**บทที่ 3**

**เทคโนโลยีที่ใช้ในการศึกษา**

ระบบการจัดการควบคุมต้นทุนโครงการก่อสร้างของบริษัท ลิงค์ อินโนว่า พร็อพเพอร์ตี้ จำกัด เป็นระบบที่อยู่ในกลุ่มเว็บแอปพลิเคชัน ได้นำเทคโนโลยีมาใช้ในการศึกษาและพัฒนา ดังนี้

3.1 ซอฟต์แวร์ที่ใช้ในการพัฒนาระบบ

3.1.1 Visual Studio Code (VSCode)

3.2 ซอฟต์แวร์ที่ใช้ในการทดสอบระบบ

3.2.1 Robot Framework

3.2.2 SeleniumLibrary

3.2.3 Chrome Driver

3.3 ซอฟต์แวร์ที่ใช้ในการทำ Test Case

3.3.1 MS Excel

**3.1 ซอฟต์แวร์ที่ใช้ในการพัฒนาระบบ**

**3.1.1 Visual Studio Code**

Visual Studio Code (VSCode) คือโปรแกรมแก้ไขและพัฒนาโค้ด (Code Editor) ที่พัฒนาโดย Microsoft ซึ่งเป็นเครื่องมือที่นักพัฒนาสามารถใช้ในการเขียนและแก้ไขโค้ด ไม่ว่าจะเป็นโค้ดภาษาโปรแกรมต่างๆ เช่น JavaScript, Python, Java, C++, HTML, CSS และอื่นๆ รวมถึงการสนับสนุนสำหรับเฟรมเวิร์ก (Framework) ต่างๆ ที่มักถูกนำมาใช้ในการพัฒนาแอปพลิเคชันต่างๆ

โดยในด้านของการทำ Automated Test (การทดสอบอัตโนมัติ), VSCode เป็นเครื่องมือที่สามารถช่วยในการพัฒนาและจัดการกับโค้ดที่ใช้ในการเขียนและทดสอบการทำงานของซอฟต์แวร์ โดยเฉพาะในส่วนของ Unit Testing (การทดสอบหน่วย), Integration Testing (การทดสอบการทำงานร่วมกันของส่วนต่างๆ), และ End-to-End Testing (การทดสอบความสมบูรณ์ของระบบทั้งหมด) นอกจากนี้ยังมี Extensions หลายตัวที่สามารถช่วยในการเขียนและเรียกใช้งาน Automated Tests ซึ่งจะช่วยในการจัดการข้อมูลการทดสอบ การรันทดสอบอัตโนมัติ การสร้างรายงานผลการทดสอบ และฟีเจอร์อื่นๆ ที่ช่วยในกระบวนการทดสอบซอฟต์แวร์อย่างมีประสิทธิภาพและมีความน่าเชื่อถือ

**A screenshot of a computer program

Description automatically generated**

**รูปที่ 3.1 ตัวอย่างหน้าการใช้งานโปรแกรม Visual Studio Code**

**การติดตั้ง Extension หรือ ส่วนขยาย** : ต้องติดตั้งส่วนขยาย (extension) ของ Robot Framework ใน VSCode ก่อนเพื่อให้สามารถเขียนและจัดการกับไฟล์สคริปต์ของ Robot Framework ได้

A screenshot of a computer

Description automatically generated

**รูปที่ 3.2 ติดตั้ง Extensions : Robot Code**

ในรูปตัวอย่างนี้คือ Extensions ที่ชื่อว่า "Robot Code" เป็นส่วนขยายที่ใช้สำหรับรองรับการเขียนโค้ด Robot Framework ใน VSCode

การสร้างไฟล์สคริปต์ Robot Framework ใน VSCode สามารถทำได้ ดังนี้

**เปลี่ยนนามสกุลไฟล์** : หลังจากที่สร้างไฟล์ใหม่ ให้เปลี่ยนนามสกุลไฟล์ให้เป็น “.robot”A screenshot of a computer program

Description automatically generated

**รูปที่ 3.3 ไฟล์ที่สร้างขึ้น จากรูปคือไฟล์ “test”**

**A screenshot of a computer program

Description automatically generated**

**รูปที่ 3.4 เติมนามสกุล “.robot” ให้กับไฟล์ “test”**

**เริ่มเขียนโค้ด** : สามารถเริ่มเขียนโค้ด Robot Framework ในไฟล์นี้ได้ โดย

เพิ่มส่วน \*\*\* Settings \*\*\*, \*\*\* Variables \*\*\*, \*\*\* Keywords \*\*\*, และ \*\*\* Test Cases \*\*\* ตามความต้องการของในทดสอบ (ในส่วนต่างๆ เหล่านี้ จะนำไปอธิบายในหัวข้อของ Robot Framework)

A screenshot of a computer

Description automatically generated

**รูปที่ 3.5 เพิ่มส่วน \*\*\* Settings \*\*\*, \*\*\* Variables \*\*\*, \*\*\* Keywords \*\*\*, และ**

**\*\*\* Test Cases \*\*\***

**การรันและทดสอบ** : หากต้องการรันและทดสอบไฟล์สคริปต์ Robot Framework ที่สร้าง เปิดเทอร์มินัลหรือหน้าต่าง Terminal ใน VSCode และใช้คำสั่ง robot เพื่อรันไฟล์สคริปต์ ตัวอย่างเช่น robot test.robot

**A screenshot of a computer program

Description automatically generated**

**รูป 3.6 เปิด Terminal และใช้คำสั่ง robot เพื่อรันไฟล์สคริปต์**

**3.2 ซอฟต์แวร์ที่ใช้ในการทดสอบระบบ**

**3.2.1 Robot Framework**

Robot Framework เป็นเครื่องมือที่ใช้ในการทดสอบอัตโนมัติ (Test Automation Framework) โดยใช้คำสั่งในการทำงาน และคำสั่งที่ใช้ พัฒนามาจากภาษา Python ซึ่งถูกพัฒนาขึ้นเพื่อช่วยในการทดสอบและตรวจสอบความถูกต้องของซอฟต์แวร์ต่างๆ ในกระบวนการพัฒนาซอฟต์แวร์ (Software Development Lifecycle)

**A screenshot of a computer

Description automatically generated**

**รูปที่ 3.7 ตัวอย่างโค้ดของ Robot Framework**

โดย Robot Framework มีส่วนหลักในไฟล์สคริปต์ของ Robot Framework (.robot) ซึ่งช่วยในการกำหนดค่าและสร้างโครงสร้างการทดสอบ ดังนี้

1. \*\*\* Settings \*\*\*

2. \*\*\* Variables \*\*\*

3. \*\*\* Keywords \*\*\*

4. \*\*\* Test Cases \*\*\*

**1. \*\*\* Settings \*\*\*** เป็นส่วนที่ใช้ในการเรียกใช้งานไลบรารีต่างๆ เพื่อให้สามารถใช้คำสั่งที่เกี่ยวข้องกับระบบปฏิบัติการได้ โดยใช้คีย์เวิร์ดและค่าที่ต้องการกำหนดให้กับการรันหรือการทำงานโดยรวมของ Robot Framework

A close up of text

Description automatically generated

**รูปที่ 3.8 ตัวอย่างการกำหนด Setting**

จากรูปตัวอย่างข้างต้น

- “Library” เป็นการกำหนดไลบลารีที่ต้องการใช้ในการรันหรือการทดสอบ เพื่อเรียกใช้ Keywords จากไลบรารีเพื่อทำงานต่าง ๆ จากตัวอย่างใช้คีย์เวิร์ด SeleniumLibrary (ในส่วนนี้จะอธิบายในหัวข้อ 3.2.2)

**2.** **\*\*\* Variables \*\*\*** เป็นการกำหนดค่าตัวแปรที่ใช้ในการเก็บข้อมูลหรือค่าที่ใช้ในการทำงานของ Test Cases หรือ Keywords ต่างๆ และช่วยให้สามารถเปลี่ยนค่าของตัวแปรได้ง่ายเมื่อต้องการปรับปรุงแก้ไขค่าในที่เดียวแทนที่จะต้องแก้ไขทุกส่วนที่ใช้ค่านั้น

ตัวอย่างการใช้งาน Variables (ตัวแปร)

A computer code with different colored text

Description automatically generated with medium confidence

**รูปที่ 3.9 ตัวอย่างการกำหนด Variables (ตัวแปร)**

A screenshot of a computer program

Description automatically generated

**รูปที่ 3.10 ตัวอย่างการใช้งาน Variables (ตัวแปร) ใน Test Case**

จากรูปตัวอย่างข้างต้น

- ${BROWSER} และ ${URL} คือตัวแปรที่ถูกกำหนดในส่วนของ Variables (ตัวแปร)

- ${BROWSER} และ ${URL} ใช้ในการเก็บค่า เบราว์เซอร์ และ URL เพื่อใช้ใน Test Cases

- ใน Test Cases “chrome\_test” สามารถเรียกใช้ค่าตัวแปรเหล่านี้ในการใช้งานได้

**3. \*\*\* Keywords \*\*\*** คือชุดคำสั่งหรือการกระทำที่กำหนดขึ้นเพื่อใช้ในการสร้างเทสเคส หรือการจัดการกับคำสั่งในเทสเคส คีย์เวิร์ดหรือคำสั่งเหล่านี้เป็นที่กำหนดโดยตัวผู้ใช้เองและสามารถรวมกันเป็นชุดของคำสั่งที่มีวัตถุประสงค์ในการทำงานเฉพาะในเนื้อหาของ Test Cases ที่ผู้ใช้เขียน

A screen shot of a computer

Description automatically generated

**รูปที่ 3.11 ตัวอย่างการใช้งาน Keywords**

จากรูปตัวอย่างข้างต้น

- Open Chrome เป็นชื่อของ Keywords ที่ผู้ใช้กำหนดเอง

- Open Browser และ Sleep เป็นคีย์เวิร์ดและคำสั่งที่ถูกใช้เพื่อดำเนินการต่าง ๆ ในการทำงานที่ Keyword กำหนด

**4. \*\*\* Test Cases \*\*\*** เป็นส่วนหนึ่งในไฟล์ที่สร้างขึ้นเพื่อเขียนและกำหนดเทสเคส (Test Cases) ที่ต้องการทดสอบในโปรแกรมหรือระบบที่กำลังพัฒนา โดยส่วนนี้จะเป็นที่ที่ระบุว่าต้องการทดสอบอะไรและวิธีการทดสอบเป็นอย่างไร โดยใช้คีย์เวิร์ดและคำสั่งที่เป็นไปตามโครงสร้างของ Robot Framework

โครงสร้างของ Test Cases จะประกอบด้วยส่วนหลัก ๆ ดังนี้

- Test Case Name เป็นชื่อของเทสเคส เพื่อระบุว่าเทสเคสนี้ทำงานอะไร ชื่อนี้จะถูกใช้ในการระบุเทสเคสเมื่อรันการทดสอบ

- Documentation (หรือ การ Comment) ส่วนนี้สามารถใช้ในการเพิ่มคำอธิบายหรือคำอธิบายเพิ่มเติมเกี่ยวกับเทสเคส เช่น วัตถุประสงค์ของการทดสอบ, ข้อมูลทดสอบที่ใช้, คาดหวังว่าจะเกิดอะไรขึ้น เป็นต้น

- Test Steps (Keywords) เป็นส่วนที่ระบุคีย์เวิร์ดและคำสั่งที่ใช้ในการทดสอบ เช่น การคลิกที่ปุ่ม, กรอกข้อมูล, การชะลอเวลาการดำเนินการถัดไป เป็นต้น

ดังนั้น หากมอง Test Cases ในโครงสร้างเบื้องต้น จะมีลักษณะดังนี้

A screenshot of a computer program

Description automatically generated

**รูปที่ 3.12** ตัวอย่างโครงสร้างของ Test Case

จากรูปตัวอย่างข้างต้น

- chrome\_test และ chrome\_test1 เป็นชื่อของเทสเคส

- Documentation (หรือ การ Comment) ใช้ในการเพิ่มคำอธิบายหรือคำอธิบายเพิ่มเติมเกี่ยวกับเทสเคส โดยใช้ เครื่องหมาย “#”

- “Open Browser”, “Sleep”, “Maximize Browser Window” และ “Close Browser” เป็นคีย์เวิร์ดและคำสั่งที่ใช้ในการทดสอบ

เครื่องหมาย “\*\*\*” ใน Robot Framework เป็นสัญลักษณ์ที่ใช้ในการกำหนดหัวข้อของส่วนต่างๆ ในไฟล์สคริปต์ และมีความหมายพิเศษที่ระบุถึงรูปแบบและบทบาทของส่วนนั้น ๆ ดังนั้นการใช้ “\*\*\*” มีเป้าหมายและประโยชน์สำคัญต่อกระบวนการเขียนและอ่านไฟล์สคริปต์ Robot Framework ดังนี้

**การแยกส่วนของไฟล์** การใช้ “\*\*\*” ช่วยแยกและแบ่งส่วนของไฟล์สคริปต์ออกเป็นส่วนต่างๆ เช่น \*\*\* Settings \*\*\*, \*\*\* Variables \*\*\*, \*\*\* Keywords \*\*\*, และ \*\*\* Test Cases \*\*\* ทำให้โค้ดมีความเรียบร้อยและง่ายต่อการอ่าน.

**ระบุบทบาทของส่วน** “\*\*\*” มีบทบาทเป็นสัญลักษณ์ที่ระบุบทบาทและประเภทของส่วนที่กำลังมาถึง ดังนั้นจะช่วยในการทำความเข้าใจว่าส่วนนั้นมีการใช้งานและการทำงานอย่างไร.

**การทำหัวข้อสัญญาณ** เครื่องหมาย “\*\*\*” เป็นสัญญาณที่ช่วยให้ผู้เขียนโค้ดรู้ว่าส่วนนี้เป็นหัวข้อหรือส่วนสำคัญในไฟล์ และช่วยให้ง่ายต่อการค้นหาข้อมูลที่ต้องการ.

**การระบุรูปแบบที่มาตรฐาน** การใช้ “\*\*\*” ระบุรูปแบบที่มาตรฐานในการเขียนไฟล์สคริปต์ Robot Framework ทำให้ผู้ใช้ทราบว่าผู้ใช้สร้างหัวข้อหรือส่วนต่าง ๆ ตามข้อกำหนดของ Robot Framework.

ดังนั้น เครื่องหมาย “\*\*\*” เป็นสัญลักษณ์ที่เพื่อการแยกและระบุส่วนต่าง ๆ ในไฟล์สคริปต์ Robot Framework โดยมีประโยชน์ต่อกระบวนการเขียนและอ่านโค้ด.

การติดตั้ง Robot Framework สามารถทำได้ด้วยขั้นตอนต่อไปนี้

**ตรวจสอบการติดตั้ง Python** : Robot Framework ใช้ Python เป็นภาษาหลักในการทำงาน ดังนั้นควรตรวจสอบว่ามี Python ติดตั้งอยู่แล้วหรือไม่ โดยใช้คำสั่ง python --version หรือ python3 --version ใน Command Line (หรือ Terminal บน macOS หรือ Linux) เพื่อตรวจสอบเวอร์ชัน Python ที่มีในเครื่อง

**ติดตั้ง pip (Package Installer for Python)** : หากยังไม่มี pip ให้ทำการติดตั้งก่อน โดยปกติแล้ว pip จะถูกติดตั้งพร้อมกับ Python สำหรับเวอร์ชัน Python 2.x แต่สำหรับ Python 3.x อาจต้องใช้ pip3 แทน

**ติดตั้ง Robot Framework** : ใช้คำสั่งต่อไปนี้ใน Command Line หรือ Terminal เพื่อติดตั้ง Robot Framework และเครื่องมือที่เกี่ยวข้องผ่าน pip

|  |
| --- |
| pip install robotframework |

**ติดตั้งเครื่องมือสำหรับการเขียน Test Cases และการรัน** : Robot Framework มีเครื่องมือหลายตัวที่ช่วยในการเขียนและรัน Test Cases ได้ง่ายขึ้น สามารถติดตั้งเครื่องมือเหล่านี้ได้ดังนี้

- RIDE (Robot Framework IDE) : เป็นโปรแกรมที่ให้ GUI สำหรับการเขียนและรัน Test Cases โดยไม่ต้องเขียนโค้ดด้วยตนเอง

|  |
| --- |
| pip install robotframework-ride |

- Robot Framework Command Line Interface (CLI) : สำหรับการรัน Test Cases ผ่าน Command Line โดยใช้คำสั่ง robot (ในส่วนของการทำงานโปรเจคนี้ ใช้ CLI ในการรัน Test Cases)

ตรวจสอบการติดตั้ง : หลังจากติดตั้งเสร็จสิ้น สามารถตรวจสอบว่า Robot Framework ได้ถูกติดตั้งอย่างถูกต้องด้วยการใช้คำสั่ง

|  |
| --- |
| robot --version |

**เริ่มใช้งาน** : สามารถเริ่มเขียนและรัน Test Cases โดยใช้เครื่องมือต่าง ๆ ที่ติดตั้งไว้ หากใช้ RIDE สามารถเปิดโปรแกรมนั้นขึ้นมาแล้วเริ่มเขียน Test Cases ได้ทันที

**3.2.2 SeleniumLibrary**

SeleniumLibrary เป็นไลบรารี (Library) สำหรับ Robot Framework ที่ใช้ในการทำออโตเมชันและการทดสอบเว็บแอปพลิเคชันผ่านเบราว์เซอร์ เป็นเครื่องมือที่มีความสามารถในการควบคุมและจำลองการกระทำของผู้ใช้งานบนเว็บเบราว์เซอร์ เพื่อใช้ในการทดสอบความสมบูรณ์และความถูกต้องของเว็บแอปพลิเคชัน

โดย SeleniumLibrary มีคุณสมบัติที่สามารถทำสิ่งต่าง ๆ ได้ ตัวอย่างเช่น

- **การเปิดและปิดเบราว์เซอร์** สามารถเปิดและปิดบราวเซอร์ที่รองรับโดย Selenium ได้ เช่น Chrome, Firefox, Edge, Safari เป็นต้น

- **การควบคุมการทำงานบนเว็บ** สามารถควบคุมการกระทำบนเว็บได้ เช่น คลิกที่ ลิงก์, กรอกข้อมูลในฟอร์ม, ตรวจสอบเนื้อหาเว็บ เป็นต้น

- **การจัดการกับไฟล์และดาวน์โหลด** สามารถอัปโหลดและดาวน์โหลดไฟล์ได้ผ่าน การใช้ SeleniumLibrary

การใช้งาน SeleniumLibrary ใน Robot Framework มีขั้นตอนหลักๆ ดังนี้

**การเรียกใช้ SeleniumLibrary** : ตรวจสอบการติดตั้ง SeleniumLibrary ด้วยคำสั่ง

|  |
| --- |
| pip install robotframework-seleniumlibrary |

หลังจากนั้นใน Test Case ที่ต้องการใช้งาน SeleniumLibrary ให้เพิ่มการเรียกใช้ไลบรารีในส่วนของ Settings ด้วยคำสั่ง

|  |
| --- |
| \*\*\* Settings \*\*\*  Library SeleniumLibrary |

**การเรียกใช้ WebDriver** : เลือก WebDriver ที่ต้องการใช้งาน เช่น ChromeDriver สำหรับ Google Chrome (ในส่วนของการทำงานโปรเจคนี้ ใช้ ChromeDriver และจะอธิบายในหัวข้อ 3.2.3), GeckoDriver สำหรับ Mozilla Firefox, และอื่นๆ ที่สามารถกำหนดให้ WebDriver ทำงานกับเบราว์เซอร์ตามที่ต้องการ สามารถทำได้โดยใช้คำสั่ง Open Browser

|  |
| --- |
| Open Browser https://www.example.com chrome |

**การใช้ Keywords ใน SeleniumLibrary** : SeleniumLibrary มี Keywords มากมายให้ใช้งาน เช่น Click Element, Input Text, Select From List by Value, Wait Until Element is Visible, เป็นต้น ซึ่งสามารถใช้ Keywords เหล่านี้ใน Test Steps เพื่อทำการตรวจสอบหรือจำลองการทำงานบนเว็บเบราว์เซอร์

|  |
| --- |
| Click Element //button[@id='submit-button']  Input Text //input[@name='username'] MyUsername  Select From List by Value //select[@name='country'] US  Wait Until Element is Visible //div[@class='loading-spinner'] |

**การปิด WebDriver** : เมื่อเสร็จสิ้นการทดสอบหรือใช้งาน WebDriver เสร็จแล้วควรทำการปิด WebDriver ด้วยคำสั่ง Close Browser

|  |
| --- |
| Close Browser |

**การสร้าง Test Cases** : สามารถเรียงเรียง Keywords ที่ใช้งานกับ SeleniumLibrary เพื่อสร้าง Test Cases ตามที่ต้องการทดสอบบนเว็บ

|  |
| --- |
| \*\*\* Test Cases \*\*\*  Test Login  Open Browser https://www.example.com chrome  Input Text //input[@name='username'] myusername  Input Text //input[@name='password'] mypassword  Click Element //button[@id='login-button']  Wait Until Page Contains Welcome, User!  Close Browser |

**การรัน Test Cases** : เมื่อเขียน Test Cases เสร็จแล้ว สามารถใช้คำสั่ง robot บน Command Line หรือ Terminal เพื่อรัน Test Cases ได้

|  |
| --- |
| robot testfile.robot |

**3.2.3 Chrome Driver**

Chrome Driver เป็นเครื่องมือหรือโปรแกรมที่เป็นส่วนหนึ่งของ Selenium WebDriver ซึ่งถูกพัฒนาขึ้นเพื่อใช้ในการควบคุมและทดสอบเบราว์เซอร์ Google Chrome อย่างอัตโนมัติ และทำการออโตเมชันบนเว็บได้

ChromeDriver ใช้เพื่อเชื่อมต่อและควบคุมการทำงานของเบราว์เซอร์ Google Chrome ผ่านอินเทอร์เฟซของ Selenium WebDriver ที่ใช้สำหรับการทดสอบและออโตเมชันเว็บ สามารถใช้งาน ChromeDriver ได้ดังนี้

- ChromeDriver ช่วยในการเปิดหน้าต่างเบราว์เซอร์ Google Chrome ให้กับการทดสอบหรือออโตเมชัน

- สามารถใช้ ChromeDriver เพื่อควบคุมการกระทำต่าง ๆ บนเบราว์เซอร์ เช่น คลิกที่ลิงค์ กรอกข้อมูลในฟอร์ม กดปุ่ม เปลี่ยนหน้าเว็บ เป็นต้น

- ChromeDriver ช่วยในการตรวจสอบสถานะและข้อมูลในเว็บได้อย่างอัตโนมัติ เช่น การตรวจสอบข้อความ การตรวจสอบค่าในฟิลด์ การตรวจสอบข้อมูลในตาราง ฯลฯ

- เมื่อทำการทดสอบหรือออโตเมชันเสร็จสิ้น สามารถใช้ ChromeDriver เพื่อปิดหน้าต่างเบราว์เซอร์ได้

ในส่วนของการติดตั้ง ChromeDriver เป็นขั้นตอนสำคัญเมื่อต้องการใช้ Selenium WebDriver เพื่อควบคุมเบราว์เซอร์ Chrome จากภาษาโปรแกรมที่ต้องการ เช่น Python, Java, หรือภาษาอื่น ๆ

**ดาวน์โหลด ChromeDriver** : ไปที่เว็บไซต์ดาวน์โหลดของ ChromeDriver ที่ https://sites.google.com/chromium.org/driver/ และเลือกเวอร์ชันของ ChromeDriver ที่ตรงกับเวอร์ชันของ Google Chrome ที่คุณมีในเครื่อง

A screenshot of a computer

Description automatically generated

**รูปที่ 3.13 หน้าเว็บของ ChromeDriver**

**A screenshot of a computer

Description automatically generated**

**รูปที่ 3.14 หน้า Downloads**

- สามารถตรวจสอบเวอร์ชันของ Chrome ที่มีในเครื่องโดยการ เข้าไปที่ “ตั้งค่า (Settings)” ของ Chrome -> เกี่ยวกับ Chrome (About Chrome)

**A screenshot of a computer

Description automatically generated**

**รูปที่ 3.15 “ตั้งค่า (Settings)” ของ Chrome**

**A screenshot of a computer

Description automatically generated**

**รูปที่ 3.16 ตรวจสอบเวอร์ชันของ Chrome ที่มีในเครื่อง**

- ในกรณีที่เวอร์ชันของ Chrome ที่มีในเครื่อง มีเวอร์ชันที่สูงกว่า หรือใหม่กว่าในหน้าเว็บ “Downloads” ให้เข้าไปที่ลิงค์ <https://googlechromelabs.github.io/chrome-for-testing/> และเลือกเวอร์ชันที่ตรงกับ Chrome ที่มีในเครื่อง

**A screenshot of a computer

Description automatically generated**

**รูปที่ 3.17 หน้าเว็บในการดาวน์โหลด ChromeDriver เวอร์ชันล่าสุด**

**เพิ่ม ChromeDriver เข้า PATH (ถ้าต้องการ)** :เพื่อให้สามารถเรียกใช้ ChromeDriver จากทุกที่ในระบบ อาจจะต้องทำการเพิ่มเส้นทางที่แสดงถึงไฟล์ ChromeDriver เข้าไปใน Environment Variables (PATH) ของระบบปฏิบัติการ

A screenshot of a computer

Description automatically generated

**รูปที่ 3.18 เพิ่ม path ที่ชื่อว่า “Path” ในส่วนของ “System variables”**



**รูปที่ 3.19 Path ที่เรียกใช้ ChromeDriver (ในที่นี้ขึ้นอยู่กับผู้ใช้ในการกำหนด Path ที่ต้องการ)**

**3.2.1 Microsoft Excel**

Microsoft Excel พัฒนาโดยบริษัท Microsoft มีการใช้งานหลากหลายในการบันทึกข้อมูลการทดสอบทั้งในลักษณะการทดสอบด้วยมือและระบบอัตโนมัติ รวมถึงการบันทึกผลลัพธ์ของกระบวนการทดสอบ การจัดการและวิเคราะห์ข้อมูลที่เกี่ยวข้องกับการทดสอบและตรวจสอบคุณภาพของซอฟต์แวร์ ได้แก่

**- การวางแผนและการจัดการกระบวนการทดสอบ** Excel สามารถใช้ในการสร้างแผนการทดสอบ กำหนดลำดับการทดสอบ รวมไปถึงการบันทึกและติดตามความคืบหน้าของกระบวนการทดสอบ

**- การจัดการข้อมูลทดสอบ** Excel ช่วยในการจัดเก็บและจัดการข้อมูลทดสอบ เช่น การสร้างชุดข้อมูลทดสอบ การบันทึกผลการทดสอบ และการจัดรูปแบบข้อมูลเพื่อให้ง่ายต่อการวิเคราะห์

**- การสร้างสคริปต์การทดสอบ (Test Scripts)** Excel ช่วยในการสร้างสคริปต์การทดสอบเบื้องต้น เช่น การกำหนดข้อมูลเข้าสู่ฟอร์ม และการคาดเดาผลลัพธ์ที่ควรได้

**- การจัดทำรายงานการทดสอบ** Excel ช่วยในการจัดทำรายงานผลการทดสอบ เพื่อแสดงข้อมูลและผลลัพธ์ของการทดสอบให้กับทีมพัฒนาหรือผู้เกี่ยวข้องอื่น ๆ

**- การจัดการข้อมูลการทดสอบอัตโนมัติ (Automated Testing)** เมื่อมีการทดสอบอัตโนมัติ ผลลัพธ์จะถูกบันทึกในรูปแบบของข้อมูล ซึ่งอาจใช้ Excel เพื่อจัดเก็บและวิเคราะห์ผลลัพธ์การทดสอบอัตโนมัติ

**- การจัดการกับข้อมูลคุณภาพ** Excel ใช้เพื่อจัดเก็บข้อมูลเกี่ยวกับคุณภาพของซอฟต์แวร์ เช่น รายชื่อบั๊ก รายงานความสำเร็จของการทดสอบ เป็นต้น

โดยมีการนำตัวอย่างเทสเคสที่มีในการใช้ในการทดสอบมาเป็นตัวอย่าง ในโปรแกรมไมโครซอฟต์เอกเซล (Microsoft Excel) ในรูปแบบของตาราง เพื่อให้ง่ายต่อการบันทึกและติดตามความคืบหน้าของกระบวนการทดสอบ

A screenshot of a computer

Description automatically generated

**รูปที่ 3.20 หน้าโปรแกรม Microsoft Excel**

ทั้งนี้ Test Case ไม่ได้มีรูปแบบที่แน่นอน เนื่องจากสามารถเขียนและออกแบบได้ในหลายลักษณะขึ้นอยู่กับประเภทของซอฟต์แวร์และวัตถุประสงค์ของการทดสอบ โดยทั่วไปแล้ว Test Case จะประกอบด้วยข้อมูลต่าง ๆ ที่เกี่ยวข้องกับการทดสอบ และขั้นตอนการทดสอบที่อธิบายวิธีการดำเนินการในกรณีที่แน่นอน ซึ่งช่วยให้ผู้ทดสอบสามารถทดสอบระบบหรือฟีเจอร์ต่างๆ ได้อย่างมีระเบียบและเป็นระบบ