

TECHNICAL ANALYSIS OF SELENIUM AND CYPRESS AS FUNCTIONAL AUTOMATION FRAMEWORK FOR MODERN WEB APPLICATION TESTING

Fatini Mobaraya and Shahid Ali

Department of Information Technology, AGI Institute, Auckland,
New Zealand

ABSTRACT

Automation testing has become increasingly needed due to the nature of the current software development project which comprises of complex application with shorter development time. Most of the companies in the industry have used Selenium extensively as functional automation tool to verify their web application's functionalities are working as expected. However, the limitation in Selenium with wait Time has significantly affect its test execution and efficiency. Thus, this research project experimenting a new automation tool in the market, Cypress, to overcome the said limitation in Selenium. This research further compares the test execution results in Selenium and Cypress to observe each tool's effectiveness in writing and executing the automation test script. The study results will be helpful towards determining a better tool in automating dynamic modern web application and providing an insight into Cypress as the future of automation testing tool.

KEYWORDS

Automation Testing, Regression Test Suite, Selenium, Cypress, JavaScript Automation Framework

1. INTRODUCTION

In the current world of technology, where every information is just a click away, web application has gained popularity since every information including the retail shops and the government infrastructures are being digitized. Web applications are becoming more complex by having multiple rich features and dynamic rendering to give a rich user experience. Due to this nature, there is a significant challenge in testing modern web application to satisfy end user's expectations.

With a complex web application and shorter product release time, automation testing came into picture. Automation testing is the uses a tool to replicate the behaviours of a real user by executing test autonomously with the goal to reduce execution time and increase test coverage.

Testing contributes 30-60% of software life cycles with a bigger percentage goes to the more complex and critical products [1]. With a shorter development time and complex web pages, modern web testing faced major development time to release with the need for fast test execution. According to [2], 29 out of 40 e-commerce sites are facing failure when being accessed by end-users.

To counter this issue, most organizations in the industry used Selenium to automate their application for functional requirements validation. However, as Selenium was developed back in 2005, the websites existed back then were much simpler than the current 2019 websites. Selenium main limitation is the difficulty in handling the current dynamic web elements, which significantly reduce test execution performance and resulting in a flaky test execution.

Thus, this research proposes Cypress as the automation framework to cater the current dynamic web applications. The main advantage of using Cypress is it simplifies asynchronous testing. Cypress defines an automatic wait in their framework where it waits for the DOM elements to finish loading or any animation to complete rendering, only then it will start looking for the web element. This is the main limitation in Selenium where automation developers need to define implicit and explicit wait to wait for the page finished loading. Testers can add waitTime or thread.sleep commands to counter this, however it will significantly affect the test execution performance.

The only drawback Cypress have is that they mainly support Chrome browser. However, according to Google Trend, as today's population is mainly using Chrome browser 90% of the time, the limitation of browser support would be a negligible drawback at this stage.

This research automates a dynamic web application – AliExpress. AliExpress is chosen as it contains various dynamic elements and rendering, as well as being accessed by over a million users worldwide, which proves it to be a reliable site.

The goals of this research are as follows:

- i. To create an automation script using Selenium and Cypress.
- ii. To develop a regression automation suites for AliExpress's main business flow which are customer's account and order checkout flow.
- iii. To compare the test execution time between Selenium and Cypress.
- iv. To compare the test efficiency and test coverage between Selenium and Cypress.

This research paper is organized as follow: Section 2 focuses on the literature review of various studies focusing on Selenium and Cypress. Section 3 is focused on the research methodology for this research. Section 4 of this research is focused on project execution. The comparative analysis of regression suite in Selenium and Cypress results are provided in section 5. Discussion to results and research findings are provided in section 6. Section 7 is dedicated towards the future work recommendations. Finally, in section 8 conclusion to the research is provided.

2. LITERATURE REVIEW

In the past, a lot of researches have been conducted to improve web application test execution by using test automation technique. One of the most common automation tools used in the past researches is Selenium. Selenium is said to be a trusted and robust automation tool due to its long existence in the market dated back to 10 years ago. However, according to [3], [4], [5] and recent studies with automation developers, Selenium possess a significant limitation in handling dynamic elements rendering, page loads and pop-up windows in the current modern web application.

This issue has not been addressed properly in the past and it affects Selenium script performance and reliability as mentioned by [6]. Thus, this research project will propose a

better way of executing automation testing by deploying Cypress automation framework with Test Driven Development (TDD) approach and Page Object Model (POM) design pattern.

TDD is one of the automation testing techniques. It focuses on test-first where developers will write automated tests followed by functional code. Based on recent studies by [7] and [8], TDD implementation will produce a simple code, increase test coverage as well as its clarity and maintainability. Furthermore, [8] described that TDD improves test execution by 21% and decrease automation code complexity by 31%.

Meanwhile, POM is a high-level abstraction that separates web pages from the test cases to encourage code reusability. It reduces coupling dependency between test cases and web pages which allows them to be independent of each other and easier for reuse in other parts of the coding. Moreover, test cases are easier to write with POM implementation [3].

The combination of TDD approach and POM design pattern will produce a maintainable test scripts which will reduce cost of maintenance later on. A clean code is necessary in automation scripts as repairing a poor code implementation will cause a huge amount of resources, both in time and energy [9].

On the other hand, test metrics which are used in this research are test execution time, test efficiency based on code effort and requirement-based test coverage. These metrics will be used in this research based on the research works of [10], [11] and [12]. These three test metrics are considered to be the main measurement that will inform us the progress of the test execution. When these three metrics are properly monitored, the high quality of the Application Under Test (AUT) can be maintained.

Besides, agile methodology will be adopted to this research project. Agile is defined as an iterative incremental process where it significantly improves project timeline and increase time to release. Amongst all available agile process, scrum is the most adaptable agile framework [13]. The advantages of scrum implementation for research-oriented project including research works optimization and high-quality research results production [14]. The details of the scrum adoption into this research is described in the following section.

3. RESEARCH METHODOLOGY

This research project will adopt agile scrum project management method. While scrum is widely used in the IT industry, there are significant studies by [15], [16], [17], and [18] showing that scrum implementation in research-based project contributes to better research outcomes. Moreover, a thesis written by [16] shows implementation of agile is possible in managing a construction project.

Agile scrum is fitted into this research project as following:

- Roles: Research owner, research team and supervisor (scrum master)
- Artefacts: Research backlog and sprint backlog
- Ceremonies: weekly scrum meeting, sprint review, sprint retrospective

Justification:

As scrum master is defined to be the person who the scrum team report to, and the one who resolves any hindrance in achieving the project goal, supervisor is appointed to be scrum master. Meanwhile, product owner is defined to be research owner – the one who knows the

requirements and specifications of this research. As this research project is evaluated as individual work, research owner and research team will be the same individual.

On the other hand, as daily stand up meeting is not possible in research-based project, sometimes due to different research findings as opposed to software development where they make significant changes in 24 hour period, as well as due to supervisor unavailability handling 10 or more researches at one time, weekly scrum meeting is seen to be more feasible than daily stand up.

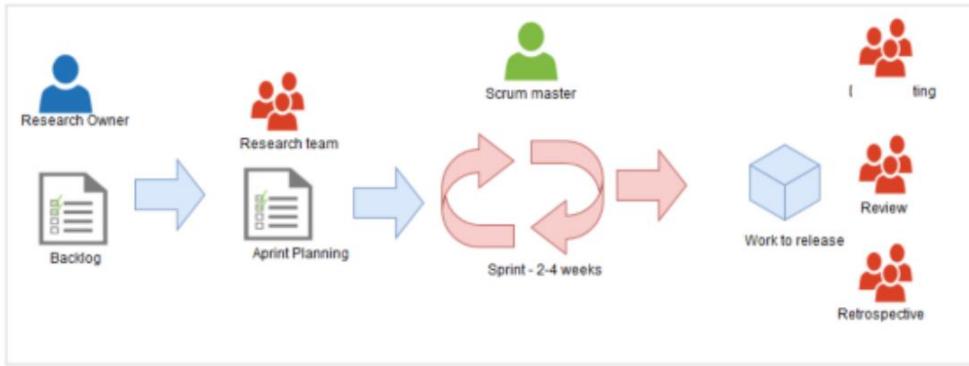


Figure 1. Research Project's Scrum Processes [15]

3.1. Selected Tool

This research uses two automation tools to develop the automation scripts which are Selenium and Cypress. Selenium is selected as it is one of the powerful and stable tools in automating web application while Cypress is selected as it offers a new way in automating modern web application. Cypress is initially a primary work of Brian Mann, a developer who felt testing dynamic websites have been tedious due to inefficient automation test execution. He then conducted a survey on the challenge's automation developers faced in testing current web application [19]. Based on the collected data, automation developers expressed that most of the debugging time was spent on synchronising wait with page loads, though the time should actually be spent on writing more test scripts. Based on these concerns, Cypress is developed and founded in year 2015.

Although there is still less published paper on Cypress due to its rather new entrance in the automation market, the statistics shown by [19] gives a promising view of Cypress's capabilities.

4. PROJECT EXECUTION

4.1. Test Automation Architecture

Both automation scripts in Selenium and Cypress implemented POM design pattern with slightly different style to cater to each programming language syntax used by each tool; automation scripts in Selenium is using Java while Cypress is using JavaScript. The page objects and test cases are separated in both tools to increase code maintainability and modularity. Figure 2 shows the architecture of automation scripts in Selenium and Figure 3 shows the architecture of Cypress. Cypress is made up of all libraries packaged together which made it easier for installation. It also works from within the browser and communicates directly with the Application Under Test (AUT), while Selenium instantiates a

WebDriver which acts as a third party to manipulate the user's actions on the AUT. Both architectures are depicted in the following figures.

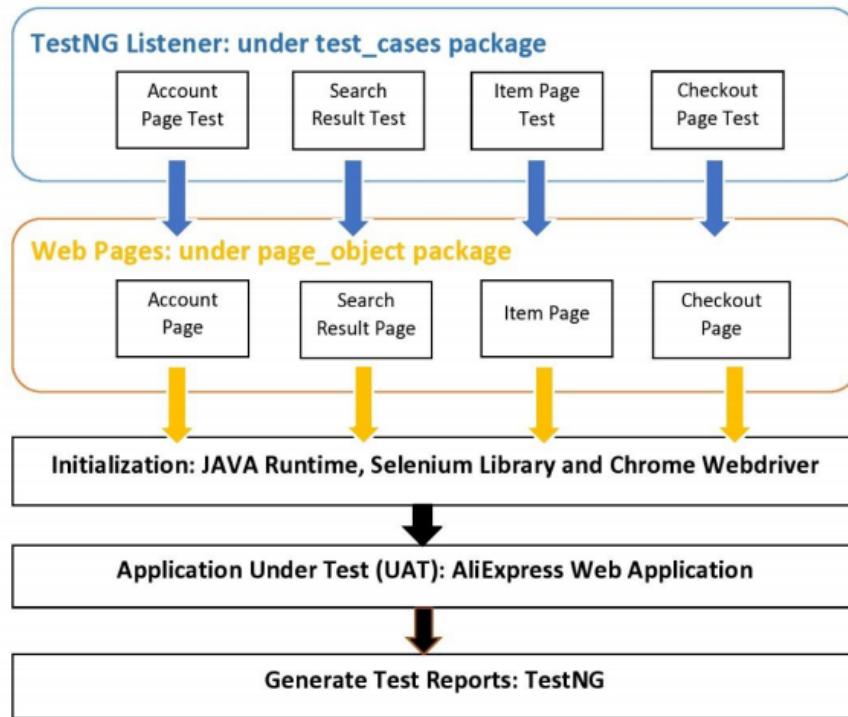


Figure 2. Architecture of Selenium

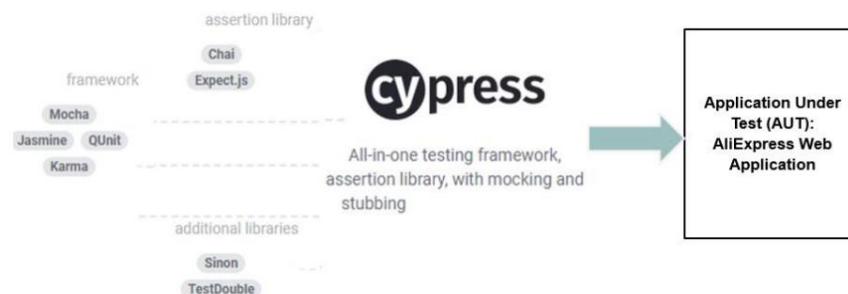


Figure 3. Architecture of Cypress

4.2. Test Scenario

Test scenarios of this research are defined as in Table 1 below while Table 2 shows the detailed test steps for test execution. As mentioned earlier, this research will automate the two key functionalities of AliExpress, the user account and order checkout. Thus, the following test scenarios are derived to cover the functionality of each features.

Table 1. Test Scenarios

| Feature | TS# | Description |
|----------|-------|--|
| Account | TS_01 | Validate that new user able to create account |
| | TS_02 | Validate that existing user able to login with valid credential |
| | TS_03 | Validate that error message is displayed for invalid login |
| | TS_04 | Validate that existing user unable to create account using the same credential |
| | TS_05 | Validate that signed in user can sign out |
| Checkout | TS_06 | Validate that user able to search for items |
| | TS_07 | Validate that user able to select variations for the desired item |
| | TS_08 | Validate that user able to add item to cart |
| | TS_09 | Validate that user able to view cart and checkout |
| | TS_10 | Validate that user need to have a registered account for checkout |

Table 2. Executed Test Steps and Results

| TS# | Test Steps | Test Data | Expected Result | Actual Result | Status |
|--------------------|--|---|---|---------------|--------|
| TS_01 - Account | 1. Open browser 2. Navigate to AliExpress URL 3. Click on Join In button 4. Enter valid email and password 5. Click Create | URL: http://www.aliexpress.com Email: era7@yahoo.com Password: Password1234 | User successfully created an account and populated to Shop Now page | As expected | PASS |
| | Account Button | | | | |
| TS_02 - Account | 1. Open browser 2. Navigate to AliExpress URL 3. Click on Sign In button 4. Enter valid email and password | URL: http://www.aliexpress.com Email: era7@yahoo.com Password: Password1234 | User successfully logged in and populated back to homepage | As expected | PASS |

| | | | | | |
|-----------------|---|---|---|-------------|------|
| | 5. Click Sign In Button | | | | |
| TS_03 - Account | 1. Open browser 2. Navigate to AliExpress URL 3. Click on Sign In button 4. Enter invalid email and password 5. Click Sign In Button | URL: http://www.aliexpress.com Email: er7@yahoo.com Password: 1234 | “Your account name or password is incorrect.” error message displayed | As expected | PASS |
| TS_04 - Account | 1. Open browser 2. Navigate to AliExpress URL 3. Click on Join In button 4. Enter an existing email and password 5. Click Create Account Button | URL: http://www.aliexpress.com Email: era7@yahoo.com Password: Password1234 | “This email already exists. Sign In >” error message is displayed | As expected | PASS |
| TS_05 - Account | Pre-condition: TS_02 is passed. 1. Hover to user icon at the top right hand corner of the page | | User is successfully logged out and populated back to the | As expected | PASS |
| | 2. Click Sign Out link | | homepage | | |

| | | | | | |
|------------------------|---|------------------|---|-------------|------|
| TS_06 - Checkout | Pre-condition: TS_02 is passed. 1. Open browser 2. Navigate to AliExpress URL 3. Type keyword at the search bar 4. Click search icon or press Enter | Keyword: Sweater | User successfully searched item and populated to Search Result page | As expected | PASS |
| TS_07 - Checkout | Pre-condition: TS_05 is passed. 1. Select the first item on the result page 2. Select any variation. | | Upon clicking the item, it opens a new tab and user is populated to the item page | As expected | PASS |
| TS_08 - Checkout | Pre-condition: TS_06 is passed. 1. Click Add To Cart button | | Item is successfully added to the cart and user is populated to recommended items page | As expected | PASS |
| TS_09 - Checkout | Pre-condition: TS_06 & TC_07 are passed. 1. Click View Cart button | | Item is successfully added to the cart and user is populated to Checkout page | As expected | PASS |
| TS_10 - Checkout | Pre-condition: TS_05 is passed. 1. Type keyword at the search bar 2. Click search icon or press Enter 3. Select the first item on the result page 4. Select any variation. 5. Click Add to Cart button | Keyword: sweater | Account frame pop Out when user click Add to Cart, which insist user to have an account before proceed checkout | As expected | PASS |

4.3. Code Snippet

The following section includes a fraction of automation scripts of this research. Figure 4 below shows the automation script for Account regression suite in Selenium. The script starts with browser instantiation and application initialization followed by five Account page test cases: existing account, valid registration, user sign out, invalid sign in, valid sign in and lastly closing the WebDriver instance.



```
1 package test_cases;
2
3@import java.util.concurrent.TimeUnit;
15
16 public class AccountTest {
17     WebDriver driver;
18     public String expected = null;
19     public String actual = null;
20     private final String APP_URL = "http://www.aliexpress.com";
21     private final String DRIVER_PATH = "E:\\Automation\\Drivers\\chromedriver.exe";
22
23     Homepage homepageObject;
24     Account accountObject;
25
26@ BeforeSuite
27     public void BrowserInstance() {
28
29         System.setProperty("webdriver.chrome.driver", DRIVER_PATH);
30         driver = new ChromeDriver();
31         driver.manage().window().maximize();
32         homepageObject = new Homepage(driver);
33     }
34
35@ BeforeTest
36     public void getURL() {
37
38         driver.get(APP_URL);
39         driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
40         homepageObject.closePopup();
41     }
42 }
```

```

43@    @Test (priority=0)
44    public void existingAcccount() {
45
46        accountObject = homepageObject.clickJoin();
47        accountObject.register("era7@yahoo.com", "");
48        String actual = "This email already exists. Sign In >";
49        Assert.assertEquals(actual, accountObject.getString());
50    }
51
52@    @Test (priority=1)
53    public void validRegistration() {
54
55        String userName = ""+(int)(Math.random()*Integer.MAX_VALUE);
56        String emailID = "User"+userName+"@ymail.com";
57
58        accountObject.register(emailID, "password1234");
59
60        try {
61            Thread.sleep(10000);
62        } catch (InterruptedException e) {
63            // TODO Auto-generated catch block
64            e.printStackTrace();
65        }
66
67        Assert.assertTrue(accountObject.returnShopNowBtn());
68        accountObject.clickShopNow();
69    }
70
71@    @Test(priority=3)
72    public void signOut() {
73        accountObject.userAccount();
74        accountObject.signOut();
75        try {
76            Thread.sleep(5000);
77        } catch (InterruptedException e) {
78            // TODO Auto-generated catch block
79            e.printStackTrace();
80        }
81        accountObject.userAccount();
82        accountObject.signIn();
83    }
84
85@    @Test (priority=4)
86    public void invalidSignIn() {
87        driver.switchTo().frame("alibaba-login-box");
88        accountObject.signIn("errr7@yahoo.com", "password1234");
89        try {
90            Thread.sleep(3000);
91        } catch (InterruptedException e) {
92            // TODO Auto-generated catch block
93            e.printStackTrace();
94        }
95        String actual = "Your account name or password is incorrect.";
96        Assert.assertEquals(actual, accountObject.getSignInError());
97    }
98
99
100@    @Test (priority=5)
101    public void validSignIn() {
102        accountObject.signIn("era7@yahoo.com", "password1234");
103    }
104
105@    @AfterSuite
106    public void tearDown() {
107        driver.close();
108        driver.quit();
109        driver = null;
110    }
111}
112}
113}

```

Figure 4. Automation Script for Account Regression Suite in Selenium (112 lines)

5. RESULTS

Test metrics serve as an important indicator to measure the progress of the automation test execution. Three metrics have been chosen to serve as an indicator of this research: total test execution time, test execution efficiency and requirement-based test coverage.

5.1. Test Execution Time

Total test execution time helps to keep track on the overall test execution progress. Figure 5 shows the time execution for each test cases in Selenium's Account regression suite while Figure 6 shows the total test execution time for the whole Account regression suite in

Selenium. Meanwhile, Figure 7 and 8 shows the same metric for Checkout regression suite in Selenium. Additionally, Figure 9 shows the total test execution time for Account regression suite in Cypress and Figure 10 shows the same test metric for Checkout regression suite in Cypress. The summary of the test execution time for both tools is depicted in the Table 3 below. A further discussion is included in the next section of this report.

Table 3. Total Test Execution Time for Regression Suite in Selenium and Cypress

| Total Execution Time (ms) | Selenium | Cypress |
|---------------------------|-----------|----------|
| Account Page | 122.41 ms | 88.91 ms |
| Checkout Page | 100.53 ms | 80.55 ms |

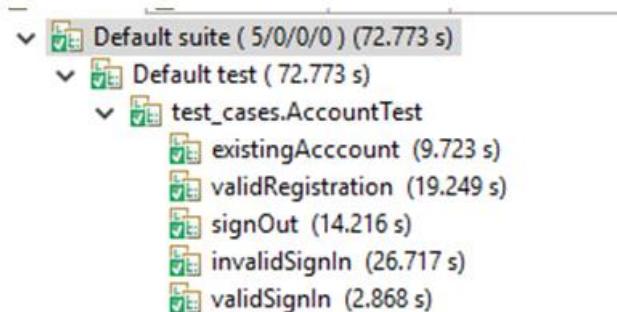


Figure 5. Test Execution Time for Account Test Cases in Selenium

| Methods in chronological order | |
|--------------------------------|-----------|
| test_cases.AccountTest | |
| BrowserInstance | 0 ms |
| getURL | 30235 ms |
| existingAcccount | 40502 ms |
| validRegistration | 51041 ms |
| signOut | 74506 ms |
| invalidSignIn | 91336 ms |
| validSignIn | 119482 ms |
| tearDown | 122407 ms |

Figure 6. Total Test Execution Time for Account Regression Suite in Selenium

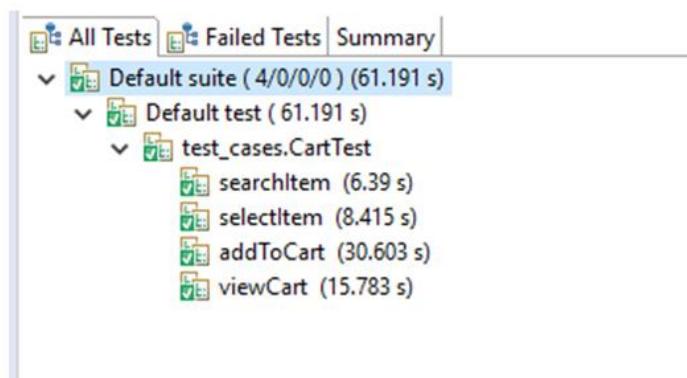


Figure 7. Test Execution Time for Checkout Test Cases in Selenium

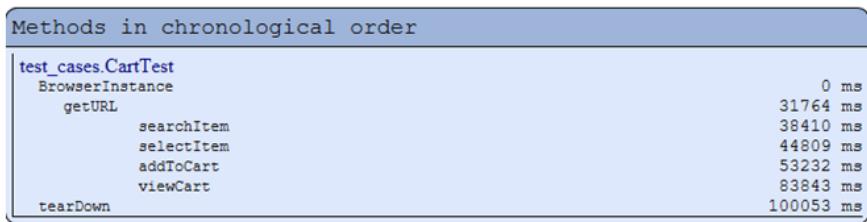


Figure 8. Total Test Execution Time for Checkout Regression Suite in Selenium

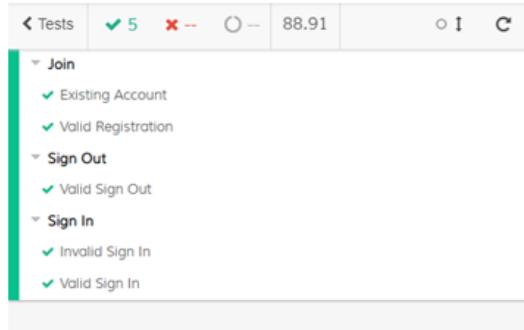


Figure 9. Total Test Execution Time for Account Regression Suite in Cypress

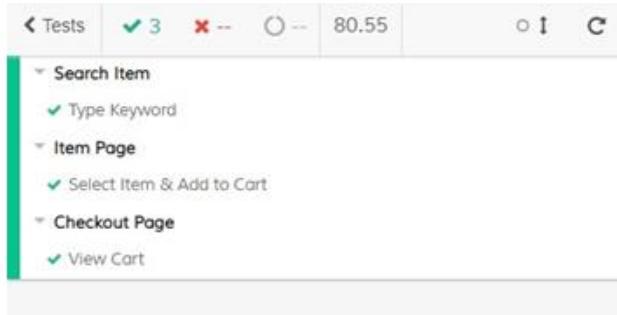


Figure 10. Total Test Execution Time for Checkout Regression Suite in Cypress

5.2. Test Efficiency

Test efficiency measures the cost-effectiveness of testing against the resources of an organization or in this research context, is the total effort in writing the automation script. The optimum test efficiency is the one that is able to reach adequate software quality standard at a lower effort. The following figures show the total line of codes for each tool's automation script. To measure this metric, total number of lines needed to be written to complete an automation scripts in Selenium and in Cypress is being tabulated as in Table 4 below.

Table 4. Total Lines of Automation Script Code in Selenium and Cypress

| Total Lines of Code | Selenium | Cypress |
|---------------------|----------|---------|
| Account Page | 112 | 67 |
| Checkout Page | 118 | 49 |

```

1 import Homepage from '../page_object/Homepage';
2 import Account from '../page_object/Account';
3
4
5 describe('Join', () => {
6   it('Existing Account', () => {
7     const home = new Homepage();
8     home.visit();
9     const register = home.clickJoin();
10    register.fillRegistEmail('era7@yahoo.com').submit();
11    cy.wait(3000);
12    cy.get('#us-xman-register-email-info').invoke('text').then((text1)=>{
13      expect(text1).to.eq('This email already exists.'+'\xa0\x0'+Sign In > ')
14    })
15  });
16});
17
18 it('Valid Registration', () => {
19   const register = new Account();
20   var email = Math.random().toString(36).substring(2, 15) + Math.random().toString(36).substring(2, 15)
21   register
22   .fillRegistEmail(email)
23   .fillRegistPassword('123456')
24   .submit();
25   cy.wait(5000);
26   cy.get('#batman-tabbed > div > div.coupon-detail-bottom > a').should('be.visible')
27   register.clickShopNow();
28 });
29 });
30
31
32 describe('Sign Out',()=>{
33   it('Valid Sign Out',()=>{
34     cy.get('div.user-account-info').trigger('mouseover')
35     cy.get('div.flyout-user-signout a').click()
36     cy.get('div.user-account-info').trigger('mouseleave')
37     cy.get('div.flyout-user-signIn > p:nth-child(2) > a').click()
38   })
39 });
40
41 describe('Sign In', () =>{
42
43   it('Invalid Sign In', () => {
44     cy.get('#alibaba-login-box').then(function ($element) {
45       const $body = $element.contents().find('body')
46       const cyElement = cy.wrap($body)
47       cyElement.find('#login-form > div.fm-btn > button').click()
48
49       const cyElement1 = cy.wrap($body)
50       cyElement1.find('div.login-error-msg').invoke('text').then((text1)=>{
51         expect(text1).to.eq('Please enter your email address or member ID.')
52       })
53     })
54   });
55
56   it('Valid Sign In',()=>{
57     cy.get('#alibaba-login-box').then(function ($element) {
58       const $body = $element.contents().find('body')
59       const cyElement = cy.wrap($body)
60       cyElement.find('#fm-login-id').type('era7@yahoo.com')
61       const cyElement1 = cy.wrap($body)
62       cyElement1.find('#fm-login-password').type('password1234')
63       const cyElement2 = cy.wrap($body)
64       cyElement2.find('#login-form > div.fm-btn > button').click()
65     })
66   });
67 });

```

Figure 11. Account Regression Suite in Cypress (67 lines)

```

1 package test_cases;
2
3 import java.util.Set;
4
5 public class CartTest {
6
7   WebDriver driver;
8   public String expected = null;
9   public String actual = null;
10  private final String APP_URL = "http://aliexpress.com";
11  private final String DRIVER_PATH = "E:\\Automation\\Drivers\\chromedriver.exe";
12
13  Homepage homepageObject;
14  SearchResult searchObject;
15  Item itemObject;
16  Account accountObject;
17  Cart cartObject;
18
19  @BeforeSuite
20  public void BrowserInstance() {
21    System.setProperty("webdriver.chrome.driver", DRIVER_PATH);
22    driver = new ChromeDriver();
23    driver.manage().window().maximize();
24
25    homepageObject = new Homepage(driver);
26    itemObject = new Item(driver);
27    accountObject = new Account(driver);
28    cartObject = new Cart(driver);
29  }
30
31  @BeforeTest
32  public void getURL() {
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
306
307
307
308
309
309
310
311
312
313
314
315
315
316
317
317
318
319
319
320
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
133
```

```

54
55     driver.get(APP_URL);
56     driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
57 }
58
59 @Test (priority=1)
60 public void searchItem() {
61     searchObject = homepageObject.clickSearchBar("Sweater");
62 }
63
64 @Test (priority=2)
65 public void selectItem() {
66     searchObject.clickItem();
67     try {
68         Thread.sleep(5000);
69     } catch (InterruptedException e) {
70         // TODO Auto-generated catch block
71         e.printStackTrace();
72     }
73     String mainwindow = driver.getWindowHandle();
74     Set<String> handles = driver.getWindowsHandles();
75     for (String handle : handles) {
76         if (!handle.equals(mainwindow)) {
77             driver.switchTo().window(handle);
78             break;
79         }
80     }
81     itemObject.selectVariation();
82 }
83
84
85 @Test (priority=3)
86 public void addToCart() {
87     itemObject.clickCart();
88     accountObject.clickSignIn();
89     driver.switchTo().frame("alibaba-login-box");
90     accountObject.signIn("era7@yahoo.com", "password1234");
91 }
92
93
94 @Test (priority=4)
95 public void viewCart() {
96     itemObject.viewCart();
97     cartObject.selectAll();
98     try {
99         Thread.sleep(3000);
100    } catch (InterruptedException e) {
101        // TODO Auto-generated catch block
102        e.printStackTrace();
103    }
104    String expected = "BUY (" + cartObject.returnTotalItem() + ")";
105    System.out.print(expected);
106    Assert.assertEquals(cartObject.returnTotalCart(), expected);
107    System.out.print(cartObject.returnTotalCart());
108 }
109
110
111 @AfterSuite
112 public void tearDown() {
113     driver.close();
114     driver.quit();
115     driver = null;
116 }
117
118 }

```

Figure 12. Checkout Regression Suite in Selenium (118 lines)

```

1 import Homepage from '../page_object/Homepage';
2 import Account from '../page_object/Account';
3 import SearchResult from '../page_object/SearchResult';
4 import Item from '../page_object/Item';
5 import Cart from '../page_object/Cart';
6
7 describe('Search Item', () => {
8     it('Type Keyword', () => {
9         const home = new Homepage();
10        home.visit();
11        const search = home.typeKeyword("Sweater");
12        search.clickItem();
13        cy.get('#root > div > div > div.main-content > div.right-menu > div > div.gallery').should('contain', 'Sweater');
14    });
15 })
16
17 describe('Item Page', ()=>{
18     it('Select Item & Add to Cart',()=>{
19         const item = new Item();
20         item.clickCart();
21         const signin = new Account();
22         signin.clickCartSignInLink();
23         cy.wait(3000);
24         cy.get('#alibaba-login-box').then(function (element) {
25             const $body = element.contents().find('body')
26             const cyElement = cy.wrap($body)
27             cyElement.find('#fm-login-id').type('era7@yahoo.com')
28             const cyElement1 = cy.wrap($body)
29             cyElement1.find( '#fm-login-password').type('password1234')
30             const cyElement2 = cy.wrap($body)
31             cyElement2.find('#login-form > div.fm-btn > button').click()
32         })
33     })
34 })

```

```

32     })
33     cy.wait(7000);
34     item.viewCart();
35   });
36 }
37
38 describe('Checkout Page', ()=>{
39   it ('View Cart', ()=>{
40     const cart = new Cart();
41     cy.get('#checkout-button')
42       .should('be.visible')
43     cy.get('div.captain h2').invoke('text').then((text1)=>{
44       cy.get('#nav-cart-num').invoke('text').should((text2)=>{
45         expect(text1).to.eq('Shopping Cart ('+text2+')')
46       })
47     })
48   })
49 });

```

Figure 13. Checkout Regression Suite in Cypress (47 lines)

5.3. Requirement-based Test Coverage

Requirement-based test coverage is measured against the number of requirements that have been covered by the test cases. This metric determines the thoroughness of the testing towards the Application Under Test (AUT). If the test coverage isn't 100%, that means that there are holes in the testing and require more test cases to be added to include all requirements.

Requirement-based test coverage can be easily measured by generating a traceability matrix. Traceability matrix gives an overview of the overall mapped requirements to test cases. Thus, ensuring all requirements have been covered during test execution phase. Furthermore, by having requirement-based test coverage, any unnecessary and redundant test cases are eliminated as test cases are derived based on the requirement. Table 5 shows the requirement traceability matrix for this research project. The *Defects* field is left empty as no defects are found in the system.

Table 5. Traceability Matrix of Requirement to Test Cases

| Requirements | Tests | Execution | Defects |
|---|-------|-----------|---------|
| User Account | TC_01 | PASS | |
| User able to create account and sign in by having a valid credential. Error message should be displayed accordingly for invalid attempt of register and sign in. A logged in user also should be able to sign out. | TC_02 | PASS | |
| | TC_03 | PASS | |
| | TC_04 | PASS | |
| | TC_05 | PASS | |
| Order Checkout | TC_06 | PASS | |
| User able to search for item, select and add desired items to cart, view cart and checkout. User shall have a registered account for checkout. The system shall insist for an account registration or account login at a point when user attempts to add items to cart without a logged in account. | TC_07 | PASS | |
| | TC_08 | PASS | |
| | TC_09 | PASS | |
| | TC_10 | PASS | |

6. DISCUSSION

6.1. Observation on Test Metrics

From the test metrics collected, it is found that the total test execution time does not have much difference between Selenium and Cypress. Based on Table 3, there is only 33.51ms differences in the total test execution time for Account regression suite between Selenium and Cypress. While 19.98ms differences for Checkout regression suite. Most of the execution time consumed in Selenium is used to instantiate the WebDriver and initialize the application. Selenium does have fast execution for each test cases (as depicted in Figure 5 and Figure 7). However, the whole regression suite execution time is slowed down by the browser initialization which contributes to the biggest number of the test execution time.

On the other hand, there is a significant difference for test efficiency between Selenium and Cypress. Based on Table 4, Cypress produces 45 lesser lines of code compared to Selenium for Account regression suite. The same situation is observed for Checkout regression suite where Cypress produces 69 lesser lines of code compared to Selenium's automation script. This indicates less effort is needed to write automation script in Cypress which significantly increase its test efficiency compared to Selenium.

As an example, Figure 14 below compares the differences between the same number of lines of code (42 lines) written in Selenium and Cypress. Most of the codes in Selenium need to be instantiated with libraries and import pages, whereas Cypress directly cater the main part of the automation script. For instance, up until line 42, Selenium is still instantiating the setup of WebDriver and importing libraries, whereas Cypress has already covered three test cases by line 39.

```

AccountTest.java
1 package test_cases;
2
3 import java.util.concurrent.TimeUnit;
4
5 public class AccountTest {
6     WebDriver driver;
7     public String expected = null;
8     public String actual = null;
9     private final String APP_URL = "http://www.aliexpress.com";
10    private final String DRIVER_PATH = "E:\\Automation\\Drivers\\chromedriver.exe";
11
12    Homepage homepageObject;
13    Account accountObject;
14
15    @BeforeSuite
16    public void BrowserInstance() {
17
18        System.setProperty("webdriver.chrome.driver", DRIVER_PATH);
19        driver = new ChromeDriver();
20        driver.manage().window().maximize();
21        homepageObject = new Homepage(driver);
22
23    }
24
25    @BeforeTest
26    public void getURL() {
27
28        driver.get(APP_URL);
29        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
30        homepageObject.closePopup();
31    }
32
33    @Test (priority=0)
34    public void existingAcccount() {
35
36
37
38
39
39
}

```

```

AccountTest.js
1 import Homepage from '../page_object/Homepage';
2 import Account from '../page_object/Account';
3
4
5 describe('Join', () => {
6     it('Existing Account', () => {
7         const home = new Homepage();
8         home.visit();
9         const register = home.clickJoin();
10        register.fillRegEmail('era@yahoo.com').submit();
11        cy.get('#div#x-man-register-email-info').invoke('text').then((text)=>{
12            expect(text).to.eq(`This email already exists.`)
13        })
14    });
15
16    it('Valid Registration', () => {
17        const register = new Account();
18        var email = Math.random().toString(36).substring(2, 15) + Math.random().toString(36)
19        register.fillRegEmail(email)
20        .fillRegPassword('123456')
21        .submit();
22        cy.wait(5000);
23        cy.get('#bottom-tabbed > div > div.coupon-detail-bottom > a').should('be.visible')
24        register.clickShopNow();
25    });
26
27    describe('Sign Out', ()=>{
28        it('Valid Sign Out', ()=>{
29            cy.get('#div.user-account-info').trigger('mouseover')
30            cy.get('#div.flyout-user-signout a').click()
31            cy.get('#div.user-account-info').trigger('mouseout')
32            cy.get('#div.flyout-user-signin > p:nth-child(2) > a').click()
33        })
34    });
35
36
37
38
39
39
})

```

Figure 14. Comparison of Automation Script in Selenium and Cypress

This shows that Cypress provides a better test efficiency compared to Selenium as Cypress takes shorter lines of code needs to write in order to complete a test scenario. Moreover, it will indirectly improve the efficiency of the effort in writing the test automation scripts.

While, for the last test metrics measured which is requirement-based test coverage, both tools able to provide 100% test coverage. Based on the requirement traceability matrix in Table 5, all the requirements are 100% covered by the test cases in Selenium and Cypress. This indicates the automation scripts written in both tools are able to execute the designed test cases accordingly.

6.2. Limitations and Workaround

As AliExpress is a dynamic web application, there are several limitations found in both tools while automating the web application. These limitations are seen to be an advantage as it serves the purpose of this research to test a challenging web application which consists a variety of dynamic elements. The significant limitation found in Selenium and Cypress while automating AliExpress is the difficulty in handling the opening of a new tab. In AliExpress, whenever a user clicked on an item, the browser will trigger a new tab handling and open the content of the item in the new tab.

The workaround for this problem is slightly different for both tools. In Selenium, new tab handling can be easily overcome by switching the tab using the keyboard commands as Selenium support multiple tab handling. However, the workaround for Cypress is trickier as Cypress doesn't support multiple tab handling. New tab handling is normally triggered when the element's attribute is set to blank target. Thus, by removing the blank target attribute, it will open the content in the same tab when user clicks on the link, which solves the issue in Cypress.

The other limitations found in Cypress are:

- Restricted web security access
- Difficulty in interacting with elements hidden in iFrames

As Cypress acts as the browser itself, most browser will restrict access as browsers adhere to strict same-origin security policy. Thus, this resulting in several links unable to be loaded due to Chrome security commands. By setting the *chromeWebSecurity* to false, it allows Cypress to display insecure content and cross-origin iframes, which works as the workaround for both issues.

6.3. Other Findings

.During this research, it is also found that test cases in Selenium is executed in alphabetical order if test case prioritization is not being assigned. Whereas in Cypress, test cases are executed in sequential order line by line. The test case written at the top will be executed first followed by the rest of the codes.

In term of code readability, automation script in Cypress is more readable than in Selenium for its shorter commands. However, it takes a more technical person to understand the commands used in Cypress whereas Selenium commands is easier to understand.

For an example, to verify the visibility of an element in Selenium, the following commands is needed:

- WebElement element = driver.findElement(By.class("checkout-button"));
 - element.isDisplayed();
- Whereas in Cypress, the commands are shorter and simpler, but is more technical to be understood:
- cy.get('#checkout-button').should('be.visible')

Moreover, the intuitive interface of Cypress made it easy to keep track of the test execution. Figure 15 and Figure 16 show the interface of Cypress where the execution progress is displayed on the left hand side of the frame while the AUT is displayed on the right hand side of the frame. This eases the test execution monitoring as tester will be able to see the commands execution as well as the AUT side by side. Figure 15 shows an assertion

which includes the expected and actual result without the need to print the output. Finally, Figure 16 shows an example of failed test cases in Cypress where it eases the tester to debug the code as Cypress points exactly on the problem in the code.



Figure 15. Example of Passed Assertion for Checkout Regression Suite in Cypress



Figure 16. Example of Failed Test Cases in Cypress

7. CONCLUSION

It has been a good experience exploring and expanding automation knowledge in other automation tool other than Selenium. Selenium is undeniably a powerful tool due to its huge community and support as it has been on the market for many years. However, Cypress also gives a promising view of how the future of the automation testing will be. It significantly eases and simplifies the automation configuration processes and produces a better and cleaner code.

With the right amount of resources and support, Cypress can be used to achieve much more. As this research relies heavily on StakeOverFlow, GitHub and Cypress official website to develop the automation scripts in Cypress, it might not be the best industry practice yet as it is conducted on the basis of self-study. It is believed that with the right mentoring, Cypress is a powerful tool in testing the ever changing and complex modern dynamic web application.

8. RECOMMENDATIONS

Several recommendations are suggested to improve and further extend this research project for future works such as:

- I. Extend the scripts to cover utility functionalities of AliExpress: Message to seller (where buyer can ask for more details with seller), save item to watch list, history of purchase and refund, and the return policy.
- II. ii. Generate a HTML test report in Selenium with Extent Report framework and Mochawesome framework for Cypress, to make it easier for people to understand the test execution status and progress.
- III. iii. As Cypress has released a new version where it supported more browsers than Chrome, it is recommended to extend the scripts for other popular browser such as Mozilla and Opera, to cover wider users.
- IV. iv. Refactor the code for unnecessary commands to increase code maintainability.

REFERENCES

- [1] Polo, M., Reales, P., Piattini, M., & Ebert, C. (2013). Test Automation. *IEEE Software'*, 84-89.
- [2] Al-Zain, S., Eleyan, D., & Hassouneh, Y. (2013). Comparing GUI Automation Testing Tools for Dynamic Web Applications. *Asian Journal of Computer and Information Systems*, 38-48.
- [3] Vila, E., Novakova, G., & Todorova, D. (2017). Automation Testing Framework for Web Applications with Selenium WebDriver: Opportunities and Threats. *Proceedings of the International Conference on Advances in Image Processing* (pp. 144-150). New York: ACM.
- [4] Bulla, A., & S, B. (2016). A Study: Automation Technique Using Selenium Web driver. *International Journal of Research*, 467-470.
- [5] Mane, D., Bhadekar, G., & Salukhe, S. (2016). Text and Keyword Driven Automation Testing using Selenium Web Driver. *International Research Journal of Engineering and Technology*, 515-519.
- [6] Colantonio, J. (2014). The #1 Killer of Selenium Script Performance and Reliability. Retrieved from Joe Colantonio: Automation Awesomeness: <https://www.joecolantonio.com/selenium-performance-reliability/>
- [7] Moe, M. M. (2019). Comparative Study of Test-Driven Development TDD, Behavior-Driven Development BDD and Acceptance Test–Driven Development ATDD. *International Journal of Trend in Scientific Research and Development*, 231-234.
- [8] Khanam, Z., & Ahsan, M. (2017). Evaluating the Effectiveness of Test Driven Development: Advantages and Pitfalls. *International Journal of Applied Engineering Research*, 7705-7716.
- [9] Leotta, M., Clerissi, D., Ricca, F., & Spadaro, C. (2013). Comparing the maintainability of selenium WebDriver test suites employing different locators: a case study. *Proceedings of the 2013 International Workshop on Joining Academia and Industry Contributions to testing Automation* (pp. 53-58). Lugano: ACM.
- [10] Damm, L.O. (2002). Evaluating and Improving Test Efficiency. Ronneby: Citeseerx.

- [11] Shahid, M., & Ibrahim, S. (2011). An Evaluation of Test Coverage Tools in Software Testing. International Conference on Telecommunication Technology and Applications (pp. 216-222). Kuala Lumpur: IACSIT Press.
- [12] Walia, M. (2012). Realizing Efficiency & Effectiveness in Software Testing through a Comprehensive Metrics Model. Bangalore: Infosys.
- [13] Hidalgo, E. S. (2019). Adapting the scrum framework for agile project management in science: case study of a distributed research initiative. *Heliyon*, 1-32.
- [14] Hicks, M., & Foster, J. S. (2010). Adopting Scrum to Managing a Research Group.
- [15] Rodriguez, D. G. (2016). Using Scrum in your research. *Crossroads: The ACM Magazine for Students*, pp. 1-5.
- [16] Pareliya, M. (2018). Implementing Agile Project Management (Scrum) Approach in the Development of Building Projects. CEPT University, Ahmedabad, India.
- [17] Marchesi, M., Mannaro, K., Uras, S., & Locci, M. (2007). Distributed Scrum in Research Project Management. 8th international conference on Agile processes in software engineering and extreme programming (pp. 240-244). Como: Springer.
- [18] Hidalgo, E. S. (2018). Management of a Multidisciplinary Research Project: Journal of Research Practice, 1-17.
- [19] Bharadwaj, S. (2018). Why Should You Switch to Cypress for Modern Web Testing? Retrieved from DZone DevOps: <https://dzone.com/articles/why-should-you-switch-to-cypress-for-modern-web-te?fromrel=true>

AUTHORS

Fatini Mobaraya graduated from University of Technology Malaysia in Bachelor of Computer Science (Software Engineering) with Honours. Her first exposure to software testing is during one of the modules back in her bachelor's degree: Software Quality & Assurance. She never looked back ever since and have found passion in software quality. She then pursued a specialized testing course, Graduate Diploma in Software Testing at AGI Institute, New Zealand; to learn in depth about testing and being a tester who can code. She is interested in projects that comprised of automation test engineering, regression testing and performance engineering.



Dr. Shahid Ali is a senior lecturer and IT program leader at AGI Education Limited, Auckland, New Zealand. He has published number of research papers on ensemble learning. His expertise and research interests include ensemble learning, machine learning, data mining and knowledge discovery.