ภาคผนวก ค
-ร่าง-
บทความฉบับสมบูรณ์
เพื่อเข้าร่วม
งานประชุมวิชาการระดับชาติ วิทยาศาสตร์ เทคโนโลยีและนวัตกรรม ครั้งที่ 6
https://sciencebase.mju.ac.th/CSTI2025/

การทดสอบระบบบริหารจัดการงานสนับสนุนภายในองค์กรด้วยการทดสอบอัตโนมัติ

Automated Testing for Issue & Support Management System

สมนึก สินธุปวน[1]*, ศิรวิทย์ หาญณรง[1], ก่องกาญจน์ ดุลยไชย[1], อลงกต กองมณี[1]
Somnuk Sindhupon[1]*, Sirawit Harnnarong[1], Kongkan Dulyachai[1], Alongkot Kongmanee[1]
[1]สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยแม่โจ้
*ผู้นิพนธ์ประสานงาน: สมนึก สิทธุปวน อีเมล: somnuk@mju.ac.th

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาและเปรียบเทียบกระบวนการทดสอบซอฟต์แวร์ระหว่าง Manual Testing และ Automated Testing โดยใช้เครื่องมือ Playwright ในการพัฒนาชุดทดสอบอัตโนมัติแบบ End-to-End (E2E Testing) และประยุกต์แนวคิด Page Object Model (POM) เพื่อเพิ่มความสามารถในการดูแลรักษาและการนำกลับมาใช้ซ้ำ อีกทั้งยังทำการประเมินความสามารถในการเข้าถึง (Accessibility) ของระบบ Issue & Support Management System เพื่อให้มั่นใจว่าสามารถรองรับผู้ใช้งานทุกกลุ่มได้

ผลการทดสอบฟังก์ชันหลักของระบบรวม 248 ครั้ง พบว่าฟังก์ชัน Login มีอัตราการผ่านร้อยละ 93.3 และ Dashboard ร้อยละ 88.9 ขณะที่ฟังก์ชันที่ซับซ้อนมากขึ้น เช่น Add Project, Projects และ Scope of Work มีอัตราการผ่านระหว่างร้อยละ 64.4–80.0 ส่วน Other Document ซึ่งเป็นฟังก์ชันที่เกี่ยวข้องกับข้อมูลจำนวนมากมีอัตราการผ่านต่ำสุดที่ร้อยละ 56.0 ข้อมูลดังกล่าวชี้ให้เห็นว่าฟังก์ชันพื้นฐานมีความเสถียรสูง ขณะที่ฟังก์ชันซับซ้อนยังต้องปรับปรุงเพิ่มเติม

เมื่อเปรียบเทียบวิธีการทดสอบ พบว่า Manual Testing มีความสะดวกและรวดเร็วในช่วงต้น แต่ Automated Testing เหนือกว่าชัดเจนในการทดสอบซ้ำและลดข้อผิดพลาดจากมนุษย์ อีกทั้งยังสามารถสร้างหลักฐานอัตโนมัติที่ตรวจสอบย้อนหลังได้ ในส่วนของการประเมิน Accessibility การทดสอบครั้งแรกได้คะแนนเฉลี่ย 91/100 อยู่ในระดับ “Good” แต่ยังพบข้อจำกัดด้าน Color Contrast และการรองรับ Screen Reader หลังจากดำเนินการปรับปรุง ผลการทดสอบครั้งที่สองมีคะแนนเฉลี่ยเพิ่มขึ้นเป็น 95/100 อยู่ในระดับ “Excellent” แสดงให้เห็นถึงพัฒนาการที่ชัดเจนในการรองรับการเข้าถึงสำหรับผู้ใช้งานทุกกลุ่ม ดังนั้น ระบบ Issue & Support Management System มีความถูกต้อง ความรวดเร็ว และความเสถียรเพียงพอต่อการใช้งานจริง การผสาน Manual, Automated และ Accessibility Testing สามารถยกระดับคุณภาพซอฟต์แวร์ให้มีความน่าเชื่อถือและรองรับผู้ใช้ได้อย่างทั่วถึง

คำสำคัญ: การทดสอบอัตโนมัติ; การทดสอบแบบแมนนวล; การประเมินความสามารถในการเข้าถึง; ระบบบริหารจัดกางานสนับสนุน;

Abstract

This research aims to study and compare software testing processes between Manual Testing and Automated Testing by employing Playwright to develop automated End-to-End (E2E) test scripts, while applying the Page Object Model (POM) approach to enhance maintainability and reusability. In addition, the study evaluates the accessibility of the Issue & Support Management System to ensure that the system can be effectively used by diverse groups of users.

The experimental evaluation was conducted across 248 test executions covering the system’s core functionalities. The results show that fundamental functions such as Login and Dashboard achieved high stability with success rates of 93.3% and 88.9%, respectively. In contrast, more complex functions such as Add Project, Projects, and Scope of Work demonstrated success rates ranging from 64.4% to 80.0%, while Other Document, which involves handling large datasets, recorded the lowest success rate at 56.0%. These findings indicate that basic functions are stable, whereas complex and data-intensive functions require further refinement.

A comparison between testing methods revealed that Manual Testing provides convenience and speed during initial phases; however, Automated Testing offers superior efficiency in repeated executions, reduces human error, and enables automatic generation of verifiable test evidence. Accessibility testing results further showed that all primary pages of the system scored above 90/100 (Excellent), although minor limitations remain regarding color contrast and screen reader compatibility. In conclusion, the Issue & Support Management System demonstrates sufficient accuracy, efficiency, and reliability for practical use. The integration of Manual Testing, Automated Testing, and Accessibility Testing significantly enhances the software's quality, dependability, and inclusiveness for all users.

Keywords: Software Testing, Automated Testing, Playwright, Page Object Model (POM), Accessibility

1. บทนำ

ซอฟต์แวร์มีบทบาทสำคัญในการสนับสนุนการดำเนินงานขององค์กรในทุกภาคส่วน การพัฒนาซอฟต์แวร์ที่มีคุณภาพและมีความปลอดภัยจึงเป็นปัจจัยสำคัญที่ช่วยเพิ่มประสิทธิภาพและความต่อเนื่องของธุรกิจ บริษัท ลูคัส สแทรททิจี จำกัด ได้พัฒนาระบบบริหารจัดการงานสนับสนุนภายในองค์กร (Issue & Support Management System) เพื่อใช้ในการติดตามและจัดการปัญหาภายในอย่างเป็นระบบ อย่างไรก็ตาม กระบวนการทดสอบที่ใช้อยู่เดิมคือการทดสอบแบบแมนนวล (Manual Testing) ซึ่งมีข้อจำกัดในด้านความล่าช้า ความเสี่ยงจากข้อผิดพลาดของมนุษย์ และความไม่เหมาะสมต่อการทดสอบซ้ำเมื่อระบบมีการปรับปรุง

ดังนั้น งานวิจัยนี้จึงมุ่งพัฒนากระบวนการทดสอบแบบอัตโนมัติ (Automated Testing) โดยใช้เครื่องมือ Playwright ซึ่งสามารถรองรับการทดสอบแบบ End-to-End (E2E Testing) และการทำงานข้ามเบราว์เซอร์ รวมทั้งสามารถจำลองพฤติกรรมการใช้งานจริงของผู้ใช้ได้อย่างแม่นยำ เพื่อยกระดับคุณภาพการทดสอบ นอกจากนี้ยังประยุกต์ใช้แนวคิด Page Object Model (POM) เพื่อจัดการโครงสร้างชุดทดสอบให้มีความยืดหยุ่นและบำรุงรักษาได้ง่าย รวมทั้งบูรณาการการทดสอบด้าน Accessibility เพื่อให้ระบบรองรับผู้ใช้งานทุกกลุ่มได้อย่างเท่าเทียม

จากผลการทดสอบเบื้องต้นจำนวน 248 ครั้ง พบว่าฟังก์ชันพื้นฐาน เช่น การเข้าสู่ระบบและแดชบอร์ด มีอัตราการผ่านการทดสอบสูงกว่า 88.9% ในขณะที่ฟังก์ชันที่มีความซับซ้อน เช่น Add Project และ Scope of Work มีอัตราการผ่านการทดสอบระหว่าง 64.4%–80.0% ส่วนฟังก์ชันที่เกี่ยวข้องกับข้อมูลปริมาณมาก เช่น Other Document มีอัตราการผ่านต่ำสุดเพียง 56.0% ผลการทดสอบเหล่านี้ยืนยันถึงความจำเป็นของการใช้ Automated Testing ควบคู่กับ Manual Testing และ Accessibility Testing เพื่อยกระดับคุณภาพ ความน่าเชื่อถือ และการเข้าถึงของระบบในสภาพแวดล้อมจริง

2. วัตถุประสงค์

1. เพื่อศึกษาแนวคิด หลักการ และกระบวนการทดสอบซอฟต์แวร์ ทั้งในรูปแบบ Manual Testing และ Automated Testing
2. เพื่อพัฒนาและดำเนินการทดสอบระบบแบบอัตโนมัติด้วยเครื่องมือ Playwright โดยเน้นการทดสอบแบบ End-to-End (E2E Testing) เพื่อจำลองพฤติกรรมการใช้งานจริงของผู้ใช้
3. เพื่อวิเคราะห์และเปรียบเทียบผลลัพธ์ของ Manual Testing และ Automated Testing ในด้านความถูกต้อง ระยะเวลา และประสิทธิภาพ
4. เพื่อประยุกต์ใช้แนวคิด Page Object Model (POM) ในการออกแบบชุดทดสอบ เพื่อเพิ่มความสามารถในการบำรุงรักษาและการนำกลับมาใช้ซ้ำ
5. เพื่อประเมินและปรับปรุงความสามารถในการเข้าถึง (Accessibility) ของระบบ Issue & Support Management System ตามมาตรฐาน WCAG 2.1

3. งานวิจัยที่เกี่ยวข้อง

การทดสอบซอฟต์แวร์ (Software Testing) ถือเป็นกิจกรรมหลักที่มีความสำคัญต่อกระบวนการพัฒนาซอฟต์แวร์ เนื่องจากมีบทบาทในการตรวจสอบความถูกต้อง (Verification) และความสอดคล้อง (Validation) ของระบบ เพื่อให้มั่นใจว่าซอฟต์แวร์ที่พัฒนาขึ้นสามารถทำงานตามข้อกำหนด (Requirements) ได้อย่างครบถ้วน และมีคุณภาพเพียงพอต่อการใช้งานจริง (Pressman & Maxim, 2019) งานวิจัยและคู่มือเชิงมาตรฐาน เช่น ISTQB (2018) ได้แบ่งประเภทการทดสอบออกเป็นหลายระดับ ได้แก่ Unit Testing, Integration Testing, System Testing และ Acceptance Testing โดยแต่ละระดับมีเป้าหมายเพื่อค้นหาข้อบกพร่องในมิติต่าง ๆ ของซอฟต์แวร์ การทดสอบแบบแมนนวล (Manual Testing) เป็นการทดสอบที่ผู้ทดสอบดำเนินการตามกรณีทดสอบ (Test Cases) ด้วยตนเอง ซึ่งมีข้อดีคือความยืดหยุ่นและสามารถประเมินปฏิสัมพันธ์ของผู้ใช้กับระบบได้อย่างละเอียด (Myers, Sandler & Badgett, 2011) อย่างไรก็ตาม ข้อจำกัดที่สำคัญคือใช้เวลานาน มีค่าใช้จ่ายสูง และไม่เหมาะกับการทดสอบซ้ำ (Regression Testing) โดยเฉพาะเมื่อซอฟต์แวร์มีการอัปเดตบ่อยครั้ง ในทางกลับกัน การทดสอบแบบอัตโนมัติ (Automated Testing) ได้รับการยอมรับว่าเป็นแนวทางที่ช่วยเพิ่มความรวดเร็วและความแม่นยำของการทดสอบ เนื่องจากสามารถลดข้อผิดพลาดจากมนุษย์ (Human Error) และสนับสนุนการทำงานแบบ Continuous Integration/Continuous Deployment (CI/CD) ได้อย่างมีประสิทธิภาพ (Fewster & Graham, 1999) นอกจากนี้ยังช่วยให้การทดสอบขนาดใหญ่ (Large-Scale Testing) และการทดสอบซ้ำในหลายสภาพแวดล้อมเป็นไปได้ง่ายขึ้น End-to-End Testing (E2E) เป็นการทดสอบที่จำลองกระบวนการทำงานจริงของผู้ใช้ ตั้งแต่การเข้าสู่ระบบ การทำงานผ่านโมดูลต่าง ๆ จนถึงผลลัพธ์สุดท้าย โดยครอบคลุมทั้งการเชื่อมโยงระหว่างระบบย่อยและการโต้ตอบกับบริการภายนอก (Meszaros, 2007) แนวทางนี้ช่วยยืนยันว่าเมื่อระบบทั้งหมดทำงานร่วมกันแล้ว สามารถตอบสนองต่อความต้องการใช้งานจริงได้ครบถ้วน เพื่อเพิ่มประสิทธิภาพของการพัฒนาสคริปต์ทดสอบ จึงได้มีการประยุกต์ใช้แนวคิด Page Object Model (POM) ซึ่งเป็นรูปแบบการออกแบบเชิงวัตถุที่ใช้สร้าง

คลาสเพื่อแทนหน้าเว็บหรือองค์ประกอบต่าง ๆ ในระบบ โดย Leotta et al. (2015) พบว่า POM สามารถช่วยลดความซ้ำซ้อนของโค้ด เพิ่มความง่ายต่อการดูแลรักษา (Maintainability) และเพิ่มความสามารถในการนำกลับมาใช้ซ้ำ (Reusability) ได้อย่างมีนัยสำคัญ ขณะที่ Stocco et al. (2017) ยังได้ชี้ว่า POM สามารถเพิ่มความคงทน (Robustness) ของสคริปต์ทดสอบต่อการเปลี่ยนแปลงของ UI ได้ดีกว่าวิธีการทั่วไป อีกประเด็นหนึ่งที่มีความสำคัญในยุคปัจจุบันคือความสามารถในการเข้าถึง (Accessibility) ของซอฟต์แวร์ เพื่อให้มั่นใจว่าผู้ใช้ทุกกลุ่ม รวมถึงผู้พิการสามารถใช้งานได้อย่างเท่าเทียม ตามมาตรฐาน Web Content Accessibility Guidelines (WCAG) ที่จัดทำโดย W3C (2018) งานวิจัยของ Sánchez-Gordón และ Luján-Mora (2017) แสดงให้เห็นว่าการบูรณาการ Accessibility Testing เข้ากับวงจรการพัฒนาซอฟต์แวร์ตั้งแต่ระยะแรก สามารถช่วยลดต้นทุนในระยะยาวและยกระดับคุณภาพของผลิตภัณฑ์ ขณะที่ Ara และ Sánchez-Gordón (2024) ได้เสนอวิธีการตรวจสอบ Accessibility โดยอัตโนมัติ ซึ่งช่วยลดภาระของผู้ทดสอบและเพิ่มความสม่ำเสมอของการประเมิน

จากการศึกษางานวิจัยที่เกี่ยวข้อง พบว่าการทดสอบซอฟต์แวร์มีความสำคัญอย่างยิ่งต่อการประกันคุณภาพและความน่าเชื่อถือของระบบ โดยแนวทางการทดสอบทั้งแบบ Manual และ Automated ต่างมีข้อดีและข้อจำกัดที่สามารถนำมาประยุกต์ใช้ร่วมกันเพื่อเพิ่มประสิทธิภาพในการตรวจสอบความถูกต้องและความเสถียรของระบบ งานวิจัยหลายชิ้นได้ยืนยันว่าเครื่องมือสำหรับการทดสอบอัตโนมัติ โดยเฉพาะ Playwright สามารถรองรับการทดสอบแบบ End-to-End ได้อย่างแม่นยำและเหมาะสมต่อการใช้งานจริง ขณะเดียวกันการประยุกต์ใช้แนวคิด Page Object Model (POM) ยังช่วยเพิ่มความสามารถในการบำรุงรักษาและการนำกลับมาใช้ซ้ำของสคริปต์ทดสอบอย่างมีประสิทธิภาพ อีกทั้งประเด็นด้าน Accessibility ก็ได้รับการยืนยันว่ามีความจำเป็นเพื่อให้ระบบรองรับผู้ใช้งานทุกกลุ่มอย่างเท่าเทียม

ดังนั้น การดำเนินการวิจัยครั้งนี้ซึ่งมุ่งเน้นการศึกษาและเปรียบเทียบการทดสอบ Manual และ Automated การพัฒนาชุดทดสอบอัตโนมัติด้วย Playwright การประยุกต์ใช้ POM ตลอดจนการประเมิน Accessibility ของระบบ Issue & Support Management System จึงมีความสอดคล้องกับทฤษฎี แนวคิด และงานวิจัยที่ผ่านมา และสามารถตอบสนองต่อวัตถุประสงค์ของการวิจัยได้ครบถ้วนทุกประการ

4. การวิเคราะห์ระบบบริหารจัดการงานสนับสนุนภายในองค์

ระบบบริหารจัดการงานสนับสนุนภายในองค์กร (Issue & Support Management System) ได้รับการออกแบบขึ้นเพื่อสนับสนุนการจัดการปัญหาและคำร้องจากลูกค้าภายในองค์กรอย่างเป็นระบบและมีประสิทธิภาพสูงสุด ระบบนี้ใช้ Use Case Diagram เป็นเครื่องมือสำคัญในการอธิบายขอบเขตการทำงานของระบบ ตลอดจนความสัมพันธ์ระหว่างผู้ใช้งานและฟังก์ชันต่าง ๆ ของระบบ การออกแบบ Use Case Diagram ช่วยให้กระบวนการทำงานของระบบมีความโปร่งใส สามารถตรวจสอบได้ และสื่อสารความเข้าใจต่อผู้พัฒนาหรือผู้ตรวจสอบระบบได้อย่างชัดเจน ระบบแบ่งผู้ใช้งานออกเป็นสองกลุ่มหลัก ได้แก่ ผู้ดูแลระบบ (Admin) และ ผู้ใช้งานทั่วไป (Staff) โดยแต่ละกลุ่มมีสิทธิ์ในการเข้าถึงฟังก์ชันที่แตกต่างกันอย่างชัดเจน ผู้ดูแลระบบมีหน้าที่ควบคุมและจัดการข้อมูลโดยรวมของระบบ ครอบคลุมการจัดการบัญชีผู้ใช้งาน การบริหารโครงการ การบันทึกเอกสารสัญญาและรายละเอียดโครงการ การตรวจสอบปัญหา และการแก้ไขปัญหาอย่างเป็นระบบ นอกจากนี้ ผู้ดูแลระบบยังสามารถสรุปผลการดำเนินงานในรูปแบบรายงานเชิงสถิติ สำหรับผู้ใช้งานทั่วไปทำหน้าที่ในฐานะผู้ปฏิบัติงาน สามารถเข้าถึงข้อมูล

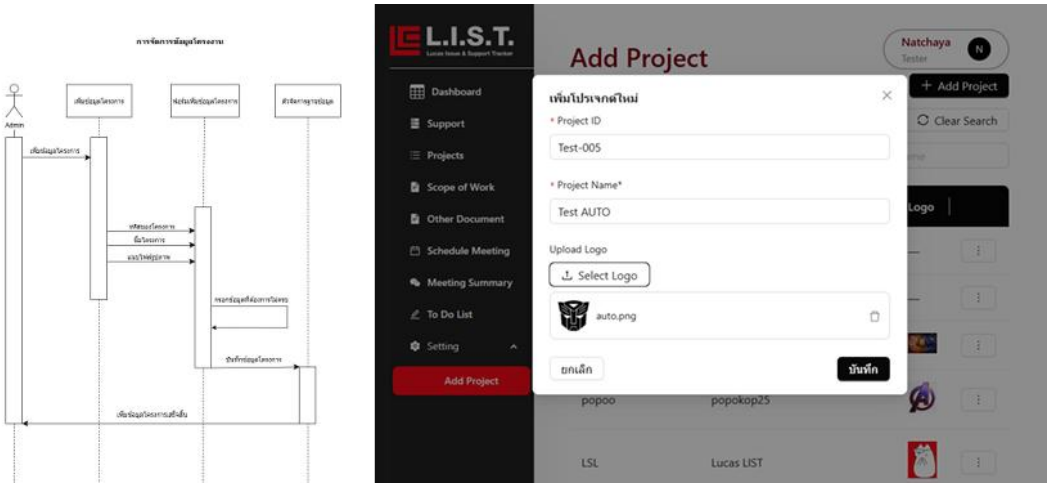
โครงการที่ตนรับผิดชอบ รายงานปัญหาที่พบ และติดตามรายละเอียดของปัญหาที่ตนแจ้งไว้ Use Case หลักของระบบ ได้แก่ การเข้าสู่ระบบ ซึ่งถือเป็นเงื่อนไขพื้นฐานที่ผู้ใช้งานทุกคนต้องดำเนินการก่อนเข้าถึงฟังก์ชันอื่น ๆ ผู้ดูแลระบบสามารถจัดการบัญชีผู้ใช้งาน จัดการโครงการ บันทึกเอกสาร และติดตามปัญหา ขณะที่ผู้ใช้งานทั่วไปสามารถรายงานปัญหาและเปิดดูรายละเอียดเพื่อการติดตามความคืบหน้า ข้อมูลที่ได้รับการแก้ไขจะถูกบันทึกและอัปเดตสถานะทันที ทำให้ผู้รายงานรับทราบถึงผลลัพธ์และความคืบหน้าของงานอย่างชัดเจน นอกจากนี้ ผู้ดูแลระบบยังสามารถออกรายงานสรุปภาพรวมของโครงการและปัญหาที่เกิดขึ้น เพื่อประกอบการวิเคราะห์และวางแผนการดำเนินงานในระดับองค์กร

รูปที่ ค.1 แสดงฟังก์ชันการทำงานของระบบ

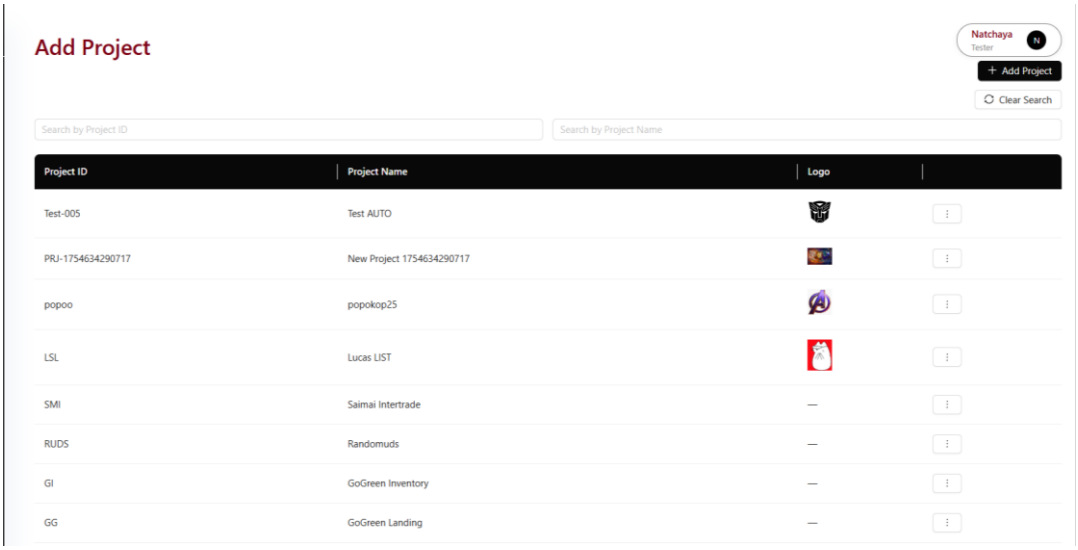
จากรูปที่ ค.1 ความสัมพันธ์ระหว่าง Use Case ระบบนี้นำแนวคิด <<include>> และ <<extend>> มาใช้เพื่อเพิ่มความชัดเจนและลดความซับซ้อนของแผนภาพ Use Case ความสัมพันธ์แบบ <<include>> แสดงถึงการพึ่งพาอาศัยกันอย่างสมบูรณ์ โดย Use Case หลักจำเป็นต้องเรียกใช้ Use Case ย่อยเสมอเพื่อให้กระบวนการดำเนินงานสำเร็จ กล่าวคือ Use Case ที่ถูกรวมเป็นขั้นตอนที่จำเป็นต้องเกิดขึ้นก่อนหรือเป็นส่วนหนึ่งของกระบวนการทั้งหมด ตัวอย่างเช่น การจัดการข้อมูลหรือเอกสารทุกประเภทจำเป็นต้องเริ่มต้นด้วย การเข้าสู่ระบบก่อนเสมอ ความสัมพันธ์ประเภทนี้ช่วยลดความซ้ำซ้อนของระบบ และทำให้ฟังก์ชันที่ต้องทำซ้ำสามารถนำไปใช้ในหลายส่วนของระบบได้อย่างมีประสิทธิภาพ ในทางตรงกันข้าม ความสัมพันธ์แบบ <<extend>> เป็นการกำหนดพฤติกรรมเสริมหรือทางเลือกที่จะเกิดขึ้นเฉพาะเมื่อมีเงื่อนไขกำหนดเท่านั้น ไม่ใช่ขั้นตอนที่จำเป็นต้องเกิดขึ้นทุกครั้งการใช้ <<extend>> ช่วยลดความซับซ้อนของ Use Case หลักโดยการแยกพฤติกรรมเสริมหรือสถานการณ์พิเศษออกจาก Use Case หลัก ตัวอย่างเช่น ผู้ดูแลระบบอาจเลือก ดูรายละเอียดของปัญหา เมื่อจำเป็นต้องแก้ไขปัญหาเท่านั้น แต่ไม่ได้หมายความว่าการแก้ไขทุกครั้งจะต้องเปิดดูรายละเอียดเสมอไป โดยระบบสะท้อนการทำงานอย่างเป็นลำดับขั้น ตั้งแต่การเข้าสู่ระบบ การจัดการข้อมูล การรายงานปัญหา การแก้ไขปัญหา จนถึงการสรุปภาพรวม ทำให้ระบบมีความโปร่งใส สามารถตรวจสอบได้ สนับสนุนประสิทธิภาพในการทำงานขององค์กร และสามารถต่อยอดเพื่อการพัฒนาระบบในอนาคตได้อย่างเหมาะสม

4.1 การเพิ่มข้อมูลค่าคงที่ในระบบ

การเพิ่มข้อมูลโครงการเป็นขั้นตอนสำคัญที่ใช้ในการสร้างข้อมูลพื้นฐานของระบบ โดยผู้ใช้งานจะต้องกรอกข้อมูลที่จำเป็น ได้แก่ ชื่อโครงการ รายละเอียดโครงการ และระยะเวลาเริ่มต้น–สิ้นสุด พร้อมทั้งสามารถแนบไฟล์ประกอบ เช่น รูปภาพหรือเอกสาร เพื่อเพิ่มความสมบูรณ์ของข้อมูล หากผู้ใช้งานละเว้นการกรอกข้อมูลที่จำเป็น ระบบจะแจ้งเตือนให้กรอกข้อมูลให้ครบถ้วน ข้อมูลโครงการที่บันทึกเรียบร้อยแล้วจะแสดงในตารางรายการโครงการ เพื่อให้ผู้ใช้งานตรวจสอบความถูกต้องได้ทันที แสดงดังรูปที่ ค.



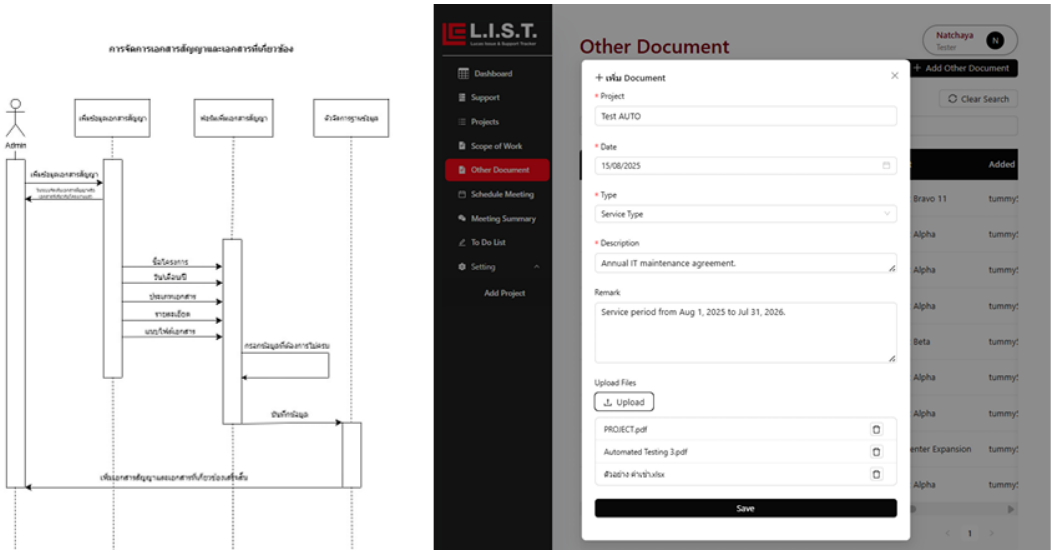
รูปที่ ค.2 ขั้นตอนการเพิ่มโครงงาน



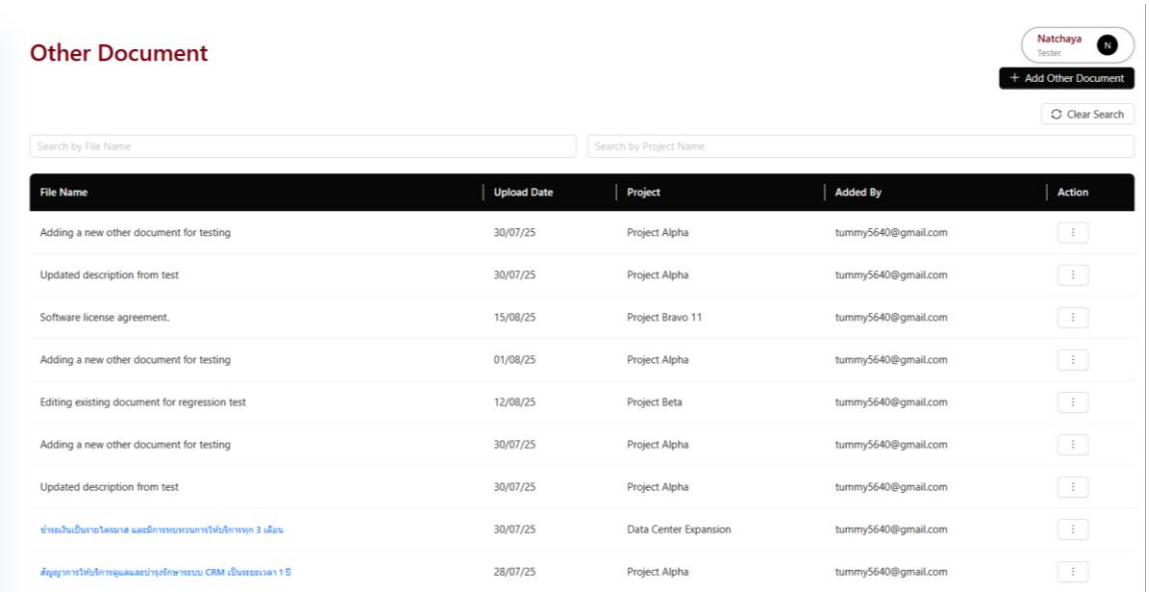
รูปที่ ค.3 แสดงข้อมูลที่เพิ่มโครงงาน

4.2 การจัดการเอกสารและรายละเอียดโครงงาน

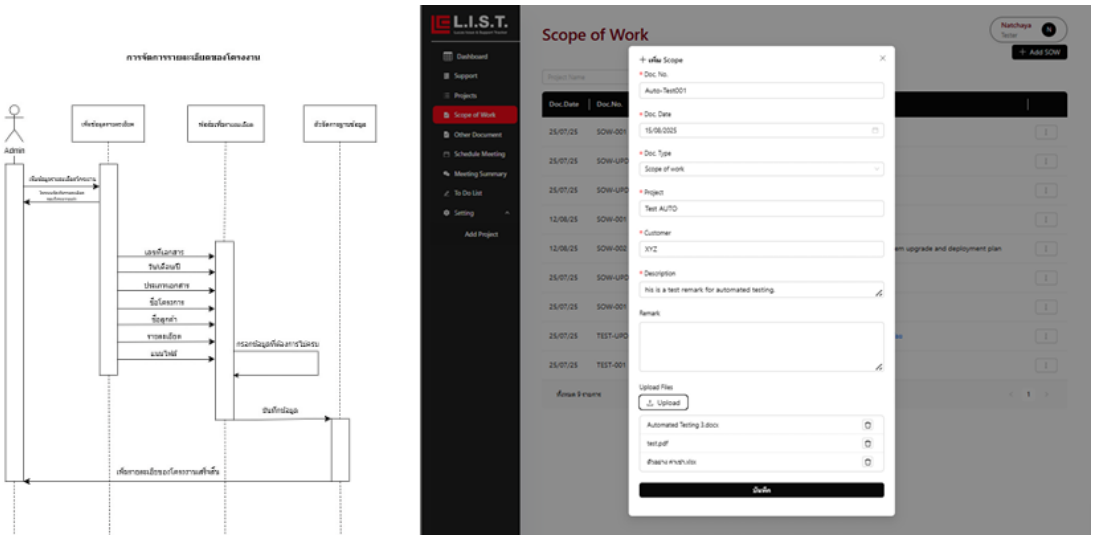
ระบบสนับสนุนการจัดเก็บและจัดการข้อมูลที่เกี่ยวข้องกับโครงการอย่างครบถ้วน ทั้งเอกสารสัญญา เอกสารประกอบอื่น ๆ และรายละเอียดของขอบเขตการดำเนินงาน ผู้ใช้งานสามารถดำเนินการเพิ่ม แก้ไข หรือลบข้อมูลได้ตามความเหมาะสม รวมถึงแนบไฟล์ที่เกี่ยวข้องเพื่อใช้เป็นหลักฐาน เมื่อบันทึกข้อมูลเรียบร้อยแล้ว รายการทั้งหมดจะแสดงในตารางที่เกี่ยวข้องเพื่อให้ง่ายต่อการตรวจสอบและติดตาม การดำเนินงานในส่วนนี้ช่วยให้ข้อมูลโครงการมีความครบถ้วน เชื่อถือได้ และสามารถใช้เป็นแหล่งอ้างอิงที่ชัดเจน แสดงดังรูปที่ ค. และ รูปที่ ค.



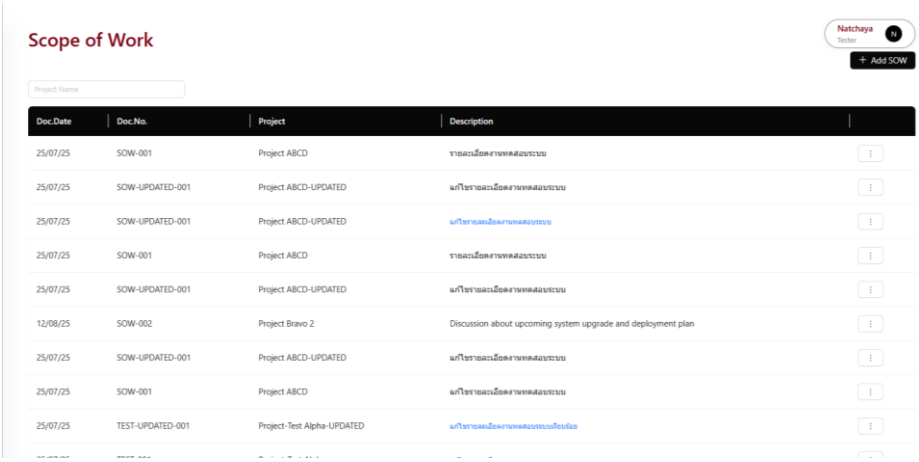
รูปที่ ค.4 ขั้นตอนการเพิ่มเอกสารสัญญาและเอกสารที่เกี่ยวข้อง



รูปที่ ค.5 แสดงข้อมูลที่เพิ่มเอกสารสัญญาและเอกสารที่เกี่ยวข้อง



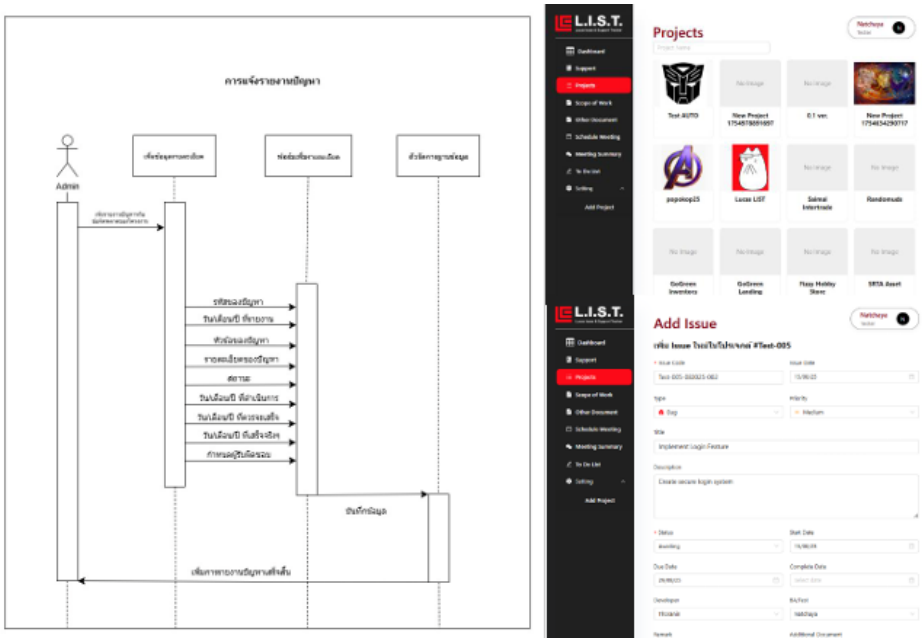
รูปที่ ค.6 ขั้นตอนการเพิ่มรายละเอียดของโครงการ



รูปที่ ค.7 แสดงข้อมูลการที่เพิ่มรายละเอียดของโครงการ

4.3 การแจ้งรายงานปัญหา

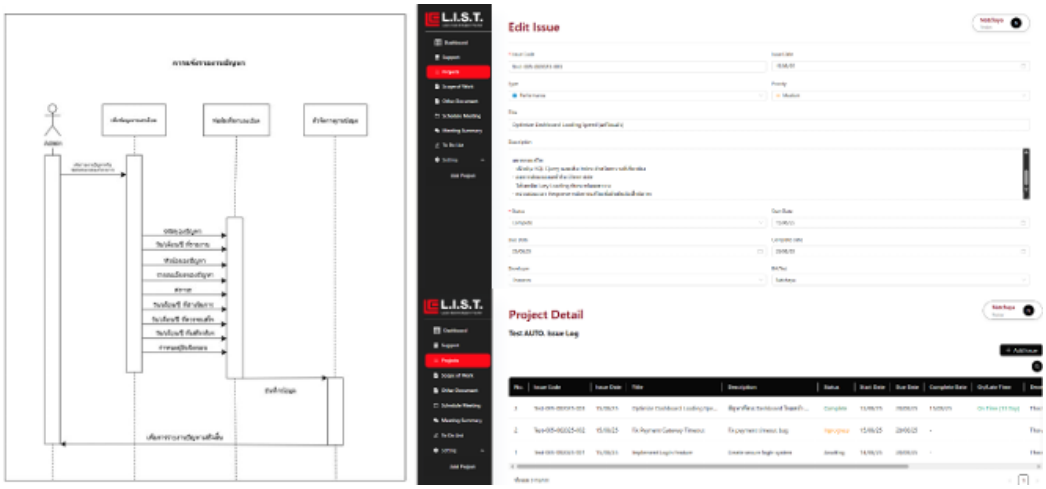
ในกรณีที่เกิดปัญหาระหว่างการดำเนินโครงการ ผู้ใช้งานสามารถบันทึกรายละเอียดของปัญหา กำหนดผู้ที่รับผิดชอบ และระบุสถานะของปัญหาได้ ข้อมูลที่บันทึกแล้วจะแสดงในตารางรายงานปัญหา โดยสถานะเริ่มต้นของปัญหาจะถูกระบุเป็น Awaiting เพื่อให้สามารถติดตามและตรวจสอบความคืบหน้าได้อย่างต่อเนื่อง การบันทึกข้อมูลในลักษณะนี้ช่วยให้การจัดการปัญหามีความเป็นระบบ โปร่งใส และสามารถตรวจสอบย้อนกลับได้



รูปที่ ค.8 ขั้นตอนการเลือโครงงาน เพื่อที่จะดำเนินการแจ้งรายงานปัญหา

4.4 การจัดการรายงานปัญหา

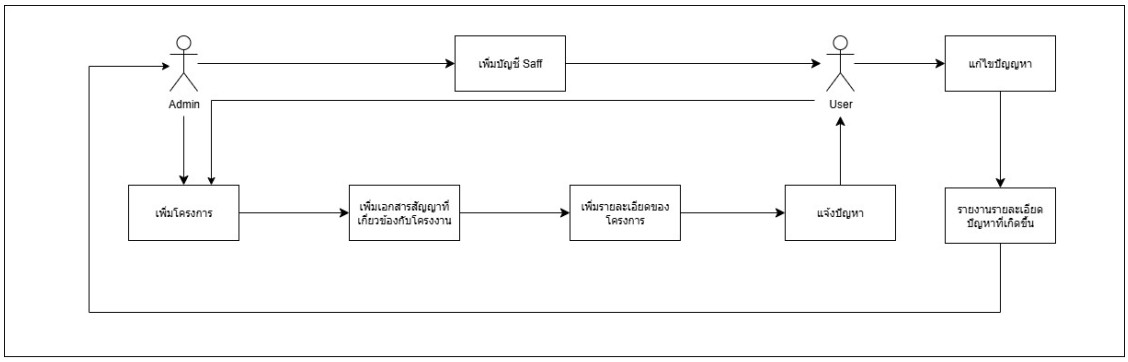
ผู้ใช้งานและผู้ดูแลระบบสามารถจัดการข้อมูลปัญหาที่ถูกบันทึกไว้ได้อย่างครบถ้วน ไม่ว่าจะเป็นการแก้ไขรายละเอียด การระบุสาเหตุเพิ่มเติม การเปลี่ยนสถานะปัญหา หรือลบข้อมูลที่ไม่จำเป็น ข้อมูลที่ปรับปรุงแล้วจะถูกอัปเดตในตารางรายงานปัญหาโดยอัตโนมัติ ทำให้การติดตามและตรวจสอบเป็นไปอย่างถูกต้องและทันเวลา ขั้นตอนนี้ช่วยให้การแก้ไขปัญหามีประสิทธิภาพมากขึ้น ลดความซ้ำซ้อน และสร้างความชัดเจนในการดำเนินงาน



รูปที่ ค.9 ขั้นตอนการตอบรายงานปัญหา

5. วิเคราะห์การออกแบบและการทดสอบระบบ

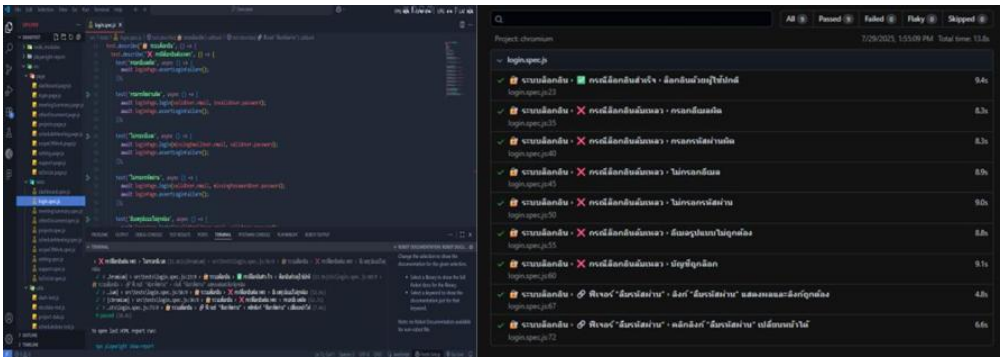
ในการทดสอบระบบภายในองค์กรนี้มีสิทธิ์การใช้งาน 2 ระดับ ได้แก่ ผู้ดูแลระบบ (Admin) และ ผู้ใช้งานทั่วไป (User) โดย Admin สามารถเพิ่มบัญชีผู้ใช้งานประเภท Staff เพื่อมอบหมายงานหรือดูแลการแก้ไขปัญหาภายในระบบ ทั้ง Admin และ User สามารถเพิ่มโครงการ ข้อมูลเอกสารสัญญา และรายละเอียดหรือขอบเขตของโครงการเข้าสู่ระบบเพื่อให้การดำเนินงานสอดคล้องกับสัญญาบำรุงรักษา เมื่อเกิดปัญหา ผู้ใช้งานสามารถแจ้งรายงาน กำหนดผู้รับผิดชอบ และติดตามสถานะการแก้ไข โดยผู้ได้รับมอบหมายสามารถดำเนินการแก้ไขและบันทึกรายละเอียดการแก้ไขลงในระบบได้ ระบบจะเก็บข้อมูลโครงการ เอกสาร รายละเอียด และประวัติการแก้ไขปัญหาอย่างครบถ้วน ทำให้การจัดการโครงการและการแก้ไขปัญหาภายในองค์กรดำเนินไปอย่างเป็นระบบ มีประสิทธิภาพและสามารถตรวจสอบความถูกต้องได้ทุกขั้นตอน



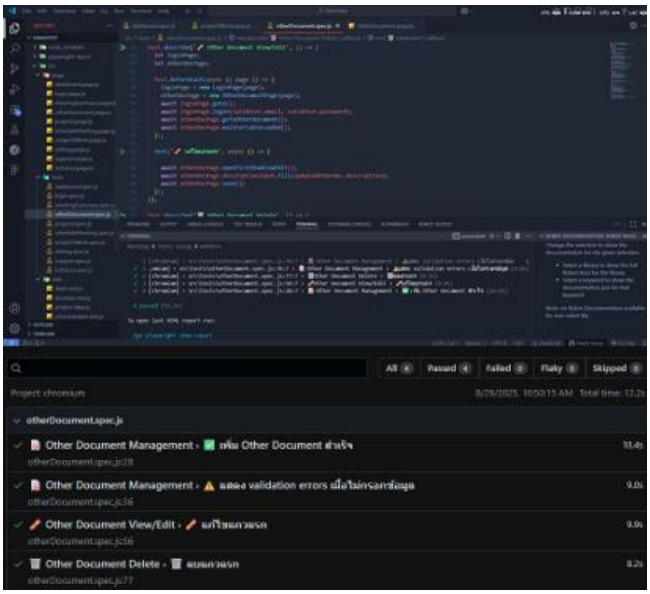
รูปที่ ค.10 แสดงฟังก์ชันการทำงานของระบบ

6. การทดสอบระบบอัตโนมัติ

เข้าสู่ระบบ ในการทดสอบการเข้าสู่ระบบเริ่มต้นจากเข้าสู่หน้าเข้าสู่ระบบ จากนั้นกรอก Email และ Password มีการตรวจสอบการกรอกข้อมูลว่าถูกกรอกลงในช่องหรือไม่ จากนั้นกดปุ่ม Sign In ถ้า Email หรือ Password ถูกต้อง จะเข้าสู่ระบบแล้วไปที่หน้า Dashboard ทันที และ ถ้า Email หรือPassword ไม่ถูกต้องจะตรวจสอบว่ามีข้อความ Error Message แสดงขึ้นมา

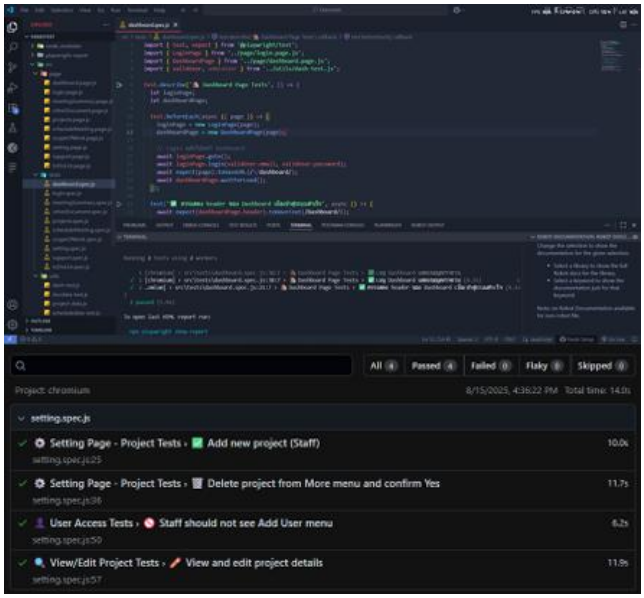


รูปที่ ค.11 ภาพแสดงผลการทดสอบระบบด้วย Playwright หน้า Login



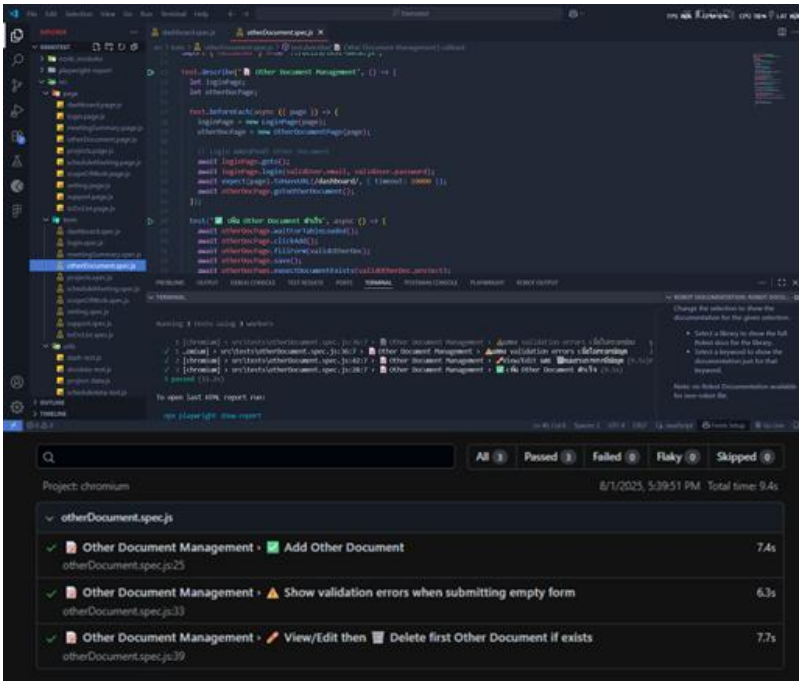
รูปที่ ค.12 ภาพแสดงผลการทดสอบระบบด้วย Playwright หน้า Dashboard

การทดสอบการเพิ่มโครงงานเริ่มจากการเข้าสู่หน้า Dashboard ของระบบ จากนั้นเลือกเมนู “Setting” และเลือกฟังก์ชัน “Add Project” เมื่อเข้าถึงหน้าจอจัดการข้อมูลโครงงานแล้ว ให้คลิกปุ่ม “Add Project” เพื่อให้ระบบแสดงฟอร์มสำหรับเพิ่มโครงงานขึ้นมา จากนั้นกรอกข้อมูลโครงงานลงในฟอร์ม และตรวจสอบว่าข้อมูลสามารถกรอกลงในช่องต่าง ๆ ได้อย่างถูกต้อง หลังจากกรอกข้อมูลครบถ้วน คลิกปุ่ม “บันทึก” (Save) และตรวจสอบว่าข้อมูลโครงงานที่เพิ่มปรากฏอยู่ในตารางรายการโครงงานอย่างถูกต้อง



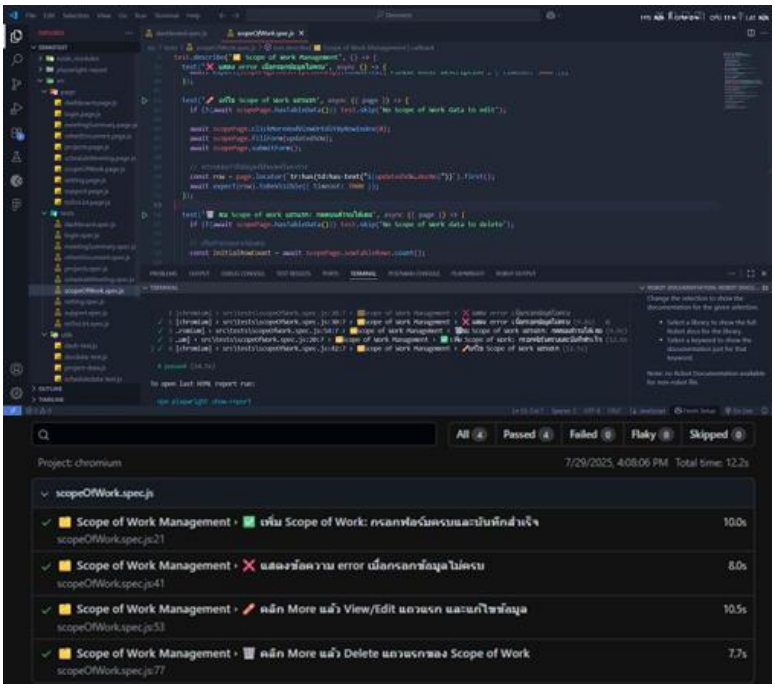
รูปที่ ค.13 ภาพแสดงผลการทดสอบระบบด้วย Playwright ในขั้นตอนการเพิ่มข้อมูลของโครงงาน

การทดสอบการเพิ่มเอกสารสัญญาและเอกสารที่เกี่ยวข้องเริ่มจากการเข้าสู่หน้า “Other Document” ของระบบ จากนั้นคลิกปุ่ม “Add Other Document” เพื่อให้ระบบแสดงฟอร์มสำหรับเพิ่มเอกสารขึ้นมา จากนั้นกรอกข้อมูลเอกสารลงในฟอร์ม และตรวจสอบว่าข้อมูลสามารถกรอกลงในช่องต่าง ๆ ได้อย่างถูกต้อง หลังจากกรอกข้อมูลครบถ้วน คลิกปุ่ม “บันทึก” (Save) และตรวจสอบว่าข้อมูลที่เพิ่มปรากฏอยู่ในตารางรายการ Other Document อย่างถูกต้อง นอกจากนี้ยังตรวจสอบความสามารถในการเปิดและดาวน์โหลดไฟล์แนบเพื่อยืนยันความสมบูรณ์ของเอกสาร



รูปที่ ค.14 ภาพแสดงผลการทดสอบระบบด้วย Playwright หน้า Other Document

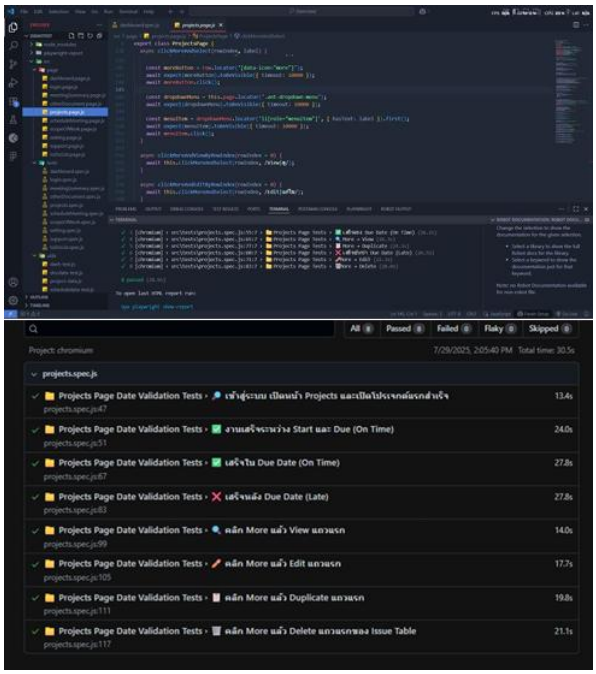
การทดสอบการเพิ่มรายละเอียดของโครงการเริ่มต้นจากการเข้าสู่หน้า “Scope of Work” ของระบบ จากนั้นคลิกปุ่ม “Add SOW” เพื่อให้ระบบแสดงฟอร์มสำหรับเพิ่มรายละเอียดของโครงการขึ้นมากรอกข้อมูลรายละเอียดของโครงการลงในฟอร์ม และตรวจสอบว่าข้อมูลสามารถกรอกลงในช่องต่าง ๆ ได้อย่างถูกต้อง หลังจากกรอกข้อมูลครบถ้วน คลิกปุ่ม “บันทึก” (Save) และตรวจสอบว่าข้อมูลที่เพิ่มปรากฏอยู่ในตารางรายการ Scope of Work อย่างถูกต้อง นอกจากนี้ยังตรวจสอบความสามารถในการเปิดและดาวน์โหลดไฟล์แนบเพื่อยืนยันความสมบูรณ์ของเอกสาร



รูปที่ ค.15 ภาพแสดงผลการทดสอบระบบด้วย Playwright หน้า Scope of Work

การทดสอบการแจ้งรายงานปัญหาเริ่มต้นจากการเข้าสู่หน้า “Projects” ของระบบ จากนั้นเลือกโครงงานที่ต้องการรายงานปัญหา เมื่อเลือกโครงงานเรียบร้อยแล้ว ให้คลิกปุ่ม “Add Issue” เพื่อเพิ่มรายงานปัญหา จากนั้นกรอกข้อมูลรายละเอียดของปัญหาที่เกิดขึ้น พร้อมกำหนดผู้รับผิดชอบโครงงาน และตรวจสอบความถูกต้องของข้อมูลที่กรอกลงในฟอร์ม หลังจากกรอกข้อมูลครบถ้วน คลิกปุ่ม “บันทึก” และตรวจสอบว่าข้อมูลรายงานปัญหาที่เพิ่มเข้ามาปรากฏอยู่ในตารางรายงานปัญหาอย่างถูกต้อง

การทดสอบการตอบรายงานปัญหาเริ่มต้นจากการเข้าสู่หน้า “Projects” ของระบบ จากนั้นเลือกโครงงานที่ต้องการตอบกลับรายงานปัญหา เมื่อเข้าถึงโครงงานเรียบร้อยแล้ว ให้เลือกรายงานปัญหาที่ต้องการตอบกลับ จากนั้นทำการแก้ไขข้อมูลที่จำเป็น เปลี่ยนสถานะของรายงานปัญหา และกรอกสาเหตุหรือรายละเอียดของปัญหา พร้อมตรวจสอบความถูกต้องของข้อมูลที่กรอกลงในฟอร์ม หลังจากกรอกข้อมูลครบถ้วน คลิกปุ่ม “บันทึก” และตรวจสอบว่าสถานะของรายงานปัญหาถูกอัปเดตในตารางรายงานปัญหาอย่างถูกต้อง



รูปที่ ค.16 ภาพแสดงผลการทดสอบระบบด้วย Playwright ในขั้นตอนแจ้งรายงานของปัญหาและตอบกลับการรายงานของปัญหา

ตารางที่ ค.1 ตารางแสดงการทดสอบระบบของหน้า Login

#	Test Scenario	Expected Result	Test Date	Test Result
Login				
1	Login สำเร็จด้วยข้อมูลถูกต้อง	เข้าสู่ระบบสำเร็จและแสดงหน้า Dashboard	11/7/68	Pass
2	Login ล้มเหลว กรอก Email หรือ Password ผิด	แสดงข้อความแจ้งเตือน "อีเมลหรือรหัสผ่านไม่ถูกต้อง"	11/7/68	Pass
3	ล้มเหลว กรอก Email หรือ Password ว่าง	แสดงข้อความแจ้งเตือน "อีเมลหรือรหัสผ่านไม่ถูกต้อง"	11/7/68	Pass
4	แสดงปุ่ม “ลืมรหัสผ่าน” และคลิกได้	นำผู้ใช้ไปยังหน้ารีเซ็ตรหัสผ่าน	11/7/68	Pass
5	ตรวจสอบการล็อกอินหลังกรอกข้อมูลครบ	เข้าสู่ระบบสำเร็จเหมือนกรณีกดปุ่ม Login	11/7/68	Pass

ตารางที่ ค.2 ตารางแสดงการทดสอบระบบของหน้า การเพิ่มโครงการ

#	Test Scenario	Expected Result	Test Date	Test Result
Add Project				
1	แสดงรายการ Project ทั้งหมด	รายการ Project ปรากฏพร้อม Project ID, ชื่อ และโลโก้ (ถ้ามี)	16/7/68	Pass
2	เพิ่มโปรเจกต์ใหม่ด้วยข้อมูลครบถ้วน	โปรเจกต์ใหม่ถูกเพิ่มและแสดงในรายการ พร้อมโลโก้ถูกต้อง	16/7/68	Pass
3	เพิ่มโปรเจกต์ใหม่โดยไม่กรอกข้อมูลจำเป็น	แสดงข้อความแจ้งเตือนให้กรอกข้อมูลจำเป็น	16/7/68	Pass
4	แก้ไขโปรเจกต์	สามารถแก้ไขข้อมูลโปรเจกต์ได้	16/7/68	Pass
5	ลบโปรเจกต์	โปรเจกต์ถูกลบออกจากรายการ	29/7/68	Pass

ตารางที่ ค.3 ตารางแสดงการทดสอบระบบของหน้า การเพิ่มบัญชีผู้ใช้งาน

#	**Test Scenario**	**Expected Result**	**Test Date**	**Test Result**
Add User				
1	แสดงรายการ User ทั้งหมด	รายการ User ปรากฏครบพร้อมข้อมูล	16/7/68	Pass
2	เพิ่ม User ใหม่ด้วยข้อมูลครบถ้วน	User ถูกเพิ่มและแสดงข้อมูลครบ	16/7/68	Pass
3	เพิ่ม User โดยไม่กรอกข้อมูลจำเป็น	แสดงข้อความแจ้งเตือนให้กรอกข้อมูลจำเป็น	16/7/68	Pass
4	แก้ไข User	ข้อมูล User ถูกแก้ไขและปรากฏในรายการ	16/7/68	Pass
5	ลบ User	User ถูกลบออกจากรายการ	16/7/68	Pass

ตารางที่ ค.4 ตารางแสดงการทดสอบระบบของหน้า การเพิ่มเอกสารสัญญาและเอกสารที่เกี่ยวข้อง

#	Test Scenario	Expected Result	Test Date	Test Result
Other Document				
1	แสดงรายการ Other Document ทั้งหมด	รายการ Other Document ปรากฏครบถ้วน	8/8/68	Pass
2	เพิ่ม Other Document พร้อมข้อมูลครบ	Other Document ถูกเพิ่มและแสดงข้อมูลครบ	8/8/68	Pass
3	เพิ่ม Other Document โดยไม่กรอกข้อมูลจำเป็น	แสดงข้อความแจ้งเตือนให้กรอกข้อมูลก่อนบันทึก	8/8/68	Pass
4	อัปโหลดไฟล์แนบ	ไฟล์ถูกอัปโหลดและชื่อไฟล์ปรากฏในรายการ	8/8/68	Pass
5	แก้ไขข้อมูล	ข้อมูลถูกแก้ไขและแสดงผล	8/8/68	Pass
6	ลบข้อมูล	ถูกลบออกจากรายการ	8/8/68	Pass

ตารางที่ ค.5 ตารางแสดงการทดสอบระบบของหน้า การเพิ่มรายละเอียดของโครงการ

#	Test Scenario	Expected Result	Test Date	Test Result
Scope Of Woek				
1	แสดงรายการ Scope of Work ทั้งหมด	รายการ Scope of Work ปรากฏครบถ้วน	24/7/68	Pass
2	เพิ่ม Scope of Work ใหม่พร้อมข้อมูลครบถ้วน	Scope of Work ถูกเพิ่มและแสดงข้อมูลครบ	24/7/68	Pass
3	เพิ่ม Scope of Work โดยไม่กรอกข้อมูลจำเป็น	แสดงข้อความแจ้งเตือนให้กรอกข้อมูลก่อนบันทึก	24/7/68	Pass
4	อัปโหลดไฟล์แนบ	ไฟล์ถูกอัปโหลดและชื่อไฟล์ปรากฏในรายการ	24/7/68	Pass
5	แก้ไขข้อมูล	ข้อมูลถูกแก้ไขและแสดงผล	24/7/68	Pass
6	ลบข้อมูล	ถูกลบออกจากรายการ	24/7/68	Pass

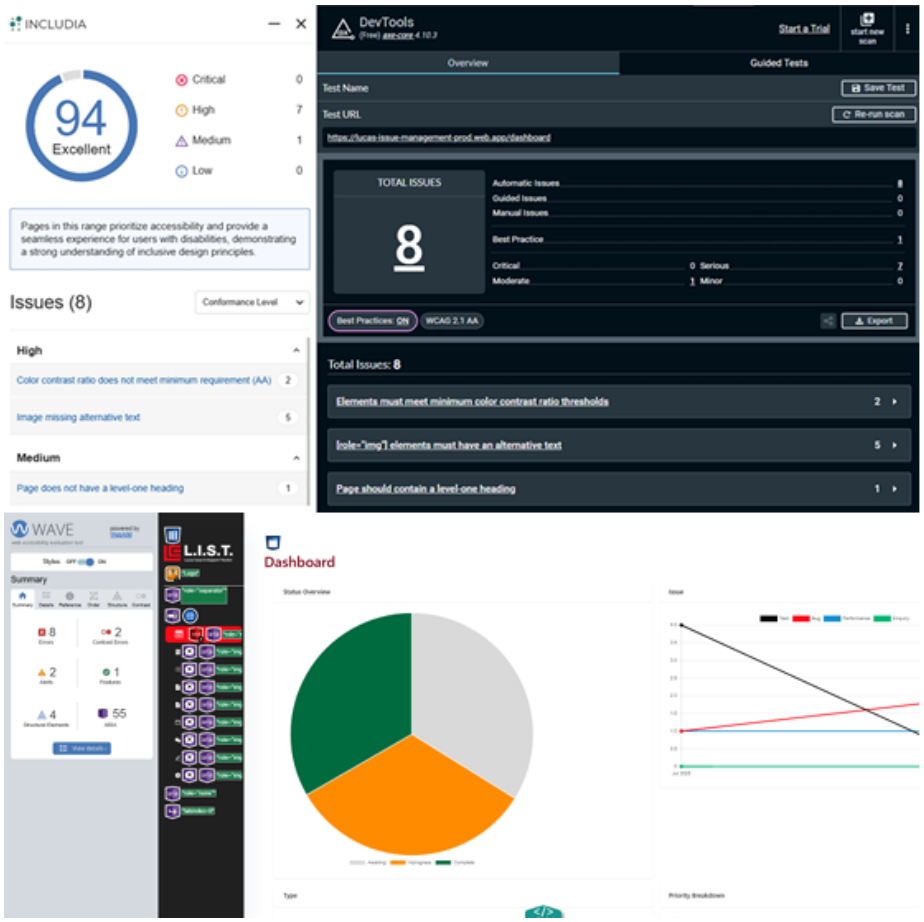
ตารางที่ ค.6 ตารางแสดงการทดสอบระบบของหน้า การแจ้งรายงานปัญหา

#	Test Scenario	Expected Result	Test Date	Test Result
Projects				
1	แสดงรายการโปรเจกต์ทั้งหมด	รายการโปรเจกต์ปรากฏครบถ้วน	29/7/68	Pass
2	แสดงรายการ Issue ของโปรเจกต์	Issue ของโปรเจกต์ปรากฏครบ	29/7/68	Pass
3	เพิ่ม Issue ใหม่พร้อมข้อมูลครบถ้วน	Issue ถูกเพิ่มและแสดงข้อมูลครบ	29/7/68	Pass
5	แก้ไขข้อมูล พร้อมอัพเดตสถานะของ Issue	ข้อมูลถูกแก้ไขและแสดงผล	29/7/68	Pass
6	สำเนาข้อมูล	ข้อมูลถูกสำเนาและแสดงผล	29/7/68	Pass
7	ลบข้อมูล	ข้อมูลถูกลบออกจากรายการ	29/7/68	Pass

7. วิเคราะห์ผลการทำ Accessibility Testing

การประเมินหน้า Dashboard ด้วยเครื่องมือ Includia Accessibility Checker, axe DevTools และ WAVE พบว่าแม้จะไม่ปรากฏปัญหาในระดับ Critical แต่ยังมีข้อจำกัดที่ส่งผลต่อผู้ใช้ที่มีความบกพร่องทางสายตา ได้แก่ ความแตกต่างของสี (Color Contrast) ที่ไม่สอดคล้องกับ WCAG 2.1 AA ทำให้ข้อความและองค์ประกอบสำคัญบางส่วนมองเห็นไม่ชัดเจน การขาดหัวข้อหลักระดับ H1 และการจัดลำดับหัวข้อที่ไม่สอดคล้องกับตรรกะ ส่งผลต่อการทำงานของ Screen Reader นอกจากนี้ยังพบการขาด alt attribute ในภาพ และการไม่มีคำอธิบายของปุ่มหรือไอคอนบางส่วน ตามมาตรฐาน WCAG 1.1.1 ทำให้ผู้ใช้ไม่สามารถรับรู้หน้าที่หรือการทำงานขององค์ประกอบได้ครบถ้วน

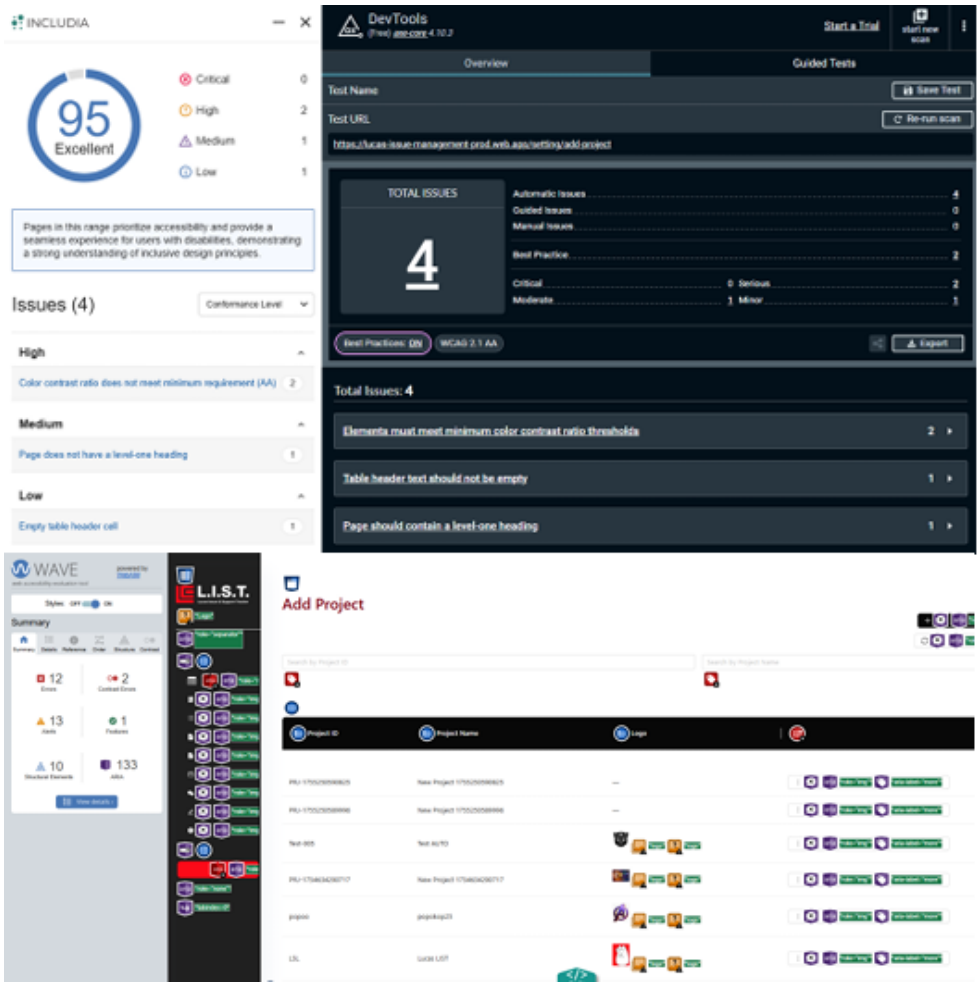
แนวทางปรับปรุงที่แนะนำ ได้แก่ การปรับค่า contrast ของสีให้สอดคล้องมาตรฐาน การจัดโครงสร้างหัวข้ออย่างถูกต้อง และการเพิ่มข้อความกำกับให้กับองค์ประกอบที่ไม่ใช่ข้อความ เพื่อให้ผู้ใช้ทุกกลุ่มสามารถเข้าถึงและใช้งานได้อย่างครบถ้วน



รูปที่ ค.17 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Dashboard

หน้าการเพิ่มข้อมูลของ โครงงาน (Add Project) แม้ไม่พบปัญหาในระดับ Critical แต่ยังพบข้อจำกัดด้านการเข้าถึงในระดับสูง ได้แก่ ความแตกต่างของสี (Color Contrast) ที่ไม่ผ่านเกณฑ์ WCAG 2.1 AA ซึ่งส่งผลให้การอ่านฟอร์มและองค์ประกอบ UI ไม่ชัดเจน นอกจากนี้ยังพบข้อบกพร่องในแบบฟอร์มและตาราง เช่น ช่องกรอกข้อความที่ไม่มี label ปุ่มที่ใช้ไอคอนโดยไม่มีข้อความกำกับ และลิงก์ที่ซ้ำกันโดยไม่ระบุบริบท ซึ่งขัดกับ WCAG 3.3.2 (Labels or Instructions) และ WCAG 2.4.4 (Link Purpose) ทำให้ผู้ใช้ Screen Reader ไม่สามารถเข้าใจหน้าที่ขององค์ประกอบได้ครบถ้วน

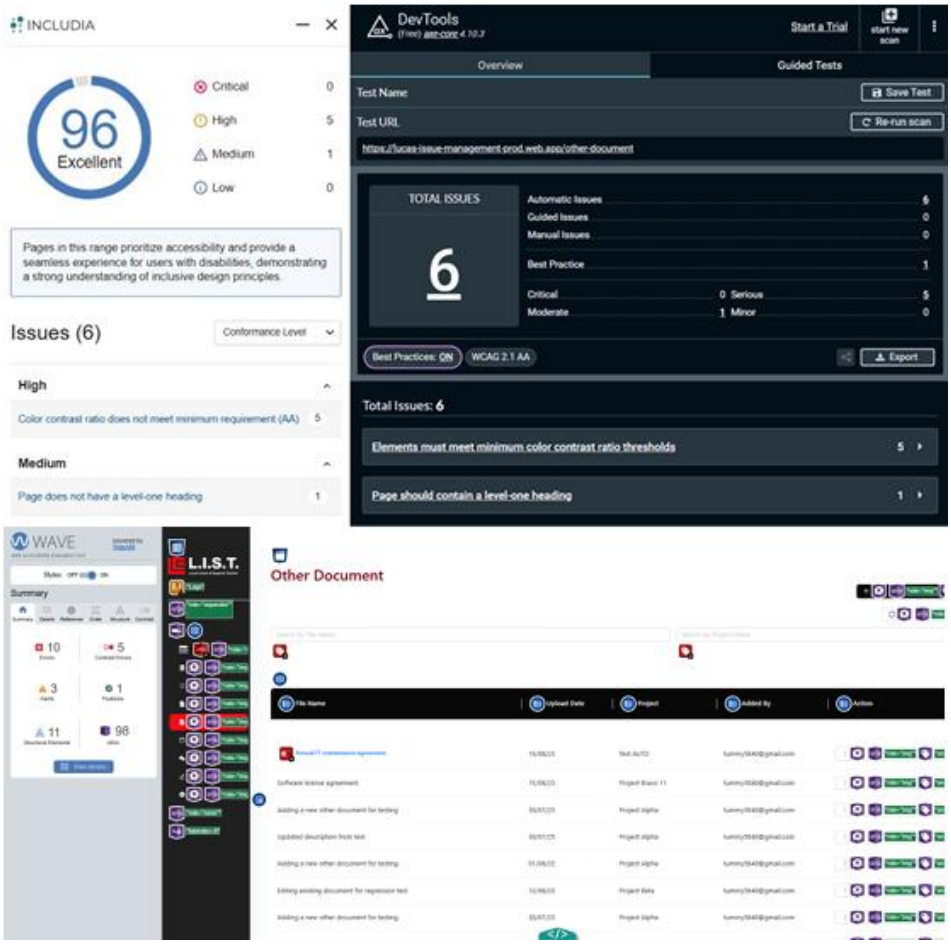
แนวทางแก้ไขที่เหมาะสม ได้แก่ การเพิ่ม label ให้ครบถ้วน การระบุข้อความกำกับปุ่มและลิงก์ให้ชัดเจน และการปรับสีของข้อความและองค์ประกอบ UI ให้สอดคล้องกับมาตรฐาน WCAG เพื่อให้ผู้ใช้ทุกกลุ่มสามารถเข้าถึงและใช้งานได้อย่างครบถ้วน



รูปที่ ค.18 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Dashboard

หน้าของ Other Document พบข้อจำกัดด้านการเข้าถึง แม้ไม่พบปัญหาในระดับ Critical แต่ยังมีประเด็นที่ส่งผลต่อผู้ใช้ โดยเฉพาะผู้ที่มีความบกพร่องทางสายตา เครื่องมือ Includia Accessibility Checker และ axe DevTools ตรวจพบปัญหาความแตกต่างของสี (Color Contrast) ที่ไม่สอดคล้องตามมาตรฐาน WCAG 2.1 AA รวมถึงการขาดหัวข้อหลักระดับ H1 ซึ่งอาจเป็นอุปสรรคต่อผู้ใช้ Screen Reader ในการรับรู้โครงสร้างเนื้อหา นอกจากนี้การตรวจสอบด้วย WAVE ยังพบว่าช่องค้นหาไม่มี label ปุ่ม Action เช่น “Edit” และ “Delete” ขาดคำอธิบายเฉพาะเจาะจง ข้อความบางส่วนมี contrast ต่ำ และตารางไม่มี caption หรือ scope ที่ชัดเจน ส่งผลให้ Screen Reader อ่านข้อมูลได้ไม่ครบถ้วน

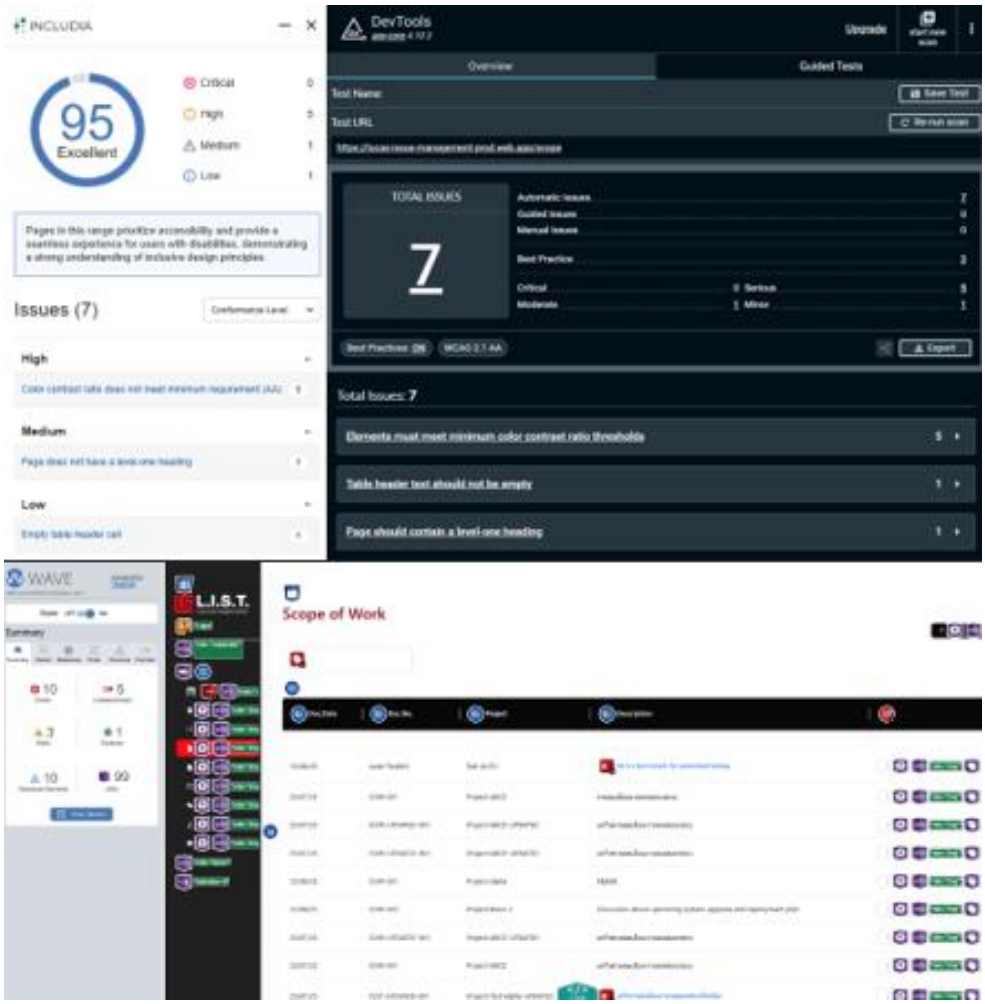
แนวทางการแก้ไขที่เหมาะสม ได้แก่ การปรับ contrast ของข้อความและองค์ประกอบ UI ให้ผ่านมาตรฐาน WCAG การเพิ่มหัวข้อ H1 ให้ชัดเจน การใส่ label ให้ช่องค้นหา การระบุข้อความปุ่ม Action ให้เฉพาะเจาะจง และการกำหนด caption และ scope ในตารางเพื่อช่วยให้ Screen Reader ตีความความสัมพันธ์ของข้อมูลได้อย่างถูกต้อง



รูปที่ ค.19 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Other Document

หน้าของ Scope of Work มีข้อจำกัดด้านการเข้าถึงในระดับที่ควรได้รับการแก้ไข แม้ไม่พบปัญหา Critical เกี่ยวกับ ARIA หรือฟอร์มที่ขาด label แต่ยังตรวจพบปัญหาความแตกต่างของสี (Color Contrast) ที่ไม่ผ่านมาตรฐาน WCAG 2.1 AA ซึ่งอาจสร้างอุปสรรคต่อผู้ใช้ที่มีความบกพร่องทางการมองเห็น นอกจากนี้ยังพบการขาดหัวข้อหลักระดับ H1 ที่จำเป็นต่อการทำความเข้าใจโครงสร้างเนื้อหาสำหรับผู้ใช้ Screen Reader รวมถึงปัญหาในโครงสร้างตาราง เช่น การเว้นช่องว่างในหัวตารางและการไม่เชื่อมโยงหัวตารางกับข้อมูลในเซลล์ ส่งผลให้ผู้ใช้ไม่สามารถรับรู้ความสัมพันธ์ของข้อมูลได้ครบถ้วน

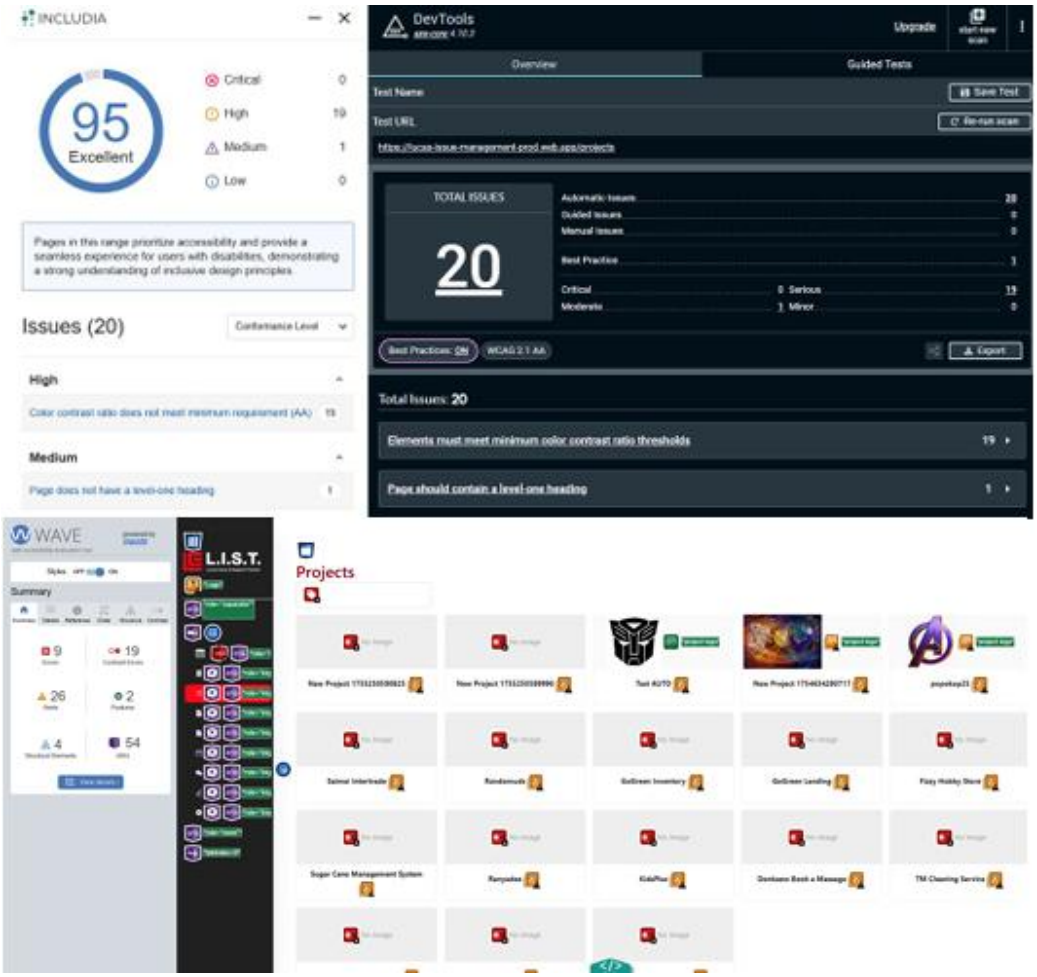
แนวทางการปรับปรุงที่เหมาะสม ได้แก่ การปรับอัตราส่วนความคมชัดของสีให้สอดคล้องตามมาตรฐาน WCAG การเพิ่มหัวข้อ H1 อย่างชัดเจน การเติมข้อความในเซลล์หัวตารางที่ว่าง และการกำหนด <th> และ scope ให้สัมพันธ์กับ <td> เพื่อช่วยให้ Screen Reader สามารถอ่านและตีความข้อมูลได้อย่างครบถ้วนและถูกต้อง



รูปที่ ค.20 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Scope of Work

หน้าของ Projects พบข้อจำกัดด้านการเข้าถึงในระดับสูง โดยเครื่องมือ axe DevTools ตรวจพบปัญหา Critical เกี่ยวกับการกำหนด ARIA Roles และ Attributes ที่ไม่สอดคล้องกับองค์ประกอบ ซึ่งอาจทำให้ผู้ใช้ Screen Reader ตีความข้อมูลผิดพลาด นอกจากนี้ยังพบความแตกต่างของสี (Color Contrast) ที่ไม่สอดคล้องตามมาตรฐาน WCAG 2.1 AA การขาดหัวข้อหลักระดับ H1 การไม่กำหนด alt text ให้กับภาพโครงการหรือโลโก้ และปุ่มบางส่วนมีข้อความซ้ำโดยไม่มีบริบทชัดเจน

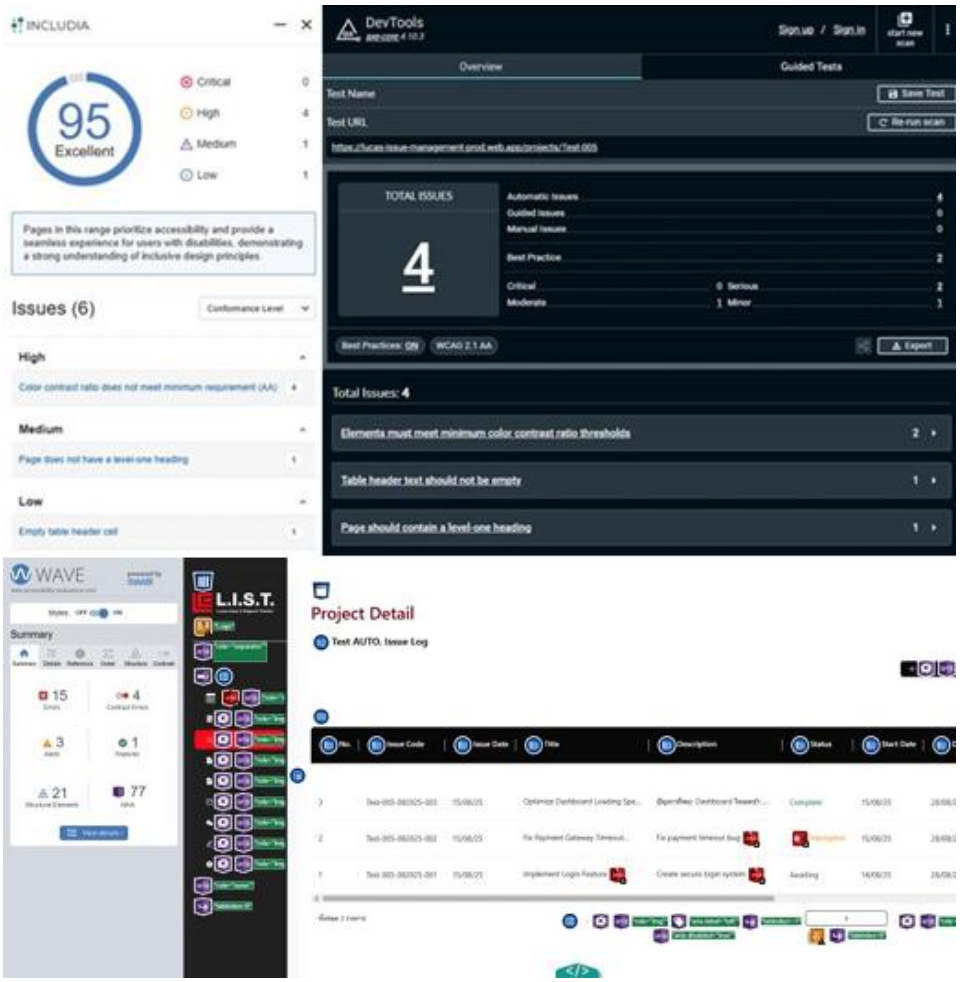
แนวทางแก้ไขที่แนะนำ ได้แก่ การปรับค่า contrast ของข้อความและองค์ประกอบ UI การกำหนดหัวข้อหลัก H1 อย่างชัดเจน การเพิ่ม alt text ให้ครบถ้วน และตรวจสอบการใช้ ARIA Roles ให้สอดคล้องกับหลัก WAI-ARIA เพื่อให้ผู้ใช้ทุกกลุ่มสามารถเข้าถึงและตีความข้อมูลได้อย่างถูกต้อง



รูปที่ ค.21 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Projects

หน้าของ Project Detail ได้รับคะแนนการประเมินระดับ Excellent (95 คะแนน) สะท้อนถึงการออกแบบที่รองรับผู้ใช้อย่างมีประสิทธิภาพ อย่างไรก็ตามยังพบข้อจำกัดด้านการเข้าถึง ได้แก่ ความแตกต่างของสี (Color Contrast) ที่ไม่ผ่านเกณฑ์ WCAG 2.1 AA การขาด Table Header หรือ Header ว่างในตาราง การไม่กำหนด Accessible Name ให้กับปุ่ม ไอคอน และภาพ รวมถึงการขาดหัวข้อหลักระดับ H1 นอกจากนี้ยังพบการใช้ ARIA role ซ้ำซ้อน การไม่กำหนด <label> ครบถ้วน และ Tabindex ที่ไม่เป็นระบบ ซึ่งส่งผลต่อผู้ใช้คีย์บอร์ดและเทคโนโลยีช่วยเหลือ

แนวทางแก้ไขที่แนะนำ ได้แก่ การปรับปรุงค่า contrast ของข้อความและองค์ประกอบ UI การเพิ่ม Table Header และ <label> การกำหนด Accessible Name ให้ครบถ้วน การใช้ ARIA role อย่างเหมาะสม และการจัด Tabindex ให้สอดคล้องกับลำดับการอ่าน เพื่อให้ผู้ใช้ทุกกลุ่มสามารถเข้าถึงและตีความข้อมูลได้อย่างครบถ้วน

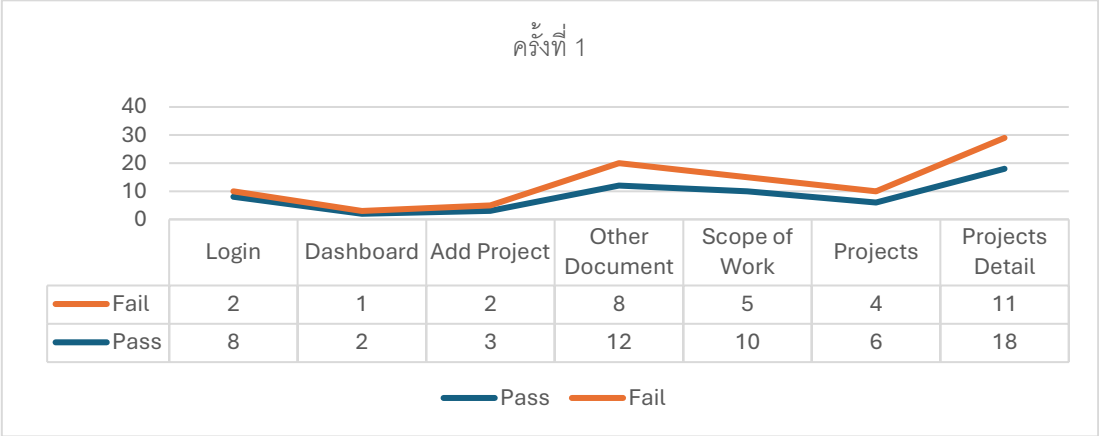


รูปที่ ค.22 การวิเคราะห์ความสามารถในการเข้าถึง (Accessibility) ของหน้า Project Detail

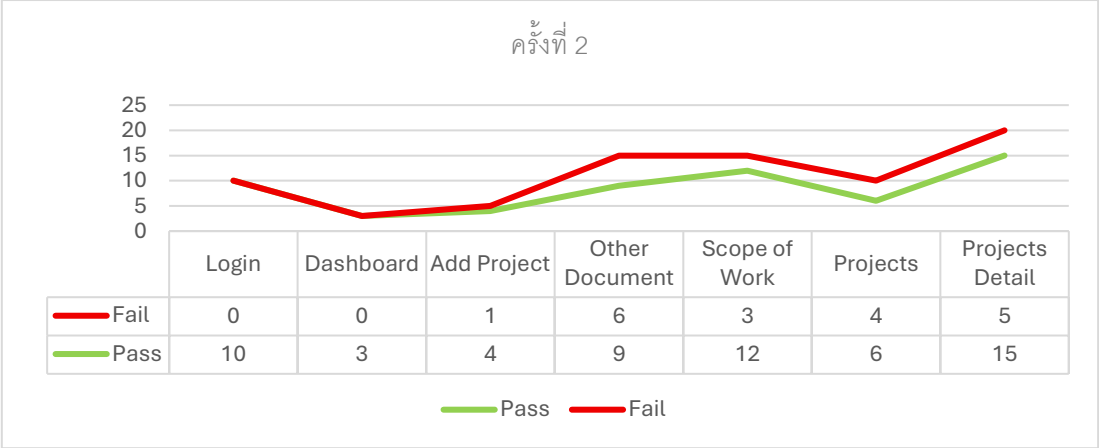
8. ผลการวิจัย

8.1 สรุปผลการทดสอบ Issue & Support Management System

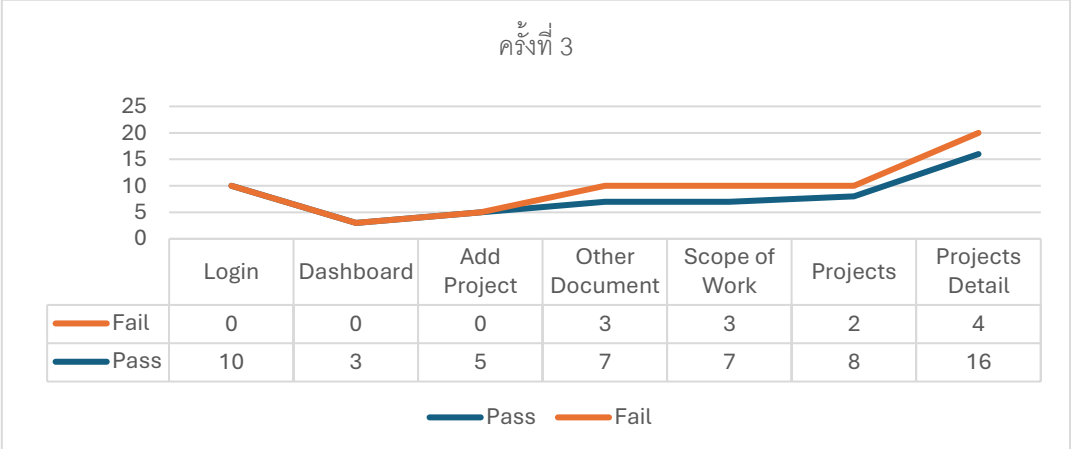
การทดสอบระบบ Issue & Support Management System มีวัตถุประสงค์เพื่อประเมินประสิทธิภาพ ความถูกต้อง และความเสถียรของการทำงานในฟังก์ชันหลัก โดยครอบคลุมตั้งแต่การเข้าสู่ระบบ (Login) การจัดการโครงการ (Project Management) การจัดการผู้ใช้งาน (User Management) การจัดการเอกสาร Scope of Work ตลอดจนการแจ้งและตอบรายงานปัญหา (Issue & Support Management) การทดสอบดำเนินการทั้งแบบแมนนวล (Manual Testing) และแบบอัตโนมัติ (Automated Testing) โดยใช้เครื่องมือ Playwright เพื่อเก็บข้อมูลเชิงปริมาณ ได้แก่ จำนวนครั้งที่ทดสอบ จำนวนครั้งที่ผ่านเกณฑ์ และจำนวนครั้งที่เกิดข้อผิดพลาด



รูปที่ ค.23 การทดสอบระบบอัตโนมัติ ครั้งที่ 1 ด้วยเครื่อง Playwright

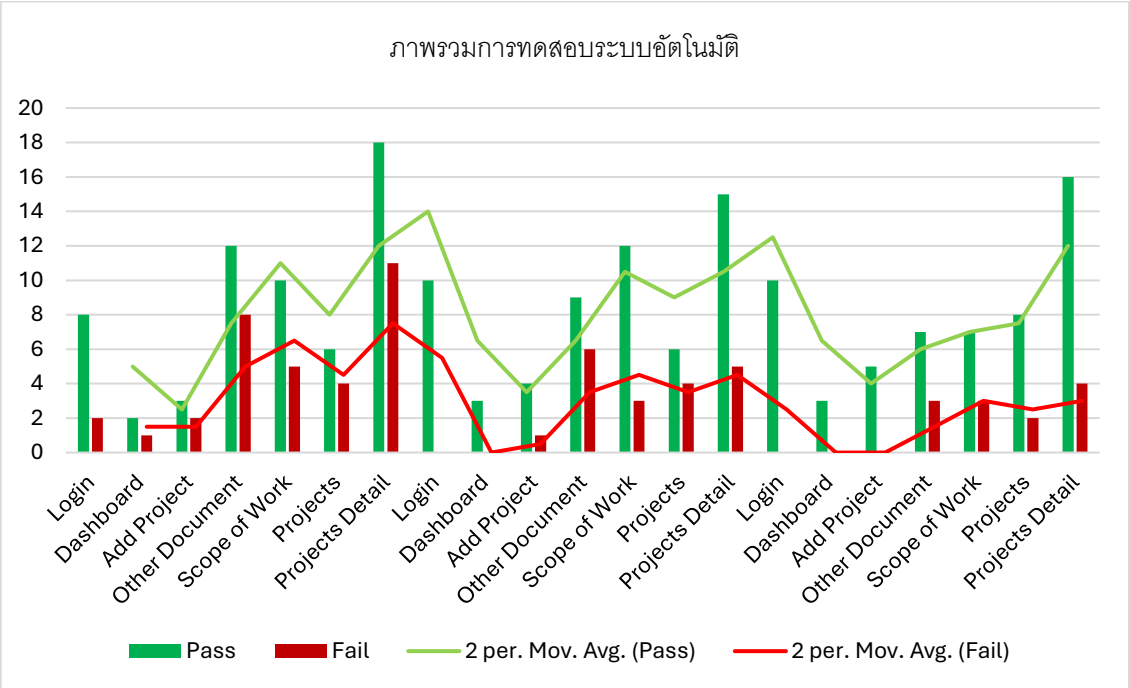


รูปที่ ค.24 การทดสอบระบบอัตโนมัติ ครั้งที่ 2 ด้วยเครื่อง Playwright



รูปที่ ค.25 การทดสอบระบบอัตโนมัติ ครั้งที่ 3 ด้วยเครื่อง Playwright

ผลการทดสอบในภาพรวมแสดงให้เห็นว่า ฟังก์ชันพื้นฐานมีความเสถียรและแม่นยำสูง ตัวอย่างเช่น ฟังก์ชันการเข้าสู่ระบบ (Login) ผ่านการทดสอบสำเร็จ 28 ครั้ง จากทั้งหมด 30 ครั้ง (ร้อยละ 93.3) และฟังก์ชันหน้าแดชบอร์ด (Dashboard) ผ่าน 8 ครั้งจาก 9 ครั้ง (ร้อยละ 88.9) ขณะที่ฟังก์ชันที่มีความซับซ้อนมากขึ้น เช่น การเพิ่มโครงการ (Add Project) มีอัตราการผ่าน 12 ครั้งจาก 15 ครั้ง (ร้อยละ 80) การจัดการเอกสารอื่น (Other Document) ผ่าน 28 ครั้งจาก 50 ครั้ง (ร้อยละ 56) และรายละเอียดโครงการ (Project Detail) ผ่าน 49 ครั้งจาก 69 ครั้ง (ร้อยละ 71) ส่วนฟังก์ชัน Scope of Work และ Project มีอัตราการผ่านที่ร้อยละ 64.4 และ 66.7 ตามลำดับ ผลดังกล่าวชี้ให้เห็นว่าฟังก์ชันที่เกี่ยวข้องกับข้อมูลจำนวนมากหรือมีหลายขั้นตอนการทำงานยังคงมีความผิดพลาด และต้องได้รับการปรับปรุงเพิ่มเติม

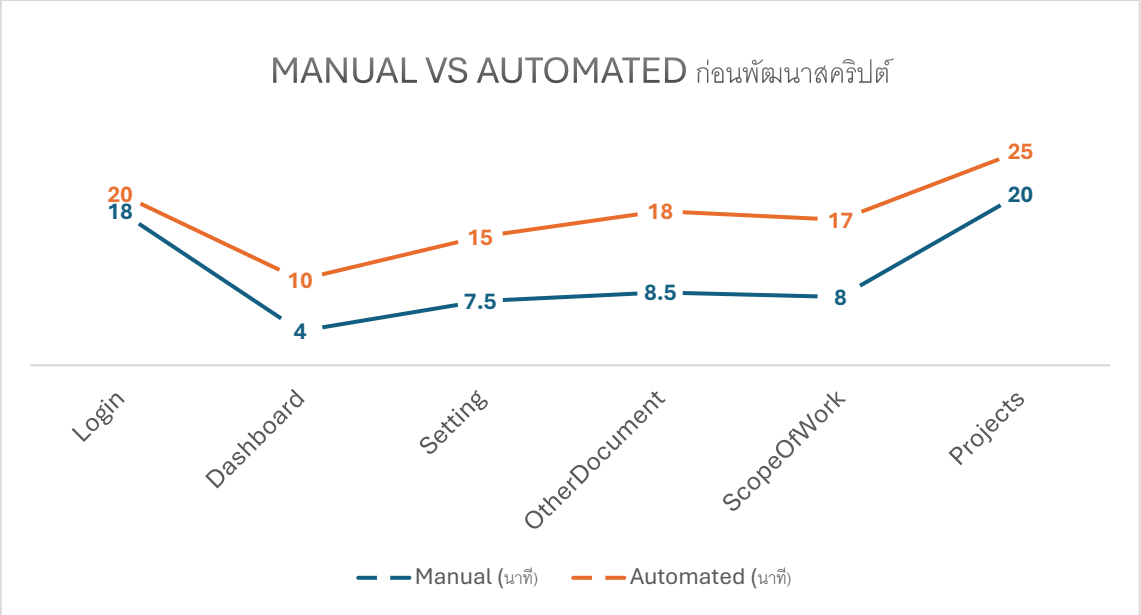


รูปที่ ค.26 ภาพรวมการทดสอบระบบอัตโนมัติ ด้วยเครื่อง Playwright

ในการเปรียบเทียบระหว่าง Manual Testing และ Automated Testing พบว่า Manual Testing มีความเหมาะสมในระยะเริ่มต้นของการทดสอบ เนื่องจากสามารถดำเนินการได้ทันทีโดยไม่ต้องใช้เวลาในการพัฒนาสคริปต์ ทั้งนี้ Manual Testing ใช้เวลาเฉลี่ย 3–20 นาทีต่อโมดูล ขณะที่ Automated Testing แม้จะสามารถรันซ้ำได้ แต่ยังไม่ครอบคลุมทุกกรณีทดสอบในระยะเริ่มต้น ส่งผลให้ไม่สามารถลดเวลาได้อย่างมีนัยสำคัญ ตัวอย่างเช่น การทดสอบฟังก์ชัน Login แบบ Manual ใช้เวลา 18 นาที ในขณะที่การทดสอบแบบ Automated ใช้เวลา 20 นาที (เพิ่มขึ้นร้อยละ 11.1) ผลดังกล่าวสะท้อนว่า Automated Testing ยังไม่เกิดประสิทธิภาพสูงสุดหากสคริปต์ยังไม่สมบูรณ์

ตารางที่ ค.7 ช่วงเริ่มต้นของการทดสอบ (Manual vs Automated ก่อนพัฒนาสคริปต์เต็ม)

หน้าที่ทำการทดสอบ	จำนวน Test Case	Manual	Automated	% ประหยัดเวลา
login	9	18 นาที	20 นาที	-11.1%
dashboard	2	4 นาที	10 นาที	150%
setting	4	7.5 นาที	15 นาที	-100%
otherDocument	4	8.5 นาที	18 นาที	-111.8%
scopeOfWork	4	8 นาที	17 นาที	-112.5%
projects	8	20 นาที	25 นาที	-25%

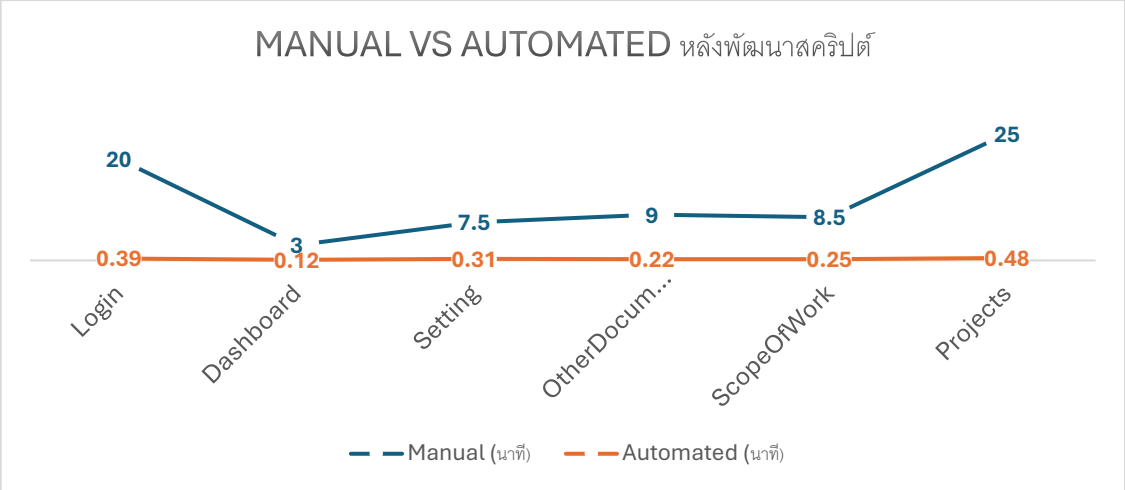


รูปที่ ค.27 Manual vs Automated ก่อนพัฒนาสคริปต์

ภายหลังจากการพัฒนาสคริปต์อัตโนมัติอย่างครบถ้วน พบว่า Automated Testing สามารถลดระยะเวลาในการทดสอบได้อย่างมีนัยสำคัญ โดยเมื่อเปรียบเทียบกับ Manual Testing แล้ว พบว่าสามารถประหยัดเวลาได้ร้อยละ 96–98 ในแต่ละโมดูล ตัวอย่างเช่น ฟังก์ชัน Login ใช้เวลาเพียง 0.39 นาทีจากเดิม 20 นาที (ลดลงร้อยละ 98) และฟังก์ชัน Projects ใช้เวลาเพียง 0.48 นาทีจากเดิม 25 นาที (ลดลงร้อยละ 98) นอกจากนี้ Automated Testing ยังช่วยลดข้อผิดพลาดที่เกิดจากมนุษย์ และสร้างหลักฐานการทดสอบที่สามารถตรวจสอบย้อนหลังได้อย่างครบถ้วน

ตารางที่ ค.8 หลังพัฒนาสคริปต์อัตโนมัติครบถ้วน (Manual vs Automated Run)

หน้าที่ทำการทดสอบ	จำนวน Test Case	Manual	Automated	% ประหยัดเวลา
login	9	20 นาที	0.39 นาที	98%
dashboard	2	3 นาที	0.12 นาที	96%
setting	4	7.5 นาที	0.31 นาที	96%
otherDocument	4	9 นาที	0.22 นาที	98%
scopeOfWork	4	8.5 นาที	0.25 นาที	97%
projects	8	25 นาที	0.48 นาที	98%



รูปที่ ค.28 Manual vs Automated หลังพัฒนาสคริปต์

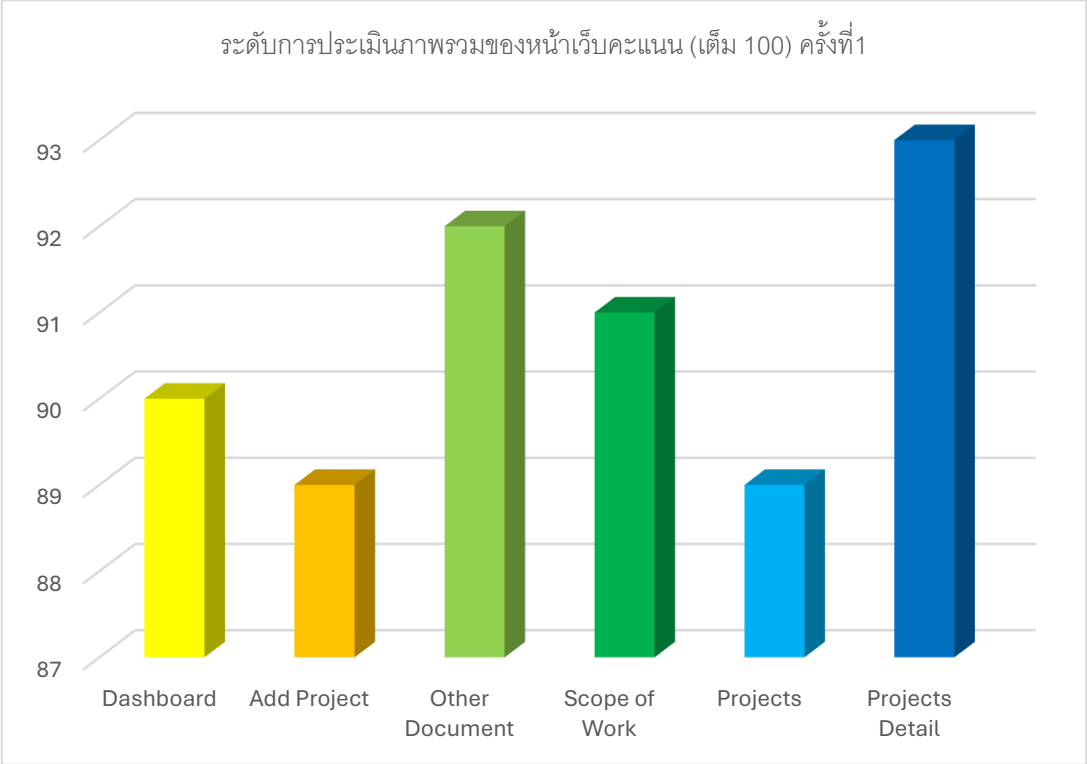
แม้ว่า Automated Testing จะมีข้อได้เปรียบด้านเวลาและความต่อเนื่อง แต่ Manual Testing ยังคงมีบทบาทสำคัญ โดยเฉพาะอย่างยิ่งในกรณีที่ต้องทดสอบฟังก์ชันใหม่ กรณีทดสอบเฉพาะ หรือสถานการณ์ที่ต้องอาศัยการวิเคราะห์เชิงลึกและการตัดสินใจของผู้ทดสอบ ซึ่งไม่สามารถดำเนินการด้วยการทดสอบอัตโนมัติได้อย่างเต็มรูปแบบ

การทดสอบระบบ Issue & Support Management System แสดงให้เห็นว่าระบบมีประสิทธิภาพในระดับสูง ทั้งด้านความถูกต้อง ความรวดเร็ว และความเสถียร สามารถตอบสนองต่อความต้องการใช้งานจริงขององค์กรได้อย่างเหมาะสม การนำ Automated Testing มาใช้สามารถยกระดับประสิทธิภาพในการทดสอบซ้ำ ลดเวลาในการดำเนินการ และเพิ่มความถูกต้องของผลการทดสอบ ในขณะที่ Manual Testing มีความเหมาะสมสำหรับกรณีการทดสอบเฉพาะ ทั้งนี้ ผลการวิจัยยังสะท้อนให้เห็นทิศทางในการปรับปรุงและพัฒนาระบบเพิ่มเติมในอนาคต เช่น การปรับปรุงการออกแบบฟอร์ม การเพิ่มประสิทธิภาพการประมวลผลข้อมูลขนาดใหญ่ และการเสริมมาตรการด้านความปลอดภัย เช่น การยืนยันตัวตนแบบสองขั้นตอน ซึ่งจะช่วยสร้างความเชื่อมั่นและเพิ่มคุณภาพการใช้งานระบบในระยะยาว

8.2 สรุปผลการทดสอบ Accessibility Testing

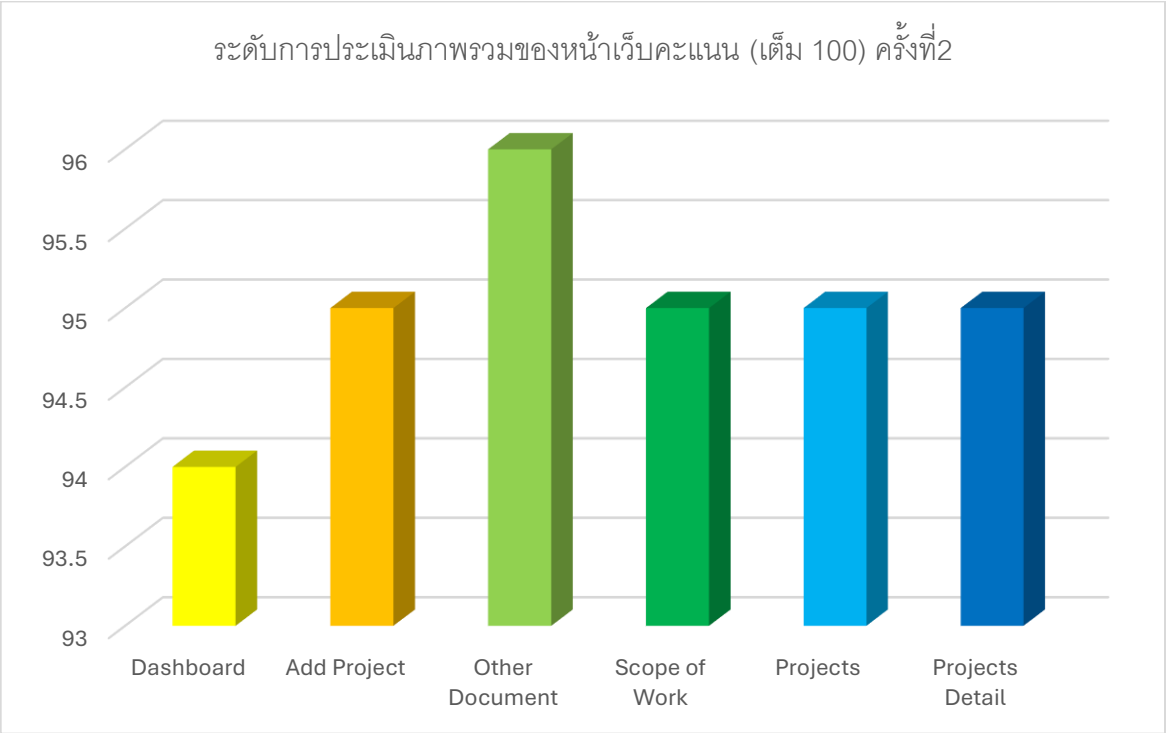
การทดสอบความสามารถในการเข้าถึงเว็บไซต์ (Web Accessibility Testing) มีวัตถุประสงค์เพื่อประเมินว่าเว็บไซต์สามารถรองรับการใช้งานของผู้ใช้ทุกกลุ่มได้อย่างเท่าเทียม โดยเฉพาะผู้ใช้ที่มีข้อจำกัดทางร่างกายหรือประสาทสัมผัส เช่น ผู้พิการทางสายตา การได้ยิน หรือการเคลื่อนไหว การทดสอบเน้นตรวจสอบความสามารถในการเข้าถึงข้อมูล ฟังก์ชันการทำงาน และเนื้อหาสำคัญของระบบ

ผลการทดสอบครั้งแรกพบข้อจำกัดบางประการ เช่น ขาดข้อความแทนภาพ (alt text) ปัญหาความคมชัดของสี (color contrast) และปุ่มบางรายการไม่สามารถเข้าถึงได้ด้วยคีย์บอร์ด ส่งผลให้การเข้าถึงเนื้อหายังไม่สมบูรณ์ อย่างไรก็ตาม คะแนนเฉลี่ยอยู่ในระดับ “Good” ได้แก่ หน้า Dashboard 90/100, Add Project 89/100, Other Document 92/100, Scope of Work 91/100, Projects 89/100 และ Projects Detail 93/100



รูปที่ ค.29 ภาพรวมระดับการประเมินภาพรวมของหน้าเว็บคะแนน (เต็ม 100) ครั้งที่ 1

หลังจากปรับปรุงองค์ประกอบที่เกี่ยวข้องกับ Accessibility ผลการทดสอบครั้งที่สองแสดงให้เห็นพัฒนาการชัดเจน โดยคะแนนในแต่ละหน้าสูงขึ้นอยู่ในระดับ “Excellent” ได้แก่ Dashboard 94/100, Add Project 95/100, Other Document 96/100, Scope of Work 95/100, Projects 95/100 และ Projects Detail 95/100



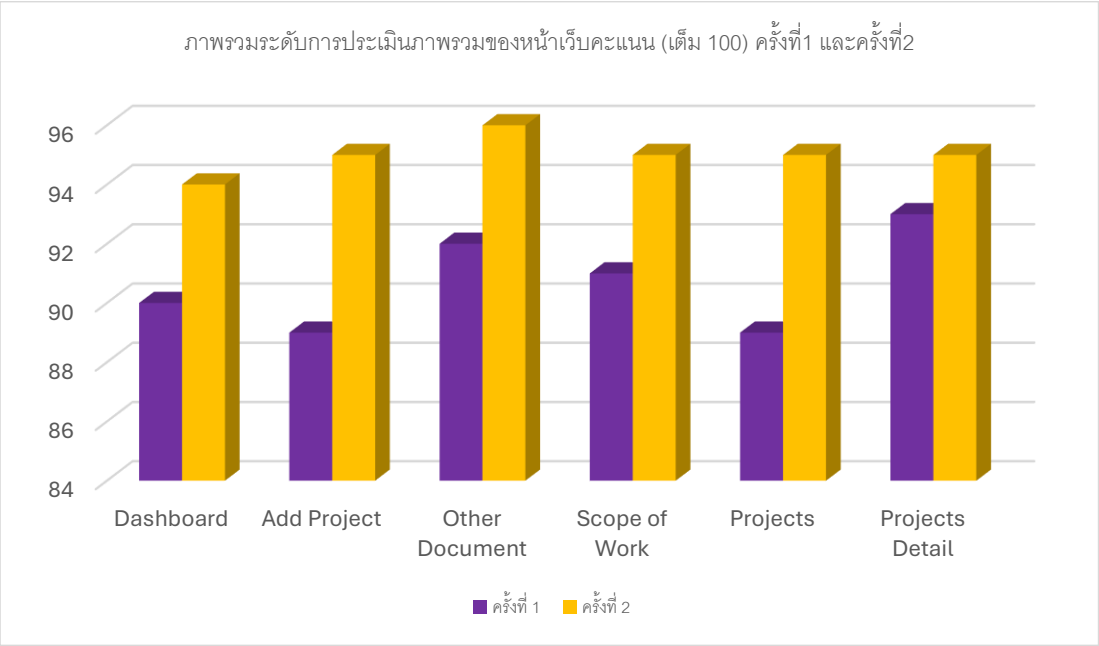
รูปที่ ค.30 ภาพรวมระดับการประเมินภาพรวมของหน้าเว็บคะแนน (เต็ม 100) ครั้งที่ 2

อย่างไรก็ตามยังพบข้อจำกัดบางประการ เช่น ความแตกต่างของสีบางส่วนที่ยังไม่สอดคล้องกับมาตรฐาน WCAG 2.1 และโครงสร้างตารางหรือการจัดเรียงข้อมูลที่อาจไม่เหมาะสมสำหรับโปรแกรมอ่านหน้าจอ การปรับปรุงประเด็นเหล่านี้จะช่วยยกระดับประสบการณ์ผู้ใช้ให้สมบูรณ์ยิ่งขึ้น

ตารางที่ ค.9 ผลการประเมิน Accessibility ก่อนและหลังปรับปรุง

หน้าเว็บ	คะแนนครั้งที่ 1	ระดับ	คะแนนครั้งที่ 2	ระดับ
Dashboard	90	Good	94	Excellent
Add Project	89	Good	95	Excellent
OtherDocument	92	Good	96	Excellent
Scope of Work	91	Good	95	Excellent
Projects	89	Good	95	Excellent

จากตารางพบว่าระบบได้รับการปรับปรุงจากระดับ “Good” ไปสู่ “Excellent” สะท้อนถึงการพัฒนาที่สอดคล้องกับมาตรฐาน WCAG โดยเฉพาะด้านการจัดโครงสร้างข้อมูลและการปรับปรุงสี อย่างไรก็ตาม ยังคงมีประเด็นที่ต้องพัฒนาเพิ่มเติม เช่น การปรับ color contrast และการใช้งานร่วมกับ screen reader



รูปที่ ค.31 ภาพรวมระดับการประเมินภาพรวมของหน้าเว็บคะแนน (เต็ม 100) ครั้งที่1 และครั้งที่2

สรุปผลการเปรียบเทียบทั้งสองครั้งพบว่าเว็บไซต์มีพัฒนาการด้าน Accessibility อย่างชัดเจน จากระดับ “Good” (คะแนนเฉลี่ยประมาณ 91/100) ในครั้งแรกไปสู่ระดับ “Excellent” (คะแนนเฉลี่ย 95/100) ในครั้งที่สอง การปรับปรุงดังกล่าวไม่เพียงแก้ไขข้อบกพร่องด้าน Accessibility เท่านั้น แต่ยังทำให้เว็บไซต์สามารถรองรับการใช้งานของผู้ใช้ทุกกลุ่มได้อย่างครอบคลุมและเท่าเทียม พร้อมทั้งสอดคล้องกับมาตรฐานสากลด้านการเข้าถึงข้อมูล (Web Content Accessibility Guidelines: WCAG) อย่างครบถ้วน

9. อภิปรายผล

ผลการวิจัยแสดงให้เห็นว่า การทดสอบแบบอัตโนมัติ (Automated Testing) มีประสิทธิภาพสูงกว่าการทดสอบแบบแมนนวล (Manual Testing) อย่างชัดเจน โดยเฉพาะในด้าน ความรวดเร็ว ความสม่ำเสมอของผลลัพธ์ และความถูกต้อง การนำเครื่องมือ Playwright ร่วมกับแนวคิด Page Object Model (POM) มาใช้ สามารถลดความซ้ำซ้อนของโค้ดทดสอบ และเพิ่มความสามารถในการบำรุงรักษาและนำกลับมาใช้ซ้ำได้อย่างมีประสิทธิภาพ ซึ่งสอดคล้องกับงานวิจัยของ Leotta et al. (2015) และ Stocco et al. (2017) ที่ยืนยันว่า POM เป็นแนวทางที่มีความคงทนและเหมาะสมต่อการออกแบบสคริปต์ทดสอบในระยะยาว

ในด้านการทดสอบการเข้าถึง (Accessibility Testing) ผลการวิจัยพบว่า ระบบสามารถรองรับผู้ใช้งานได้ในหลายมิติ แต่ยังคงมีข้อบกพร่องบางประการ เช่น ความแตกต่างของสี (Color Contrast) โครงสร้างของตาราง (Table Structure) และการรองรับการใช้งานด้วยคีย์บอร์ด ซึ่งสอดคล้องกับข้อกำหนดตามมาตรฐาน Web Content Accessibility Guidelines (WCAG, 2018) การปรับปรุงในประเด็นดังกล่าวจึงเป็นสิ่งจำเป็นในการพัฒนาระบบให้ตอบสนองผู้ใช้งานทุกกลุ่มอย่างเท่าเทียม

ตารางที่ ค.10 การเปรียบเทียบ Manual Testing และ Automated Testing

ประเด็น	Manual Testing	Automated Testing
ความรวดเร็ว (Short-term)	ใช้เวลาน้อยในระยะแรก	ต้องใช้เวลาเขียนสคริปต์ แต่ทดสอบซ้ำได้รวดเร็ว
ความถูกต้องและสม่ำเสมอ	อาจมีข้อผิดพลาดจากมนุษย์	ลด Human Error ผลลัพธ์สม่ำเสมอ
ประสิทธิภาพ	เหมาะกับฟังก์ชันใหม่หรือทดสอบเฉพาะกิจ	เหมาะกับฟังก์ชันซ้ำซ้อนหรือใช้งานบ่อย
หลักฐานอ้างอิง	ต้องบันทึกด้วยมือ	สร้างรายงานอัตโนมัติ
ข้อจำกัด	ใช้แรงงานมาก ทดสอบซ้ำจำกัด	ต้องใช้ทักษะและเวลาในการพัฒนาเริ่มต้น

จากตาราง พบว่า Automated Testing มีประสิทธิภาพมากกว่าในระยะยาว โดยเฉพาะการทดสอบซ้ำและการจัดเก็บหลักฐานอัตโนมัติ ขณะที่ Manual Testing ยังคงจำเป็นสำหรับฟังก์ชันใหม่ ๆ หรือการตรวจสอบเชิง Usability ที่ Automated Testing ยังไม่สามารถครอบคลุมได้

อย่างไรก็ตาม แม้การทดสอบแบบอัตโนมัติจะช่วยลดเวลาและลดความผิดพลาดที่เกิดจากมนุษย์ (Human Error) ได้อย่างมีนัยสำคัญ แต่ก็ยังไม่สามารถแทนที่การทดสอบแบบแมนนวลได้ทั้งหมด โดยเฉพาะในการประเมินประสบการณ์ผู้ใช้ (Usability Testing) และการทดสอบกรณีการใช้งานที่ซับซ้อนซึ่งต้องอาศัยการวิเคราะห์เชิงบริบทของผู้ทดสอบ ผลลัพธ์นี้สอดคล้องกับข้อสังเกตของ Fewster และ Graham (1999) ที่ชี้ให้เห็นว่า Automated Testing ควรถูกมองว่าเป็น “เครื่องมือสนับสนุน” ไม่ใช่การแทนที่ Manual Testing อย่างสมบูรณ์

10. สรุปผลการวิจัย

การวิจัยครั้งนี้สามารถบรรลุวัตถุประสงค์ที่กำหนดไว้ โดยสรุปผลที่สำคัญได้ดังนี้

1. การทดสอบแบบอัตโนมัติด้วย Playwright และการประยุกต์ใช้แนวคิด POM มีประสิทธิภาพสูงกว่าการทดสอบแบบแมนนวล ทั้งในด้าน ความรวดเร็ว ความแม่นยำ และความสม่ำเสมอ โดยเฉพาะในการทดสอบซ้ำ (Regression Testing) และการทดสอบในระบบที่มีการเปลี่ยนแปลงบ่อย

2. การประเมินด้านการเข้าถึง (Accessibility) พบว่าระบบมีความสามารถรองรับผู้ใช้งานได้ในหลายมิติ แต่ยังคงต้องปรับปรุงเพื่อให้เป็นไปตามมาตรฐาน WCAG อย่างสมบูรณ์ เช่น การปรับปรุงสี ข้อความกำกับ และการโต้ตอบผ่านคีย์บอร์ด

3. แม้ว่าการทดสอบแบบอัตโนมัติจะช่วยเพิ่มประสิทธิภาพได้อย่างมาก แต่ Manual Testing ยังคงมีความสำคัญ ในการทดสอบฟังก์ชันใหม่ ๆ และการประเมินประสบการณ์การใช้งานจริงของผู้ใช้

4. การบูรณาการ Automated Testing และ Accessibility Testing เข้าสู่กระบวนการพัฒนาซอฟต์แวร์ ช่วยยกระดับคุณภาพของระบบ เพิ่มความเสถียร และรองรับการใช้งานได้อย่างครอบคลุมมากยิ่งขึ้น

กล่าวโดยสรุป งานวิจัยนี้ยืนยันว่า การนำ Automated Testing มาประยุกต์ใช้ร่วมกับแนวทางมาตรฐานด้าน Accessibility เป็นแนวทางที่สามารถช่วยให้องค์กรเพิ่มคุณภาพของซอฟต์แวร์ ลดต้นทุนด้านเวลา และทำให้ระบบมีความเหมาะสมต่อการใช้งานจริงในสภาพแวดล้อมที่ซับซ้อนและเปลี่ยนแปลงอยู่เสมอ

11. เอกสารอ้างอิง

Fewster, M., & Graham, D. (1999). *Software test automation*. Addison-Wesley.

Meszaros, G. (2007). *xUnit test patterns: Refactoring test code*. Addison-Wesley.

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing* (3rd ed.). Wiley.

Leotta, M., Clerissi, D., Ricca, F., & Spadaro, C. (2015). Improving test suites maintainability with the Page Object pattern: An industrial case study. In *2015 IEEE 8th International Conference on Software Testing, Verification and Validation Workshops (ICSTW)* (pp. 1–8). IEEE. https://doi.org/10.1109/ICSTW.2015.7107464

Sánchez-Gordón, M. L., & Luján-Mora, S. (2017). Accessibility considerations in software development processes: A systematic literature review. *Computer Standards & Interfaces, 54*, 140–164. https://doi.org/10.1016/j.csi.2017.01.002

Stocco, A., Micucci, D., & Tonella, P. (2017). Precision test reuse for GUI regression testing. In *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '17)* (pp. 209–220). ACM.https://doi.org/10.1145/3092703.3092724

International Software Testing Qualifications Board. (2018). *Certified tester foundation level syllabus*. ISTQB. https://www.istqb.org/certifications/certified-tester-foundation-level

World Wide Web Consortium. (2018). *Web content accessibility guidelines (WCAG) 2.1*. W3C. https://www.w3.org/TR/WCAG21/

Pressman, R. S., & Maxim, B. R. (2019). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill.

Ara, R., & Sánchez-Gordón, M. L. (2024). Automated accessibility testing: A systematic review. *Journal of Web Engineering, 23*(4), 511–534.

Automation Testing คืออะไร. (2025, มิถุนายน 13). *Medium*.
แหล่งที่มา https://medium.com/@thedeeppub/e5f4543624d8

CircleCI. (2025, มิถุนายน 18). What is E2E? A guide to end-to-end testing. *CircleCI Blog*. สืบ
แหล่งที่มาhttps://circleci.com/blog/what-is-end-to-end-testing

Narinrit. (2025, มิถุนายน 18). E2E testing ง่ายๆ ด้วย Playwright! *Medium*.
แหล่งที่มา https://medium.com/@narinritnj/091825fd5ea1

Playwright Tutorial for Beginners. (2025, มิถุนายน 23). *YouTube*.
แหล่งที่มา https://www.youtube.com/watch?v=wBf4vx03Tw4

Microsoft. (2025, มิถุนายน 25). Playwright documentation. *Playwright.dev*.
แหล่งที่มา https://playwright.dev/docs/intro

Praserts, P. (2025, มิถุนายน 25). POM นั้นสำคัญไฉน. *Medium*.
แหล่งที่มา https://medium.com/@ppraserts/7cbd092610ea

Microsoft. (2025, มิถุนายน 25). Page Object Model (POM) in Playwright. *Playwright.dev*.
สืบค้นจาก https://playwright.dev/docs/pom

Upskill UX. (2025, กรกฎาคม 30). Accessibility Evaluation: การประเมินการช่วยในการเข้าถึง. *Medium*.
แหล่งที่มา https://medium.com/upskill-ux/c590c40ef3b2

Yay Bouu. (2025, กรกฎาคม 30). WCAG 2.2 คืออะไร? ไม่ทำคนทำ UX/UI ควรต้องรู้. *Medium*.
แหล่งที่มา https://medium.com/@yay.bouu/35b70dbc9b3f