครื่องมือหลัก ส่วนการดำเนินงานด้านการเข้ารหัส (Cryptographic Operations) โดยเฉพาะการตรวจสอบลายเซ็นดิจิทัล (Signature) ของ ID Token ซึ่งเป็นรูปแบบ RS256 นั้นได้ใช้ไลบรารี phpseclib (PHP Secure Communications Library) (Wigginton, n.d.) เข้ามาจัดการ รวมทั้งเลือกใช้ไลบรารี php-jwt สำหรับการสร้างและจัดการ JWT (Firebase, 2023) ด้วยเช่นกัน

กระบวนการประมวลผลการตอบรับข้อมูล (HandleCallback) ในบริการ SSO-Authen ถูกพัฒนาให้ทำงานตามลำดับอย่างรัดกุมเพื่อให้เกิดความเชื่อมั่นสูงสุด ดังที่ปรากฏในภาพที่ 5 แผนภาพลำดับขั้นตอน (Sequence Diagram) ซึ่งสรุปขั้นตอนการทำงานได้ ดังนี้



ภาพที่ 5 แผนภาพลำดับการทำงานของบริการกลาง SSO-Authen

- 1. การตรวจสอบสถานะความปลอดภัย (State Verification):** ตรวจสอบค่าสถานะ (State) ที่ส่งกลับมาจากเซิร์ฟเวอร์เพื่อป้องกันการโจมตีประเภทปลอมแปลงคำขอจากเว็บไซต์อื่น (CSRF)
- 2. การแลกเปลี่ยนข้อมูลยืนยันตัวตน:** การนำรหัสอนุญาตชั่วคราว (Authorization Code) ไปแลกเปลี่ยนเป็นข้อมูลยืนยันตัวตนในรูปแบบโทเค็น (ID Token และ Access Token) ผ่านช่องทางสื่อสารส่วนหลัง (Back-Channel) ที่ปลอดภัย
- 3. การตรวจสอบความถูกต้องของลายเซ็น:** ดำเนินการตรวจสอบลายเซ็นดิจิทัลและความถูกต้องของเงื่อนไขต่าง ๆ ภายในโทเค็นที่ได้รับจากผู้ให้บริการ (Provider)
- 4. การจัดทำให้เป็นบรรทัดฐานของข้อมูล (Claim Mapping):** เรียกใช้ฟังก์ชันแปลงข้อมูล (NormalizeClaims) เพื่อเปลี่ยนรูปแบบข้อมูลที่หลากหลายให้กลายเป็นมาตรฐานข้อมูลผู้ใช้ของมหาวิทยาลัย
- 5. การประสานงานแอปพลิเคชันลูกข่าย:** สื่อสารไปยังจุดเชื่อมต่อบริการ (API Endpoint) ของแอปพลิเคชันลูกข่ายเพื่อดำเนินการจัดการสถานะผู้ใช้ในระบบปลายทาง
- 6. การสร้างและส่งมอบสิทธิ์การใช้งาน (JWT Generation):** สร้างข้อมูลในรูปแบบ JSON Web Token ที่บรรจุข้อมูลผู้ใช้ซึ่งผ่านการรับรองแล้ว และดำเนินการส่งผู้ใช้ไปยังหน้าเป้าหมาย (Redirect) เพื่อเริ่มต้นการใช้งานแอปพลิเคชันอย่างสมบูรณ์



ภาพที่ 6 หน้าจอรบบจัดการแอปพลิเคชันของบริการกลาง SSO-Authen ที่ผู้ใช้สามารถลงทะเบียน Client ได้เอง

ระยะที่ 4 ทดสอบระบบ (System Testing)

วางแผนการทดสอบ SSO-Authen ออกเป็น 2 ส่วน เพื่อประเมินประสิทธิภาพและความยืดหยุ่นของสถาปัตยกรรมบริการกลางที่พัฒนาขึ้น ได้แก่

1. การทดสอบเชิงแนวคิด (Proof of Concept) ทดสอบการเชื่อมต่อกับผู้ให้บริการ OIDC สาธารณะที่มีมาตรฐานสูงอย่าง Google, Auth0, และ Okta รวมถึงผู้ให้บริการ OIDC ที่เป็นเป้าหมายจากสำนักนวัตกรรมดิจิทัลและระบบอัจฉริยะมหาวิทยาลัยสงขลานครินทร์ เพื่อพิสูจน์ว่าสถาปัตยกรรม Provider-Based และกลไกแปลงข้อมูล (Claim Mapping) สามารถทำงานกับผู้ให้บริการที่มีความแตกต่างกันได้จริง

2. การทดสอบการบูรณาการจริง (Practical Integration Testing) นำบริการกลางไปทดลองใช้งานจริงกับเว็บแอปพลิเคชัน 3 ประเภทที่มีสถาปัตยกรรมแตกต่างกัน ได้แก่ ระบบ PHP ทั่วไป ระบบจัดการเนื้อหา (CMS) WordPress และระบบ Single-Page Application ที่ใช้ JavaScript Framework สมัยใหม่ (Modern Web App) อย่าง React และ Node.JS เพื่อเก็บข้อมูลเชิงปริมาณและคุณภาพสำหรับประเมินประสิทธิภาพของบริการในสภาพแวดล้อมจริง เพื่อพิสูจน์ว่าสถาปัตยกรรมแบบไร้สถานะ (Stateless) ที่ใช้ JWT สามารถรองรับเทคโนโลยีที่หลากหลายได้จริงตามที่ออกแบบไว้ โดยที่การประเมินเชิงปริมาณ ผู้วิจัยกำหนดเกณฑ์การวัดผลเป็น 2 ด้าน คือ (1) ระยะเวลาในการพัฒนา (Development Time) เปรียบเทียบระหว่างเวลาที่คาดการณ์ตามมาตรฐานการทำงานเดิมของหน่วยงาน (Baseline) กับเวลาที่ใช้จริงในการบูรณาการผ่าน SSO-Authen และ (2) ความซับซ้อนเชิงโครงสร้างซอฟต์แวร์ (Software Complexity) ใช้วิธีวิเคราะห์รหัสคำสั่งแบบสถิติก (Static Code Analysis) ตามหลักวิศวกรรมซอฟต์แวร์ เพื่อเปรียบเทียบความแตกต่างของภาระงานก่อนและหลังการใช้บริการกลาง โดยการมอบหมายให้นักพัฒนาในทีม จำนวน 3 ท่าน เป็นผู้ดำเนินการติดตั้งระบบตัวอย่างเข้ากับบริการกลาง เพื่อลดอคติที่อาจเกิดขึ้นหากผู้วิจัยเป็นผู้ดำเนินการเองและเพื่อให้ได้ข้อมูลที่สะท้อนระยะเวลาในการเรียนรู้และใช้งานจริงมากที่สุด สำหรับการประเมินเชิงคุณภาพด้านประสบการณ์ของนักพัฒนา (Developer Experience) ได้เลือกกลุ่มตัวอย่างแบบเจาะจง (Purposive Sampling) ซึ่งประกอบด้วย ทีมนักพัฒนาขององค์กร จำนวน 8 คน เพื่อรวบรวมข้อมูลและความคิดเห็นเชิงลึกจากผู้ที่มีประสบการณ์ในบริบทของระบบงานโดยตรง

ผลการวิจัยและอภิปรายผลการวิจัย (Result and Discussion)

ผลการวิจัย

จากการดำเนินการวิจัยเพื่อบรรลุวัตถุประสงค์ข้อที่ 1 ผลลัพธ์ที่ได้ คือ บริการ SSO-Authen ซึ่งเป็นบริการยืนยันตัวตนกลางที่พร้อมใช้งาน ประกอบด้วยองค์ประกอบหลัก คือ คลาส SSOHandler ที่จัดการ OIDC Flow และระบบจัดการแอปพลิเคชัน (Admin Panel) สำหรับนักพัฒนา (ดังแสดงในภาพที่ 6) ซึ่งเป็นพื้นฐานสำหรับการทดสอบในส่วนต่อไป

(1) ผลการทดสอบเชิงแนวคิด

ผลการทดสอบในส่วนนี้ดำเนินการเพื่อประเมินการบรรลุวัตถุประสงค์ข้อที่ 2 ด้านความยืดหยุ่นของสถาปัตยกรรม โดยมีวัตถุประสงค์เพื่อพิสูจน์ความยืดหยุ่นของสถาปัตยกรรมแบบ Provider-Based และกลไกการแปลงข้อมูล (Claim Mapping) ผลการทดสอบ พบว่า บริการ SSO-Authen สามารถเชื่อมต่อกับผู้ให้บริการ OIDC ที่มีประเภทแตกต่างกันได้สำเร็จ ทั้ง Google (ผู้ให้บริการสาธารณะ) Auth0 และ Okta (แพลตฟอร์ม Identity as a Service) และ PSU SSO (ผู้ให้บริการระดับองค์กร) โดยการแก้ไขเพียงไฟล์คอนฟิกของผู้ให้บริการเท่านั้น ซึ่งเป็นการยืนยันว่าสถาปัตยกรรมที่ออกแบบไว้มีความยืดหยุ่นและเป็นไปตามมาตรฐานสากลจริง ดังแสดงในตารางที่ 1 ซึ่งเปรียบเทียบชื่อฟิลด์ข้อมูล (Claims) ที่แตกต่างกันของผู้ให้บริการแต่ละรายและความสามารถของกลไกการแปลงข้อมูล (Claim Mapping) ที่สามารถจัดการและรวบรวมข้อมูลจากแหล่งที่แตกต่างกันให้กลายเป็นมาตรฐานข้อมูลอัตลักษณ์ชุดเดียวกันได้อย่างสมบูรณ์

ตารางที่ 1 เปรียบเทียบ Claim ของผู้ให้บริการ OIDC และการตั้งค่า Claim_Mapping ของ SSO-Authen

Key มาตรฐานของ SSO-Authen	Google	Auth0	Okta	PSU SSO
id	sub	sub	sub	psu_id
username	email	nickname	preferred_username	preferred_username
name	name	name	name	display_name_th
firstName	given_name	given_name	given_name	first_name_th
lastName	family_name	family_name	family_name	last_name_th
email	email	email	email	email

(2) ผลการทดสอบการบูรณาการจริง ผลการทดสอบในส่วนนี้เป็นการประเมินประสิทธิภาพของระบบต้นแบบ เพื่อตอบวัตถุประสงค์ข้อที่ 1 และ 3 ด้านการสร้างบริการกลางและแนวปฏิบัติที่ดีสำหรับการบูรณาการ โดยได้มุ่งเน้นการวัดผลกระทบของบริการ SSO-Authen ต่อกระบวนการพัฒนาของฝั่ง Client Application ใน 4 ด้านหลัก คือ การลดระยะเวลาในการพัฒนา การลดความซับซ้อนทางวิศวกรรมซอฟต์แวร์ ประสบการณ์ของนักพัฒนา และความยืดหยุ่นในการบำรุงรักษา

(3) การลดระยะเวลาในการพัฒนา (Development Time Reduction) ผลการทดลอง พบว่า บริการ SSO-Authen สามารถลดระยะเวลาที่นักพัฒนาต้องใช้ในการสร้างกลไกรับยืนยันตัวตนลงได้อย่างมีนัยสำคัญ เนื่องจากนักพัฒนาไม่จำเป็นต้องศึกษาความซับซ้อนของโปรโตคอล OIDC แต่จะมุ่งเน้นเพียงการพัฒนาส่วนเชื่อมต่อกับรูปแบบที่กำหนดไว้เท่านั้น โดยเปลี่ยนจากระยะเวลาประเมินเฉลี่ย 51.6 ชั่วโมง (สำหรับการพัฒนาขึ้นเองแบบเดิม) เหลือเพียงระยะเวลาที่ใช้จริงเฉลี่ย 13.3 ชั่วโมง หรือลดลงเฉลี่ย 78% ซึ่งเป็นหลักฐานเชิงประจักษ์ที่ยืนยันว่าการใช้บริการกลางช่วยเร่งกระบวนการส่งมอบบริการดิจิทัลใหม่สู่ผู้ใช้ได้รวดเร็วยิ่งขึ้นกว่าการไม่ใช้บริการกลางถึงประมาณ 38 ชั่วโมงต่อหนึ่งโครงการพัฒนา

เพื่อให้ผลการเปรียบเทียบระยะเวลามีความน่าเชื่อถือเชิงวิชาการ ผู้วิจัยได้กำหนดวิธีการและที่มาของข้อมูลการประเมินเชิงประจักษ์ (Empirical Evaluation) ดังนี้

1. **ที่มาของเวลาประเมิน (Baseline)** อ้างอิงจากข้อมูลสถิติประวัติการทำงานย้อนหลังของโครงการพัฒนาระบบสารสนเทศภายในหน่วยงานจำนวน 9 โครงการ (แบ่งเป็นแอปพลิเคชัน 3 ประเภท ประเภทละ 3 โครงการ)

2. **ผู้ประเมินและผู้ทดสอบ** ดำเนินการทดสอบโดยทีมนักพัฒนาของหน่วยงาน จำนวน 3 ท่าน ซึ่งเป็นตัวแทนของผู้ที่มีระดับประสบการณ์แตกต่างกัน ได้แก่ นักพัฒนาระดับอาวุโส (ประสบการณ์ 8 ปี รับผิดชอบระบบ PHP แบบดั้งเดิม) นักพัฒนาระดับกลาง (ประสบการณ์ 5 ปี รับผิดชอบระบบ CMS WordPress) และนักพัฒนาระดับปฏิบัติการ (ประสบการณ์ 3 ปี รับผิดชอบระบบ Single-Page Application)

3. **วิธีการประเมิน** ใช้การบันทึกเวลาปฏิบัติงานจริงนับตั้งแต่ขั้นตอนการศึกษาคู่มือ API Contract การเขียนโค้ดเชื่อมต่อ การทดสอบระบบ ไปจนถึงการนำขึ้นใช้งานจริง

การควบคุมตัวแปรและวิธีการวัดผลที่ชัดเจนนี้ ช่วยยืนยันว่าข้อมูลระยะเวลาการพัฒนาที่ลดลงสามารถสะท้อนประสิทธิภาพของการใช้บริการ SSO-Authen ในสภาพแวดล้อมการทำงานจริงได้อย่างเป็นรูปธรรม (ตารางที่ 2)

ตารางที่ 2 เปรียบเทียบระยะเวลาในการพัฒนาระบบยืนยันตัวของแอปพลิเคชันประเภทต่าง ๆ

ประเภทแอปพลิเคชัน	เวลาประเมิน (ชั่วโมง)	เวลาที่ใช้จริง (ชั่วโมง)	ระยะเวลาที่ลดลง (%)
PHP	40–60	8–12	80%
WordPress	50–70	12–18	76%
React/Node.JS	45–65	10–16	78%

(4) **การลดความซับซ้อนทางวิศวกรรมซอฟต์แวร์ (Software Complexity Reduction)** เพื่อให้การประเมินมีความน่าเชื่อถือตามหลักวิชาการวิศวกรรมซอฟต์แวร์ นอกเหนือจากการนับจำนวนบรรทัดของโค้ด (Lines of Code – LOC) ผู้วิจัยได้นำทฤษฎีความซับซ้อนไซโคลเมตริก (Cyclomatic Complexity: M) ของ McCabe (1976) และศาสตร์แห่งซอฟต์แวร์ของฮัลสเตด (Halstead, 1977) เข้ามาใช้วิเคราะห์โครงสร้างการทำงานฝั่งไคลเอนต์ร่วมด้วย โดยเปรียบเทียบระหว่างการเขียนโค้ดเพื่อเชื่อมต่อ OIDC โดยตรงกับการเชื่อมต่อผ่านบริการ SSO-Authen พบว่า บริการกลางสามารถลดความซับซ้อนในทุกมิติทางวิศวกรรมซอฟต์แวร์ได้อย่างมีนัยสำคัญ (ตารางที่ 3)

ตารางที่ 3 เปรียบเทียบดัชนีชี้วัดความซับซ้อนของซอฟต์แวร์ฝั่งแอปพลิเคชันไคลเอนต์ (Software Metrics)

มิติและตัวชี้วัดทางวิศวกรรมซอฟต์แวร์	โค้ดเชื่อมต่อ OIDC แบบเดิม (Baseline)	โค้ดเชื่อมต่อผ่าน SSO-Authen	อัตราการลดลง (%)
จำนวนบรรทัดคำสั่ง (LOC)	เฉลี่ย 420 บรรทัด	เฉลี่ย 80 บรรทัด	81%
ความซับซ้อนไซโคลเมตริก (Cyclomatic Complexity: M)	$M = 9$ (ซับซ้อนปานกลาง)	$M = 2$ (ซับซ้อนต่ำมาก)	78%
ขนาดของโปรแกรม (Halstead Program Volume: V)	2,840	480	83%
ความพยายามในการพัฒนา (Halstead Program Effort: E)	18,200	1,950	89%

ผลการทดสอบแสดงให้เห็นความสัมพันธ์ระหว่างการออกแบบสถาปัตยกรรมและประสิทธิภาพการทำงานอย่างชัดเจน โดยกลไกการแปลงข้อมูล (Claim Mapping) ในตารางที่ 1 เป็นปัจจัยหลักที่ช่วยลดภาระของนักพัฒนา ความสะดวกลดลงออกมาในระยะเวลาการพัฒนา (ตารางที่ 2) ที่ลดลงเฉลี่ยถึง 78% และความซับซ้อนทางวิศวกรรมซอฟต์แวร์ (ตารางที่ 3)

ที่ดัชนีทุกตัวลดลงกว่า 80% เนื่องจากความซับซ้อนของการจัดการ OIDC Flow และ JSON Web Token (JWT) ถูกห่อหุ้มและจัดการโดยคลาส SSOHandler ทั้งหมด ทำให้นักพัฒนามุ่งเน้นเพียงการเรียกใช้ API Contract ที่กำหนดไว้เท่านั้น

(5) **ประสบการณ์ของนักพัฒนา (Developer Experience – DX)** เพื่อให้การประเมินผลประสบการณ์การทำงานของนักพัฒนามีความน่าเชื่อถือเชิงวิชาการตามมาตรฐานวิศวกรรมซอฟต์แวร์ ผู้วิจัยได้ประยุกต์ใช้กรอบแนวคิด SPACE Framework (Forsgren et al., 2021) ซึ่งเป็นกรอบการวัดประสิทธิภาพและประสบการณ์การทำงานของนักพัฒนาที่ได้รับ การยอมรับในระดับสากล มาใช้เป็นโครงสร้างในการออกแบบเกณฑ์การประเมินร่วมกับทีมนักพัฒนาที่ทำการทดสอบทั้ง 8 ท่าน โดยได้จัดกลุ่มข้อคำถามการประเมินให้สอดคล้องกับมิติของ SPACE Framework ดังนี้

1. **มิติด้านความพึงพอใจและสุขภาวะ (Satisfaction and Well-Being - S)** ประเมินจากความชัดเจนของเอกสาร และความปลอดภัยในการใช้งาน ซึ่งส่งผลต่อความมั่นใจและลดความเครียดในการเชื่อมต่อบริการ

2. **มิติด้านกิจกรรม (Activity - A)** ประเมินจากความง่ายในการสร้าง API Endpoint ซึ่งสะท้อนถึงการลดลงของปริมาณภาระงาน (Workload) ที่นักพัฒนาต้องเขียนโค้ดเพิ่มเติม

3. **มิติด้านประสิทธิภาพและความลื่นไหล (Efficiency and Flow - E)** ประเมินจากความง่ายในการติดตั้งและตั้งค่า ซึ่งสะท้อนถึงความคล่องตัวในการทำงานโดยไม่เกิดความสับสน

จากผลการประเมิน (ตารางที่ 4) พบว่า นักพัฒนาส่วนใหญ่มีความพึงพอใจในระดับ “ดีมาก” ในทุกมิติ โดยเฉพาะ มิติด้านความพึงพอใจและสุขภาวะ (S) ที่นักพัฒนาให้ความเห็นสอดคล้องกันว่า API Contract ที่ออกแบบไว้ และความชัดเจนของเอกสารคู่มือ ช่วยให้นักพัฒนามั่นใจในความปลอดภัยและสามารถนำทอเค็น (JWT) ไปใช้งานต่อในแอปพลิเคชันได้ทันที โดยไม่ต้องลองผิดลองถูก

ตารางที่ 4 ผลการประเมินความพึงพอใจของนักพัฒนา (จำนวนคน)

เกณฑ์การประเมิน	ดีมาก (5)	ดี (4)	พอใช้ (3)
ความง่ายในการติดตั้งและตั้งค่า	6	2	0
ความง่ายในการสร้าง API Endpoint	5	3	0
ความปลอดภัยในการใช้งาน	7	1	0
ความชัดเจนของเอกสาร	7	1	0

(6) **ความยืดหยุ่นในการบำรุงรักษา (Maintainability)** จุดเด่นที่สุดของสถาปัตยกรรมบริการกลาง SSO-Authen คือ ความง่ายในการบำรุงรักษา จากการจำลองสถานการณ์เปลี่ยนผู้ให้บริการยืนยันตัวตนจาก PSU SSO เป็น Google ผลการทดลองยืนยันว่าสามารถเปลี่ยนผู้ให้บริการได้สำเร็จภายในระยะเวลาเพียง 5 นาที ด้วยการแก้ไขไฟล์คอนฟิกในบริการกลางเพียงจุดเดียว โดยที่ไม่ต้องแก้ไขโค้ดของเว็บแอปพลิเคชันที่มาเชื่อมต่อเลยแม้แต่บรรทัดเดียว ซึ่งเป็นการพิสูจน์ถึงความยืดหยุ่นของสถาปัตยกรรมอย่างชัดเจน (ภาพที่ 7)

```

<?php
// PHP
// ก่อน (PSU): config/providers/psu.php
return [
    'clientID' => 'psu_client_id',
    'clientSecret' => 'psu_secret',
    'providerURL' => 'https://psusso.psu.ac.th/',
    'claim_mapping' => [
        'id' => 'psu_id',
        'name' => 'display_name_th',
        'email' => 'email'
    ]
];

// หลัง (Google): config/providers/google.php
return [
    'clientID' => 'google_client_id',
    'clientSecret' => 'google_secret',
    'providerURL' => 'https://accounts.google.com/',
    'claim_mapping' => [
        'id' => 'sub',
        'name' => 'name',
        'email' => 'email'
    ]
];

// config/config.php - เปลี่ยนแค่บรรทัดเดียว
$activeProvider = 'google'; // เปลี่ยนจาก 'psu'

```

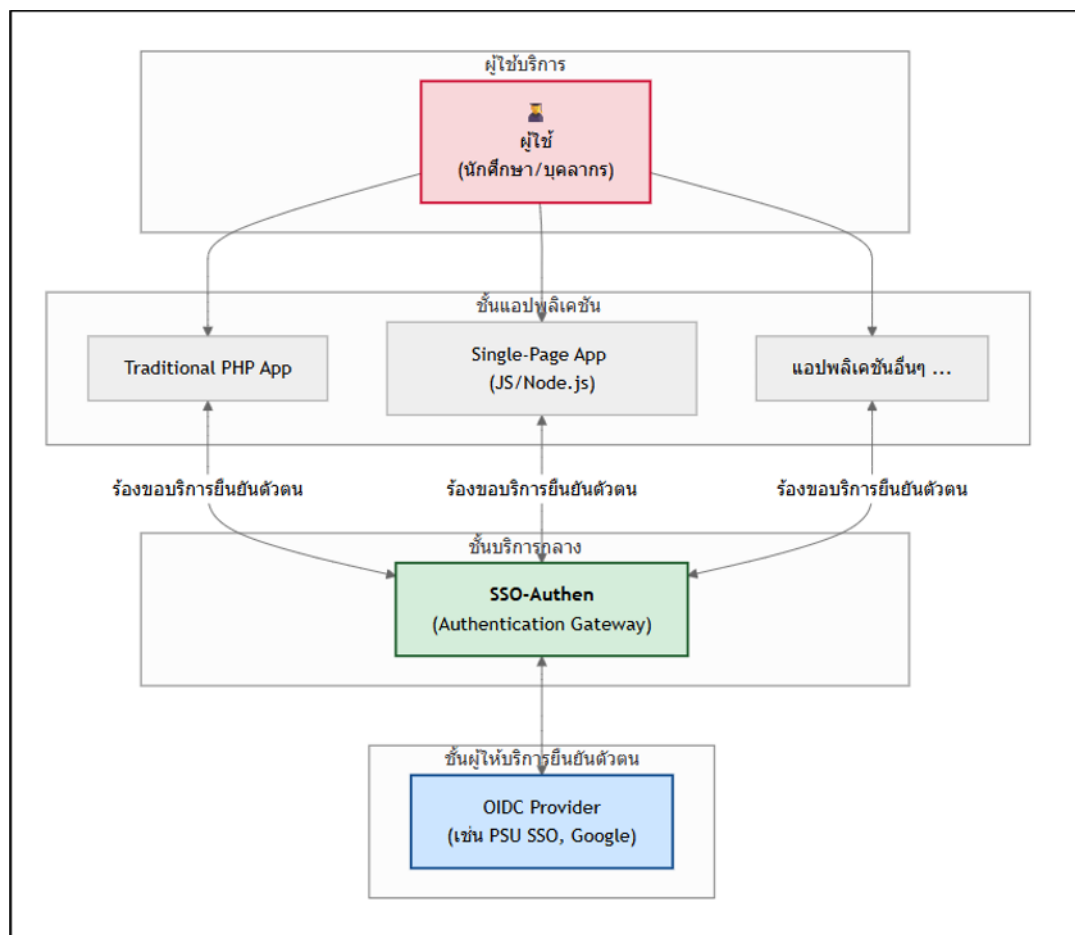
ภาพที่ 7 การสับเปลี่ยน IdP จาก PSU เป็น Google ของระบบ SSO-Authen

อภิปรายผลการวิจัย

ผลการทดลองใช้งานจริงได้แสดงให้เห็นถึงประสิทธิภาพของบริการ SSO-Authen ในการแก้ปัญหาการยืนยันตัวตนที่กระจัดกระจายได้อย่างเป็นรูปธรรม การลดระยะเวลาในการพัฒนาได้เฉลี่ยถึง 78% และลดความซับซ้อนของโค้ดที่ฝั่ง Client Application ลงกว่า 81% ถือเป็นการเปลี่ยนแปลงกระบวนการพัฒนาย่างมีนัยสำคัญ การลดลงของความซับซ้อนนี้ได้รับการยืนยันความน่าเชื่อถือผ่านการวิเคราะห์ด้วยทฤษฎีวิศวกรรมซอฟต์แวร์ (ตารางที่ 3) โดยตามทฤษฎีของ McCabe พบว่า ระบบแบบเดิมมีค่าความซับซ้อนไซโคลเมตริกสูงถึง $M = 9$ เนื่องจากแอปพลิเคชันโคลเอนด์ต้องจัดการเงื่อนไขการตัดสินใจ (If-Else) หลายเส้นทาง เช่น การตรวจสอบค่าสถานะและการตรวจสอบลายเซ็นดิจิทัล ในขณะที่แอปพลิเคชันที่ใช้ SSO-Authen มีค่า M ลดเหลือเพียง 2 เนื่องจากโครงสร้างโค้ดถูกเปลี่ยนเป็นรูปแบบเส้นตรง (Linear Flow) สอดคล้องกับดัชนีของฮัลสเต็ดที่ค่าความพยายามในการพัฒนา (E) ลดลงถึง 89% ซึ่งสะท้อนให้เห็นว่าสถาปัตยกรรมของ SSO-Authen ประสบความสำเร็จในการผลักดันความซับซ้อนทางเทคนิค (Complexity Offloading) จากฝั่งลูกข่ายไปบริหารจัดการแบบรวมศูนย์ที่จุดเดียว เมื่อเปรียบเทียบกับเกณฑ์เวลาประเมินมาตรฐาน (Baseline) ในตารางที่ 2 ซึ่งระบุว่าพัฒนาระบบยืนยันตัวตนแบบเดิมต้องใช้เวลาเฉลี่ยประมาณ 45–65 ชั่วโมงต่อโครงการ การที่ SSO-Authen สามารถลดเวลาการทำงานจริงลงเหลือเพียงประมาณ 8–18 ชั่วโมง (หรือลดลงเฉลี่ย 38 ชั่วโมงต่อโครงการ) เป็นหลักฐานเชิงประจักษ์ที่ตอบโจทยความต้องการขององค์กรในการเร่งส่งมอบบริการดิจิทัลใหม่ ๆ ผู้ใช้ได้รวดเร็วยิ่งขึ้น ปัจจัยสำคัญที่ช่วยลดระยะเวลานี้ คือ ระบบ Admin Panel ที่ช่วยให้นักพัฒนาสามารถลงทะเบียนแอปพลิเคชันและรับ Credential ได้ทันทีด้วยตนเองโดยไม่ต้องรอผู้ดูแลระบบกลางดำเนินการให้ ช่วยให้นักพัฒนามุ่งเน้นไปที่การพัฒนาฟังก์ชันหลักของแอปพลิเคชันได้โดยตรง

สถาปัตยกรรมบริการกลางนี้ได้เข้ามาตอบโจทย์ประเด็นปัญหาที่กล่าวถึงในบทนำได้อย่างตรงจุด ทั้งในด้านผู้ใช้บริการที่ไม่จำเป็นต้องจดจำรหัสผ่านสำหรับแต่ละระบบงานอีกต่อไป ด้านความปลอดภัยที่ลดความเสี่ยงจากการมีฐานข้อมูล Credential หลายแห่ง และในด้านการพัฒนาที่ช่วยลดภาระของนักพัฒนาลงได้อย่างมาก ซึ่งสะท้อนผ่านผลการประเมินประสิทธิภาพของนักพัฒนา (DX) ที่อยู่ในระดับที่ดีมาก อย่างไรก็ตาม การส่งมอบข้อมูลผ่าน JSON Web Token (JWT) ในสถาปัตยกรรมแบบไร้สถานะมีความท้าทายสำคัญเรื่องความมั่นคงปลอดภัย ซึ่งงานวิจัยนี้ได้เสนอแนวทางการจัดการผ่านกลไกความปลอดภัย 2 ระดับ คือ 1) การใช้ลายเซ็นดิจิทัลแบบ RS256 ร่วมกับการกำหนดอายุการใช้งานของโทเค็น (Short-Lived Token) เพื่อจำกัดช่วงเวลาความเสี่ยงหากเกิดกรณีโทเค็นหลุดรอด และ 2) การบังคับใช้งานผ่านโปรโตคอล HTTPS ร่วมกับการจัดเก็บโทเค็นแบบ HttpOnly Cookie เพื่อป้องกันการถูกดักจับข้อมูลและการโจมตีผ่านสคริปต์จากภายนอกหรือ Cross-Site Scripting (XSS) ซึ่งถือเป็นแนวปฏิบัติที่ช่วยยกระดับความปลอดภัยให้สูงกว่าการปล่อยให้ นักพัฒนาแต่ละรายออกแบบระบบเอง

ความสำเร็จในการบูรณาการกับระบบที่มีความหลากหลาย ตั้งแต่ PHP แบบดั้งเดิม, WordPress, ไปจนถึง Single-Page Application สมัยใหม่ ได้พิสูจน์ให้เห็นถึงความแข็งแกร่งของสถาปัตยกรรมที่ออกแบบไว้ ซึ่งไม่ใช่เพียงการสื่อสารผ่าน API Endpoint ที่ยืดหยุ่นเท่านั้น แต่ยังมียอดประกอบที่สำคัญอีกส่วนหนึ่ง นั่นคือการสร้างสถาปัตยกรรมแบบไร้สถานะ (Stateless) ที่ใช้ JSON Web Token (JWT) เป็นกลไกในการส่งมอบสถานะการล็อกอิน การเปลี่ยนแปลงนี้ช่วยลดข้อจำกัดทางเทคโนโลยีได้อย่างสิ้นเชิง ทำให้ SSO-Authn สามารถทำหน้าที่เป็นโครงสร้างพื้นฐานด้านการยืนยันตัวตน (Authentication Infrastructure) ที่เป็นมาตรฐานให้แก่เว็บแอปพลิเคชันทุกรูปแบบในองค์กรได้อย่างยั่งยืน ดังโครงสร้างที่แสดงในภาพที่ 8



ภาพที่ 8 โครงสร้างสถาปัตยกรรมการยืนยันตัวตนใหม่ที่มี SSO-Authn เป็นสื่อกลาง

สรุปผลการวิจัย (Conclusion)

การพัฒนาบริการ SSO-Authen ประสบความสำเร็จตามวัตถุประสงค์ที่ตั้งไว้ทุกประการ โดยได้สร้างบริการยืนยันตัวตนกลางที่เป็นมาตรฐาน ยืดหยุ่น และสามารถบูรณาการเข้ากับเว็บแอปพลิเคชันที่หลากหลายได้จริง ผลจากการทดสอบได้พิสูจน์แล้วว่าสถาปัตยกรรมที่พัฒนาขึ้นสามารถลดต้นทุนทั้งด้านเวลาและความซับซ้อนในการพัฒนาได้อย่างมหาศาล หัวใจสำคัญของความสำเร็จนี้อยู่ที่การออกแบบสถาปัตยกรรมให้เป็นบริการกลางที่ทำงานแบบไร้สถานะ (Stateless) และใช้ JSON Web Token (JWT) เป็น "สัญญา" ในการบูรณาการ สิ่งนี้ทำให้ SSO-Authen สามารถทำหน้าที่เป็นโครงสร้างพื้นฐานด้านการยืนยันตัวตน (Authentication Infrastructure) ที่พร้อมรองรับการเติบโตของบริการดิจิทัลในอนาคต โดยไม่ถูกจำกัดด้วยแพลตฟอร์มการพัฒนาอีกต่อไป

ข้อเสนอแนะ (Suggestions)

1. ข้อเสนอแนะในการนำผลการวิจัยไปใช้

การสร้างสถาปัตยกรรม Authentication Gateway ในองค์กร

สถาปัตยกรรมของ SSO-Authen ได้ถูกออกแบบมาเพื่อนำไปติดตั้งและใช้งานในฐานะ "Authentication Gateway" หรือ Portal กลางสำหรับทั้งองค์กร (เช่น auth.your-org.com) โดย Portal กลางนี้จะจัดการ OIDC Client เพียงชุดเดียวที่เชื่อมต่อกับผู้ให้บริการยืนยันตัวตน (Provider) จากนั้น เว็บแอปพลิเคชันภายในทั้งหมดที่ไม่ว่าจะพัฒนาด้วยเทคโนโลยีใดก็จะทำการเชื่อมต่อกับ Portal กลางนี้ผ่าน Application Contract ที่กำหนดไว้ แทนที่แต่ละแอปจะต้องเชื่อมต่อกับผู้ให้บริการโดยตรง สถาปัตยกรรมนี้จะช่วยรวมศูนย์การจัดการ ลดภาระของนักพัฒนา เร่งความเร็วในการพัฒนาระบบใหม่ และที่สำคัญคือช่วยลดพื้นผิวการโจมตี (Attack Surface) ให้กับขององค์กรได้อย่างมีนัยสำคัญ

2. ข้อเสนอแนะในการวิจัยครั้งต่อไป

การพัฒนาต่อยอด

- พัฒนาเป็น Private Composer Package เพื่อให้การติดตั้งและอัปเดตเวอร์ชันของบริการ SSO-Authen ในอนาคตทำได้ง่ายและเป็นมาตรฐานยิ่งขึ้นผ่านคำสั่ง Composer

- การรองรับ Single Sign-Out โดยการเพิ่มฟังก์ชันเพิ่มเติมเพื่อจัดการการออกจากระบบจากทุกแอปพลิเคชันพร้อมกัน เมื่อผู้ใช้ออกจากระบบที่ Provider กลาง (Jones & Bradley, 2022)

การประยุกต์ใช้ในวงกว้าง

สถาปัตยกรรม Multi-Provider ของ SSO-Authen สามารถใช้เป็นต้นแบบในการพัฒนาระบบ "Identity Broker" ที่เชื่อมต่อนับระบบ SSO ของมหาวิทยาลัยหลายแห่งเข้าไว้ด้วยกัน เพื่อการให้บริการยืนยันตัวตนข้ามสถาบัน (Primbs & Menth, 2024) เช่น การสร้างบัตรนักศึกษาเสมือน (Virtual Card) สำหรับเข้าใช้บริการห้องสมุดที่เป็นสมาชิกในข่ายงานห้องสมุดมหาวิทยาลัยส่วนภูมิภาค (PULINET) ได้ต่อไป

กิตติกรรมประกาศ (Acknowledgements)

ผลงานวิจัยนี้สำเร็จลงได้ด้วยดีโดยได้รับความอนุเคราะห์และการสนับสนุนจากคณะผู้บริหาร หัวหน้าฝ่าย หัวหน้ากลุ่มงาน และบุคลากรกลุ่มงานพัฒนาระบบสารสนเทศและเทคโนโลยีสารสนเทศ สำนักวิทยบริการ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตปัตตานี ผู้วิจัยขอขอบคุณ ผู้ช่วยศาสตราจารย์ ดร.สมพร ช่วยอารีย์ ผู้อำนวยการสำนักวิทยบริการ และผู้ช่วยศาสตราจารย์ ดร.กิตติพงศ์ ช่างลอยเลื่อน รองผู้อำนวยการสำนักวิทยบริการ ฝ่ายดิจิทัลและนวัตกรรมการเรียนรู้ สำหรับคำปรึกษาและข้อเสนอแนะอันเป็นประโยชน์ยิ่ง และขอขอบคุณ นางจุฑารัตน์ ปานผดุง หัวหน้าฝ่ายหอสมุดจอห์น เอฟ. เคนเนดี ที่ได้อนุเคราะห์ตรวจสอบความถูกต้องของเนื้อหาและรายการอ้างอิง ตลอดจนสำนักวิทยบริการ ที่ได้ให้การสนับสนุนด้านทรัพยากรในการดำเนินงานวิจัยครั้งนี้

รายการอ้างอิง (References)

- กุลประวีณ์ พรหมอารีย์. (2568). การพัฒนากระบวนการเบิกจ่ายเงินด้วยระบบอิเล็กทรอนิกส์สู่สำนักงานอภัยุริยะโดยประยุกต์ใช้ Design Thinking และ ADKAR Model. *PULINET Journal*, 12(1), 89-103.
- ปิติ สุวรรณาคม และอุทาน บุณศักดิ์ศรี. (2566). การพัฒนาระบบพิสูจน์ตัวตนโดยไม่ใช้รหัสผ่านบนโปรโตคอล OpenID Connect. *วิจัยและพัฒนาบนฐานเศรษฐกิจ BCG สู่การพัฒนาประเทศอย่างยั่งยืน. การประชุมวิชาการระดับชาติ ครั้งที่ 15 มหาวิทยาลัยราชภัฏนครปฐม* (น. 548-556). สถาบันวิจัยและพัฒนา มหาวิทยาลัยราชภัฏนครปฐม.
- ปิยะบุตร ปัญญาดี และอภิสิทธิ์ วรรณ. (2567). การพัฒนาระบบตรวจสอบการเชื่อมโยงข้อมูลทรัพยากรสารสนเทศอิเล็กทรอนิกส์ของสำนักหอสมุด มหาวิทยาลัยเชียงใหม่. *PULINET Journal*, 12(1), 158-173.
- วิไล จันทรแก้ว, เจนจิรา คำหล่อ, และสุดา ใจแก้ว. (2567). ระบบบริหารจัดการ “บริการบอร์ดเกม (Board Game)”. *PULINET Journal*, 12(1), 1-14.
- สำนักงานพัฒนารัฐบาลดิจิทัล (องค์การมหาชน). (2562). คู่มือประกอบการพัฒนาระบบ: การพัฒนาระบบสารสนเทศภาครัฐให้เชื่อมโยงกับระบบยืนยันตัวบุคคลกลางแบบรวมศูนย์ (Single Sign-On). <https://www.dga.or.th/document-sharing/40/>
- สำนักงานพัฒนารัฐบาลดิจิทัล (องค์การมหาชน). (2566). แผนพัฒนารัฐบาลดิจิทัลของประเทศไทย พ.ศ. 2566-2570. <https://www.dga.or.th/policy-standard/policy-regulation/dga-019/dga-027/dg-plan-2566-2570/>
- Firebase. (2023). *firebase/php-jwt*. GitHub. <https://github.com/firebase/php-jwt>
- Forsgren, N., Storey, M. A., Maddila, C., Zimmermann, T., Houck, B., & Ford, J. (2021). The SPACE of developer productivity: There's more to it than you think. *Queue*, 19(1), 20-48. <https://doi.org/10.1145/3454122.3454124>
- Gupta, A. (2024). *An Introduction to OpenID Connect*. IDPro. https://www.researchgate.net/publication/387173144_An_Introduction_to_OpenID_Connect
- Halstead, M. H. (1977). *Elements of software science*. Elsevier Science Inc.
- Jett, M. (2024, September 13). *OpenID Connect PHP v1.0.2*. GitHub. <https://github.com/jumbojett/OpenID-Connect-PHP>
- Jones, M., & Bradley, J. (2022). *OpenID Connect back-channel logout 1.0 - draft 07*. OpenID Foundation. https://openid.net/specs/openid-connect-backchannel-1_0-07.html
- McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on Software Engineering*, SE-2(4), 308-320. <https://doi.org/10.1109/TSE.1976.233837>
- Primbs, J., & Menth, M. (2024). OIDC2: Open identity certification with OpenID Connect. *IEEE Open Journal of the Communications Society*. <https://doi.org/10.1109/OJCOMS.2024.3376193>
- Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., & Mortimore, C. (2023). *OpenID Connect Core 1.0 incorporating errata set 2*. OpenID Foundation. https://openid.net/specs/openid-connect-core-1_0.html
- Wigginton, J. (n.d.). *Why phpseclib?*. phpseclib. <https://phpseclib.com/docs/why>