**บทที่ 4**

**การวิเคราะห์และออกแบบระบบ**

การพัฒนาโครงงาน เรื่อง การทดสอบอัตโนมัติสำหรับระบบเว็บแอพพลิเคชั่นการจัดการควบคุมต้นทุนโครงการก่อสร้าง มีการวิเคราะห์ออกแบบระบบ (System Analysis) ซึ่งเป็นการศึกษางานเพื่อออกแบบระบบการทำงาน สิ่งที่ต้องการปรับปรุงและแก้ไขงานให้มีทิศทางที่ดีขึ้น โดยมีขั้นตอนการวิเคราะห์ระบบ ดังนี้

**4.1 การวิเคราะห์ระบบ**

การทดสอบระบบการจัดการควบคุมต้นทุนโครงการก่อสร้าง และกระบวนการบันทึกค่าใช้จ่าย ระบบที่เหมาะสมสำหรับการทดสอบและขั้นตอนในการทดสอบ คือ ระบบการทดสอบแบบอัตโนมัติ (Automated Testing) ดังนี้

**4.1.1 ระบบการจัดการควบคุมต้นทุนโครงการก่อสร้าง** เป็นระบบที่ใช้ในการบันทึกและจัดเก็บข้อมูลเกี่ยวกับค่าใช้จ่ายที่เกี่ยวข้องกับโครงการก่อสร้าง รวมถึงค่าวัสดุที่ใช้ในการก่อสร้าง ค่าแรงงานที่ใช้จ้าง ค่าใช้จ่ายในการจัดหาวัสดุ ค่าใช้จ่ายเพิ่มเติม เป็นต้น

**4.1.2 การทดสอบแบบอัตโนมัติ (Automated Testing)** เป็นการใช้เครื่องมือหรือสคริปต์เพื่อสร้างกรณีทดสอบและทำการทดสอบโดยอัตโนมัติ ซึ่งสามารถช่วยลดเวลาและความผิดพลาดในการทดสอบได้ เพื่อให้การทดสอบมีประสิทธิภาพและถูกต้องมากยิ่งขึ้น

ระบบการจัดการควบคุมต้นทุนโครงการก่อสร้างที่ถูกพัฒนาขึ้นจะประกอบด้วยองค์ประกอบต่างๆ เช่น

**4.1.3 โมดูลการบันทึกค่าใช้จ่าย** ระบบที่ช่วยในการบันทึกและจัดเก็บข้อมูลเกี่ยวกับค่าใช้จ่ายต่างๆ ที่เกี่ยวข้องกับโครงการ เช่น ค่าวัสดุ ค่าแรงงาน ค่าใช้จ่ายในการจัดหาวัสดุ เป็นต้น

**4.1.4 โมดูลการวิเคราะห์ผลกำไรและขาดทุน** ระบบที่ช่วยในการวิเคราะห์และประเมินผลกำไรหรือขาดทุนของโครงการ โดยพิจารณาค่าใช้จ่ายทั้งหมดที่เกิดขึ้นในโครงการ

**4.1.5 โมดูลการบริหารจัดการโครงการ** ระบบที่ช่วยในการจัดการโครงการเพื่อให้มีประสิทธิภาพ เช่น การวางแผนการใช้ทรัพยากร การติดตามความคืบหน้า การจัดการเวลา เป็นต้น

การทดสอบมีหลากหลายรูปแบบในการทดสอบ เช่น

**- Unit Testing** การทดสอบแต่ละโมดูลของระบบเพื่อตรวจสอบว่ามีความพร้อมที่จะทำงานได้ถูกต้องหรือไม่

**- Integration Testing** การทดสอบการทำงานร่วมกันของหลายๆ โมดูล เพื่อตรวจสอบว่าการติดต่อกันของส่วนต่างๆ ถูกต้อง

**- Functional Testing** การทดสอบระบบเพื่อตรวจสอบว่าส่วนต่างๆ ทำงานตามฟังก์ชันที่กำหนดไว้หรือไม่

**- Regression Testing** การทดสอบเพื่อตรวจสอบว่าการเปลี่ยนแปลงในระบบไม่ได้ทำให้ส่วนอื่นๆ เสียหาย

**- E2E Testing (End-to-End Testing)** การทดสอบซอฟต์แวร์หรือแอปพลิเคชันที่ครบทุกขั้นตอนของการใช้งานจริงโดยจำลองการทำงานข้ามทุกส่วนของระบบเพื่อตรวจสอบความถูกต้องและความสมบูรณ์ของระบบในสถานการณ์การใช้งานจริงของผู้ใช้

**4.2 ขั้นตอนการวิเคราะห์ระบบ**

การทดสอบระบบการจัดการควบคุมต้นทุนโครงการก่อสร้าง มีขั้นตอนในการวิเคราะห์ระบบ ดังนี้

4.2.1 การออกแบบการทดสอบระบบ

4.2.1.1 เป้าหมายและความต้องการของการทดสอบ (Testing Objectives and Requirements) กำหนดวัตถุประสงค์ของการทดสอบ และรายละเอียดของความต้องการที่ควรให้ระบบทดสอบรองรับ เช่น ประเภทของซอฟต์แวร์ที่ต้องการทดสอบ ฟีเจอร์ที่ต้องการทดสอบ และการรองรับการทดสอบแบบอัตโนมัติ

4.2.1.2 การวิเคราะห์และออกแบบเครื่องมือและสิ่งอำนวยความสะดวก (Tooling and Infrastructure Design) กำหนดว่าจะใช้เครื่องมือและสิ่งอำนวยความสะดวกใดในการทดสอบอัตโนมัติ เช่น เลือกเฟรมเวิร์ค (framework), ไลบลารีที่ใช้ หรือเครื่องมือที่เหมาะสม และกำหนดโครงสร้างพื้นฐาน (infrastructure) ที่จำเป็น

4.2.1.3 ออกแบบ Test Cases สร้างรายการทดสอบ ซึ่งเป็นขั้นตอนที่ต้องทำเพื่อตรวจสอบว่าซอฟต์แวร์ทำงานถูกต้องตามความต้องการหรือไม่ ต้องพิจารณากรณีที่ควรทดสอบและกรณีที่ควรหลีกเลี่ยง

* กรณีที่ควรหลีกเลี่ยง ได้แก่

- การทดสอบซ้ำซ้อน (Redundant Testing) การทดสอบโดยใช้ชุดทดสอบที่มีรายการทดสอบเหมือนกันซ้ำๆ

- การทดสอบที่ไม่มีความสำคัญ (Unnecessary Testing) การทดสอบรายละเอียดที่ไม่สำคัญหรือไม่มีผลต่อความสามารถหรือการทำงานของซอฟต์แวร์ ควรเน้นการทดสอบในส่วนที่สำคัญและมีความเสี่ยงสูง

- การทดสอบที่ไม่ครอบคลุม (Incomplete Testing) การทดสอบที่ไม่ครอบคลุมรายการทดสอบที่จำเป็นสำหรับความสมบูรณ์ของระบบ ควรรับรองว่าทุกฟังก์ชันหลักถูกทดสอบอย่างครอบคลุม

- การทดสอบโดยไม่มีแผน (Ad-hoc Testing) การทดสอบโดยไม่มีแผนหรือรายการทดสอบที่ชัดเจน จะทำให้ยากต่อการติดตามและรายงานผลการทดสอบ

- การทดสอบที่ไม่เกี่ยวข้อง (Irrelevant Testing) การทดสอบฟังก์ชันหรือรายการทดสอบที่ไม่เกี่ยวข้องกับความต้องการหรือการทำงานของระบบ ควรเน้นที่การทดสอบที่สอดคล้องกับความต้องการของผู้ใช้และฟังก์ชันหลัก

- การทดสอบที่ซับซ้อนเกินไป (Over-Testing) การทดสอบที่เน้นไปกับรายละเอียดเฉพาะเจาะจงที่ไม่เป็นส่วนสำคัญ สามารถทำให้เสียเวลาและทรัพยากรที่ไม่จำเป็น

- การทดสอบที่ไม่คิดถึงประสิทธิภาพ (Inefficient Testing) การทดสอบที่ไม่มีแผนหรือไม่ถูกนำเสนอในลักษณะที่เป็นประสิทธิภาพ ควรวางแผนการทดสอบให้มีโครงสร้างและการใช้ทรัพยากรที่มีประสิทธิภาพ

- การทดสอบที่ไม่พิจารณาภายนอก (No External Perspective) การทดสอบที่ไม่พิจารณาการใช้งานจริงของผู้ใช้ ควรพิจารณาจากมุมมองของผู้ใช้เพื่อให้การทดสอบมีความสมจริง

- การทดสอบที่ไม่สนใจประสิทธิภาพ (Performance Blindness) การทดสอบที่ไม่สนใจประสิทธิภาพและประสิทธิภาพของระบบ ควรทดสอบความสามารถในการจัดการกับภาระและประสิทธิภาพของระบบในสภาวะจริง

- การทดสอบโดยไม่คำนึงถึงการปรับปรุง (Lack of Improvement) การทดสอบที่ไม่คำนึงถึงการปรับปรุงและการเรียนรู้จากข้อผิดพลาดที่พบในการทดสอบ ควรใช้ข้อมูลจากการทดสอบในการปรับปรุงซอฟต์แวร์

4.2.1.4 เลือกและติดตั้งเครื่องมือทดสอบ (Select and Set Up Testing Tools) เลือกและติดตั้งเครื่องมือที่เหมาะสมสำหรับการทดสอบอัตโนมัติ เช่น Selenium WebDriver สำหรับการทดสอบเว็บแอปพลิเคชัน

4.2.1.5 เขียนและดำเนินการทดสอบ (Write and Execute Tests) สร้างสคริปต์หรือรหัสทดสอบตามที่ออกแบบไว้ และดำเนินการทดสอบ

4.2.1.6 การจัดการข้อมูลและการตั้งค่า (Data Management and Configuration) กำหนดการจัดการข้อมูลทดสอบและการตั้งค่าที่จำเป็นสำหรับการทดสอบ และสร้างข้อมูลทดสอบเพื่อใช้ในกระบวนการทดสอบ

4.2.1.7 การรายงานผลการทดสอบ (Reporting) กำหนดวิธีรายงานผลการทดสอบอย่างชัดเจน โดยรวมถึงการรายงานข้อผิดพลาดและข้อมูลที่เกี่ยวข้อง

4.2.1.8 การบำรุงรักษาและปรับปรุง (Maintenance and Improvement) ปรับปรุงระบบทดสอบเมื่อมีความเปลี่ยนแปลงในซอฟต์แวร์หรือความต้องการการทดสอบ และรักษาโครงสร้างทดสอบให้เป็นไปอย่างมีประสิทธิภาพ

4.2.1.9 การทดสอบและประเมิน (Testing and Evaluation) ทดสอบระบบทดสอบเพื่อให้แน่ใจว่าระบบทดสอบทำงานอย่างถูกต้องและตรงตามความต้องการ

4.2.1.10 การดูแลรักษาระบบทดสอบ (Test System Maintenance) ดูแลและรักษาระบบทดสอบเพื่อให้มีประสิทธิภาพตลอดเวลา

* Diagram ที่ใช้ในการวิเคราะห์ระบบ

- Use case diagram เป็นแผนภาพที่ใช้ในการแสดงและอธิบายวิธีการใช้ระบบซอฟต์แวร์หรือแอปพลิเคชันที่ถูกพัฒนาขึ้นโดยมุ่งเน้นที่ประสิทธิภาพการทำงานของระบบจากมุมมองของผู้ใช้งาน (ลูกค้าหรือผู้ใช้งานจริง) ซึ่งช่วยให้ทราบถึงความต้องการของระบบและการกระทำที่ผู้ใช้งานสามารถทำได้กับระบบนั้น ๆ

- Sequence diagram (แผนภาพลำดับเหตุการณ์) เป็นแผนภาพที่ใช้ในการแสดงแบบกราฟิคและแสดงลำดับของการเรียกใช้วัตถุหรือออบเจ็กต์ในระบบซอฟต์แวร์ โดยสร้างแผนภาพที่แสดงการกระทำและการโต้ตอบระหว่างวัตถุหรือออบเจ็กต์เวลาที่แน่นอน

- Activity diagram (แผนภาพกิจกรรม) เป็นแผนภาพที่ใช้ในการแสดงกระบวนการหรือลำดับขั้นตอนของกิจกรรมในระบบซอฟต์แวร์ โดยเน้นการแสดงกระบวนการทำงานและสามารถใช้ในการอธิบายรายละเอียดของการกระทำในแต่ละขั้นตอนและความสัมพันธ์ระหว่างกิจหกรรมต่าง ๆ ในระบบ

- ER Diagram หรือ Entity-Relationship Diagram (แผนภาพ Entity-Relationship) เป็นเครื่องมือที่ใช้ในการแสดงแบบกราฟิคและสัญลักษณ์เพื่อแสดงความสัมพันธ์ระหว่างข้อมูลหรือแต่ละองค์ประกอบในฐานข้อมูล โดยมักถูกใช้ในการวางแผนและออกแบบระบบฐานข้อมูล