TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**

–––––––––––––––––––––––––––––––



**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**

**NGÀNH HỆ THỐNG THÔNG TIN QUẢN LÝ**

**CHUYÊN NGÀNH QUẢN TRỊ HỆ THỐNG THÔNG TIN**

**NGHIÊN CỨU KIỂM THỬ TỰ ĐỘNG**

**CHO ỨNG DỤNG WEB GURU99**

**BẰNG CÔNG CỤ ROBOT FRAMEWORK**

Sinh viên thực hiện : Lê Nhật Phi

Lớp : 45K21.1

Đơn vị thực tập : TMA Solution Bình Định

Cán bộ hướng dẫn : Huỳnh Thanh Nhã

Giảng viên hướng dẫn : ThS.Cao Thị Nhâm

**Đà Nẵng, 8/2022**

**NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP**

Họ và tên sinh viên:

Lớp: Khoa: Trường:

Thực tập từ ngày: …./……/ 2022 đến ngày: ........./ .……./ 2022

Tại:

Địa chỉ:

Sau quá trình thực tập tại đơn vị của sinh viên, chúng tôi có một số nhận xét, đánh giá như sau:

**1. Về thái độ, ý thức, đạo đức, kỷ luật**

**2. Kiến thức chuyên môn**

**3. Khả năng hòa nhập và thích nghi với công việc**

**4. Trách nhiệm, sáng tạo trong công việc**

**5. Các nhận xét khác**

**Đánh giá chung:**

**Điểm:**

……….., ngày .......tháng ......năm 2022

**Xác nhận của đơn vị thực tập**

# LỜI CẢM ƠN

# LỜI CAM ĐOAN

Quy định:

Soạn thảo trên trang A4 (trang dọc, lề trái: 3.5cm; trên, phải, dưới: 2.5 cm), Font Times New Roman, canh đều 2 bên, size 13, cách dòng 1.5, cách đoạn trên 6pt, cách đoạn dưới 3pt, hàng đâu tiên lùi vào 1.27 cm; hình và bảng soạn theo caption, chèn trích dẫn chéo (Cross-reference) cho bảng và hình; các danh mục hình, bảng, mục lục làm tự động; đánh số trang như file mẫu (bìa không có số trang, danh mục + mục lục số trang theo i, ii, iii…, nội dung chính theo 1,2,3…)

# MỤC LỤC

[LỜI CẢM ƠN i](#_Toc109907839)

[LỜI CAM ĐOAN ii](#_Toc109907840)

[MỤC LỤC iv](#_Toc109907841)

[DANH MỤC HÌNH ẢNH vii](#_Toc109907842)

[DANH MỤC BẢNG BIỂU viii](#_Toc109907843)

[DANH MỤC CÁC TỪ VIẾT TẮT ix](#_Toc109907844)

[LỜI MỞ ĐẦU 1](#_Toc109907845)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 2](#_Toc109907846)

[1.1. Tổng quan về kiểm thử phần mềm. 2](#_Toc109907847)

[1.1.1. Kiểm thử phần mềm là gì? 2](#_Toc109907848)

[1.1.2. Lợi ích của kiểm thử phần mềm. 2](#_Toc109907849)

[1.1.3. Vòng đời kiểm thử phần mềm (Software Testing Life Cycle). 2](#_Toc109907850)

[1.1.4. 7 nguyên tắc kiểm thử phần mềm. 3](#_Toc109907851)

[1.1.5. Error, Fault, Failure, Bug và Defect là gì? 3](#_Toc109907852)

[1.1.6. Verification và Validation là gì? 4](#_Toc109907853)

[1.1.7. Phân biệt giữa QA và QC. 5](#_Toc109907854)

[1.2. Vòng đời phát triển phần mềm (Software Development Life Cycle). 5](#_Toc109907855)

[1.2.1. SDLC là gì? 5](#_Toc109907856)

[1.2.2. Các mô hình phổ biến của SDLC. 6](#_Toc109907857)

[1.2.3. Các phương pháp của SDLC. 8](#_Toc109907858)

[1.3. Loại kiểm thử phần mềm. 9](#_Toc109907859)

[1.3.1. Kiểm thử thủ công. 9](#_Toc109907860)

[1.3.2. Kiểm thử tự động. 9](#_Toc109907861)

[1.3.3. Kiểm thử bảo mật. 9](#_Toc109907862)

[1.3.4. Kiểm thử API. 9](#_Toc109907863)

[1.4. Kỹ thuật kiểm thử phần mềm. 10](#_Toc109907864)

[1.4.1. Kiểm thử tĩnh. 10](#_Toc109907865)

[1.4.2. Kiểm thử động. 10](#_Toc109907866)

[1.4.3. Kiểm thử hộp trắng. 10](#_Toc109907867)

[1.4.4. Kiểm thử hộp đen. 10](#_Toc109907868)

[1.4.5. Kiểm thử hộp xám 10](#_Toc109907869)

[1.5. Cấp độ kiểm thử phần mềm. 11](#_Toc109907870)

[1.5.1. Kiểm thử đơn vị. 11](#_Toc109907871)

[1.5.2. Kiểm thử tích hợp. 11](#_Toc109907872)

[1.5.3. Kiểm thử hệ thống. 11](#_Toc109907873)

[1.5.4. Kiểm thử chấp nhận 11](#_Toc109907874)

[CHƯƠNG 2. Tổng quan về công cụ kiểm thử selenium và ROBOT FRAMEWORK 12](#_Toc109907875)

[2.1. Công cụ kiểm thử tự động Selenium 12](#_Toc109907876)

[2.1.1. Giới thiệu chung về Selenium 12](#_Toc109907877)

[2.1.2. Các thành phần của Selenium. 13](#_Toc109907878)

[2.2. Tổng quan về Robot FrameWork. 15](#_Toc109907879)

[2.2.1. Robot Framework là gì? 15](#_Toc109907880)

[2.2.2. Các tính năng nổi bật của Robot FrameWork. 16](#_Toc109907881)

[2.2.3. Các thư viện hỗ trợ trong Robot Framework. 16](#_Toc109907882)

[2.3. Tổng quan về Xpath. 17](#_Toc109907883)

[2.3.1. XPath là gì? 17](#_Toc109907884)

[2.3.2. Cú pháp của XPath. 18](#_Toc109907885)

[2.4. Cài đặt môi trường thực hiện kiểm thử tự động. 18](#_Toc109907886)

[2.4.1. Cài đặt Python. 18](#_Toc109907887)

[2.4.2. Cài đặt Robot FrameWork bằng pip. 21](#_Toc109907888)

[2.4.3. Cài đặt thư viện Selenium2Library. 22](#_Toc109907889)

[2.4.4. Cài đặt Webdriver Manager. 23](#_Toc109907890)

[2.4.5. Cài đặt ChromeDriver 25](#_Toc109907891)

[CHƯƠNG 3. TRIỂN KHAI KIỂM THỬ TỰ ĐỘNG ỨNG DỤNG WEB GURU99 TRÊN ROBOT FRAMEWORK 26](#_Toc109907892)

[3.1. Mục 3.1 26](#_Toc109907893)

[3.1.1. Mục 3.1.1 26](#_Toc109907894)

[3.1.2. Mục 3.1.2 26](#_Toc109907895)

[3.2. Mục 3.2 26](#_Toc109907896)

[CHƯƠNG 4. KẾT QUẢ 27](#_Toc109907897)

[4.1. Mục 4.1… 27](#_Toc109907898)

[4.2. Mục 4.2… 27](#_Toc109907899)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 28](#_Toc109907900)

[TÀI LIỆU THAM KHẢO 29](#_Toc109907901)

[PHỤ LỤC 30](#_Toc109907902)

*(Mục lục này chỉ là ví dụ)*

# DANH MỤC HÌNH ẢNH

[Hình 1.1.3‑1 Hình ảnh vòng đời kiểm thử 3](#_Toc109907903)

[Hình 1.2.1‑1 Hình ảnh giai đoạn vòng đời phát triển phần mềm. 6](#_Toc109907904)

[Hình 2.1.1‑1 Hình ảnh thành phần của công cụ Selenium 12](#_Toc109907905)

[Hình 2.2.3‑1 Ví dụ về thư viện CalculatorLibrary 17](#_Toc109907906)

[Hình 2.3.2‑1 Cú pháp của XPath 18](#_Toc109907907)

[Hình 2.4.1‑1 Cài đặt Python 3.9.1 19](#_Toc109907908)

[Hình 2.4.1‑2 Thiết lập PYTHON\_HOME 19](#_Toc109907909)

[Hình 2.4.1‑3 Thiết lập Path 20](#_Toc109907910)

[Hình 2.4.1‑4 Cài đặt Python thành công 21](#_Toc109907911)

[Hình 2.4.2‑1 Cài đặt robot framework thành công 21](#_Toc109907912)

[Hình 2.4.2‑2 Cài đặt wxPython==4.1.1 thành công 21](#_Toc109907913)

[Hình 2.4.2‑3 Cài đặt RIDE thành công 22](#_Toc109907914)

[Hình 2.4.2‑4 Giao diện RIDE khi mở sau cài đặt thành công 22](#_Toc109907915)

[Hình 2.4.3‑1 Cài đặt Selenium2Library thành công 23](#_Toc109907916)

[Hình 2.4.4‑1 Cài đặt Webdriver – Manager thành công 23](#_Toc109907917)

[Hình 2.4.4‑2 Cài đặt webdrivermanager 24](#_Toc109907918)

[Hình 2.4.4‑3 cài đặt webdrivermanager chrome thành công 24](#_Toc109907919)

[Hình 2.4.4‑4 Thiết lập đường dẫn sau khi cài Webdrivermanager Chrome 24](#_Toc109907920)

[Hình 2.4.5‑1 Cài đặt Chromedriver 25](#_Toc109907921)

[Hình 3.1.2‑1 Kiến trúc của mô hình 27](#_Toc109907922)

# DANH MỤC BẢNG BIỂU

[Bảng 1.1.6‑1 Bảng phân biệt giữa Verification và Validation 4](#_Toc109907923)

[Bảng 1.1.7‑1 Bảng phân biệt giữa QA và QC. 5](#_Toc109907924)

[Bảng 1.2.2‑1 Ưu – nhược điểm của mô hình thác nước. 7](#_Toc109907925)

[Bảng 1.2.2‑2 Ưu – nhược điểm của mô hình chữ V. 7](#_Toc109907926)

[Bảng 1.2.2‑3 Ưu - nhược điểm của mô hình xoắn ốc. 8](#_Toc109907927)

[Bảng 1.2.2‑4 Ưu – nhược điểm của mô hình Agile. 8](#_Toc109907928)

[Bảng 2.1.1‑1 Ưu – Nhược điểm của công cụ Selenium 13](#_Toc109907929)

[Bảng 2.1.2‑1. Ưu – nhược điểm của Selenium IDE 14](#_Toc109907930)

[Bảng 2.1.2‑2. Ưu – nhược điểm của Selenium Webdriver 14](#_Toc109907931)

[Bảng 2.1.2‑3 Ưu – nhược điểm của Selenium RC 15](#_Toc109907932)

[Bảng 3.1.2‑1 Kiến trúc 27](#_Toc109907933)

# DANH MỤC CÁC TỪ VIẾT TẮT

**AI** : Artificial Intelligence

**…**

# LỜI MỞ ĐẦU

1. **Mục tiêu nghiên cứu của đề tài**

* Đề tài này nghiên cứu …

1. **Nhiệm vụ của đề tài**

* Nghiên cứu ...
* ...

1. **Phương pháp nghiên cứu**

* ...

1. **Đối tượng và phạm vi nghiên cứu**

* ...

1. **Kết cấu của đề tài**

Đề tài được tổ chức gồm phần mở đầu, x chương nội dung và phần kết luận...

* Mở đầu
* **Chương 1**:
* **Chương 2**:
* **Chương 3**:
* **Chương 4**:
* Kết luận và hướng phát triển

# CƠ SỞ LÝ THUYẾT

## Tổng quan về kiểm thử phần mềm.

### Kiểm thử phần mềm là gì?

Kiểm thử phần mềm là phương pháp kiểm tra xem sản phẩm phần mềm đó trên thực thế có phù hợp với các yêu cầu đặt ra hay không, và đảm bảo rằng không có lỗi hay khiếm khuyết.

Mục đích của kiểm thử phần mềm là xác định các lỗi, khiếm khuyết hoặc các yêu cầu còn thiếu của phần mềm so với yêu cầu thực tế.

### Lợi ích của kiểm thử phần mềm.

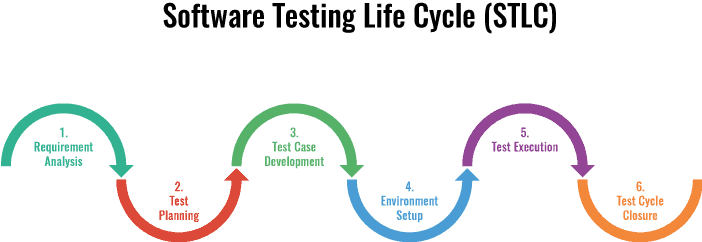
* **Hiệu quả về chi phí:** việc tìm ra các lỗi/khiếm khuyết sẽ giúp công ty tiết kiệm, việc xác định lỗi trong giai đoạn đầu sẽ giúp quá trình sửa chữa tốn ít chi phí hơn.
* **Bảo mật:** kiểm thử giúp loại bỏ các rủi ro và vấn đề trong sản phẩm.
* **Chất lượng sản phẩm:** giúp công ty cung cấp các sản phẩm chất lượng cho khách hàng.
* **Sự hài lòng của khách hàng:** kiểm thử giúp sản phẩm đầy đủ và phù hợp với các yêu cầu của khách hàng.

### Vòng đời kiểm thử phần mềm (Software Testing Life Cycle).

Vòng đời kiểm thử phần mềm (STLC) là quy trình kiểm thử được thực hiện theo hệ thống và có kế hoạch rõ ràng. Trong quá trình kiểm thử, rất nhiều giai đoạn khác nhau được thực hiện một cách tuần tự. Mỗi giai đoạn đều có đầu vào và đầu ra khác nhau nhưng đều hướng tới mục tiêu cuối cùng là đảm bảo chất lượng sản phẩm.

Vòng đời kiểm thử phần mềm (STLC) gồm 6 giai đoạn:

* Phân tích yêu cầu.
* Lập kế hoạch kiểm thử.
* Thiết kế kịch bản kiểm thử.
* Thiết lập môi trường kiểm thử.
* Thực hiện kiểm thử.
* Đóng chu trình kiểm thử.



Hình 1.1.3‑1 Hình ảnh vòng đời kiểm thử

### 7 nguyên tắc kiểm thử phần mềm.

* Kiểm thử đưa ra lỗi.
* Kiểm thử cạn kiệt là không thể.
* Kiểm thử càng sớm càng tốt.
* Sự tập trung của lỗi.
* Nghịch lí thuốc trừ sâu.
* Kiểm thử phụ thuộc vào ngữ cảnh.
* Không có lỗi – Sai lầm.

### Error, Fault, Failure, Bug và Defect là gì?

**Error** là hành động của con người dẫn đến kết quả sai.

**Fault** là lỗi xảy ra khi làm sai các bước, quy trình hoặc chuẩn bị dữ liệu.

**Failure** là lỗi khi có kết quả sai lệch so với yêu cầu đặc tả, là sự khác biệt giữa kết quả thực tế trên màn hình và kết quả mong đợi của một thành phần, hệ thống hoặc service nào đó.

**Bug** là một khiếm khuyết trong một thành phần hoặc hệ thống mà nó có thể làm cho thành phần hoặc hệ thống này không thực hiện đúng chức năng yêu cầu của nó.

**Defect** là lỗi trong quá trình phát triển hoặc lỗi logic làm cho chương trình hoạt động sai yêu cầu đề ra.

### Verification và Validation là gì?

Verification là một quá trình đánh giá các sản phẩm làm việc trung gian của một vòng đời phát triển phần mềm để kiểm tra xem liệu rằng chúng ta có đi đúng hướng để tạo ra sản phẩm cuối cùng. Cũng có thể hiểu, Verification là một quá trình để đánh giá các sản phẩm trung gian của phần mềm để kiểm tra xem sản phẩm có đáp ứng các điều kiện đặt ra trong khi bắt đầu một giai đoạn.

Validation là quá trình đánh giá sản phẩm cuối cùng để kiểm tra xem phần mềm có đáp ứng được yêu cầu nghiệp vụ không.

|  |  |
| --- | --- |
| Verification | Validation |
| Đánh giá sản phẩm trung gian để kiểm tra xem nó có đáp ứng các yêu cầu cụ thể của từng giai đoạn không. | Đánh giá sản phẩm cuối cùng để kiểm tra xem nó có đáp ứng yêu cầu nghiệp vụ không. |
| Kiểm tra xem sản phẩm có được xây dựng đúng yêu cầu và đặc điểm kỹ thuật thiết kế không. | Xác định xem phần mềm có phù hợp với nhu cầu sử dụng và đáp ứng yêu cầu nghiệp vụ không. |
| Kiểm tra “Chúng tôi xây dựng sản phẩm đúng không?”. | Kiểm tra “Chúng tôi xây dựng đúng sản phẩm”. |
| Thực hiện mà không cần chạy phần mềm. | Thực hiện cùng với việc chạy phần mềm. |
| Bao gồm tất cả các kỹ thuật kiểm thử tĩnh. | Bao gồm tất cả các kỹ thuật kiểm thử động. |

Bảng 1.1.6‑1 Bảng phân biệt giữa Verification và Validation

### Phân biệt giữa QA và QC.

Quality Assurance (QA) là một tập hợp các hoạt động có kế hoạch và có hệ thống cần thiết để cung cấp sự tin tưởng đầy đủ rằng các sản phẩm và dịch vụ sẽ phù hợp với các yêu cầu cụ thể và đáp ứng nhu cầu của người sử dụng.

Quality Control (QC) là quá trình mà chất lượng sản phẩm được so sánh với các tiêu chuẩn áp dụng và hành động được thực hiện khi phát hiện thấy sự không phù hợp.

|  |  |
| --- | --- |
| QA | QC |
| Mục tiêu: ngăn ngừa khiếm khuyết. | Mục tiêu: xác định và cải thiện các khiếm khuyết. |
| Không liên quan đến thực hiện chương trình. | Luôn luôn liên quan đến việc thực hiện chương trình. |
| Là quy trình để tạo ra phần mềm. | Là quá trình để xác minh phần mềm. |
| Tất cả thành viên nhóm chịu trách nhiệm. | Đội kiểm thử chịu trách nhiệm. |

Bảng 1.1.7‑1 Bảng phân biệt giữa QA và QC.

## Vòng đời phát triển phần mềm (Software Development Life Cycle).

### SDLC là gì?

Vòng đời phát triển phần mềm (SDLC) là một quy trình phát triển phần mềm, được chia nhỏ thành nhiều giai đoạn từ ý tưởng đến triển khai. SDLC được sử dụng để giúp các nhóm phát triển tối ưu quy trình làm việc của họ và làm cho nó có thể mở rộng mà vẫn đảm bảo chất lượng.



Hình 1.2.1‑1 Hình ảnh giai đoạn vòng đời phát triển phần mềm.

SDLC gồm có 6 giai đoạn:

* Thu thập và phân tích yêu cầu.
* Thiết kế.
* Phát triển.
* Kiểm thử.
* Triển khai.
* Bảo trì.

### Các mô hình phổ biến của SDLC.

#### Mô hình thác nước (Water Model)

Là mô hình SDLC được chấp nhận rộng rãi. Theo cách tiếp cận này, toàn bộ quá trình phát triển phần mềm được chia thành các giai đoạn khác nhau, kết quả của một giai đoạn đóng vai trò là đầu vào cho giai đoạn tiếp theo.

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Dễ hiểu, dễ quản lý. * Nhóm dự án hiểu rõ các yêu cầu của khách hàng * Có nhiều tài liệu cung cấp thông tin cho khách hàng. * Thích hợp cho các dự án đơn giản và nhỏ. | * Không cho phép thay đổi phạm vi dự dán. * Không cho phép thay đổi yêu cầu của dự án. * Chỉ nhìn thấy sản phẩm sau khi hoàn thành dự án. * Không dễ xử lý rủi ro bất ngờ. |

Bảng 1.2.2‑1 Ưu – nhược điểm của mô hình thác nước.

#### Mô hình chữ V (V-Model)

Là mô hình mở rộng của mô hình thác nước, trong mô hình này, các giai đoạn được lập kế hoạch song song. Vì vậy, có giai đoạn xác minh của SDLC ở phía bên kia và giai đoạn xác nhận ở phía bên kia. V-Model tham gia theo gia đoạn lập trình.

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Dễ dàng quản lý. * Có tính tổ chức và hệ thống. * Phù hợp với các dự án có quy mô vừa và nhỏ. | * Không dễ thay đổi các yêu cầu thường xuyên. * Có sự không chắc chắc và có tính rủi ro. |

Bảng 1.2.2‑2 Ưu – nhược điểm của mô hình chữ V.

#### Mô hình xoắn ốc.

Là quy trình phát triển định hướng rủi ro cho các dự án phần mềm. Kết hợp thế mạnh của các mô hình khác và giải quyết của các mô hình trước còn tồn tại.

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Các yêu cầu được thay đổi linh hoạt. * Hạn chế được rủi ro. * Phần mềm được sản xuất sớm trong vòng đời của nó. | * Không thích hợp cho dự án có quy mô nhỏ. * Đòi hỏi quản lý phải có chuyên môn cao. * Không thể xác định trước thời gian và chi phí của dự án. |

Bảng 1.2.2‑3 Ưu - nhược điểm của mô hình xoắn ốc.

#### Mô hình Agile

Là phương pháp thực hành nhằm thúc đẩy sự tương tác liên tục của quá trình phát triển và kiểm thử trong quá trình SDLC của bất kỳ dự án nào. Trong Agile, toàn bộ dự án được chia thành các bản xây dựng nhỏ tăng dần.

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Phù hợp với những dự án thường xuyên thay đổi yêu cầu hoặc không có yêu cầu rõ ràng. * Khách hàng có thể xem trước từng phần dự án trong suốt quá trình phát triển. * Tỉ lệ thành công cao. | * Khó lên được kế hoạch dự án. * Cần sự hợp tác của các thành viên cao. * Chi phí cao so với các mô hình khác. * Ít tài liệu hướng dẫn cho nhân viên. |

Bảng 1.2.2‑4 Ưu – nhược điểm của mô hình Agile.

### Các phương pháp của SDLC.

#### Phương pháp Scrum

Scrum là bộ khung làm việc (framework) giúp các công ty, tổ chức chia nhỏ công việc thành những phần nhỏ hơn, để quản lý dễ dàng hơn và được hoàn thành bởi một nhóm liên chức năng trong khoảng thời gian quy định từ 2-4 tuần.

Lợi ích của phương pháp Scrum:

* Cải thiện chất lượng phần mềm.
* Rút ngắn thời gian phát hành phần mềm.
* Nâng cao tinh thần đồng đội.
* Tăng mức độ hài long khách hàng, …

#### Phương pháp Kanban.

Kanban là công cụ được sử dụng để giúp ác công ty tổ chức đạt hiệu quả cao trong công việc, là công cụ kiểm soát sản xuất, dung nhiều màu để chỉ định nguyên liệu và các công đoạn khác nhau. Kanban giới hạn số lượng công việc cho phép trong một điều kiện nhất định.

## Loại kiểm thử phần mềm.

### Kiểm thử thủ công.

Kiểm thử thủ công là một loại kiểm thử phần mềm trong đó các trường hợp kiểm thử được người kiểm thử thực hiện theo cách thủ công mà không sử dụng bất kỳ công cụ tự động nào. Mục đích là xác định các lỗi, sự cố và khiếm khuyết trong ứng dụng phần mềm.

### Kiểm thử tự động.

Kiểm thử tự động là quá trình sử dụng các công cụ, script và phần mềm để thực hiện các trường hợp kiểm thử, bằng cách lặp lại những hành động được xác định trước để thực hiện các test case.

### Kiểm thử bảo mật.

Kiểm thử bảo mật là một loại kiểm thử phần mềm nhằm khám phá các lỗ hổng, mối đe dọa, rủi ro trong một ứng dụng phần mềm và ngăn chặn các cuộc tấn công độc hại từ những kẻ xâm nhập. Mục đích của kiểm thử bảo mật là xác định tất cả các lỗ hổng và điểm yếu có thể có của hệ thống phần mềm có thể dẫn đến việc mất thông tin, doanh thu, danh tiếng dưới tay của nhân viên hoặc người ngoài của tổ chức.

### Kiểm thử API.

Kiểm thử API là một loại kiểm thử nhằm kiểm tra trực tiếp server mà không cần các giao diện lập trình ứng dụng. Kiểm thử API cũng là một phần của kiểm thử tích hợp có vai trò xác định xem phần mềm có đáp ứng mong đợi về chức năng, độ tin cậy, hiệu suất và bảo mật hay không.

## Kỹ thuật kiểm thử phần mềm.

### Kiểm thử tĩnh.

Kiểm thử tĩnh (Static testing): Kiểm thử một thành phần hoặc hệ thống ở mức độ đặc tả hoặc thực nghiệm mà không chạy phần mềm, ví dụ: nhận xét hoặc phân tích code tĩnh.

### Kiểm thử động.

Kiểm thử động (Dynamic testing): Là kiểm thử có tiến hành chạy phần mềm, thực hiện các bước để kiểm thử xem các chức năng đã thực hiện có chạy đúng với kết quả mong đợi hay không.

### Kiểm thử hộp trắng.

Kiểm thử hộp trắng (White Box Testing) là kỹ thuật kiểm thử phần mềm trong đó cấu trúc bên trong, thiết kế và mã hóa của phần mềm được kiểm tra để xác minh luồng đầu vào – đầu ra và cải thiện thiết kế, khả năng sử dụng và bảo mật.

### Kiểm thử hộp đen.

Kiểm thử hộp đen (Black Box Testing) là 1 phương pháp kiểm thử phần mềm được thực hiện mà không biết được cấu tạo bên trong của phần mềm, là cách mà các tester kiểm tra xem hệ thống như một chiếc hộp đen, không có cách nào nhìn thấy bên trong của cái hộp.

### Kiểm thử hộp xám

Kiểm thử hộp xám (Gray box testing) là sự kết hợp giữa kiểm thử hộp đen và kiểm thử hộp trắng. Với kỹ thuật này, cấu trúc bên trong sản phẩm được biết một phần.

## Cấp độ kiểm thử phần mềm.

### Kiểm thử đơn vị.

Kiểm thử đơn vị (Unit testing), mức độ kiểm thử phần mềm này do các developer phụ trách, họ sẽ kiểm tra các module, các hàm, các phương thức,… mà họ viết ra nhằm gia tăng sự tin cậy cho các chức năng mà mình viết.

Kiểm thử đơn vị nằm trong phạm vi của kiểm thử hộp trắng.

### Kiểm thử tích hợp.

Kiểm thử tích hợp (Integration testing) là kiểm thử sự tương tác giữa các chức năng với nhau trong hệ thống và được thực hiện bởi các Tester.

Kiểm thử tích hợp có 2 hướng tiếp cận phổ biển: tích hợp từ dưới đi lên và tích hợp từ trên đi xuống.

### Kiểm thử hệ thống.

Kiểm thử hệ thống (System testing) là kiểm thử một hệ thống đã hoàn thành, đã tích hợp đầy đủ các chức năng nhằm kiểm tra xem hệ thống phần mềm đó có đáp ứng đầy đủ các yêu cầu chức năng theo bản đặc tả yêu cầu phần mềm (SRS) hay không và được thực hiện bởi các Tester.

Kiểm thử hệ thống nằm trong phạm vi kiểm thử hộp đen.

Một số kiểm thử thường được thực hiện trong kiểm thử hệ thống: kiểm thử chức năng, kiểm thử hiệu năng, kiểm thử cơ sở dữ liệu, kiểm thử bảo mật,….

### Kiểm thử chấp nhận

Kiểm thử chấp nhận (Acceptance testing) là kiểm tra xem hệ thống có đá ứng đúng nhu cầu và mong đợi của khách hàng hay không và được thực hiện bởi người dung hoặc khách hàng.

Kiểm thử chấp nhận gồm 2 loại:

* Kiểm thử Alpha.
* Kiểm thử Beta.

# Tổng quan về công cụ kiểm thử selenium và ROBOT FRAMEWORK

## Công cụ kiểm thử tự động Selenium

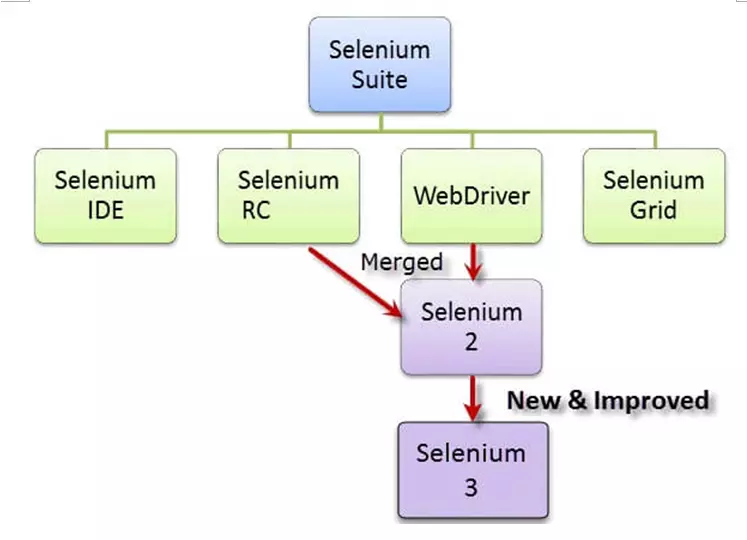
### Giới thiệu chung về Selenium

Selenium là bộ kiểm thử tự động miễn phí (mã nguồn mở) dành cho các ứng dụng web trên các trình duyệt và nền tảng khác nhau. Nó khá giống với HP Quick Test Pro (QTP bây giờ là UFT) chỉ khác là Selenium thì tập trung vào việc tự động hóa các ứng dụng dựa trên nền tảng web.

Kiểm thử được thực hiện bằng cách sử dụng công cụ Selenium thường được gọi là kiểm thử Selenium.

Selenium không chỉ là một công cụ độc lập mà còn là một bộ công cụ của phần mềm, mỗi bộ đều đáp ứng được nhu cầu kiểm thử khác nhau của một tổ chức. Nó gồm 4 thành phần:

* Selenium Integrated Development Environment (Selenium IDE)
* Selenium Remote Control (Selenium RC)
* WebDriver
* Selenium Grid



Hình 2.1.1‑1 Hình ảnh thành phần của công cụ Selenium

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Mã nguồn mở, miễn phí sử dụng. * Khả năng mở rộng cao. * Có thể chạy kiểm thử trên các trình duyệt khác nhau. * Hỗ trợ các hệ điều hành khác nhau. * Có thể thực hiện kiểm thử song song. * Có thể thực hiện kiểm thử khi trình duyệt đang bị thu nhỏ | * Chỉ có thể kiểm thử ứng dụng web. * Không có kho lưu trữ đối tượng được xây dựng sẵn. * Kiểm thử tự động với tốc độ chậm. * Không thể truy cập phần tử bên ngoài ứng dụng web đang được kiểm thử. * Thông số có thể được thực hiện thông qua lập trình nhưng rất khó thực hiện. * Không hỗ trợ tạo báo cáo test/bug. |

Bảng 2.1.1‑1 Ưu – Nhược điểm của công cụ Selenium

### Các thành phần của Selenium.

#### Selenium IDE

Là một công cụ được xây dựng dưới dạng Add-ons của Firefox. Đây là cách tiện ích để xây dựng các ca kiểm thử, gồm các phần tử giao diện giúp cho ta có thể thực hiện thao tác. Selenium IDE là khuôn khổ đơn giản nhất trong bộ Selenium và là cách đơn giản nhất để học. Nó chỉ nên sử dụng như là một công cụ tạo mẫu.

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Dễ dàng cài đặt và sử dụng. * Không yêu cầu kinh nghiệm lập trình, chỉ cần hiểu về HTML và DOM. * Có thể xuất ra các kiểm thử có định dạng sử dụng được trong Selenium RC và WebDriver. * Có module built-in help và báo cáo kết quả kiểm thử | * Chỉ dùng được cho FireFox và Google Chrome. * Thực thi kiểm thử chậm hơn Selenium RC và WebDriver * Được thiết kế để tạo prototypes của kiểm thử. * Không hỗ trợ cho hoạt động lặp lại và điều kiện. |

Bảng 2.1.2‑1. Ưu – nhược điểm của Selenium IDE

#### Selenium WebDriver

thực hiện tiếp cẩn và ổn định hơn trong tự động hóa các hành động của trình duyệt. Nó không phụ thuộc vào JavaScripts cho việc kiểm thử tự động hóa và điều khiển trình duyệt bằng cách liên lạc trực tiếp với nó. Các ngôn ngữ được hỗ trợ như Selenium RC.

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Cài đặt đơn giản. * Giao tiếp và tương tác trực tiếp với trình duyệt. * Không yêu cầu các thành phần khác như RC Server. * Thời gian thực thi nhanh hơn Selenium IDE và RC | * Yêu cầu kiến thức lập trình. * Không có cơ chế built-in cho việc ghi lại các thông điệp trong thời gian chạy và sinh ra kết quả kiểm thử. * Không sẵn sang hỗ trợ các trình duyệt mới. |

Bảng 2.1.2‑2. Ưu – nhược điểm của Selenium Webdriver

#### Selenium RC

Là framework thử nghiệp hàng đầu của toàn bộ dự án Selenium trong một thời gian dài. Đây là công cụ kiểm tra web tự động đầu tiên cho phép người dung sử dụng ngôn ngữ lập trình mà họ thích. Selenium RC có thể hỗ trợ các ngôn ngữ lập trình như: Java, Python, PHP, C#, ….

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Chạy trên nhiều trình duyệt, hệ điều hành. * Hỗ trợ các hoạt động lặp lại và điều khiển. * Thực thi nhanh hơn Selenium IDE. * Có thể hỗ trợ data-driven testing. | * Cài đặt phức tạp. * Yêu cầu có Selenium RC Server. * Ít tương tác trình duyệt. * Thời gian thực thi chậm hơn WebDriver. * Phải có kiến thức lập trình. |

Bảng 2.1.2‑3 Ưu – nhược điểm của Selenium RC

#### Selenium Grid

Là một công cụ được sử dụng cùng với Selenium RC để chạy thử nghiệm song song trên các máy khác nhau và các trình duyệt khác nhau cùng một lúc. Tính năng của nó cho phép chạy đồng thời thử nghiệm trong nhiều trình duyệt và môi trường, tiết kiệm thời gian to lớn, sử dụng khái niệm hub-and-nodes.

## Tổng quan về Robot FrameWork.

### Robot Framework là gì?

Là một khung kiểm thử tự động hóa hướng từ khóa và hướng dữ liệu dựa trên Python, có thể mở rộng để kiểm thử chấp nhận, phát triển theo hướng kiểm thử chấp nhận (ATDD). Nó cung cấp mọi thứ cần thiết để xây dựng và phát triển một kịch bản kiểm thử, gồm điều kiện đầu vào/kết thúc, báo cáo kết quả, …. Các tester không cần quan tâm đến các thuật toán lập trình cơ bản nhất.

### Các tính năng nổi bật của Robot FrameWork.

* Giúp chúng ta thực hiện kiểm thử tự động với kịch bản ở dạng bảng một cách dễ dàng.
* Đưa ra kết quả thực thi các kịch bản kiểm thử và các log ở dạng html giúp chúng ta đọc và phân tích kết quả nhanh chóng và dễ dàng hơn.
* Có hỗ trợ chức năng đánh dấu các kịch bản kiểm thử, cho phép chúng ta lựa chọn kịch bản kiểm thử tiện lợi và nhanh chóng.
* Khả năng chạy trên nhiều khác nhau mà không cần chỉnh sửa kịch bản kiểm thử hay các từ khóa ở tầng dưới.

### Các thư viện hỗ trợ trong Robot Framework.

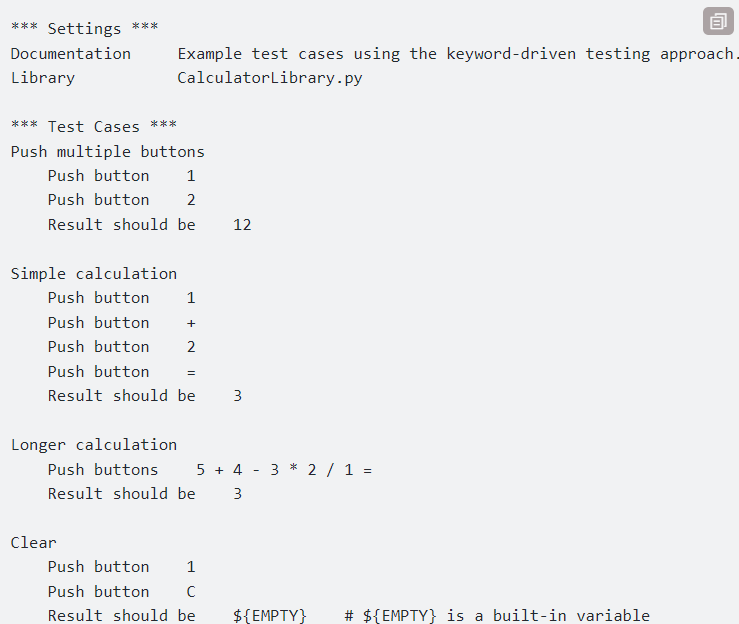
#### Selenium2Library

* Là ngôn ngữ rất sát với ngôn ngữ thực tế của người dùng, bạn mong muốn action gì chỉ cần gõ từ khó tương ứng.
* Được sử dụng để kiểm thử trên nền Web, được thừa hưởng từ SeleniumLibrary và được bổ sung để sử dụng Selenium 2 và WebDriver.
* Hoạt động ở hầu hết các trình duyệt như Safari, Chrome, …và trên cả Python.
* Để chạy được Selenium2Library cần phải cài đặt Robot FrameWork, và cài đặt bằng câu lệnh:

pip install robotframework-selenium2library

#### CalculatorLibrary

* Là thưu viện về tính toán đơn giản, nó chỉ chứa logic nghiệp vụ chứ không chỉ bao gồm phần UI.
* Để sử dụng CalculatorLibrary, đầu tiên phải tải các keywords đã được viết sẵn bản demo, sau đó giải nén ra sẽ được một số file trong đó bao gồm file thư viện.



Hình 2.2.3‑1 Ví dụ về thư viện CalculatorLibrary

## Tổng quan về Xpath.

### XPath là gì?

Là một ngôn ngữ được thiết kế ra với mục đích giúp cho ứng dụng có thể “di chuyển” bên trong các tệp XML để truy xuất các giá trị, thuộc tính của các element.

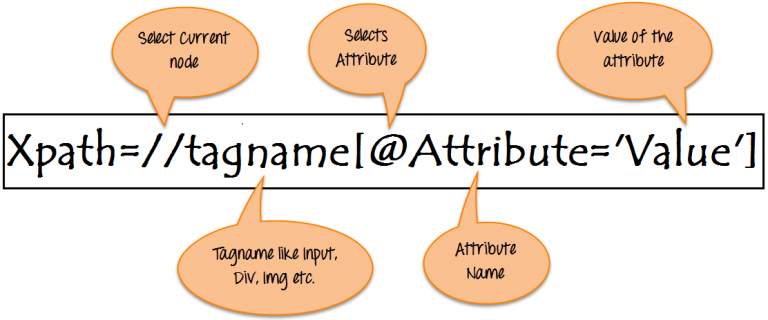
XPath trong Selenium là một đường dẫn XML được sử dụng để điều hướng thông qua cấu trúc HTML của trang. Có thể hiểu, XPath là một cú pháp hoặc ngôn ngữ để tìm kiểm bất kì element nào trên trang bằng cách sử dụng biểu thức đường dẫn XML hoặc sử dụng cấu trúc HTML DOM.

XPath có 2 loại:

* XPath tương đối/ XPath không ổn định: tìm kiếm các element ở bất kỳ đâu trên trang web và không cần một đường dẫn dài, đầy đủ.
* XPath tuyệt đối/ XPath ổn định: là cách trực tiếp và chính xác nhất để tìm ra vị trí của element.

### Cú pháp của XPath.

XPath chứa đường dẫn của element nằm trên trang web, sau đây là cấu trúc tiêu chuẩn để tạo XPath:



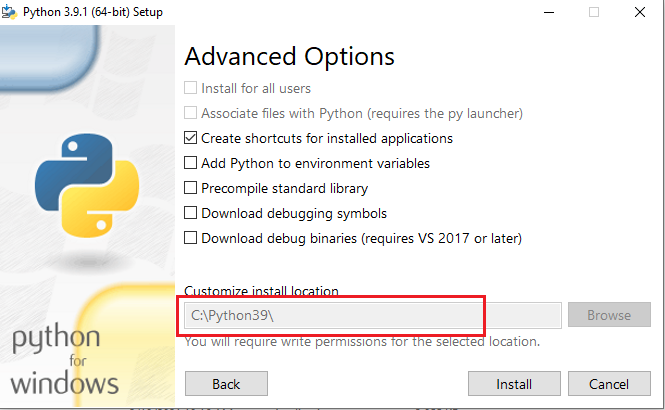
Hình 2.3.2‑1 Cú pháp của XPath

Để định vị được chính xác element, chúng ta cần phải có nhiều thông tin, yếu tố hay thuộc tính của element đó để xác định như: ID, Classname, Name, Link text, ….

## Cài đặt môi trường thực hiện kiểm thử tự động.

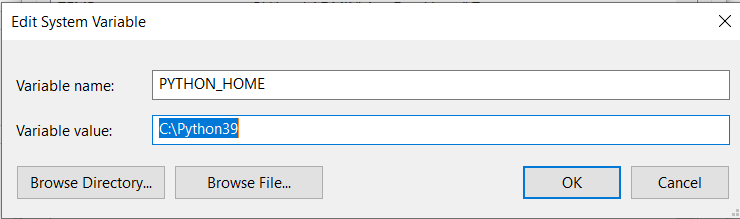
### Cài đặt Python.

Đây là bước bắt buộc cần có để sử dụng được Robot FrameWork. Cài đặt Python phiên bản 3.9.1 về máy tính, thư mục Python được lưu ở ổ đĩa C.



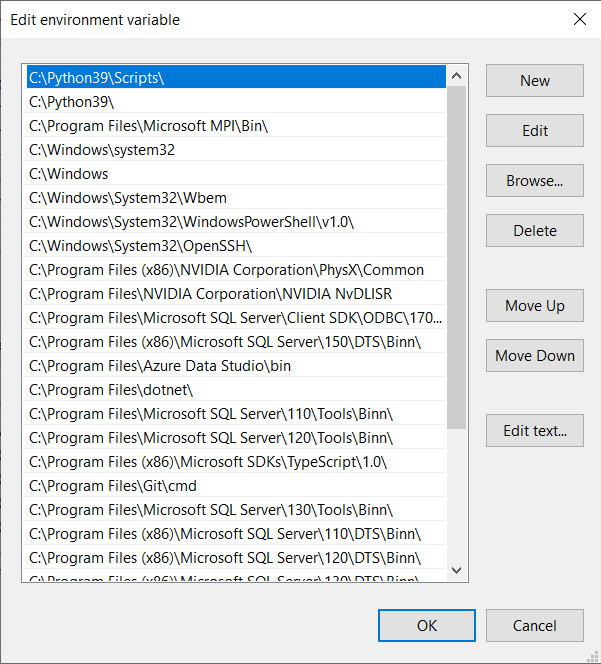
Hình 2.4.1‑1 Cài đặt Python 3.9.1

Thiết lập môi trường PYTHON\_HOME tại Advanced symtem setting.



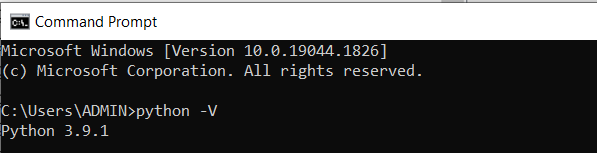
Hình 2.4.1‑2 Thiết lập PYTHON\_HOME

Thiết lập Path của Python tại Advanced symtem setting.



Hình 2.4.1‑3 Thiết lập Path

Kiểm tra Python đã cài đặt thành công.

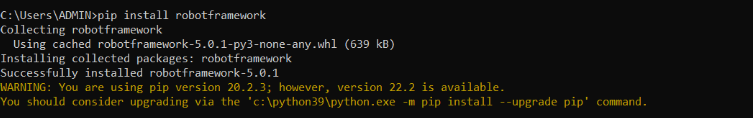


Hình 2.4.1‑4 Cài đặt Python thành công

### Cài đặt Robot FrameWork bằng pip.

Cài đặt Robot FrameWork sử dụng cmd trong C:\Python39\Script với lệnh:

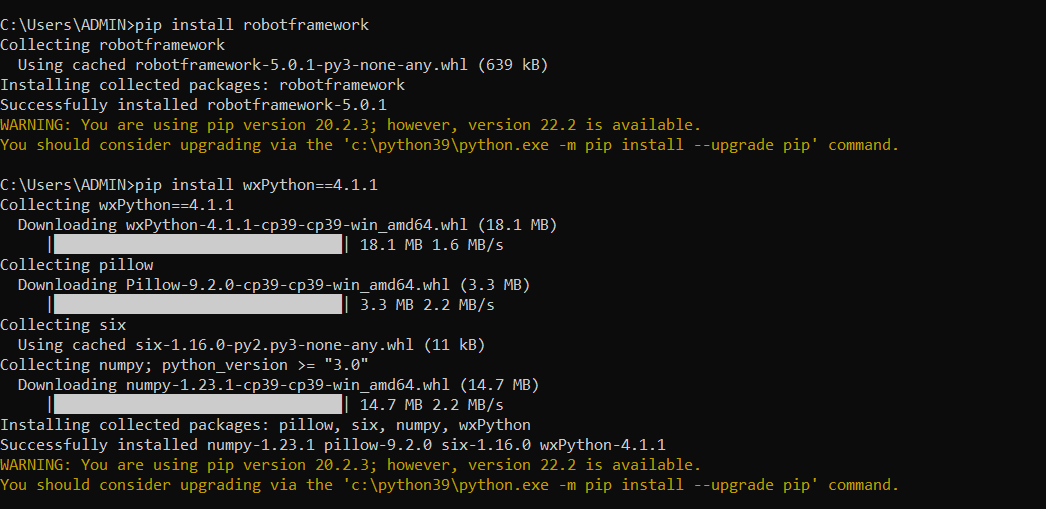
**pip install robot framework**

****

Hình 2.4.2‑1 Cài đặt robot framework thành công

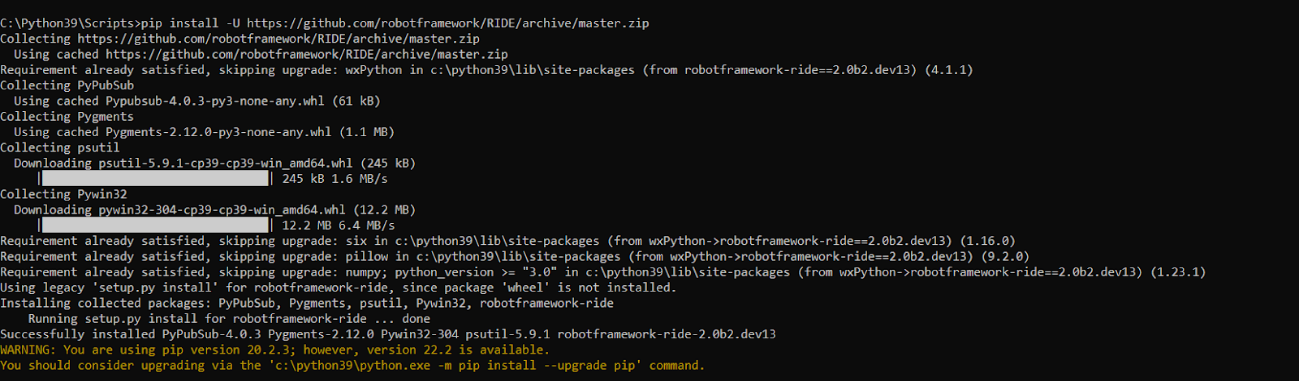
Cài đặt wxPython sử dụng cmd trong C:\Python39\Script với lệnh:

**pip install wxPython==4.1.1**



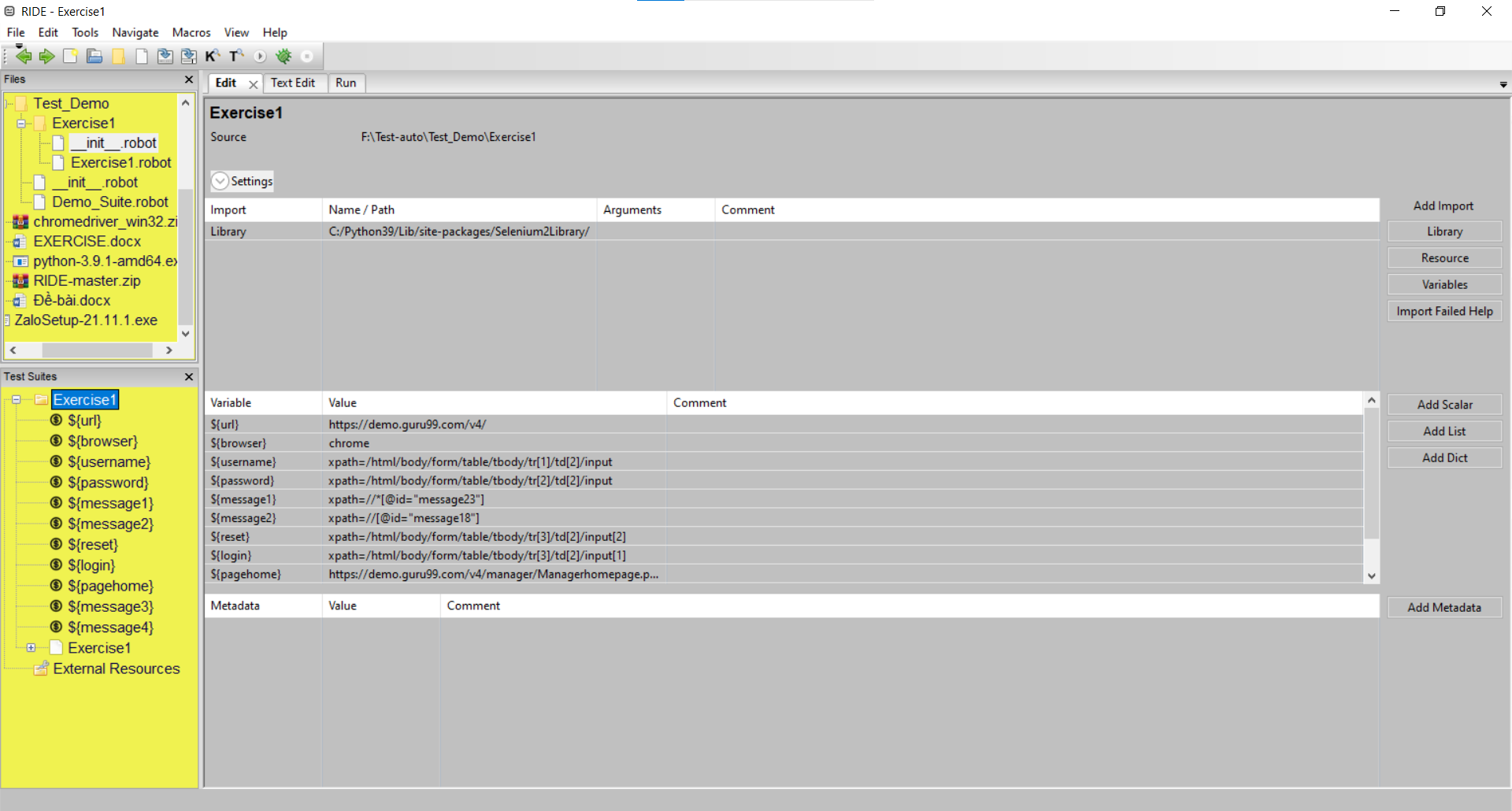
Hình 2.4.2‑2 Cài đặt wxPython==4.1.1 thành công

Cài đặt RIDE (tool editor) sử dụng cmd trong C:\Python39\Script với lệnh:



Hình 2.4.2‑3 Cài đặt RIDE thành công

Mở RIDE bằng file Ride.py trong C:\Python39\Script

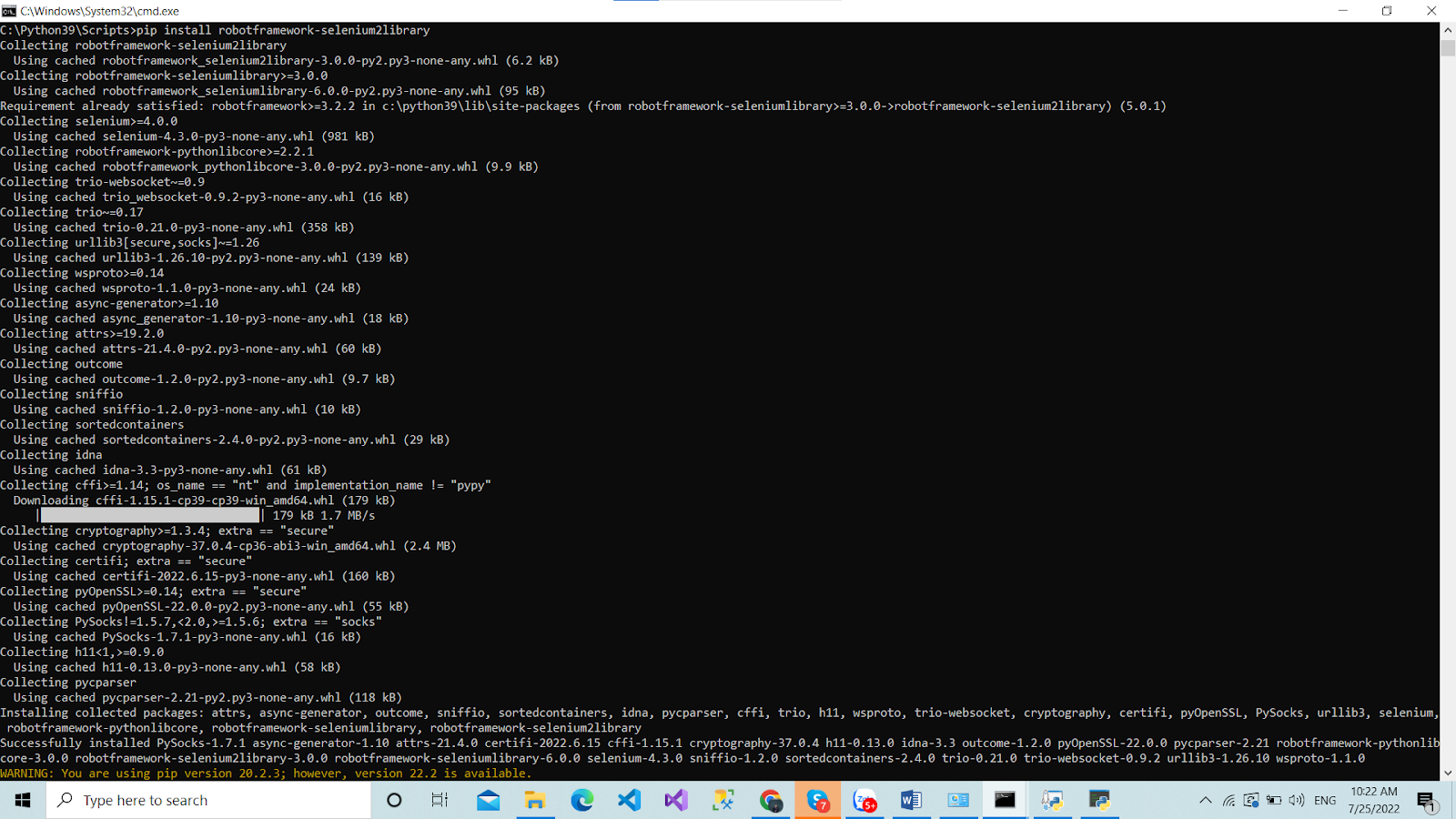


Hình 2.4.2‑4 Giao diện RIDE khi mở sau cài đặt thành công

### Cài đặt thư viện Selenium2Library.

Cài đặt Selenium2Library sử dụng cmd trong C:\Python39\Script với lệnh:

**pip install robotframework-selenium2library**

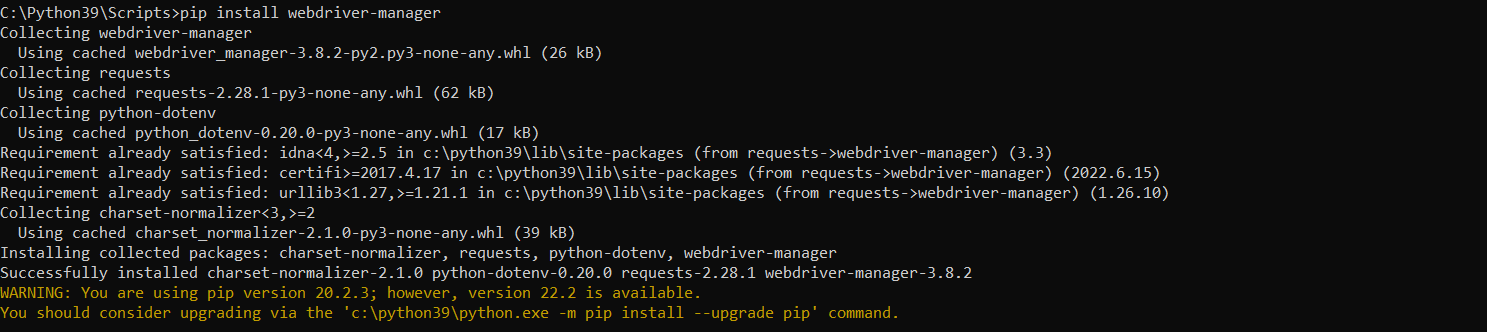


Hình 2.4.3‑1 Cài đặt Selenium2Library thành công

### Cài đặt Webdriver Manager.

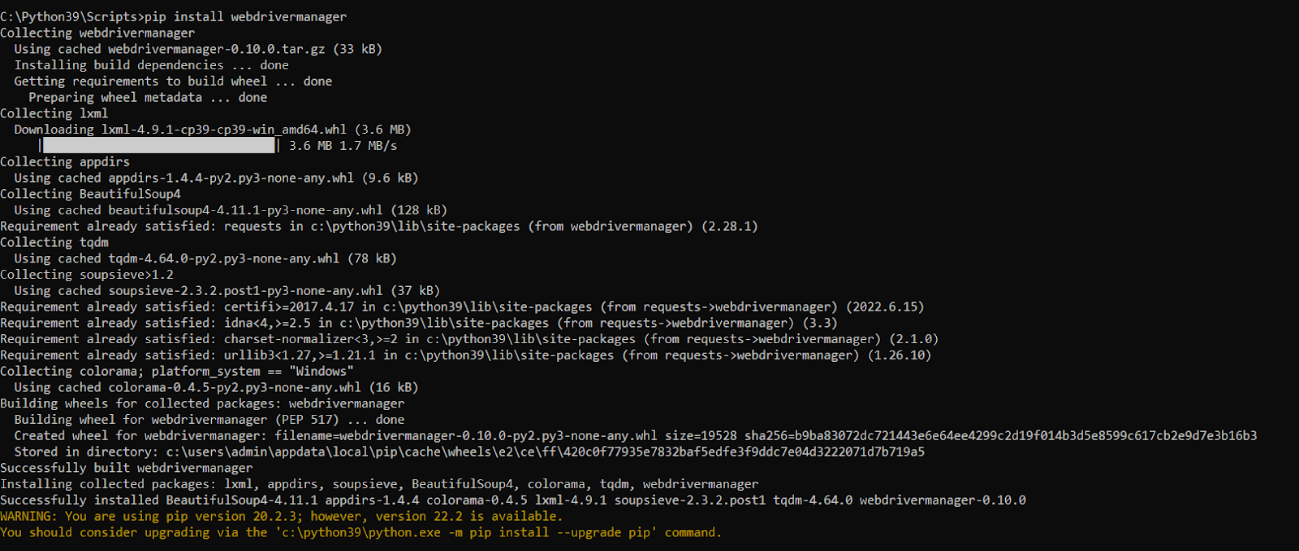
Cài đặt Webdriver Manager sử dụng cmd trong C:\Python39\Script với lần lượt từng lệnh:

**pip install webdriver-manager**



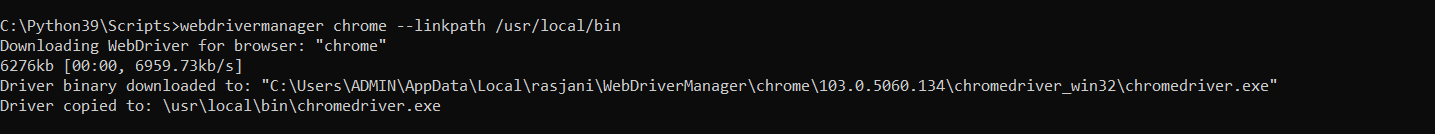
Hình 2.4.4‑1 Cài đặt Webdriver – Manager thành công

**pip install webdrivermanager**



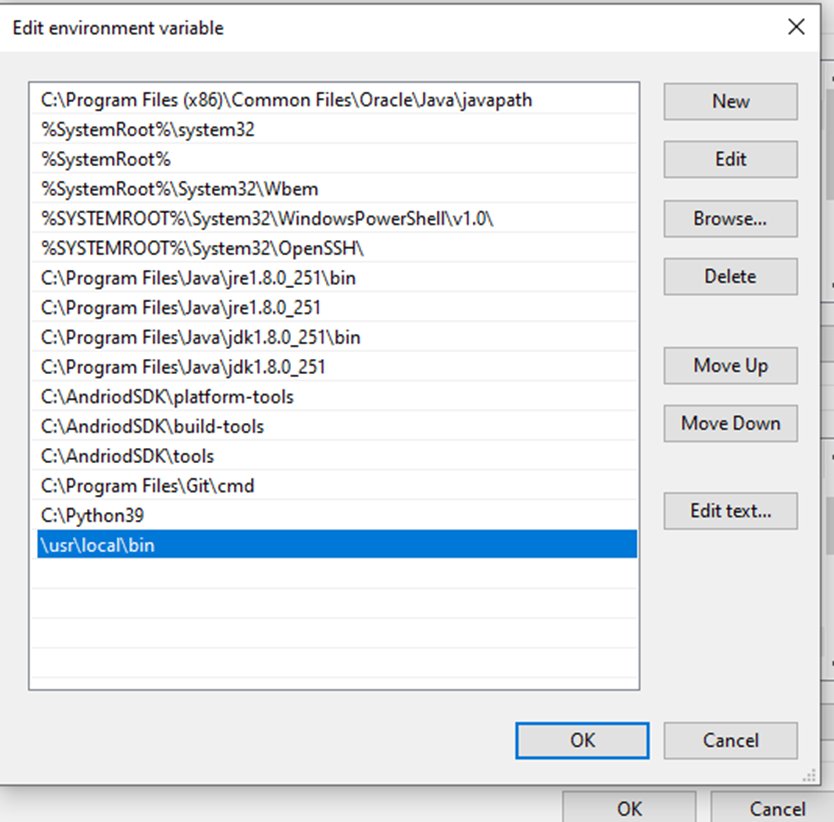
Hình 2.4.4‑2 Cài đặt webdrivermanager

**webdrivermanager chrome --linkpath /usr/local/bin**



Hình 2.4.4‑3 cài đặt webdrivermanager chrome thành công

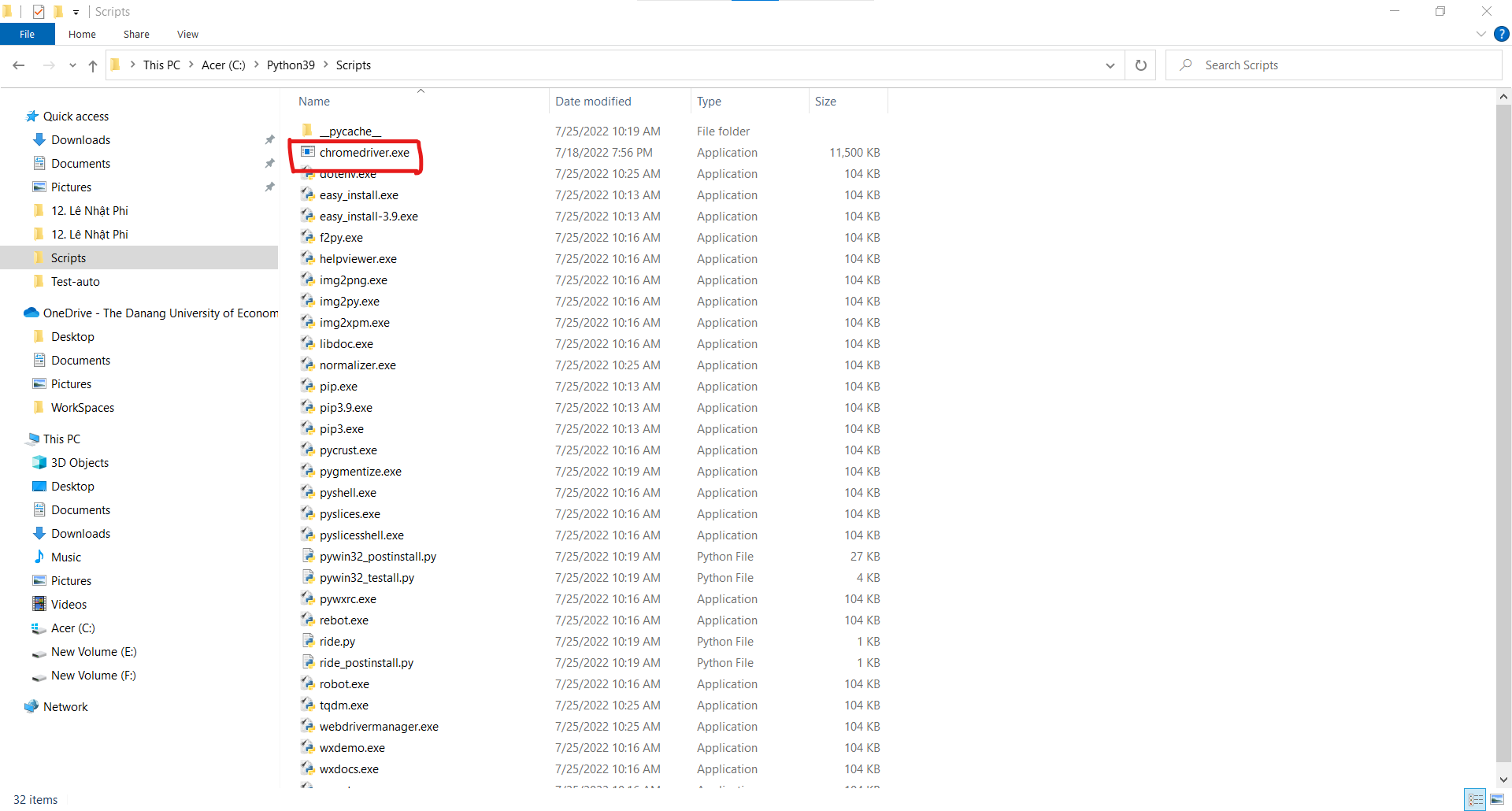
Thiết lập đường dẫn **\usr\local\bin**



Hình 2.4.4‑4 Thiết lập đường dẫn sau khi cài Webdrivermanager Chrome

### Cài đặt ChromeDriver

Tải ChromeDriver phù hợp với phiên bản Google Chrome ở máy tính và sao chép vào C:\Python39\Script



Hình 2.4.5‑1 Cài đặt Chromedriver

# TRIỂN KHAI KIỂM THỬ TỰ ĐỘNG ỨNG DỤNG WEB GURU99 TRÊN ROBOT FRAMEWORK

## Mục 3.1

### Mục 3.1.1

### Mục 3.1.2

## Mục 3.2

# KẾT QUẢ

## Mục 4.1…

## Mục 4.2…

Kết quả được xây dựng dựa trên ngôn ngữ lập trình thể hiện như (Bảng 4.1).

Bảng 3.1.2‑1 Kiến trúc

|  |  |  |
| --- | --- | --- |
| Lớp | Ý nghĩa | Tham số |
|  |  |  |
|  |  |  |
|  | | |

Hình 3.1.2‑1 Kiến trúc của mô hình

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Đề tài đã thực hiện được …

# TÀI LIỆU THAM KHẢO

* + - 1. Tên tác giả, Tên tác giả… *Tên công trình*. Nơi xuất bản, năm, trang
      2. Tên tác giả, Tên tác giả… *Tên công trình*. Nơi xuất bản, năm, trang

# PHỤ LỤC