Robot Framework

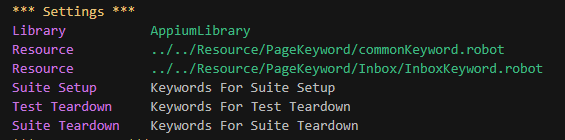
Introduction to Robot Framework

Robot Framework คืออะไร

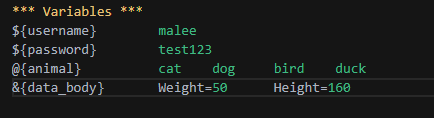
[Robot Framework](https://robotframework.org/) คือ Generic test automation framework ออกแบบมาเพื่อ Acceptance testing และ Aceptance test-driven development (ATDD) โดย Test script มีลักษณะที่เข้าใจง่าย อ่านแล้วเข้าใจว่าต้องการจะทำอะไร เป็น Tools อย่างหนึ่งที่เอาไว้ทำ automation testing

Robot Framework Structure มีอะไรบ้าง

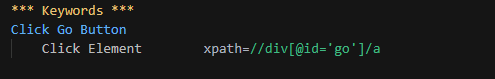
\*\*\* Settings \*\*\*

เป็นส่วนที่ใช้ในการ เรียก file Resource, import library , ใส่รายละเอียด document description, การทำ Setup/Teardown

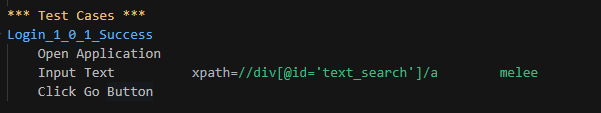
\*\*\* Variables \*\*\*

ใช้ในการประกาศตัวแปรต่าง ๆ

\*\*\* Keywords \*\*\*

ใช้ในการรวมคำสั่งที่ทำ action ต่างๆ และสามารถตั้งชื่อให้ผู้อื่นเข้าใจได้ทันทีว่า action นี้ทำอะไร คล้ายกับ Function ในภาษาอื่น ๆ 

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

ใช้ในการเขียน Test Script เป็นขั้นตอนของ test case นั้น ๆ ประกอบไปด้วย 2 ส่วน

1.ชื่อ test cases

Login\_1\_0\_1\_Success

2.คำสั่ง (Keyword)

Open Application

Input Text

Click Go Button

Robot Framework Run อย่างไร

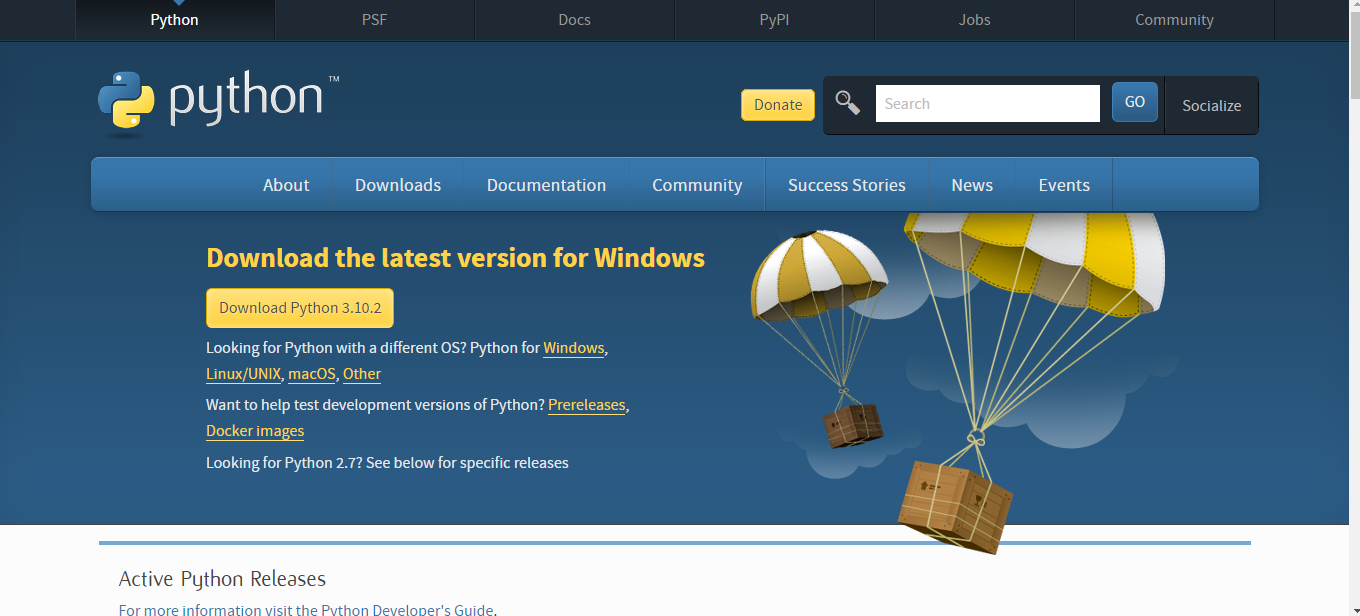
ในการสั่ง Run Test ใน Robot Framework นั้น สามารถทำได้ผ่านทั้งทาง TREMINAL และ Command โดยต้องทำการ (เข้าให้ถึงไฟล์ที่เราเก็บไฟล์โรบอทไปไว้อยู่ cd เข้าไป ไม่รู้จะเขียนว่ายังไง) แล้วพิมพ์คำสั่ง robot (ชื่อไฟล์).robot แล้วกด Enter 

เมื่อรันเสร็จ จะแสดงหน้าดังภาพ เราจะได้ไฟล์สรุปผล 3 ไฟล์ (output.xml, log.html report.html) โดยจะบอกรายละเอียดที่เกี่ยวข้องกับการทดสอบของเรา ผ่านหรือไม่ ใช้เวลาเท่าไหร่ในการทดสอบ เป็นต้น

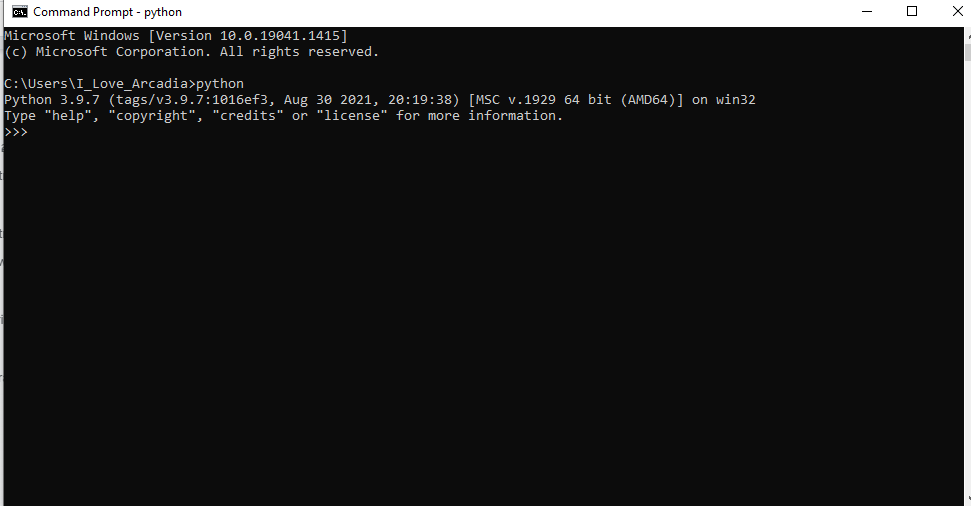
Install Robot Framework

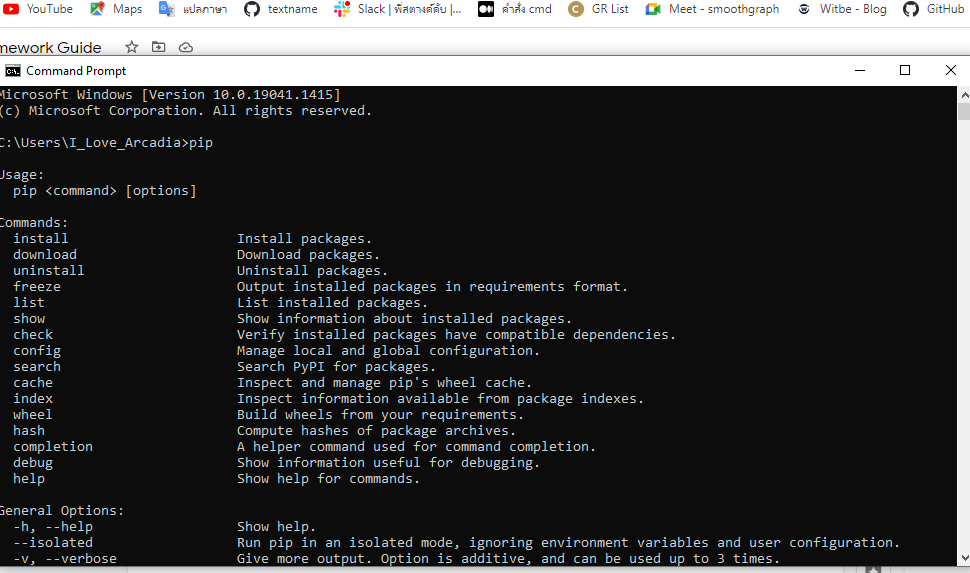
Pre-requisite Download

**Python 3.8 ขึ้นไป**

* Download from: <https://www.python.org/downloads/>
* คลิกเลือก Add Python 3.10 to PATH แล้วคลิก Install Now



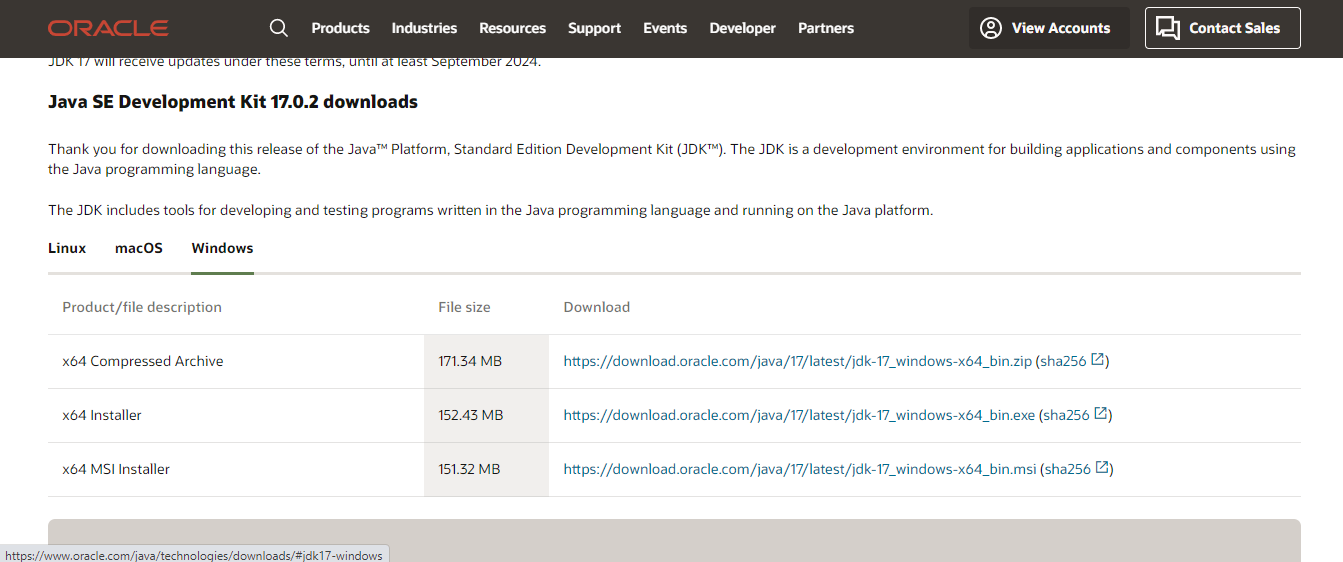
Check point: พิมพ์คำสั่ง python และ pip ใน cmd/terminal จะได้ผลลัพท์ดังภาพ ถือว่าติดตั้งสำเร็จ

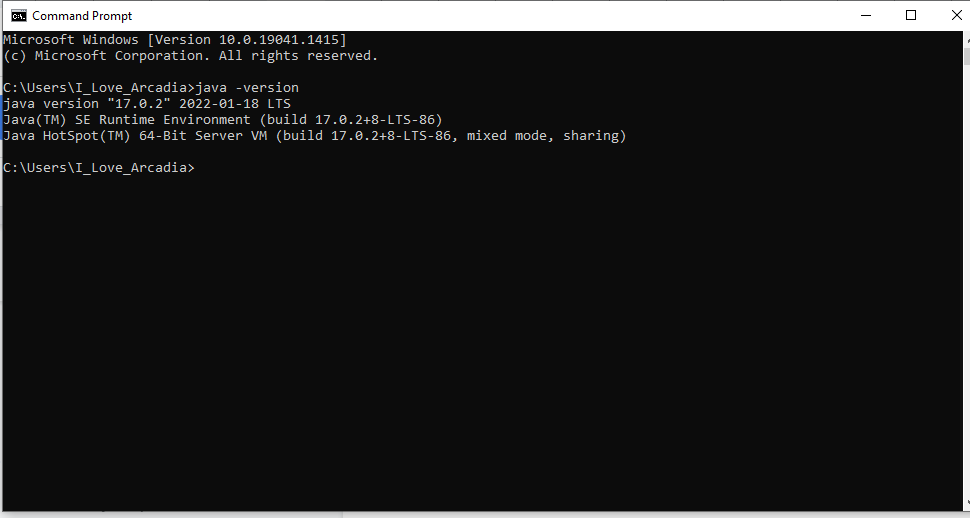


## 

## **Java JDK**

* Download from: <https://www.oracle.com/java/technologies/downloads/>



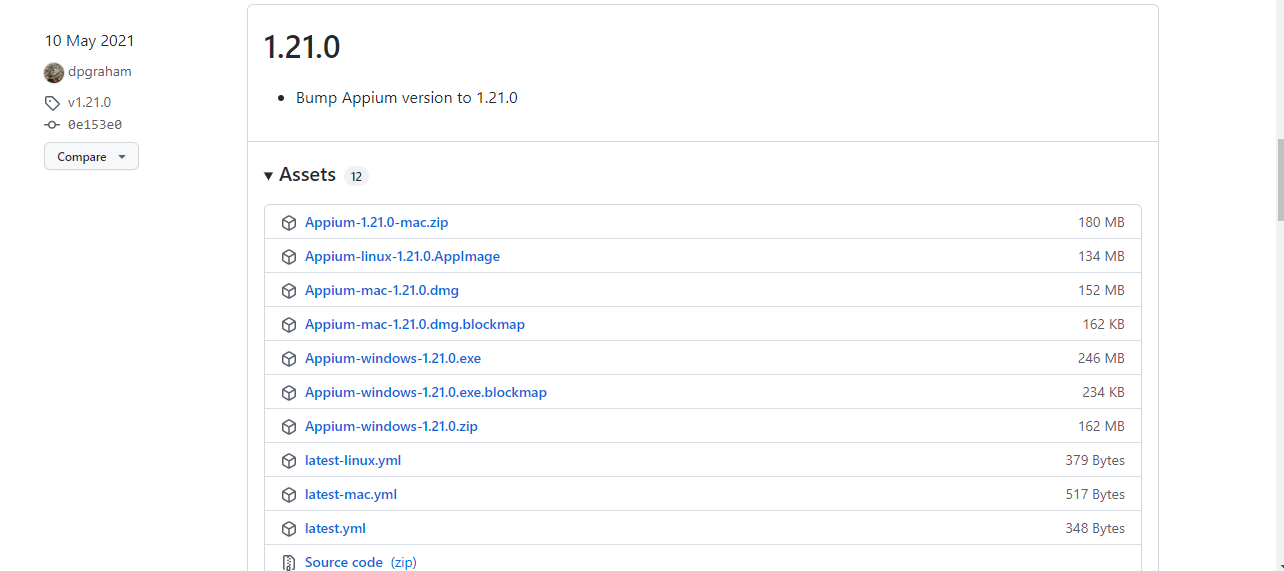
Check point: พิมพ์คำสั่ง java -version ใน cmd/terminal จะได้ผลลัพท์ดังภาพ ถือว่าติดตั้งสำเร็จ

## **Android Studio**

* Download from: <https://developer.android.com/studio>
* คลิกเลือก I have read and agree with the above terms and conditions

## **Appium Desktop App**

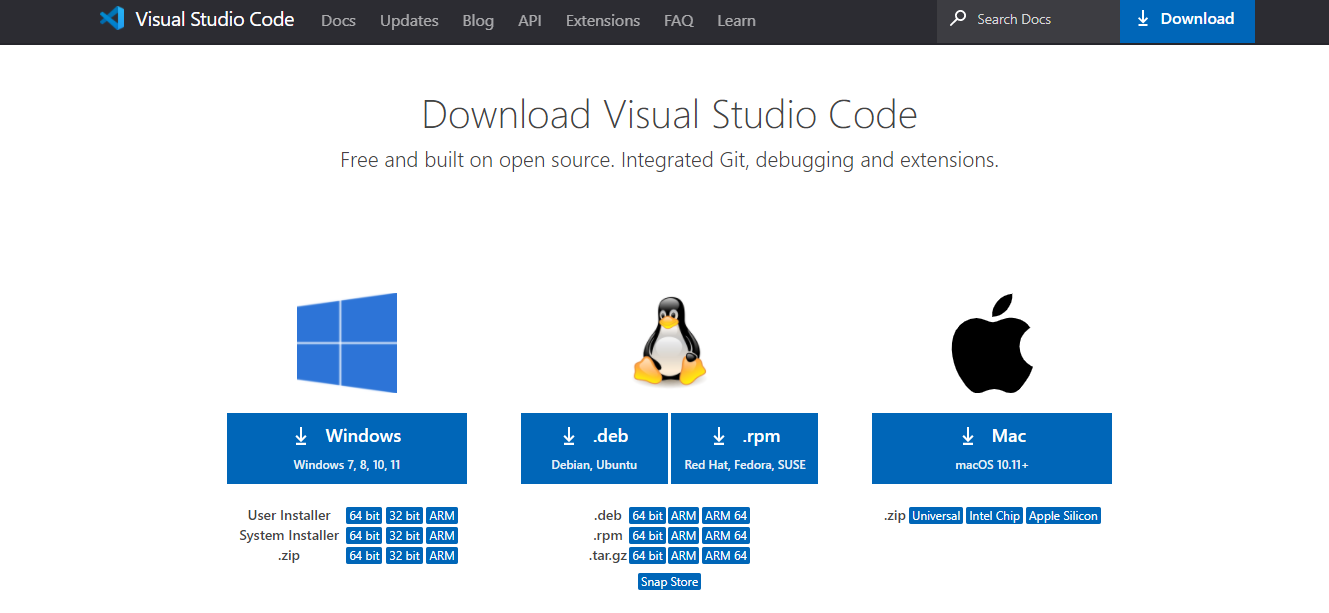
* Download Version 1.21.0 from: <https://github.com/appium/appium-desktop/releases>



## **ImageMagick**

* Download from : https://imagemagick.org/script/download.php

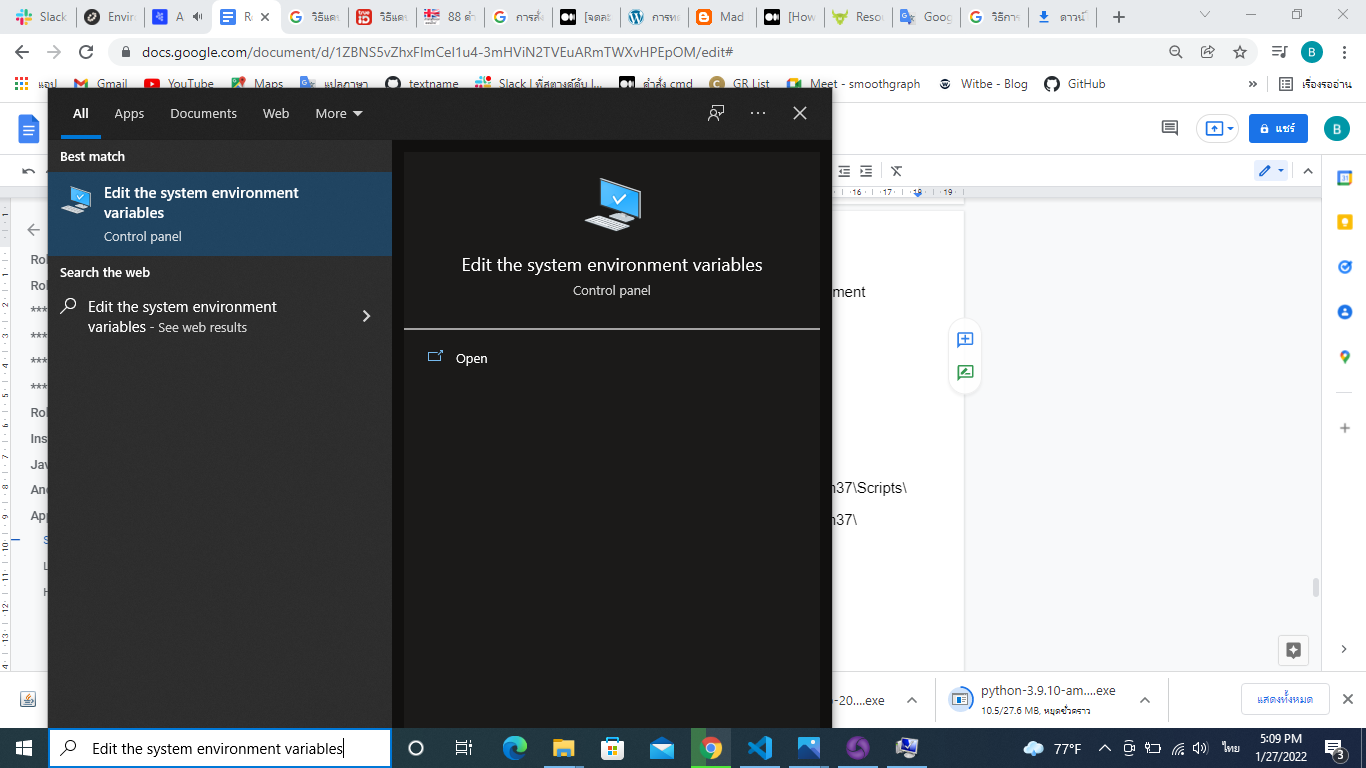
**Visual Studio Code**

* Download Version 1.21.0 from: <https://code.visualstudio.com/download>

Set Environment variables path

หลังจากที่เราได้ติดตั้งโปรแกรม ทั้งหมดแล้ว เราจะมา set environment variables path โดยมีขั้นตอนดังนี้

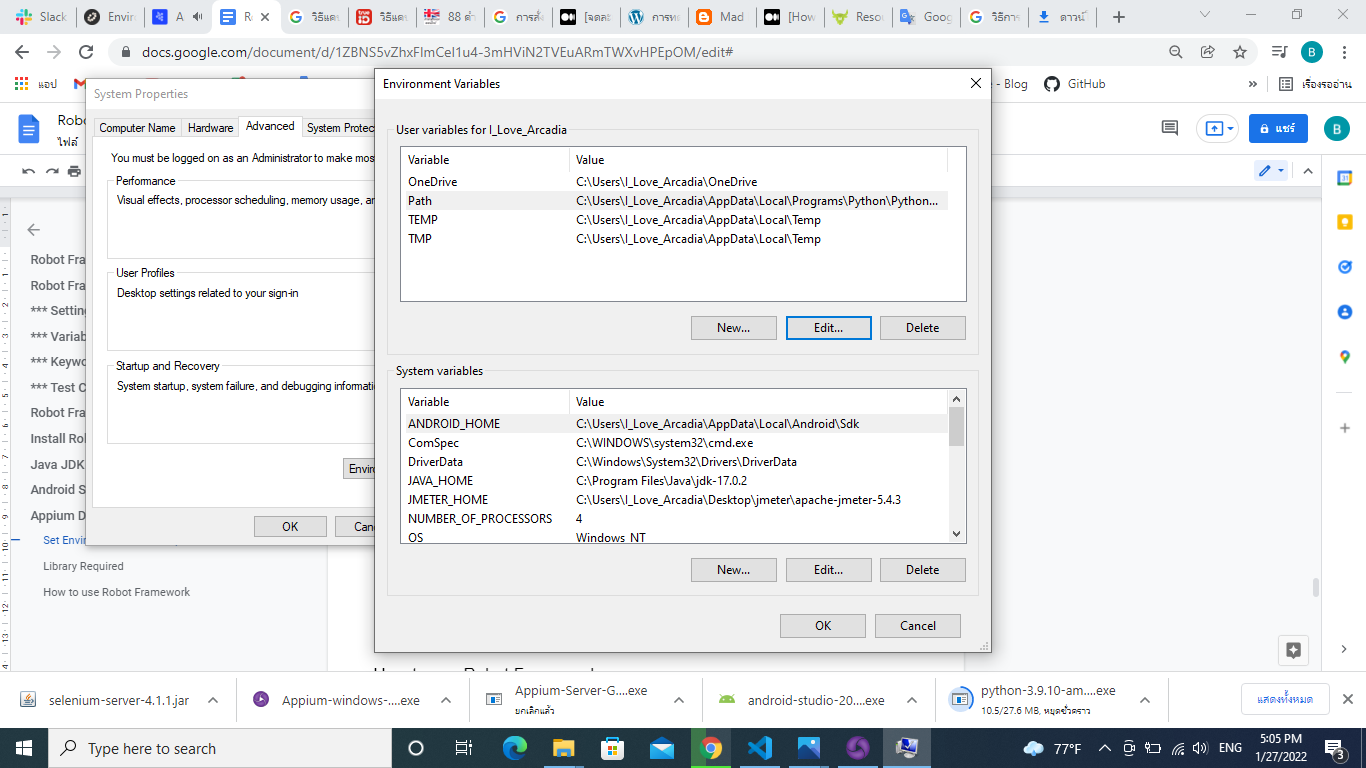
1. Search Edit the system environment variables ที่ tap bar



1. คลิก Advanced
2. คลิก Environment variables



1. คลิก Path ที่ส่วนของ User variables และ คลิก Edit



1. คลิก New และใส่ path ดังนี้

* **Python**

C:\Users\MyUser\AppData\Local\Programs\Python\Python37\Scripts\

C:\Users\MyUser\AppData\Local\Programs\Python\Python37\

\*\* ”Python37” คือเวอร์ชั่นที่เราได้ดาวโหลดมาเปลี่ยนให้ตรงกับไฟล์ที่ติดตั้ง

* **Android SDK**

%**ANDROID\_HOME%**\platform-tools

* **Java JDK**

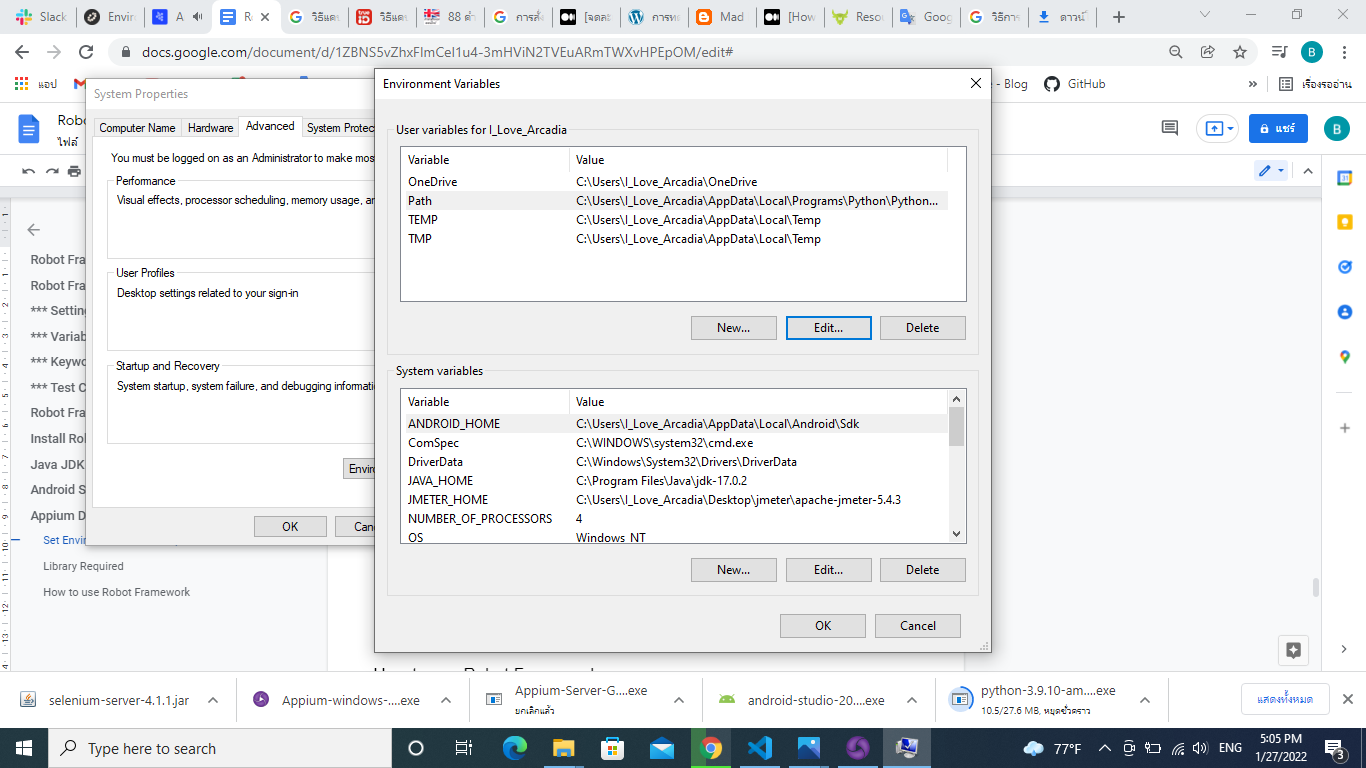
**%JAVA\_HOME%\bin**

1. คลิก Path ที่ส่วนของ system variables และ คลิก Edit

C:\Program Files\Java\jdk1.8.0\_91\bin

\*\* jdk จะแสดงค่า version ตามที่ติดตั้ง

1. คลิก New ที่ System variables



1. ใส่ Variable Name และ Value ดังนี้

* **ANDROID\_HOME**

Variable Name: ANDROID\_HOME

Value: C:\Users\ MyUser\AppData\Local\Android\sdk

* **JAVA\_HOME**

Variable Name: JAVA\_HOME

Value: C:\Program Files\Java\jdk1.8.0\_91

**Library Required**

พิมพ์คำสั่ง pip ดังต่อไปนี้ ใน Command

Install Robot Framework version 5.0 ขึ้นไป

* pip install robotframework

Install Browser & Appium library

* pip install robotframework-appiumlibrary (Mobile Testing)
* pip install robotframework-browser (Web Testing)
  + rfbrowser init

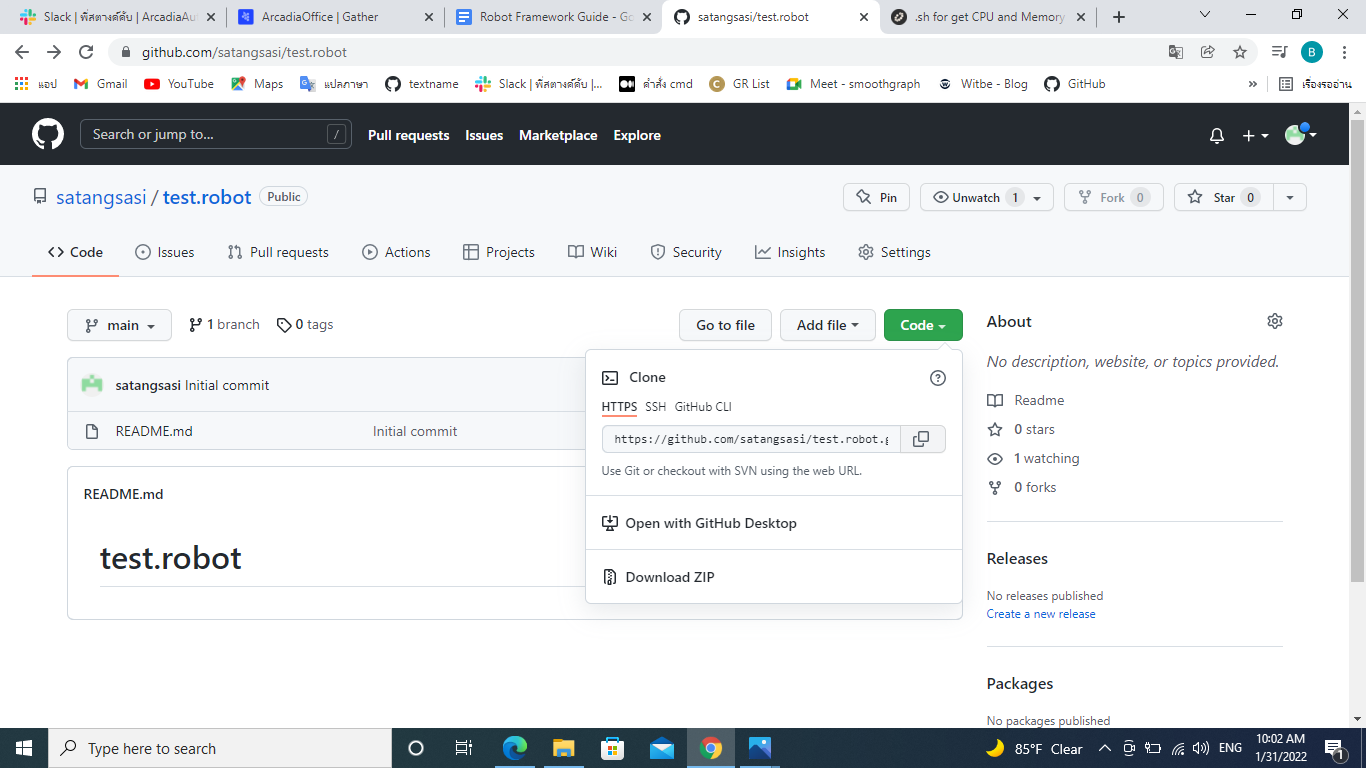
Install Requests library API Testing

* pip install robotframework-requests
* pip install robotframework-jsonlibrary

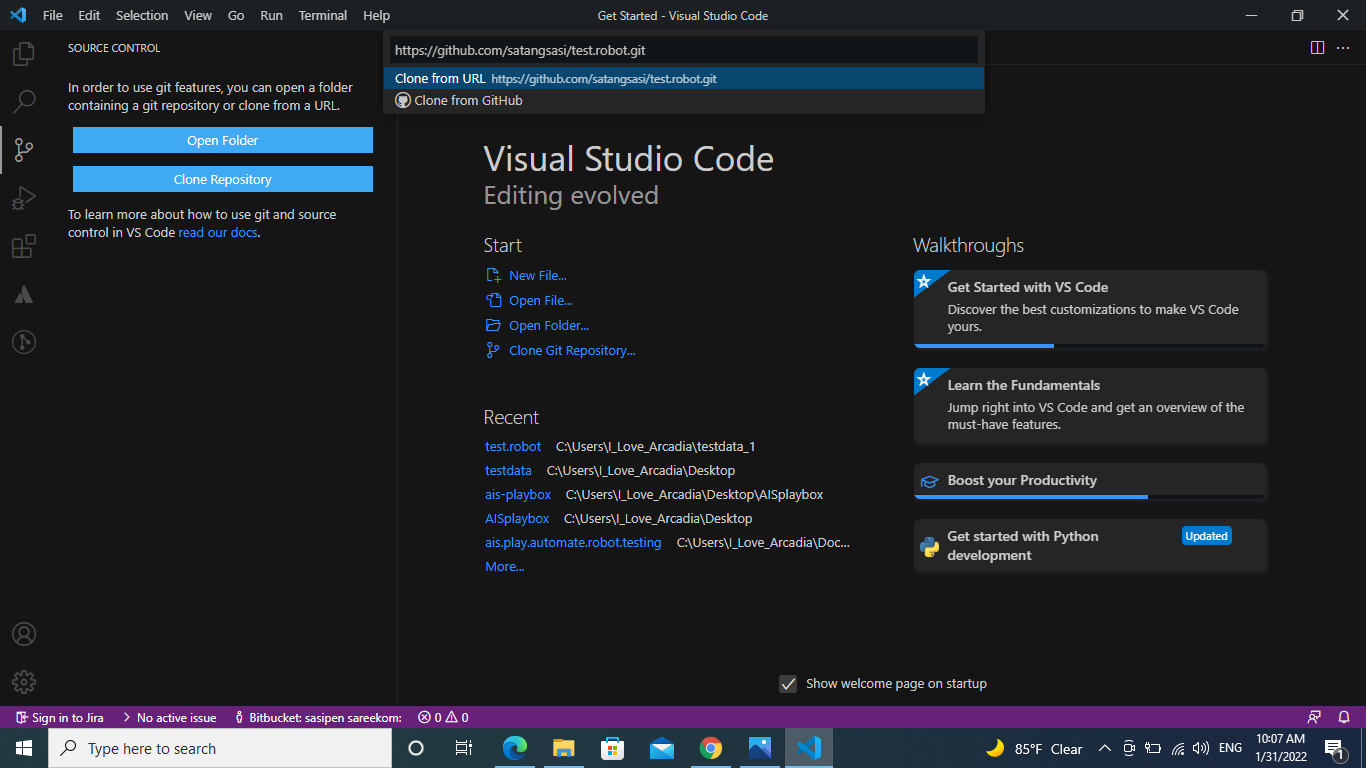
git clone/push/pull

เมื่อเราสร้าง repositories ไว้แล้วเราจะมาทำการ clone ไฟล์ไปไว้ที่เครื่องของเรากัน เพื่อที่เวลาเราทำงานเราจะได้ update งานต่างๆขึ้นมาเก็บไว้ ถ้าทำงานร่วมกับผู้อื่นก็จะสามารถ update การเปลี่ยนแปลงและทำงานร่วมกันได้ โดยมีขั้นตอนดังนี้

1. คลิกที่ code และ copy https.git ไว้

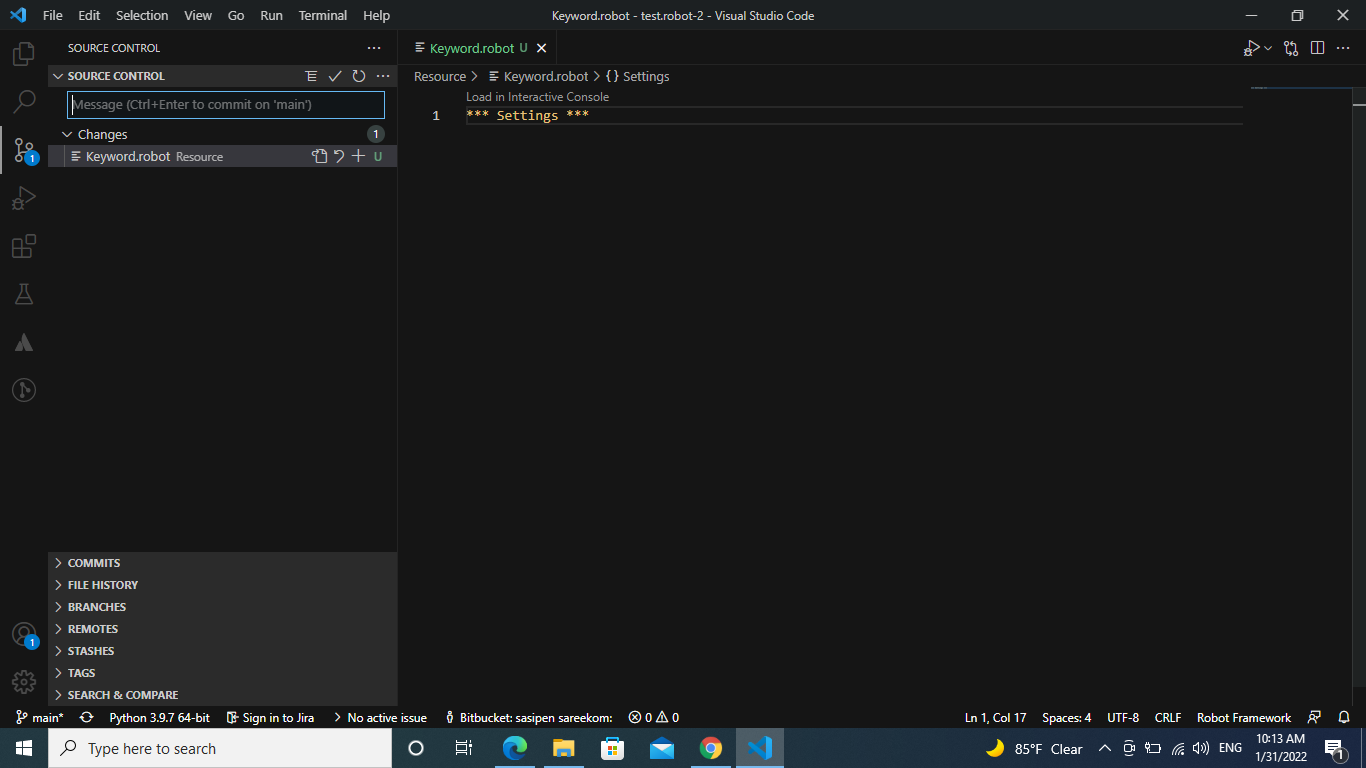


1. เปิด VS code คลิกที่ Source Control และเลือก Clone Repository และวางตัว https.git ที่เราcopy มา แล้วกด Enter

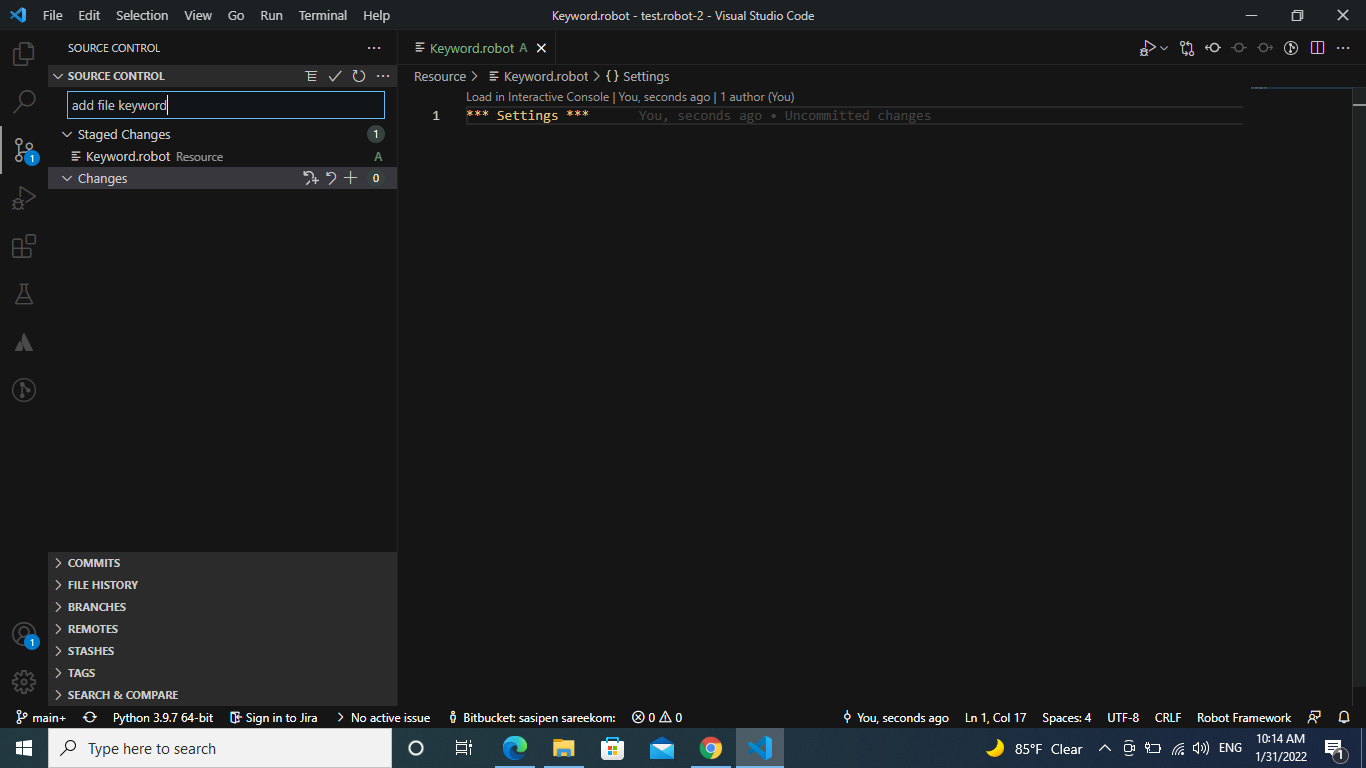


เราก็จะได้ File ที่เราสร้างไว้และสามารถอัพเดทงานขึ้นไปเก็บไว้ได้ โดยเมื่อเรามีการอัพเดทงาน หรือแก้ไขงานก็สามารถอัพเดทความเปลี่ยนแปลงได้ดังนี้

1. คลิกที่ Source Control จะเห็นความเปลี่ยนแปลงต่างๆอยู่ที่ Change

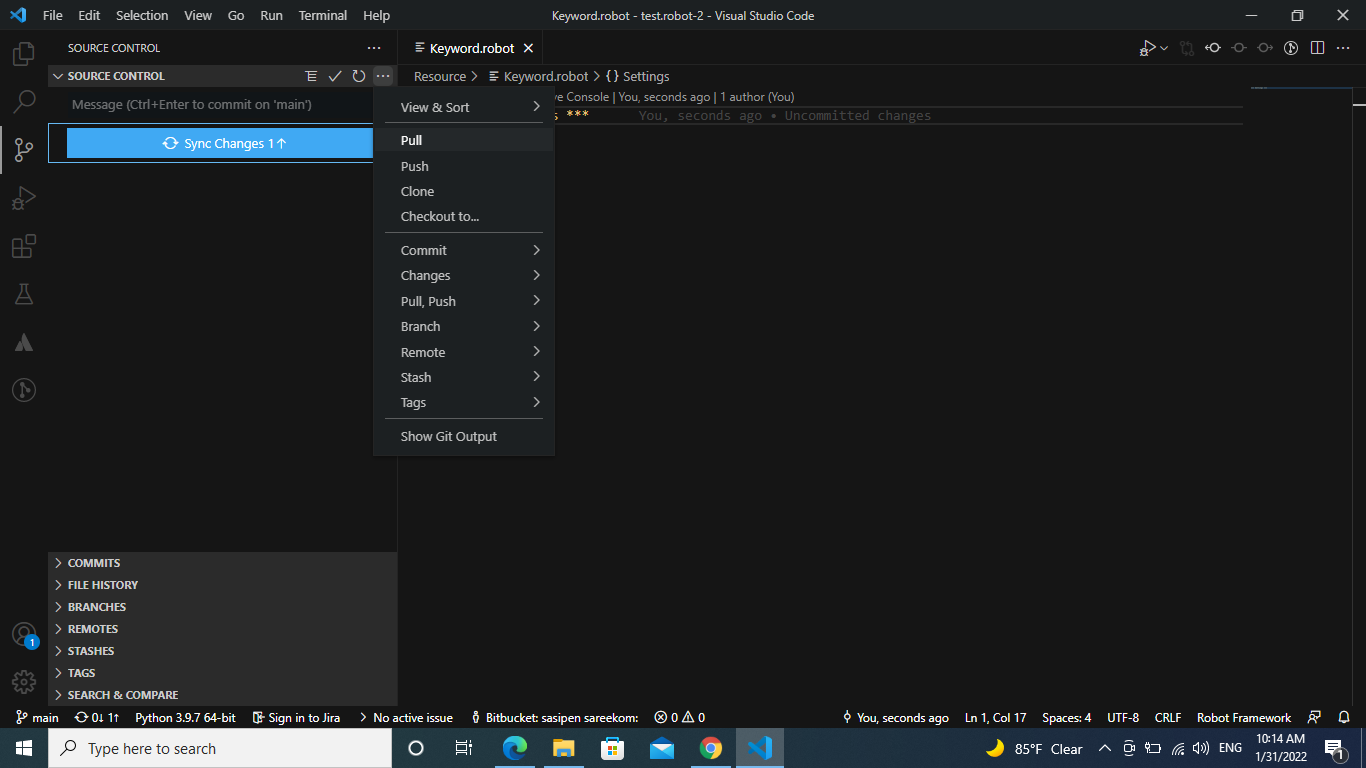


1. คลิกเลือกที่เครื่องหมาย + เพื่อเลือกไฟล์ไปที่ Staged Change และ comment เพื่ออัพเดทว่าเราได้ทำการเปลี่ยนแปลงหรือแก้ไขอะไรบ้าง



1. คลิกที่เครื่องหมาย ถูกเพื่อเป็นการคอนเฟิร์ม และ คลิกที่ … จะมีหัวข้อที่เราต้องใช้อยู่ 2 ข้อ คือการ push กับ pull

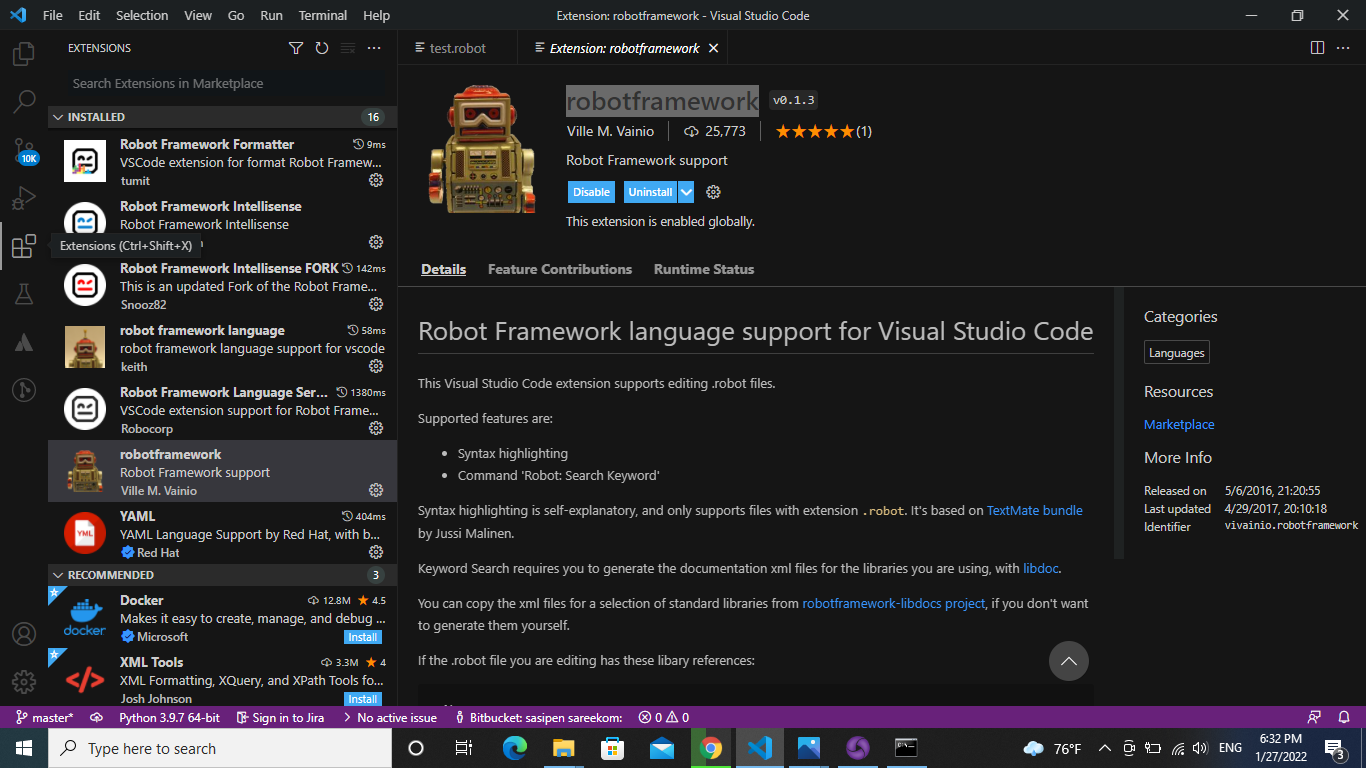
* push ใช้เมื่อเราต้องการอัพเดทไฟล์ที่เราเลือกขึ้นไปเก็บไว้
* pull ใช้เมื่อเราต้องการอัพเดทว่ามีไฟล์ใดเปลี่ยนแปลงไปบ้าง อัพเดทงานให้เป็นปัจจุบันก่อน (จะเป็นประโยชน์มากเมื่อเราทำงานร่วมกับผู้อื่นในไฟล์เดียวกัน เราเมื่อ pull เราจะเห็นว่าผู้อื่นเพิ่มหรือลบอะไร ตรงไหนเข้ามาในงานบ้าง จะได้ทำงานได้ไม่ทำซ้ำกัน หรือใช้ในสิ่งที่ลบไปแล้ว)



How To Use Robot Framework

* Install EXTENSION

ก่อนที่เราจะเริ่มเขียน test script บน VS code เรามา download extension ไว้ใช้งานเพื่อให้เรา เขียน test script ได้สะดวกและรวดเร็วยิ่งขึ้นกันก่อน ที่แนะนำให้ download มีดังนี้

* Python
* robotframework
* robot framework language
* Robot Framework Intellisense
* Robot Framework Language Server
* Robot Framework Intellisense FORK
* Relative Path
* Robot Framework Formatter
* Create File .robot

ต่อมาเราจะไปสร้าง File.robot กัน โดยที่ไป File > New File > Save as > ตั้งชื่อไฟล์และตามด้วย .robot เช่น test.robot

เมื่อเรามีไฟล์ .robot แล้วเราก็จะสามารถเขียน test script ตามโครงสร้าง Structure ของ robot framework ได้แล้ว

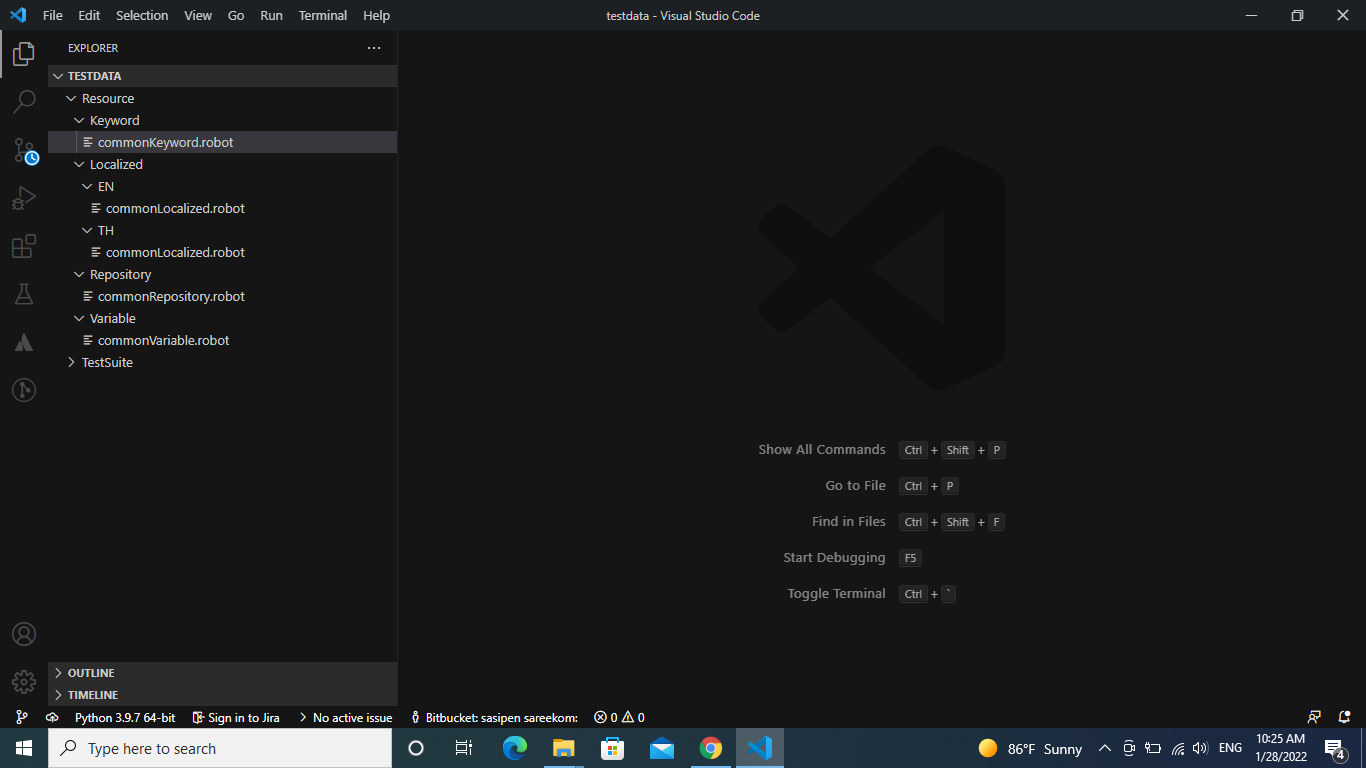
โครงสร้าง Robot Framework

โดยเราจะสร้าง Folder สำหรับงานเทสขึ้นมา และจะมี Folder ด้านในอยู่ 2 ส่วนคือ

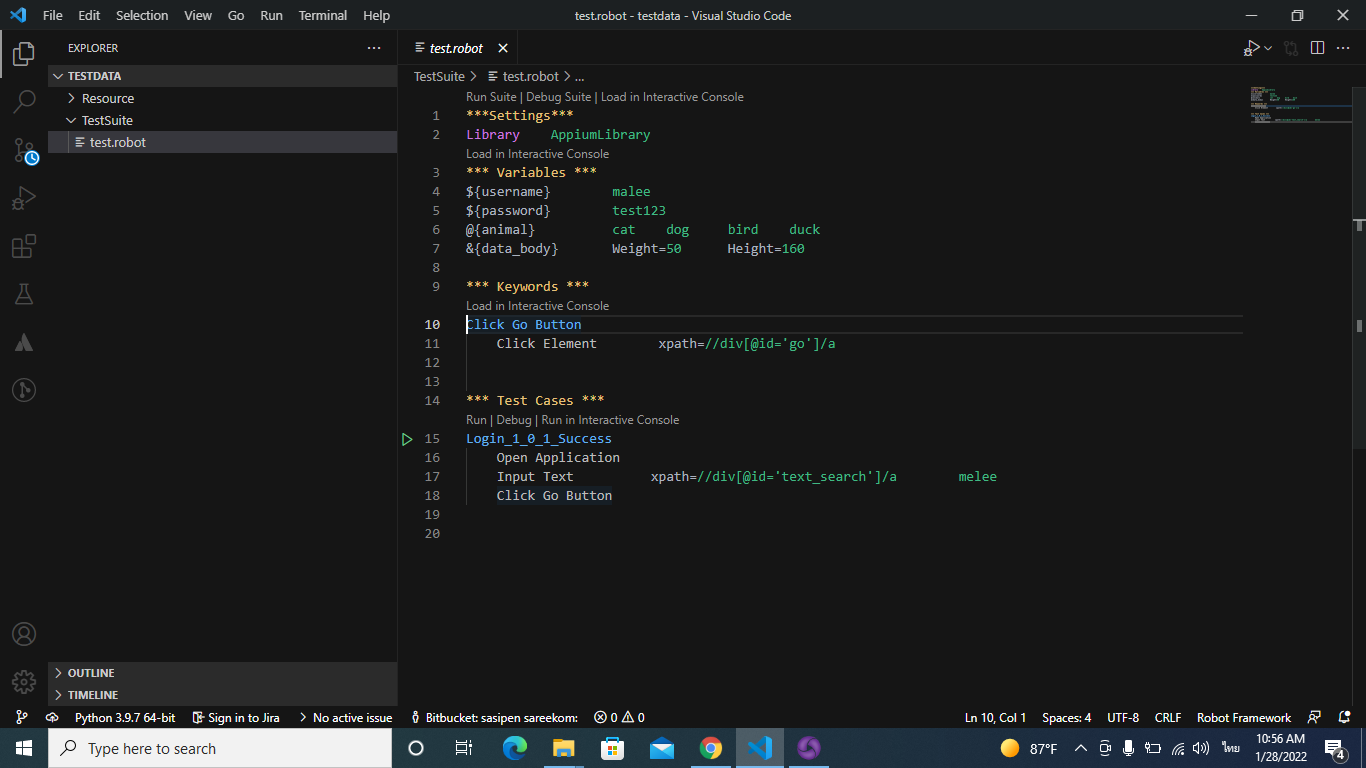
1. Resource เป็นเหมือนแหล่งข้อมูลที่เราจะเอามาใช้ในการเขียน test script ขึ้น จะแบ่งเป็น Folder ย่อยๆอีก 4 อัน คือ

* Localized > ใช้การเก็บตัวแปรคำศัพท์ ที่มีทั้งภาษาไทยและอังกฤษ
* Keyword > ใช้ในการเก็บ คำสั่งต่างๆ
* Repository > ใช้ในการเก็บ locater (xpath,id,accessibility id,etc.)
* Variable > ใช้ในการประกาศตัวแปร

โดยทั้ง 4 Folder นั้นจะมีไฟล์.robot อยู่ด้านในแยกตาม feature ที่เราจะเทส และมีไฟล์ common\_\_\_\_.robot เป็นไฟล์หลักในการเก็บค่าต่างๆที่สามารถใช้ร่วมกันได้ในหลายๆ feature



1. TestSuite เป็น Folder ที่เราจะเก็บไฟล์ test script ของ feature ต่างๆที่เราจะทำการ test



ตัวอย่างการเขียนตัวแปร เขียนคำสั่ง

**ตัวแปร variables**

robot framework จะมีตัวแปรอยู่ 3 ประเภทด้วยกัน คือ

1. ${**Scalar**} ใช้ในการเก็บค่าตัวแปรที่มีค่าเดียว เช่น

${username} malee

${password} test123

1. @{**List**} ใช้ในการเก็บค่าตัวแปรที่มีหลายๆค่า เช่น

@{animal} cat dog bird duck

@{colour} green red orange

1. &{**Dictionary**} ใช้ในการเก็บค่าหลายๆค่า ซึ่งมีความคล้ายคลึงกับตัวแปรแบบ List แต่ต่างกันที่ สามารถใช้ key แทนการระบุ index ได้ เช่น

&{allanimal} key1=cat key2=dog key3=bird

ถ้าเรานำ ${allanimal}[key3] มาใช้เราก็จะได้ค่าเป็น bird

**คำสั่ง Keyword**

Robot framework จะใช้คำสั่งใน Library ต่างๆ เช่น AppiumLibrary SeleniumLibrary BuiltIn เป็นต้น โดยเราสามารถนำคำสั่งที่มีอยู่มาใช้ได้เลย หรือจะนำคำสั่งต่างๆมารวมกัน แล้วสร้างเป็นคำสั่งใหม่ของเราก็ได้ ตัวอย่างเช่น

คำสั่งใน AppiumLibrary

Wait Until Element Is Visible

Click Element

Capture Page Screenshot

คำสั่งที่เราสร้างขึ้นใหม่ โดยการเอาคำสั่งที่มีอยู่เดิมมารวมกันแล้วตั้งชื่อเป็นคำสั่งใหม่ขึ้นมา

Select Go Button

Wait Until Element Is Visible

Click Element

Capture Page Screenshot

Run Test Script

ในการ run test script นั้น จากข้างต้นสามารถ run ได้โดยการ พิมพ์คำสั่ง robot ชื่อไฟล์.robot ที่ TREMINAL หรือ Command ก็ได้จะเป็นการ run test case ทั้งหมดที่มีในไฟล์นั้น แต่ถ้าหากเราต้องการ run แค่ test case บางส่วนหรือจำกัดการรันต่างๆก็สามารถทำได้เหมือนกัน โดยการใส่ tag ต่างๆเพิ่มเติม เช่น

* -v ชื่อตัวแปร : ตัวแปรที่ต้องการ > (สามารถใส่ -v กี่ตัวก็ได้)

-v "deviceName:192.168.1.103:6555" -v ar\_OS:Android -v ar\_LANG:EN

* -t ชื่อเทสเคสที่ต้องการรัน > (สามารถใส่กี่ -t กี่ตัวก็ได้)

-t Login\_1\_1\_001\_Valid\_Username

* -o ชื่อไฟล์ ที่ต้องการให้เก็บผล output ไว้ >

-o 1\_1\_001\_Valid\_Username

ถ้าเราสั่ง run test case แยกโดยใช้ -t แล้วตามด้วยชื่อtest case ผล output จะเก็บไว้เพียง test case ที่เรา run ครั้งล่าสุดเท่านั้น แต่ถ้าเราใส่ -o ด้วยผลการ run แต่ละรอบจะถูกเก็บไว้แยกกัน

* -d path/ชื่อโฟล์เดอร์ ที่เราต้องการให้ผลการรันต่างๆไปเก็บไว้

-d Result ผลการรันเทสจะเก็บไว้ใน Folder : Result ที่เดียวกับที่เราเก็บ test script

-d C:\Users\MyUser\Desktop\Result ผลการรันเทสจะเก็บไว้ใน Folder : Result ในหน้า Desktop

* -i tag ที่เราได้ตั้งไว้ > ใช้ในการ run test caes ที่มี tag เดียวกัน

-i login

* -e tag ที่เราได้ตั้งไว้ > ใช้ในการข้าม test caes ที่มี tag ที่เราระบุไว้

-e On-Hold

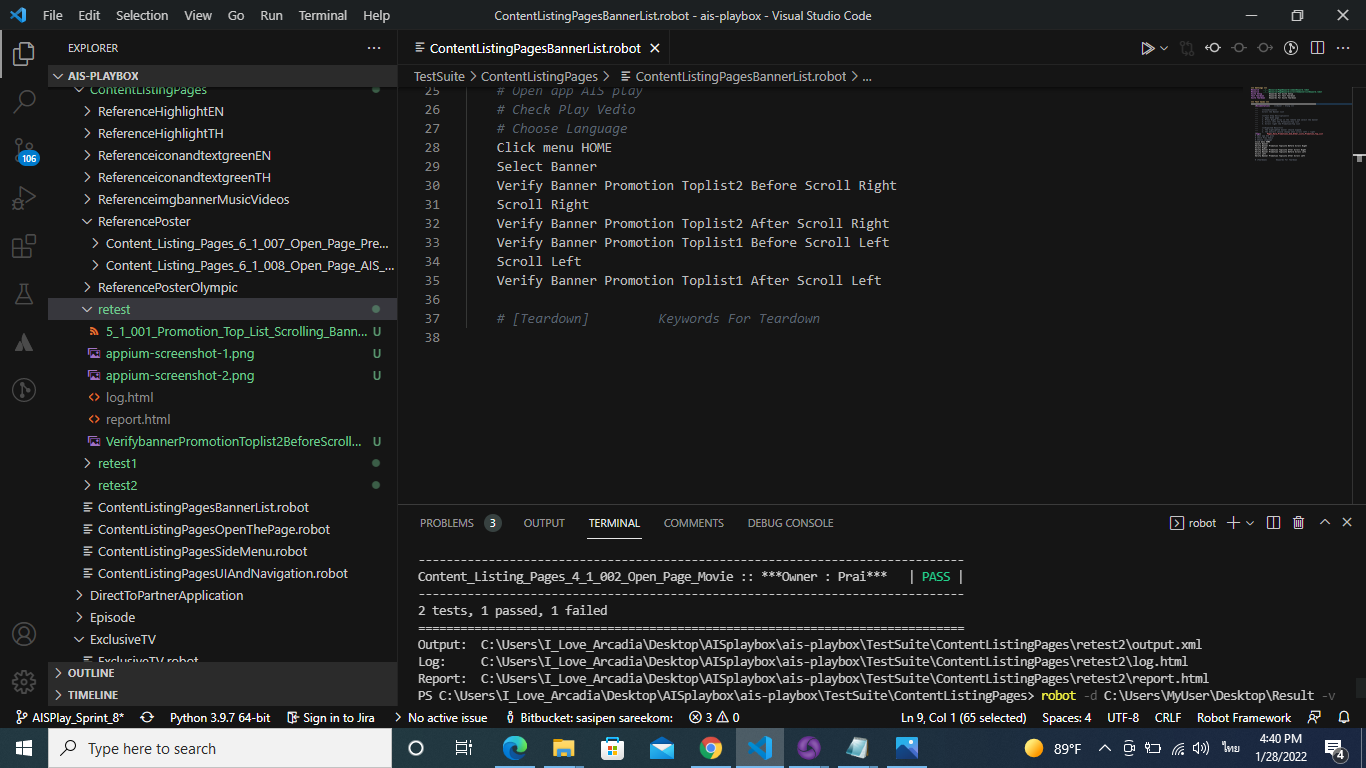
ตัวอย่างการสั่ง run test script

robot -d C:\Users\MyUser\Desktop\Result -v "deviceName:192.168.1.103:6555" -v images\_dir:ReferencePoster -t Login\_1\_1\_001\_Valid\_Username -o 1\_1\_001\_Valid\_Username -v ar\_OS:Android -v ar\_LANG:EN test.robot

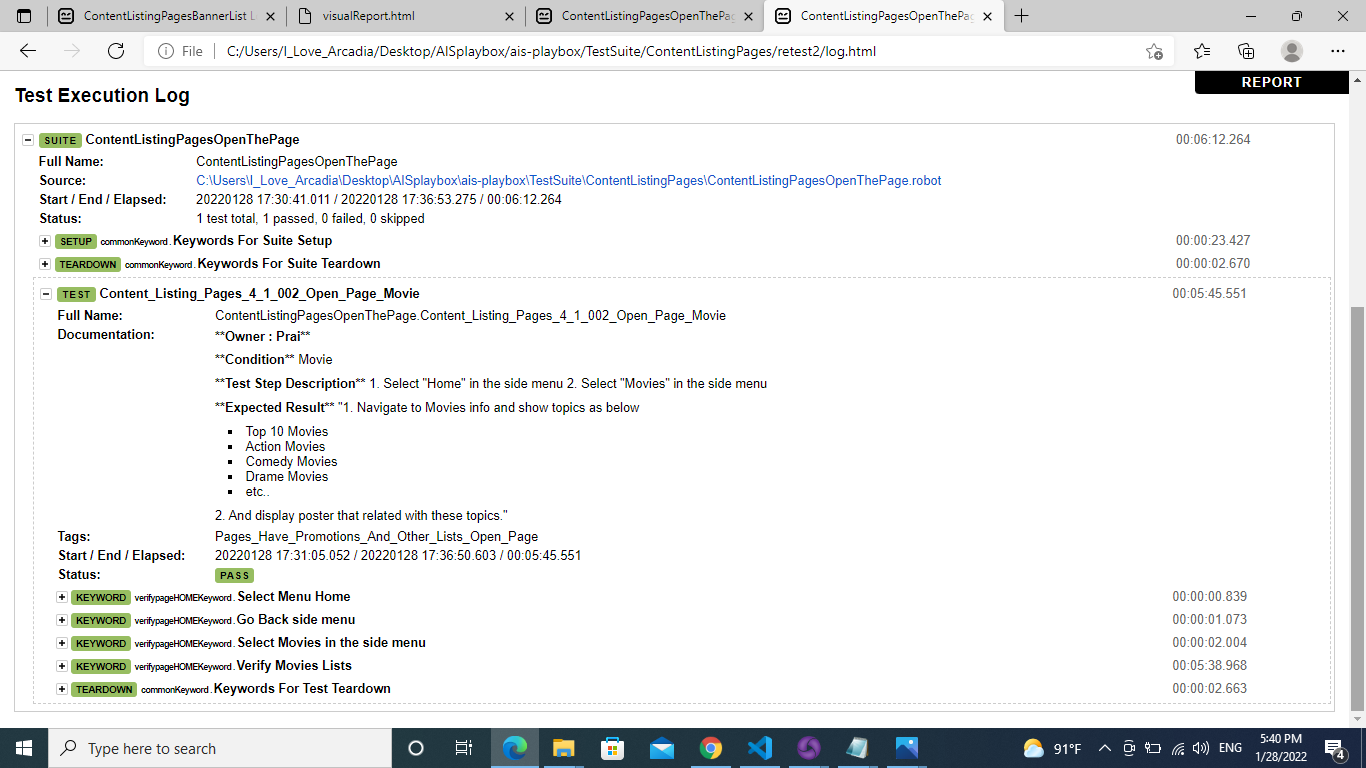
robot --nostatusrc -d "C:\\Andromeda\\admd.automate.robot.testing\\Result" -v "DOMAIN:${test\_domain}" -v "flag\_set\_proxy:${set\_proxy}" -v "REGRESSION:${regression}" -v "LANG:${language}" -e "On-Hold" C:\\Andromeda\\admd.automate.robot.testing\\Testsuite"""

การอ่าน Log report

ในการ run test case แต่ละครั้งนั้นจะได้ ไฟล์ Log ออกมา ซึ่งจะช่วยบอกเราได้ว่า test caes ของเราผ่านหรือไม่ ใช้เวลา run ทั้งหมดเท่าไหร่ แต่ละคำสั่งใช้เวลาเท่าไหร่ มีคำสั่งอะไรบ้าง ภาพ หรือ ข้อความที่ได้เก็บไว้เป็นอย่างไร



ตัวอย่าง Log



1

3

2

4

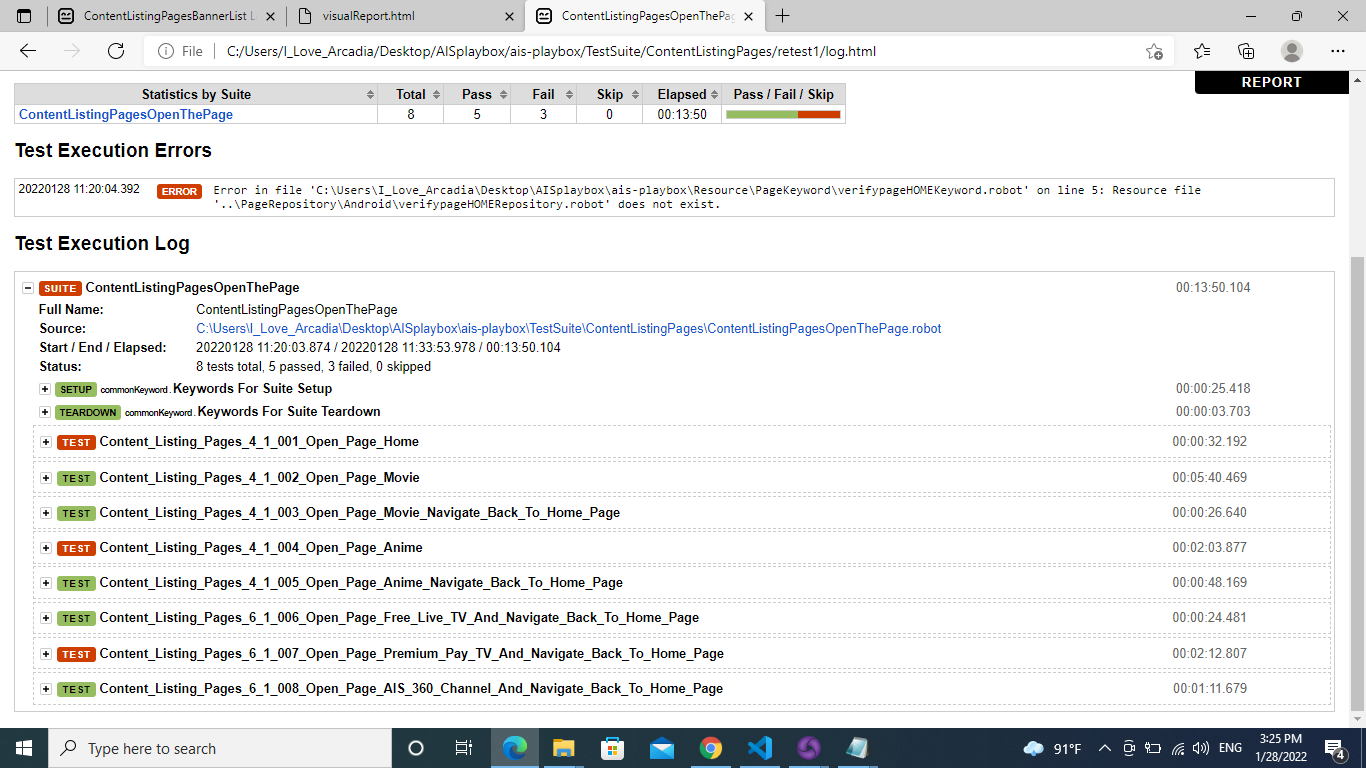
ส่วนประกอบของ Log

ส่วนที่ 1 จะแสดง ชื่อ test case และข้อมูลเกี่ยวกับ test caseข้อนั้น

ส่วนที่ 2 จะแสดง สถานะว่า test case ข้อนั้นผ่าน หรือ ไม่ผ่าน

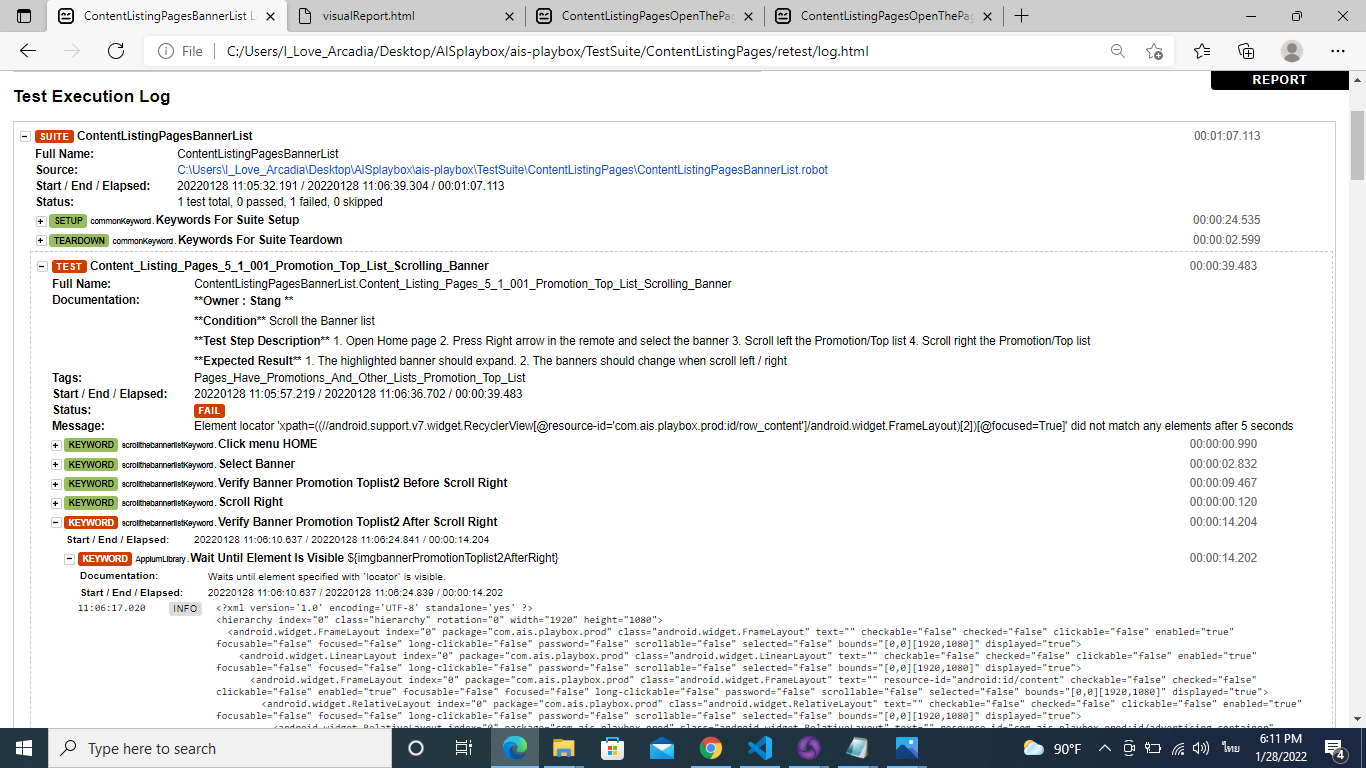
ส่วนที่ 3 จะแสดง คำสั่งต่างๆที่ใช้ใน test case นี้

ส่วนที่ 4 จะแสดง เวลาที่ใช้ในการ run test ย่อยไปถึงเวลาในการ run แต่ละคำสั่งด้วย

จากภาพจะเห็นได้ว่า test case ข้อที่ run ผ่านจะเป็น สีเขียว ข้อที่ไม่ผ่าน จะเป็นสีแดง

หลังจากที่เราเรียนรู้เกี่ยวกับส่วนประกอบที่จะแสดงอยู่ใน Log แล้วต่อมาเราจะมาดูกันว่า ถ้าเรา run test แล้วไม่ผ่าน จะดูได้อย่างไรว่า test script นั้นผิดที่ตรงไหน มีอะไรไม่ตรงตามที่เราคาดการณ์ไว้บ้าง

ตัวอย่างที่ 1



1

2

ในส่วนที่ 1 จะบอกเราเกี่ยวกับว่า test case นี้ run ไม่ผ่าน ข้อความที่แจ้งว่ามีลักษณะดังนี้

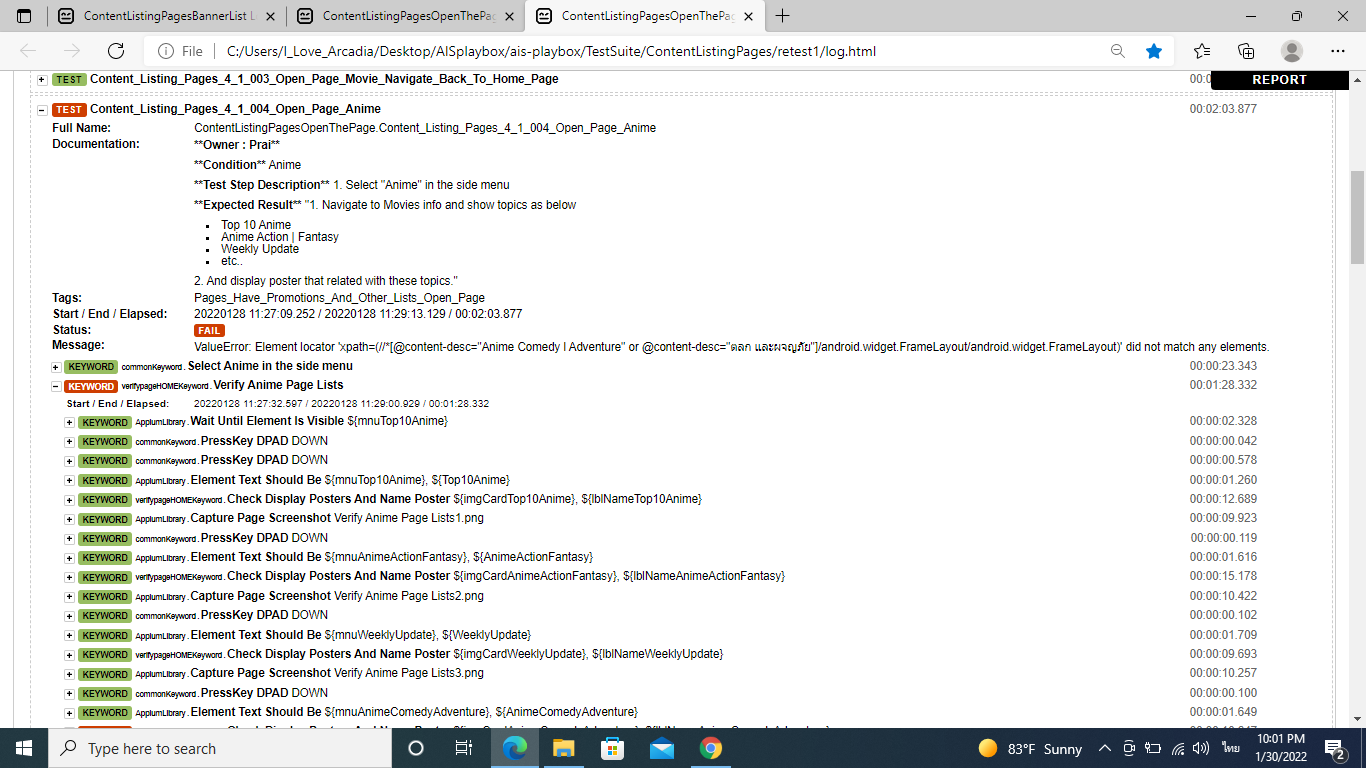
|  |  |
| --- | --- |
| **Message:** | Element locator 'xpath=((//android.support.v7.widget.RecyclerView[@resource-id='com.ais.playbox.prod:id/row\_content']/android.widget.FrameLayout)[2])[@focused=True]' did not match any elements after 5 seconds |

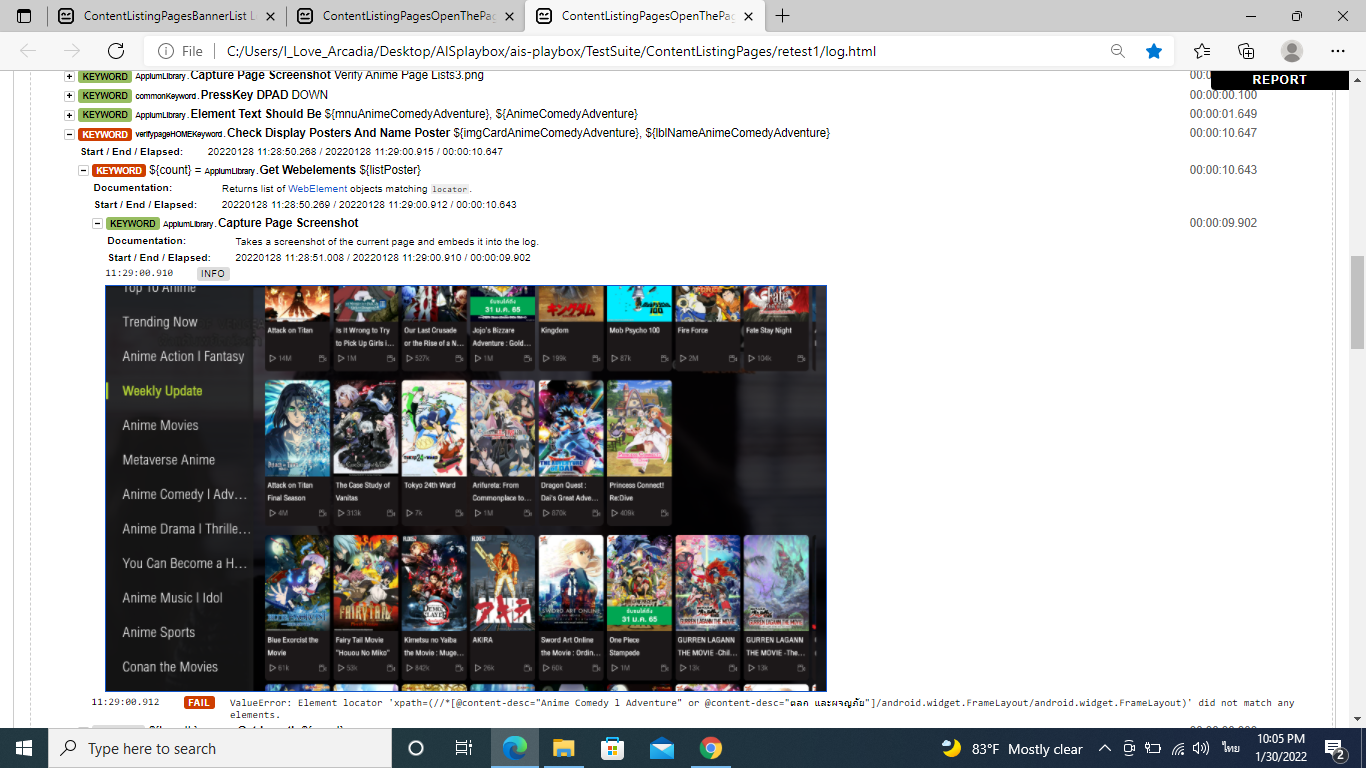
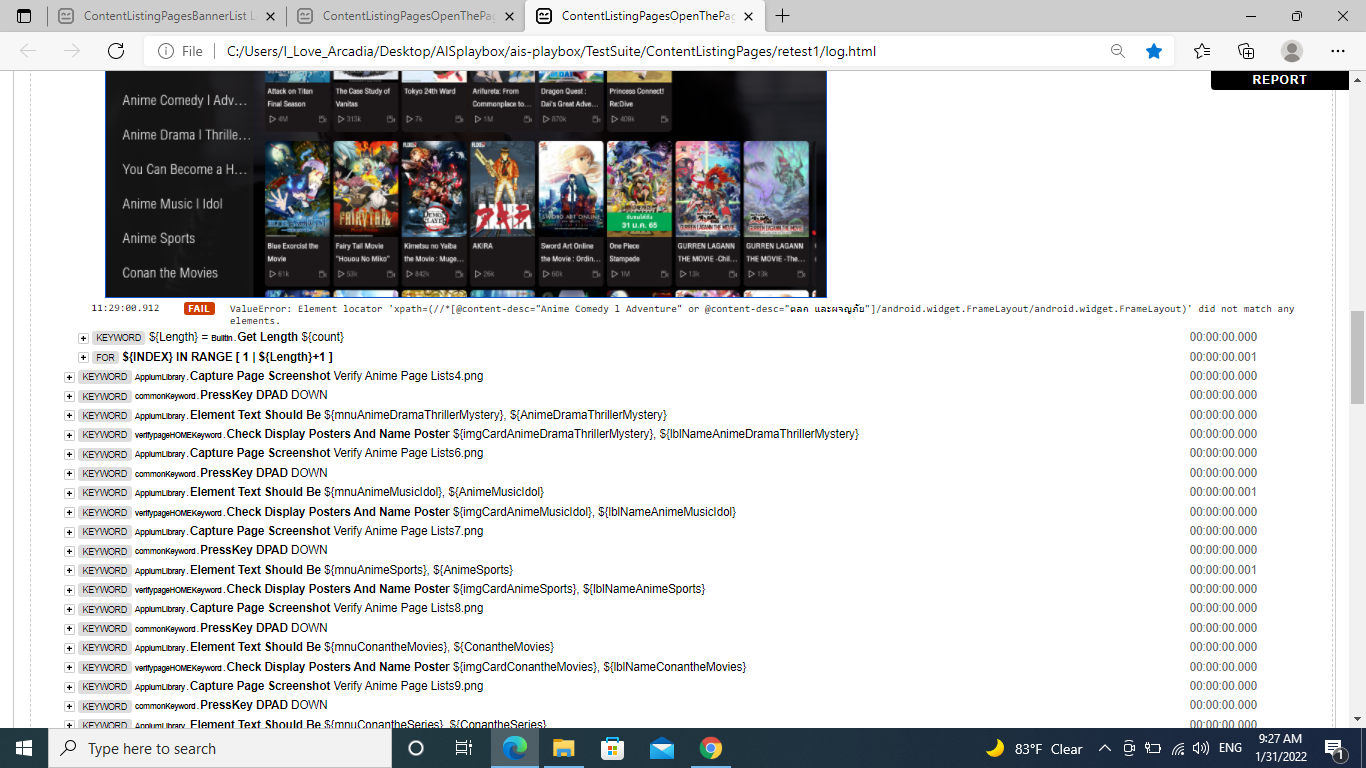
จะหมายถึง ไม่สามารถหา element ที่มี xpath ที่ระบุไว้ได้ ภายใน 5 วินาที

ในส่วนที่ 2 จะบอกเราว่า element ที่หาไม่เจอนั้นอยู่ที่คำสั่งใด และเป็น xpath ของตัวแปรใด



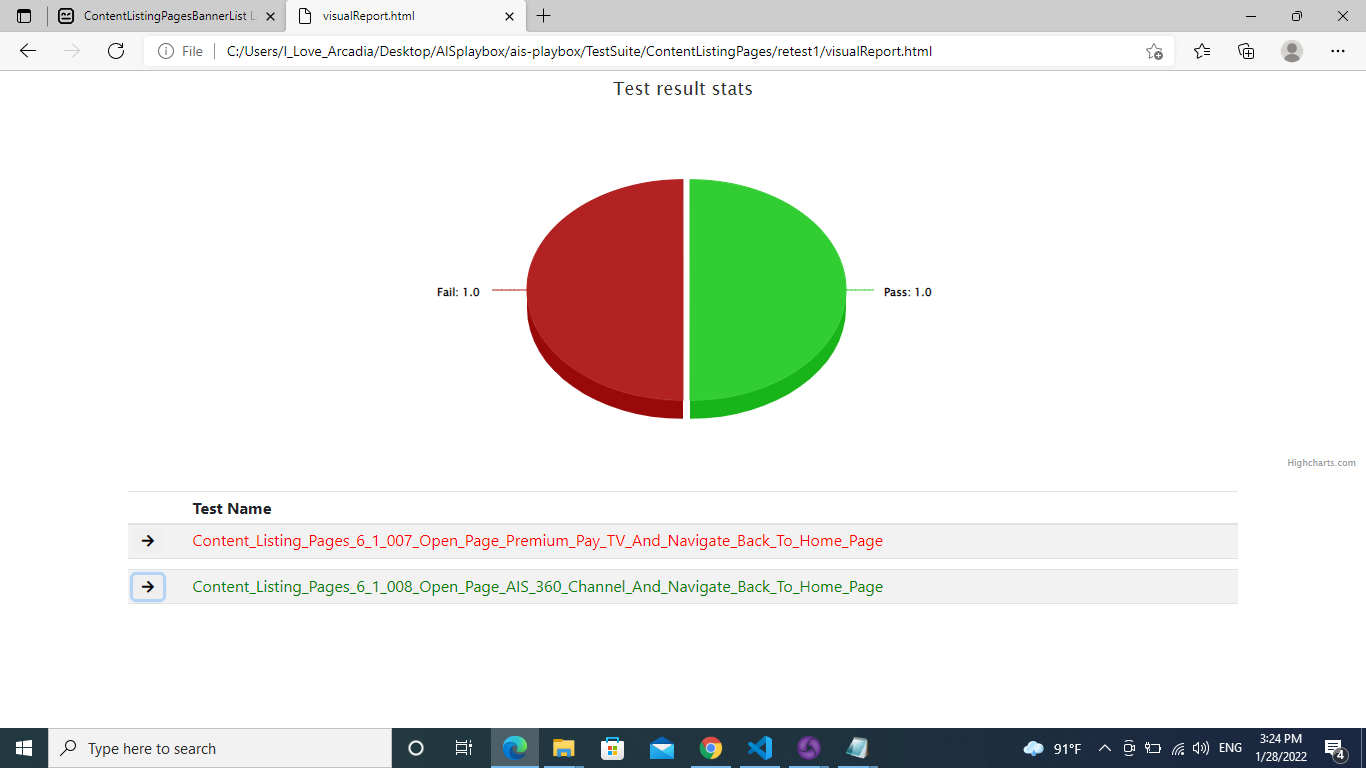
จะบอกเราว่า xpath ที่หาไม่เจออยู่นั้นอยู่ที่คำสั่ง Verify Banner… และอยู่ในคำสั่งย่อย Wait Until Element Is Visible และ xpath นั้น เป็นของตัวแปรชื่อ ${imgbannerPromotionToplist2AfterRight} ซึ่งเราก็จะไปตรวจสอบว่า xpath ของตัวแปรนั้นหาไม่เจอเนื่องจากเหตุใด และทำการแก้ไขต่อไป

ตัวอย่างที่ 2

จากภาพจะเห็นได้ว่า test case นี้ไม่ผ่านที่คำสั่ง  **Check Display Posters And Name Poster คำสั่งย่อย Get Webelements ${listPoster}** มีข้อความที่แสดงคือ

หมายความว่า ไม่เจอ**ตัวแปร ${listPoster}** xpath=........ตามที่เราระบุไว้ ในหน้านี้ ถ้าเราดูจากภาพเราก็จะคิดว่า ก็มี poster อยู่ หัวข้อ Anime Comedy l Adventure ตลก และผจญภัย ก็ verify ผ่าน ไม่น่ามีจุดผิดอะไร แต่ถ้าเราสังเกตดีๆ เราจะพบว่า ลำดับของ poster ด้านขวาที่แสดงอยู่นั้นจะเป็นของหัวข้ออื่น ที่เรา verify หัวข้อ Anime Comedy l Adventure ตลก และผจญภัย ผ่านเนื่องจากทางเมนูด้านซ้ายเจอหัวข้อนี้อยู่จริงๆ แต่ poster ด้านขวาจะเป็นของ หัวข้อ Weekly Update ลำดับของเมนูด้านซ้าย มีการเปลี่นแปลงไป ทำให้ poster ที่เกี่ยวข้องกับหัวข้อ Anime Comedy l Adventure ตลก และผจญภัย ยังไม่แสดงขึ้นมา

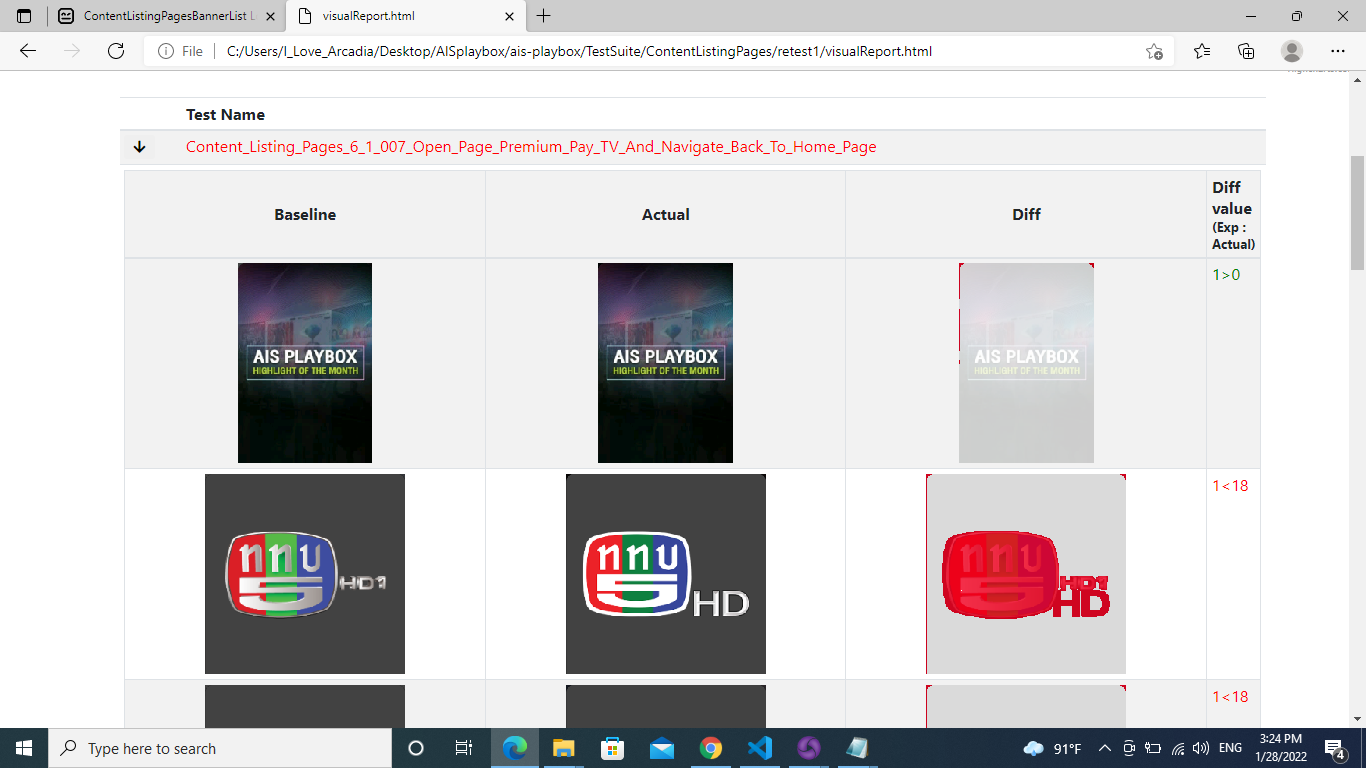
ในการดูว่า test case นั้นผิดพลาดที่ส่วนไหน เราจะต้องทราบด้วยว่าสิ่งที่ควรจะเป็นในจุดที่ run test ไม่ผ่านควรจะเป็นอย่างไร ต้องตรวจสอบคำสั่งก่อนหน้าจุดที่ run test ไม่ผ่านด้วย ว่าทำงานถูกต้องจริงๆหรือไม่ Log จะทำให้เราสามารถหาส่วนที่ต้องแก้ไข้ หรือส่วนที่มีความเปลี่ยนแปลงไปได้ง่ายขึ้นนั่นเอง

ตัวอย่างที่ 3

Log ที่แสดงการ เปรียบเทียบรูปภาพ ของคำสั่ง Compare Image ใน Library Robot Eye จากภาพจะเป็นการ run test 2 ข้อ มีผ่าน 1 ข้อและไม่ผ่าน 1 ข้อ ถ้าเรากดดูรายละเอียดจะเป็นดังนี้

ข้อที่ 1 : ภาพในกรอบสีแดง จะมีความแตกกัน 18 จุดเกินค่าที่เรายอมรับได้ที่ตั้งไว้ 1

ข้อที่ 2 : จุดที่แตกต่างกันของภาพที่เราเปรียบเทียบเป็น 0 ไม่เกินค่าที่เราตั้งไว้



|  |  |
| --- | --- |
|  |  |