**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC KINH TẾ**

🙣 🕮 🙡

****

**BÁO CÁO CHUYÊN ĐỀ**

**THỰC TẬP TỐT NGHIỆP**

**ỨNG DỤNG CÔNG CỤ SELENIUM**

**THỰC HIỆN KIỂM THỬ WEBSITE ĐẶT VÉ TÀU**

**SAFE RAILWAY**

SVTH: Trần Thị Diệu Linh

Lớp: 44K14

GVHD: ThS. Cao Thị Nhâm

***Đà Nẵng, 02/2022***

LỜI CẢM ƠN

Lời đầu tiên em xin chân thành cảm ơn cô Cao Thị Nhâm - giáo viên đã hướng dẫn, hỗ trợ và giúp đỡ em trong suốt quá trình thực hiện đề tài.

Em xin chân thành cảm ơn tất cả quý thầy cô khoa Thống kê – Tin học cùng các thầy cô trường Đai học Kinh tế – Đại học Đà Nẵng đã tận tình giảng dạy, truyền cảm hứng cho chúng em trong suốt bốn năm qua.

Cuối cùng em xin chân thành LogiGear Việt Nam – chi nhánh Đà Nẵng đã tạo điều kiện tốt nhất để em hoàn thành đề tài tốt nghiệp của mình.

Em xin chân thành cảm ơn!

Sinh viên

Trần Thị Diệu Linh

LỜI CAM ĐOAN

*Tôi xin cam đoan đề tài “Ứng dụng công cụ Selenium thực hiện kiểm thử tự động website đặt vé tàu Safe Railway” tại công ty LogiGear Việt Nam – chi nhánh Đà Nẵng là bài làm của riêng cá nhân tôi.  
Tôi xin hoàn toàn chịu trách nhiệm về tính trung thực của đề tài.*

*Sinh viên thực hiện*

**Trần Thị Diệu Linh**

MỤC LỤC

[PHẦN MỞ ĐẦU 1](#_Toc96808326)

[I. SỰ CẦN THIẾT CỦA ĐỀ TÀI 1](#_Toc96808327)

[II. NỘI DUNG CỦA ĐỀ TÀI 3](#_Toc96808328)

[III. PHƯƠNG PHÁP THỰC HIỆN 3](#_Toc96808329)

[IV. BỐ CỤC CỦA BÁO CÁO 3](#_Toc96808330)

[CHƯƠNG 1: GIỚI THIỆU CÔNG TY LOGIGEAR 4](#_Toc96808331)

[1.1. Giới thiệu chung: 4](#_Toc96808332)

[1.2. Lĩnh vực hoạt động: 4](#_Toc96808333)

[1.3. Sứ mệnh: 5](#_Toc96808334)

[1.4. Chi nhánh: 5](#_Toc96808335)

[1.5. Kênh truyền thông chính thức: 6](#_Toc96808336)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 8](#_Toc96808337)

[2.1. Tổng quan về kiểm thử phần mềm: 8](#_Toc96808338)

[2.1.1 Phần mềm là gì? 8](#_Toc96808339)

[2.1.2 Vòng đời phát triển phần mềm và kiểm thử: 8](#_Toc96808340)

[2.1.2.1. Vòng đời phát triển phần mềm: 8](#_Toc96808341)

[2.1.2.2. Cách kiểm thử phần mềm bị ảnh hưởng như thế nào theo từng SDLC được chọn? 8](#_Toc96808342)

[2.1.3 Các giai đoạn kiểm thử và Milestones: 8](#_Toc96808343)

[2.1.4 Các nhóm kiểm thử: 8](#_Toc96808344)

[2.1.5 Mục tiêu của kiểm thử phần mềm: 8](#_Toc96808345)

[2.1.6 Test Coverage: 8](#_Toc96808346)

[2.1.7 Phương thức kiểm thử: 8](#_Toc96808347)

[2.2. Thiết kế các trường hợp kiểm thử và tìm kiếm bugs: 8](#_Toc96808348)

[2.2.1. Test Requirement: 8](#_Toc96808349)

[2.2.2. Phương pháp kiểm thử và kỹ thuật thiết kế các trường hợp kiểm thử 8](#_Toc96808350)

[2.2.3. Lỗi phần mềm: 8](#_Toc96808351)

[2.3. Kiểm thử thủ công: 8](#_Toc96808352)

[2.4. Kiểm thử tự động: 8](#_Toc96808353)

[2.4.1. Kiểm thử tự động là gì? 8](#_Toc96808354)

[2.4.2. Những lợi ích và cạm bẫy của kiểm thử tự động: 8](#_Toc96808355)

[2.4.3. Sử dụng hiệu quả kiểm thử tự động: 8](#_Toc96808356)

[2.4.4. Sự phát triển của phương pháp tự động hóa kiểm thử: 8](#_Toc96808357)

[2.4.5. Test Design cho kiểm thử tự động: 8](#_Toc96808358)

[2.4.5.1. Mục tiêu của Test Design: 8](#_Toc96808359)

[2.4.5.2. Thiết kế Test Module: 8](#_Toc96808360)

[2.4.5.3. Action Design: 8](#_Toc96808361)

[CHƯƠNG 3: CÔNG CỤ SELENIUM 10](#_Toc96808362)

[3.1. Selenium là gì? 10](#_Toc96808363)

[3.2. Định vị User Interface Elements: 10](#_Toc96808364)

[3.3. Page Object Design Pattern 10](#_Toc96808365)

[3.4. Các khái niệm chung về Test Automation Framework: 10](#_Toc96808366)

[CHƯƠNG 4: PHÂN TÍCH HỆ THỐNG 11](#_Toc96808367)

[4.1. Tổng quan về hệ thống: 11](#_Toc96808368)

[4.2. Phân tích yêu cầu: 11](#_Toc96808369)

[CHƯƠNG 5: THIẾT KẾ TESTCASES VÀ THỰC HIỆN KIỂM THỬ 11](#_Toc96808370)

[TÀI LIỆU THAM KHẢO 12](#_Toc96808371)

[PHỤC LỤC 13](#_Toc96808372)

DANH SÁCH HÌNH ẢNH

DANH SÁCH TỪ VIẾT TẮT

|  |  |
| --- | --- |
| **Từ viết tắt** | **Diễn giải** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# PHẦN MỞ ĐẦU

## SỰ CẦN THIẾT CỦA ĐỀ TÀI

Sau đây là những tai nạn đáng tiếc gây liên lụy nhiều người, và thiệt hại hàng tỷ đô la đã xảy ra trong lịch sử để thấy được tầm quan trọng của việc kiểm thử phần mềm:

Thứ nhất, vào tháng 4 năm 2015, Bloomberg ở London đã gặp sự cố do trục trặc phần mềm ảnh hưởng đến hơn 300.000 thương nhân trên thị trường tài chính. Nó buộc chính phủ phải hoãn việc bán nợ 3 tỷ bảng.  
  
Thứ hai, Nissan đã phải triệu hồi hơn 1 triệu xe khỏi thị trường do lỗi phần mềm trong các máy dò cảm ứng túi khí. Đã có báo cáo hai tai nạn do lỗi phần mềm này.  
  
Thứ ba, Starbucks đã buộc phải đóng cửa khoảng 60% các cửa hàng ở Mỹ và Canada do lỗi phần mềm trong hệ thống POS của mình vì họ không thể xử lý giao dịch.  
  
Thứ tư, một số nhà bán lẻ bên thứ ba của Amazon, thấy giá sản phẩm của họ giảm xuống còn 1p do trục trặc phần mềm. Họ đã bị tổn thất nặng nề  
  
Thứ năm, lỗ hổng trong Window 10. Lỗi này cho phép người dùng thoát khỏi các hộp bảo mật sandbox thông qua một lỗ hổng trong hệ thống win32k.  
  
Thứ sáu, năm 2015, máy bay chiến đấu F-35 trở thành nạn nhân của lỗi phần mềm, lỗi xảy ra khiến nó không thể phát hiện mục tiêu chính xác.  
  
Thứ bảy, máy bay Air300 A300 của China Airlines bị rơi do lỗi phần mềm vào ngày 26 tháng 4 năm 1994, vụ tai nạn đã cướp đi mạng sống của 264 người vô tội  
  
Thứ tám, năm 1985, máy xạ trị Therac-25 của Canada bị trục trặc do lỗi phần mềm và đã truyền liều gây chết người cho bệnh nhân, khiến 3 người chết và 3 người khác bị thương nặng.  
  
Thứ chín, vào tháng Tư năm 1999, một lỗi phần mềm gây ra sự thất bại của vụ phóng vệ tinh quân sự trị giá 1,2 tỷ đô la, tai nạn tốn kém nhất trong lịch sử.  
  
Thứ mười, vào tháng 5 năm 1996, một lỗi phần mềm đã khiến tài khoản ngân hàng của 823 khách hàng của một ngân hàng lớn ở Hoa Kỳ được ghi có 920 triệu đô la Mỹ.  
  
Thứ mười một, gần đây, lỗi phần mềm đã khiến rất nhiều hệ thống xảy ra vấn đề dữ liệu bị mất cắp như Facebook, thegioididong,...  
  
Thứ mười hai, trong tháng 9/2018, vụ việc Zalo và các dịch vụ khác của VNG cũng gặp sự cố không thể truy cập được.

Và ngoài mười hai tai nạn đáng tiếc nói trên, thì còn có năm vụ lỗi phần mềm đã đi vào lịch sử.

Thứ nhất, sự kiện đầu tiên sinh ra bug. Vào ngày 9/9/1949, vì một lý do nào đó mà một máy tính Mark II tại trường Đại học Havard đã không hoạt động. Sau khi kiểm tra, các kỹ sư đã phát hiện ra nguyên nhân là một con bướm đã mắc kẹt vào bên trong máy tính, vào lúc 3 giờ 45 phút chiều sự cố này được ghi nhận, trong đó có đoạn: “Con bọ (bug) thực thế đầu tiên được tìm thấy”.

Thứ hai, sự kiện Y2K. Các nhà phát triển phần mền đã không bao giờ nghĩ rằng phần mềm của họ có thể tồn tạo qua năm 1999. Nhiều lập trình viên cho rằng viết “19” trước biến “năm” là không cần thiết, chính vì thế hầu hết các biến “năm” chỉ được lưu bằng hai số. Ví dụ 1999 sẽ lưu thành 99. Do đó khi sắp bước sang năm 2000, các phần mềm sẽ cập nhật về năm 1900 thay vì năm 2000. Chính vì thế mà nhiều người đã lo lắng rằng tên lửa hạt nhân sẽ được phóng ra, máy bay chở hàng trăm hành khách sẽ rơi...Nhưng điều đó đã không xảy ra vì chúng ta vẫn ở đây. Tuy nhiên, đã có hàng tỷ đô la được chi ra để tiến hành nâng cấp các phần mềm và thực tế đã ghi nhận: ở Tây Ban Nha, một trang web về thời tiết đã chuyển sang ngày 1-1-1900 và ở Úc, một số máy bán vé xe buýt đã không hoạt động.

Thứ ba, vụ tên lửa Dhahran làm chết 28 lính Mỹ. Vào tháng 2/1991, trong cuộc Chiến tranh vùng Vịnh (hay còn được gọi là Chiến tranh vịnh Ba Tư hay là Chiến dịch Bão Táp Sa Mạc là cuộc xung đột giữa Iraq và liên minh gần 30 quốc gia) một tên lửa của Iraq đã đi lọt vào căn cứ Mỹ ở Dhahran làm chết 28 lính Mỹ. Sự việc này sau đó được xác minh là nguyên do tính toán giờ của hệ thống phòng thủ tên lửa bị chậm do không tính toán đến sai số do pin gây ra. Sau mỗi giờ thì đồng hồ lại sai 1 mili giây và sau 100 giờ thì đã sai 0,33 giây. Với 0,33 giây thì radar có thể phát hiện được tên lửa nhưng hệ thống phòng thủ không kịp khai hỏa. Và hậu quả dấn tới là 28 lính Mỹ thiệt mạng.

Thứ tư, sai số hơn 100km do lập trình theo mét nhưng điều khiển lại dùng đơn vị feet. Năm 1998, NASA phóng con tàu Mars Climate Orbiter để thăm dò khí hậu sao hỏa. Nhưng nó vĩnh viễn không thực hiện thành công nhiệm vụ của mình vì: Người điều khiển mặt đất sử dụng đơn vị feet trong khi phần mềm lập trình chỉ sử dụng đơn vị mét. Việc này dẫn tới sai số quỹ đạo hơn 100 km. Dẫn đến việc bay quá gần khí quyển sao hỏa, và Mars Climate Orbiter đã biến thành bụi vũ trụ.

Thứ năm, thiệt hại 370 triệu USD do lỗi phần mềm chuyển từ 64 bit sang 16 bit. Vào ngày 4/6/1996, chỉ sau 30 giây được phóng lên, tên lửa Ariane 5 đã nổ tung như pháo hoa. Sự cố này sau đó được mô phỏng lại với một điều kiện tượng nhằm tìm ra nguyên nhâ đã cho thấy phần mềm máy tính đã chuyển dữ liệu sử dụng biến kiểu số động 64-bit sang biến sử dụng số 16-bít để biểu diễn phương nằm ngang. Những biến này đã sử dụng một kích thước khác trong bộ nhớ là nguyên nhân hàng loạt lỗi ảnh hưởng đến tất cả máy tính trong tên lửa làm tê liệt hệ thống và đã kích hoạt hệ thống tự phá hủy. Sự cố này đã làm bay màu 370 triệu USD chỉ trong tích tắc.

Qua 12 vụ tai nạn và 5 vụ lỗi phần mềm đi vào lịch sử được nhắc tới kể trên thì chúng ta đều thấy dù chỉ là một sai sót rất nhỏ đều có thể dẫn đến một hậu quả nặng nề vượt qua sức tưởng tượng của con người, nó không chỉ ảnh hưởng đến thời gian, tiền bạc mà thậm chí còn gây chết người. Từ đó chúng ta giúp chúng ta nhận ra sự quan trọng của khâu kiểm thử trong quá trình phát triển dự án.

Qua đề tài “Ứng dụng công cụ Selenium thực hiện kiểm thử website đặt vé tàu Safe Railway”, em muốn đưa ra cái nhìn tổng quan hơn về kiểm thử phần mềm, đặt biệt là áp dụng việc kiểm thử tự động trong quá trình kiểm tra. Ngoài ra còn làm rõ ra được những tốt và chưa tốt khi kiểm thử tự động được thực hiện.

## NỘI DUNG CỦA ĐỀ TÀI

* Nghiên cứu các giai đoạn trong kiểm thử phần mềm, các hoạt động cần thiết trong từng giao đoạn và các kỹ thuật áp dụng cho việc kiểm thử phần mềm.
* Mô tả phương pháp kiểm thử trong thực tế cho một vài sản phẩm
* Nghiên cứu công cụ thực hiện kiểm thử website tự động Selenium
* Ứng dụng các phương pháp đã được học để thực hiện kiểm thử tự động.

## PHƯƠNG PHÁP THỰC HIỆN

* Tìm hiểu về quy trình và các kĩ thuật kiểm thử phần mềm
* Tìm hiểu về công cụ Selenium
* Phân tích hệ thống Safe Railway
* Viết test plan, test cases cho từng chức năng có trong Safe Railway
* Thực hiện kiểm thử hệ thống Safe Railway

## BỐ CỤC CỦA BÁO CÁO

Bài báo cáo bao gồm 5 chương và 4 phần với các nội dung như sau:

**Phần mở đầu**

**Chương 1: Giới thiệu về công ty Logigear**

**Chương 2: Cơ sở lý thuyết**

**Chương 3: Giới thiệu về công cụ Selenium**

**Chương 4: Phân tích hệ thống**

**Chương 5: Thiết kế testcases và thực hiện kiểm thử**

**Phần kết luận và hướng phát triển**

**Tài liệu tham khảo**

**Phụ lục**

# CHƯƠNG 1: GIỚI THIỆU CÔNG TY LOGIGEAR

## Giới thiệu chung:

LogiGear được thành lập tại Thung lũng Silicon vào năm 1994 bởi các nhà lãnh đạo tư tưởng hàng đầu về Kiểm thử và Phát triển Phần mềm. LogiGear đã hoàn thành hơn 3.000 dự án cho hàng trăm công ty trên toàn thế giới. Khách hàng của Logigear đa dạng về lĩnh vực cũng như là tầm vóc:

Business Services: Penske Logistics, EBSCO, RealEC technologies,…

Computer and Networking Equiments: Cisco, Philip, Commscope,…

Consumer Electronics, Games, Entertainments: Disney, Perfect World, Sonos,…

eCommerce, Retail, Distrubition: Mersedes-BEN, World Market, Navtech,…

Education and Publishing: Children’s Progress, Cnet, Kaplan,…

Enterprise Software: Adobe, Energy Hub, Sprint,…

Financial Services: VISA, Fidelity, Sumimoto Bank,…

Government and Legal: New York State and Department of Transportation, Internal, Revenue Service, BAL,…

Healthcare and Pharmaceutial: Pfizer, Standford hospital and clinics, Imprivata,..

Insurance: Singapore life, Auda Explore, Allstate,…

Interactive Agencies: Barefoot, Hard Associates Inc, Level,...

Life science industry: AB applied biosystems, DNA Script,…

Construction: Taylor Morrison.

LogiGear Group bao gồm LogiGear, MK Partners và TPP Technology. Được thành lập vào năm 2021, LogiGear Group có thể cung cấp các giải pháp chuyển đổi kỹ thuật số và kinh doanh đầy đủ. Từ kỹ thuật phần mềm kỹ thuật số, dịch vụ tư vấn và triển khai Salesforce đến xác thực cấp độ hệ thống đầu cuối đầy đủ, Kiểm tra phần mềm và dịch vụ QA, có thể hỗ trợ bạn trong tất cả các khía cạnh của việc triển khai Chuyển đổi kỹ thuật số.

LogiGear Việt Nam được thành lập vào năm 2005, có ba chi nhánh ở Huế, Đà Nẵng và Thành phố Hồ Chí. LogiGear đã định vị mình là một trong những công ty phát triển và kiểm thử phần mềm lớn nhất tại Việt Nam.

## Lĩnh vực hoạt động:

Hiên nay, LogiGear đang hoạt động ở 4 lĩnh vực:

* Software Development:
* Custom Software Development Services
* FullStack Development
* Quality Assurance:
* Offshore and Onshore QA
* QA Strategy and Consulting
* QA training and Curriculum
* Test Automation
* Salesforce Consulting Services:
* Salesforce Development Support
* Business Operations:
* Robotic Process Automation

## Sứ mệnh:

“LogiGear Corporation cung cấp các công nghệ và chuyên môn Kiểm tra phần mềm tiên tiến hàng đầu, cùng với các dịch vụ Phát triển Phần mềm cho phép khách hàng của chúng tôi đẩy nhanh tốc độ phát triển kinh doanh trong khi vẫn tin tưởng vào phần mềm mà họ cung cấp.”

## Chi nhánh:

* LogiGear USA – trụ sở chính
* Địa chỉ: 1730 S. Amphlett Blvd. Suite 200, San Mateo, CA 94402
* Số điện thoại:
* +1(800) 322-033 (Sales)
* +1(650) 572-1400 (Support)
* Fax: +1 650 572 2822
* LogiGear USA – Seattle
* 600 Stewart Street Suite 400, Seattle WA 98101
* LogiGear Việt Nam - Sài Gòn
* Địa chỉ: 462 Phan Xích Long, phường 2, quận Phú Nhuận, TP.Hồ Chí Minh
* Số điện thoại: +84 28 3995 4072
* Fax: + 84 28 3995 4076
* LogiGear Việt Nam – Đà Nẵng
* Địa chỉ: 346 đường 2/9, quận Hải Châu, Đà Nẵng
* Số điện thoại: +84 236 3655 333
* Fax: +84 28 3995 4076
* LogiGear Việt Nam – Huế
* Địa chỉ: 53 Nguyễn Huệ, quận Vĩnh Ninh, Thừa Thiên Huế
* LogiGear Japan – Tokyo
* Địa chỉ: Hatsudai Center Bldg, 1-51-1, Hatsudai, Shibuya, Tokyo 151-0061
* Số điện thoại: +81-3-5400-8702

## Kênh truyền thông chính thức:

* Website: [www.logigear.com](http://www.logigear.com)
* Facebook: www.facebook.com/logigearcorporation
* LogiGear Việt Nam
* Đà Nẵng: [www.facebook.com/logigeardanang](http://www.facebook.com/logigeardanang)
* Thành phố Hồ Chí Minh: [www.facebook.com/logigearhcmc](http://www.facebook.com/logigearhcmc)

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1. Tổng quan về kiểm thử phần mềm:

### Phần mềm là gì?

### Vòng đời phát triển phần mềm và kiểm thử:

#### 2.1.2.1. Vòng đời phát triển phần mềm:

#### 2.1.2.2. Cách kiểm thử phần mềm bị ảnh hưởng như thế nào theo từng SDLC được chọn?

### Các giai đoạn kiểm thử và Milestones:

### Các nhóm kiểm thử:

### Mục tiêu của kiểm thử phần mềm:

### Test Coverage:

### Phương thức kiểm thử:

## 2.2. Thiết kế các trường hợp kiểm thử và tìm kiếm bugs:

### 2.2.1. Test Requirement:

### 2.2.2. Phương pháp kiểm thử và kỹ thuật thiết kế các trường hợp kiểm thử

### 2.2.3. Lỗi phần mềm:

## 2.3. Kiểm thử thủ công:

## 2.4. Kiểm thử tự động:

### 2.4.1. Kiểm thử tự động là gì?

### 2.4.2. Những lợi ích và cạm bẫy của kiểm thử tự động:

### 2.4.3. Sử dụng hiệu quả kiểm thử tự động:

### 2.4.4. Sự phát triển của phương pháp tự động hóa kiểm thử:

### 2.4.5. Test Design cho kiểm thử tự động:

#### 2.4.5.1. Mục tiêu của Test Design:

#### 2.4.5.2. Thiết kế Test Module:

#### 2.4.5.3. Action Design:

# CHƯƠNG 3: CÔNG CỤ SELENIUM

## 3.1. Selenium là gì?

## 3.2. Định vị User Interface Elements:

## 3.3. Page Object Design Pattern

## 3.4. Các khái niệm chung về Test Automation Framework:

# CHƯƠNG 4: PHÂN TÍCH HỆ THỐNG

## 4.1. Tổng quan về hệ thống:

## 4.2. Phân tích yêu cầu:

# CHƯƠNG 5: THIẾT KẾ TESTCASES VÀ THỰC HIỆN KIỂM THỬ

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

# TÀI LIỆU THAM KHẢO

# PHỤC LỤC