**Lab 4 Report**

張嗣岱

107598069

5/14

1. **Test Plan**
   1. **Summary/Scope**

Testing the simple function of keystone web can operate successfully (e.g., create/remove/update/delete item for the main terms)

* 1. **Feature to be tested**

1. Post Features

* Create post on the Admin UI page
* Edit post on the Admin UI page
* Delete post on the Admin UI page
* Search post by keyword on the Admin UI page

1. Comment Features

* Create comment on the Admin UI page
* Edit comment on the Admin UI page
* Delete comment on the Admin UI page

1. Category Features

* Create category on the Admin UI page
* Show posts of the specific category by pressing category name on the “Blog” page

1. Enquiry Features

* Create enquiry on the “Contact” page
* Delete enquiry on Admin UI page

1. User Features

* Create new user on the Admin UI page
  1. **Success criteria of completing the test**

All test cases must pass.

* 1. **Test environment and/or infrastructure**

1. **Python 3.6.2**
2. **Selenium 3.141.0**
3. **ChromeDriver** [ChromeDriver 74.0.3729.6](https://chromedriver.storage.googleapis.com/index.html?path=74.0.3729.6/)
4. [Docker Desktop for Windows](https://docs.docker.com/docker-for-windows/install/) & [Docker Hub](https://hub.docker.com/r/ntutselab/keystonejs)
   1. **Test approaches**
   2. **Testing tasks**

To implement the proposed strategy, the following activities are planned to perform.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Activity Name** | **Plan hours** | **Schedule Date** |
| 1 | Study Python, selenium | 4 | 5/8 |
| 2 | Familiar with keystone | 2 | 5/8 |
| 3 | Design test cases | 3 | 5/9 |
| 4 | Implement test cases | 15 | 5/9-5/13 |
| 5 | Perform tests | 3 | 5/14 |
| 6 | Complete Lab4 report | 4 | 5/15 |

1. **Test Design**

**2.1**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Create post on the Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下未有Post |
| **Success Guarantee** | 可以建立一則post |
| **Main Success Scenario** | 1. 登入keystone 2. 按下Post旁的”+”的圖示 3. 輸入Post的名稱 4. 按下Create按鈕 5. 按下Save按鈕 6. 點選Post 分頁 7. 驗證名稱是否被新增成功 |
| **Extensions** |  |

**2.2**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Edit post on the Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post |
| **Success Guarantee** | 可以編輯一則post |
| **Main Success Scenario** | 1. 登入keystone 2. 點Post的超連結 3. 驗證CreatePost的Author欄位下沒有”Demo User” 4. 點CreatePost並將Author欄位選擇”Demo User”後儲存 5. 點選Post分頁 6. 驗證Author欄位值 |
| **Extensions** |  |

**2.3**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Delete post on the Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post |
| **Success Guarantee** | 沒有任何post |
| **Main Success Scenario** | 1. 登入keystone 2. 點Post的超連結 3. 點選CreatePost旁的垃圾桶 4. 選擇確認刪除 5. 驗證Post是否為”No posts found...” |
| **Extensions** |  |

**2.4**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Search posts by keyword on the Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post |
| **Success Guarantee** | 可以編輯一則post |
| **Main Success Scenario** | 1. 登入keystone 2. 點Post的超連結 3. 點選搜尋欄並輸入”CreatePost” 4. 驗證”CreatePost”有存在 5. 清空搜尋欄並輸入”expectNoResult” 6. 驗證”expectNoResult”不存在 |
| **Extensions** |  |

**2.5**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Create comment on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post，而comment欄未有comment |
| **Success Guarantee** | 可以一則comment並指定給一則post |
| **Main Success Scenario** | 1. 登入keystone 2. 按下comment旁的”+”的圖示 3. 選擇下拉式選單Author是Demo User 4. 選擇下拉式選單Post是CreatePost 5. 按下Create按鈕 6. 按下Save按鈕 7. 點選comment分頁 8. 用ID驗證comment是否被新增成功 |
| **Extensions** |  |

**2.6**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Edit comment on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post，有一則指定給CreatePost的comment |
| **Success Guarantee** | 可以新增一則comment並指定給一則post |
| **Main Success Scenario** | 1. 登入keystone 2. 點comment的超連結 3. 點comment ID 4. 選擇下拉式選單CommentState是Archived 5. 按下Save按鈕 6. 點選comment分頁 7. 驗證CommentState是否被編輯為”Archived” |
| **Extensions** |  |

**2.7**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Delete comment on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且每個Post 欄位下一個有名為CreatePost的Post，有一則指定給CreatePost的comment |
| **Success Guarantee** | 可以刪除一則comment |
| **Main Success Scenario** | 1. 登入keystone 2. 點comment的超連結 3. 點選commentID旁的垃圾桶 4. 選擇確認刪除 5. 驗證comment是否為”No comments found...” |
| **Extensions** |  |

**2.8**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Create category on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且Category 沒有東西 |
| **Success Guarantee** | 可以建立一個Category |
| **Main Success Scenario** | 1. 登入keystone 2. 按下Categories旁的”+”的圖示 3. 輸入category的名稱 4. 按下Create按鈕 5. 點選category 分頁 6. 驗證category名稱是否被新增成功 |
| **Extensions** |  |

**2.9**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Show posts of the specific category by pressing category name on the "Blog" page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且Category欄位下一個有名為CreateCategory的categroy |
| **Success Guarantee** | 可以在Blog下看到有什麼Post被放在哪個Category裡 |
| **Main Success Scenario** | 1. 登入keystone 2. 按下Post旁的”+”的圖示 3. 輸入Post的名稱 4. 按下Create按鈕 5. 選擇下拉式選單State是Published 6. 選擇下拉式選單Categories是CreateCategory 7. 按下Save按鈕 8. 到Blog的頁面下 9. 驗證CreatePost有被放到CreateCategory裡 |
| **Extensions** |  |

**2.10**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Create enquiry on the "Contact" page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且Enquiry沒有東西 |
| **Success Guarantee** | 可以建立一則enquirie |
| **Main Success Scenario** | 1. 點選”Contact” 2. 輸入enquirie的名稱、信箱、電話、種類、訊息 3. 按下Create按鈕 4. 登入keystone 5. 點Enquiries的分頁 6. 驗證enquirie名稱 |
| **Extensions** |  |

**2.11**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Delete enquiry on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/>，並且enquirie 欄位下一個有名為testName的enquirie |
| **Success Guarantee** | 可以刪除一則enquirie |
| **Main Success Scenario** | 1. 登入keystone 2. 點Enquiries的分頁 3. 點選testName旁的垃圾桶 4. 選擇確認刪除 5. 驗證enquiries是否為”No enquirie found...” |
| **Extensions** |  |

**2.12**

|  |  |
| --- | --- |
| **Use Case Section** | **Comment** |
| **Use Case Name** | Create a new user on Admin UI page |
| **Precondition** | 可以使運行在<http://127.0.0.1:3000/> |
| **Success Guarantee** | 可以新增一個User |
| **Main Success Scenario** | 1. 登入keystone 2. 按下Users旁的”+”的圖示 3. 輸入user的名稱、信箱、密碼、確認密碼 4. 按下Create按鈕 5. 輸入user的電話 6. 按下Save的按鈕 7. 點選User的分頁 8. 驗證user的Email是否為剛剛新建的Email |
| **Extensions** |  |

To fulfill the test requirements listed in section 1.1, the following methods are selected and corresponding test cases are designed.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Class** | **Method** | **CFG** | **Basis Path** | **Inputs** | **Expected Outputs** |
| 1 | Base32 | decodeBase32ength |  | P1:{1,2,3,4,5,2,7,9}  P2:{1, 2, 7, 8, 9}  P3: {1, 2, 3, 6} | "1b" | 42 |
| 2 | Base32 | padLeftWithZeroToLength |  | |  |  | | --- | --- | | |  | | --- | | P1: {1, 2, 4, 5, 6, 7, 5,  8}, P2: {1, 3} | | | "abc",4 | 0abc |
| 3 | Base32 | getCharIndex |  | P1: {1, 2, 3},  P2: {1, 2, 4} | '1' | 1 |
| 4 | GeoHash | fromLongToString |  | |  |  | | --- | --- | | |  | | --- | | P1: {1, 2a, 2b, 4, 5, 6,  7, 8, 9}, P2: {1, 2a, 3}, P3: {1, 2b, 3} | | | 1 | “0” |
| 5 | GeoHash | gridAsString |  | P1: {1, 2, 3, 4, 5, 6, 7, 9, 10, 5, 6, 7, 8, 9, 10, 5, 11, 12, 3, 13} | "d",-1,-1,1,1,[f] | "c F g \n" + "9 d e \n"+ "3 6 7 \n" |
| 6 | GeoMem | createRegionFilter |  | |  |  | | --- | --- | | |  | | --- | | P1: {1, 2a, 4}, P2: {1, 2a, 2b, 4}, P3: {1, 2a, 2b, 2c, 4}, P4: {1, 2a, 2b, 2c, 2d,  4}, P5: {1, 2a, 2b, 2c, 2d,  3} | | | 0,0,-100,100 | true |

The details of the design are given below:

[The Excel file of test cases…](https://github.com/derry95922/STV/blob/master/LabReport/Lab3/Lab3%20(Basis%20Path%20%20test%20case%20design).xlsx)

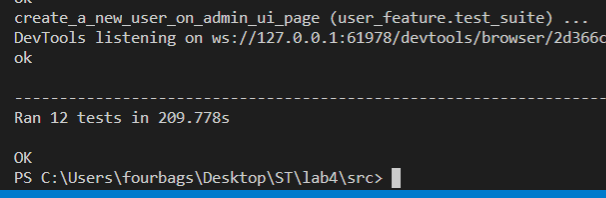
1. **Test Implementation**

The design of test cases specified in Section 1.2 was implemented using selenium with python. The test scripts of 3 selected test cases are given below. The rest of the test script implementations can be found in the [link](https://github.com/derry95922/STV/tree/master/lab4/src) (or python files).

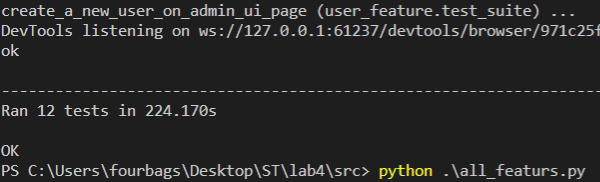
|  |  |  |
| --- | --- | --- |
| **No.** | **Test method** | **Source test code** |
| 1 | Create\_post\_on\_the\_admin\_ui\_page | def create\_post\_on\_the\_admin\_ui\_page(self):  print('---Create post on the Admin UI page---')    wait\_plus\_icon\_is\_visible(self.driver, 'Posts', 'posts').click()  wait\_element\_is\_visible(self.driver, "//\*[@data-screen-id='modal-dialog']")  input\_text(self.driver, "//\*[@name='name']", "CreatePost")  wait\_element\_is\_visible(self.driver, "//\*[@data-button-type='submit']").click()  wait\_element\_is\_visible(self.driver, "//\*[@class='css-2960tt']")  wait\_element\_is\_visible(self.driver, "//\*[text()='Save']").click()  wait\_element\_is\_visible(self.driver, "//\*[@data-list-path='posts']").click()  wait\_element\_is\_visible(self.driver, "//\*[@class='ItemList-wrapper']")  assert "CreatePost" in self.driver.find\_element\_by\_xpath("//\*[text()='CreatePost']").text    delete\_post(self.driver) |
| 2 | Delete\_comment\_on\_admin\_ui\_page | def delete\_comment\_on\_admin\_ui\_page(self):  print('---delete\_comment\_on\_admin\_ui\_page---')  create\_comment(self.driver)  wait\_element\_is\_visible(self.driver, "//\*[@class='octicon octicon-home']").click()  wait\_web\_herf\_is\_visible(self.driver, "comments").click()  wait\_element\_is\_visible(self.driver, "//\*[@class='ItemList-wrapper']")  wait\_element\_is\_visible(self.driver, "//\*[contains(@class,'ItemList\_\_value--id')]/../preceding-sibling::\*").click()  wait\_element\_is\_visible(self.driver, "//\*[@data-screen-id='modal-dialog']")  wait\_element\_is\_visible(self.driver, "//\*[@data-button-type='confirm']").click()  wait\_element\_is\_visible(self.driver, "//\*[text()='No comments found...']")  noPosts = self.driver.find\_element(By.XPATH, "//\*[text()='No comments found...']").text  assert "No comments found..." == noPosts |
| 3 | Create\_category\_on\_admin\_ui\_page | def create\_category\_on\_admin\_ui\_page(self):  print('---create\_category\_on\_admin\_ui\_page---')  wait\_plus\_icon\_is\_visible(self.driver, 'Posts', 'categories').click()  wait\_element\_is\_visible(self.driver, "//\*[@data-screen-id='modal-dialog']")  input\_text(self.driver, "//\*[@name='name']", "CreateCategory")  wait\_element\_is\_visible(self.driver, "//\*[@data-button-type='submit']").click()  wait\_element\_is\_visible(self.driver, "//\*[contains(@data-list-path,'categories')]").click()  wait\_element\_is\_visible(self.driver, "//\*[@class='ItemList-wrapper']")  assert "CreateCategory" in self.driver.find\_element(By.XPATH, "//\*[contains(@class,'ItemList\_\_value--text')]").text  delete\_category(self.driver) |

1. **Test Results**
   1. **Test Suite result snapshot**

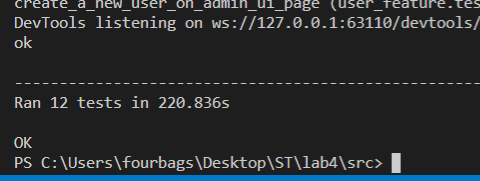
**#1**



**#2**



**#3**



1. **Summary**

In Lab 4,12 **test cases have been designed and implemented using selenium with python and the basis path/graph coverage technique**. The **execution results of the 12 test cases are all passed**. Thus, the test requirements described in Section 1 are satisfied. I think that design by user scenario is very complicated because there are a lot of situation must be considered. And I also think that it’s a very good experience to try to use selenium with python to testing with webs.