Software Testing Document

group 1 team 2

1.Testing plan:

* Tools:

1. Selenium WebDriver and PyUnit to coding automation testing cases
2. Pivotal Tracker to write user stories testing templates
3. Github Issue Tracker and Slack to report bugs/ communicating with developers
4. Software Testing Document for recording

* Testing types:

Unit testing, UI functional testing(automatically and manually),

Integration testing

* Metrics:

1. planned metrics

**a）product metrics**

*Complexity*: the number of lines of code (LOC), number of features of production, cost of time

*Defect density*: number of defects relative to the software size

Mean time of failure: (according to our communication tool) interruption frequency, signal emission failure

*Customer problems:* total number of problems encountered by customer while using the product

*Customer satisfaction*: the survey from professor, and other classmates

**b）process metrics**

Project milestones: time of breakthroughs in difference

phrases, to determine whether project is on schedule.

Rate of test case execution & number of passed test cases: determining whether testing is proceeding on schedule.

Defect detection rate: number of defect detected per period of time

Defect resolution: number of solved defect compared with detects

b. results

a.testing coverage: all features from user stories

b.number of tests pass: 15 unit testing cases & 10 functional

testing cases

c.number of bugs detected: 10

d.defect resolution: 10/10 all solved

2.Testing cases:

* Unit Testing cases:

3 testing classes:TestInfrastructure(), TestAPI(), and TestSocketIO()

a.

TestingInfrastructure(): make sure our dependencies in place and all services

are running as expected.

cases: test\_cdn, test\_gunicorn\_running,test\_node\_running,test\_nginx\_running, test\_reverse\_proxy,test\_Python\_dependencies

b.

TestAPI(): sending request to URL, then check if it will send back

right data

cases:

test\_get\_user: send request to <http://localhost/api/users/?format=json> to get userlist, then check if http status code is 200 to make sure request is fulfilled, at last to check if get a userlist that contains users

test\_get\_message: request to <http://localhost/api/messages/?format=json> to get message list, then check if http status code is 200 to make sure request is fulfilled.

test\_get\_rooms: send request to <http://localhost/api/rooms/?format=json> to get the room list, then check if http status code is 200 to make sure request is fulfilled.

test\_create\_public\_module: creating a room object, then add the object to check if api can get data well

c.

TestSocketIO(): testing the connecting to the socket

cases:

test\_global\_namespace\_REST\_endpoint: connect to socket

as user, then check status code to see if it is fulfilled.

test\_global\_namespace\_connect: we connect to the socket as a user and make sure that we can see it in the active list of users.we use that same endpoint to show which users are currently online

* UI functional testing cases

a.Create New Team Testing

Test name: Create New Team testing

Test items: “create new teams” button, pop-up dialogue, team name input area, “cancel” button, “save” button, “Chatting Room” list

Test priority: high

Preconditions: login to the system

input data:

· Button click: “create new teams” button, “cancel” button, “save” button

· Input text: team name

Test steps:

· Click the “create new teams” Button

· Check the pop-up dialogue

· Input team name or input nothing

· Click the “cancel” Button or “save” button

· Check the Chatting Room list

Postcondition: the successfully created room can be clicked and entered

Expected output:

· After clicking “create new teams” button, the “Create New Team” pop-up dialogue will display and other areas on UI will become disabled

· If nothing input, then click “save” and “cancel” button, no new room will show in “Chatting Room” list

· If input team name, then click “save”, the new team name will show in “Chatting Room” list

· If input team name, then click “cancel”, the no new room will show in “Chatting Room” list

Actual output: same with expected

Pass or Fail: pass

b.Browser Compatibility

Test name: Browser Compatibility

Test items: layout, all button function

Test priority: high

Preconditions: login to the system, have IE, Chrome, Safari, Firefox browsers.

input data:

· none

Test steps:

· First using browser like IE and observe the layout of the UI and check if there is some button or place missing

· Change another Browser like Chrome and repeat the steps

Postcondition: All the layout is correct without button or model missing

Expected output:

· When using different browser, the layout of each part is correct and without button missing.

Actual output: same with expected

Pass or Fail: pass

c.Responsive UI test

Test name: Responsive UI test

Test items: Layout of UI

Test priority: high

Preconditions: login to the system

input data:

· none

Test steps:

· Change the size of the browser to Full-size and observe the layout of the UI.

· Narrow the size of the browser to middle size (tablet size) and observe the layout of UI.

· Narrow the size of the browser to small size (mobile size) and observe the layout of UI.

Postcondition: the layout of the UI is different according to the size of browser.

Expected output:

· When the size is full size, the layout is like this, there is a side-bar in the left with Room-list, User-list and quick link to other two tools, and the conversation details will show in the right. These two parts are showing in the same time.

· When the size is tablet size or mobile size, only the conversation details place is showing, and the side bar with Room-list, User-list and quick link to other two tools cannot display.

· If click “Back”, the side bar with Room-list, User-list and quick link to other two tools can display, and the conversation details place will disappear.

· If click each room in the room list, the side bar with Room-list, User-list and quick link to other two tools cannot display, and the room which has been click will be open and the conversation details place of that room will display.

Actual output: same with expected

Pass or Fail: pass

d.Button Group test

Test name: Button Group test

Test items: “Send“ button, “Search” button in the option menu, “Search” button in the search bar , “Back” button, Chatting room list button, emoji button

Test priority: high

Preconditions: login to the system

input data:

· Button click: “Send“ button, “Search” button in the option menu, “Search” button in the search bar , “Back” button, Chatting room list button, emoji button

· Input text: message in the message bar, search bar

Test steps:

· Input something in the message bar and click the “send” button

· Check if there is the message which is just input displaying in the message place.

· Click the emoji button and click send button.

· Check if there is a emoticon in the message place which is just chosen to send.

· Click “Search” button in the option menu

· Check if there is a message bar displaying above the message place.

· Input something in the search bar and click the “search” button in the search place.

· Check if there is a pop up model displaying the search results.

· Click the room list button

· Check if the title of the room has changed and the conversation has changed into the room chosen

Postcondition: the successfully achieve the function of each button

Expected output:

· After clicking “send” button, the message that input in the message bar before has display.

· After clicking the emoji button and send button, the emoticon that has been chosen display in the message place.

· After clicking “Search” button in the option menu, the search bar display above the message place.

· After input something and click search in the search bar, the pop up window display in the front and show the results.

· After click the room list button, the title will change into the room we have chosen and the message detail will change into the room we have chosen.

Actual output: same with expected

Pass or Fail: pass

e.GroupMessageSending test

Test name: group message sending test

Test items: message input text, message list area, message send button

Test priority: high

Preconditions: user log into system successfully

input data:

message in message input text box.

Test steps:

· type the message in message input box

· click the send button·

click the send button

repeat for each of the user room in room list

Expected output:

·After typing the message, and click the send button, message will appear in the message list area.

after click the send button , the message will appear in the message list area.

the message will not appear in other group’s message list area

Pass or Fail: pass

f.Emoticons test

Test name: emoticons test

Test items: message input text, message list area, message send button, emoticon buttton

Test priority: medium

Preconditions: user log into system successfully

input data:

message in message input text box.

Test steps:

· type the message in message input box

· click the emoticon button

· choose a emoticon you want to send

click the send button

repeat for each of the emoticon in emoticon list

Expected output:

· after selecting the emoticon button , a pop out window listing all emoticons will appear

the message will only appear in the group you are in.

Actual output: same with expected

Pass or Fail: pass

g.Message Search test

Test name: message search test

Test items: message search button, message input text, message list area, message pop out window

Test priority: medium

Preconditions: user log into system successfully, different messages exist in message list area.

input data:

string you want to search

Test steps:

· click search button.

· type the string you want to search in the pop out window search text box.

· click search button.

Expected output:

· after the search button, a pop out window listing the message contains or equal to the string in search area will be listed

ordered by time.

Actual output: same with expected

Pass or Fail: pass

h.Upload File test

Test name: upload file test

Test items: upload file button, message input text, message list area

Test priority: medium

Preconditions: user log into system successfully, file being uploaded exist in local disk.

input data:

different types of file no more than 25MB(system maximum required)

Test steps:

· user log in system.

· click upload file button

· choose the file you want to upload.

click send button

repeat first three steps and click cancel button

repeat first four steps for each type of file

Expected output:

· after clicking the send button, file name will appear in the message list area.

when you click the file name, you will be asked just open it or download in your local disk.

when you choose open, file will be opened immediately and when you download, it will be downloaded in the path

you selected.

after click the cancel button, the file will be dropped.

each type of the file will be uploaded successfully after repeating the first four steps

Actual output: same with expected

Pass or Fail: pass

i.Create New Chatting Room test

Test name: create new chatting room test

Test items: create new chatting room button, chat room input area, “log in” button, cancel button, save button

Test priority: high

Preconditions: user log into system successfully, create new room button

input data:

· chat room input area

Test steps:

· user log in system successfully

· click create new room button

· type new room name in textbox

click save button

click create new room button

· type new room name in textbox

click cancel button

Postcondition: users log in successfully.

Expected output:

· after clicking the save button, new chat room name will appear in the group name llistbox.

and when you click the new room label, a new chatting room window appears.

after clicking the cancel button, new chat room name will become non-exist.

Actual output: same with expected

Pass or Fail: pass

j.User Online Status test

Test name: User online status test

Test items: create new teams button, team name input area, “log in” button, user list

Test priority: medium

Preconditions: login to the system, have users online/offline

input data:

· user list

different user account

Test steps:

· create two users, one user log into system.

· check the user list area

· log out this user account

· check the user list area

Postcondition: two users can be successfully created and log in /log out successfully.

Expected output:

· after one user log in, the name color becomes blue.

after log out, the name color become grey.

when there is no users, the user list is empty.

Actual output: same with expected

Pass or Fail: pass

Automation Testing Code attachment

Unit Testing:

|  |
| --- |
| import unittest |
| import commands |
| import json |
| from socketIO\_client import SocketIO, BaseNamespace |
|  |
| ### General |
| class TestInfrastructure(unittest.TestCase): |
|  |
| output = commands.getoutput('ps -A') |
|  |
| def test\_cdn(self): |
| self.assertTrue(requests.get('https://cdnjs.cloudflare.com').status\_code == 200) |
|  |
| def test\_gunicorn\_running(self): |
| self.assertTrue('gunicorn' in self.output) |
|  |
| def test\_node\_running(self): |
| self.assertTrue('node' in self.output) |
|  |
| def test\_nginx\_running(self): |
| self.assertTrue('nginx' in self.output) |
|  |
| def test\_reverse\_proxy(self): |
| website\_text = requests.get('http://localhost').text |
| title = 'Communication tool' |
| self.assertTrue(title in website\_text) |
|  |
| def test\_python\_dependencies(self): |
|  |
| pip\_output = commands.getoutput('pip freeze').split('\n') |
| available = [line.strip() for line in pip\_output if '==' in line] |
| requirements\_file = [line.strip() for line in open('/home/cs673/Comm-Tool/requirements.txt')] |
|  |
| for line in requirements\_file: |
| self.assertTrue(line in available) |
|  |
| def test\_node\_dependencies(self): |
| pass |
|  |
| class TestAPI(unittest.TestCase): |
|  |
| def test\_get\_users(self): |
| res = requests.get('http://localhost/api/users/?format=json') |
| users = res.json() |
| self.assertTrue(res.status\_code == 200) |
| self.assertTrue( len(users) > 0 ) |
|  |
| def test\_get\_messages(self): |
| res = requests.get('http://localhost/api/messages/?format=json') |
| messages = res.json() |
| self.assertTrue(res.status\_code == 200) |
|  |
| def test\_get\_rooms(self): |
| res = requests.get('http://localhost/api/rooms/?format=json') |
| rooms = res.json() |
| self.assertTrue(res.status\_code == 200) |
| self.assertTrue( len(rooms) > 0 ) |
|  |
| def test\_create\_public\_room(self): |
| room\_data = { |
| 'name': 'test\_room', |
| 'description': 'testing room API', |
| 'public': True |
| } |
| pass |
|  |
| def test\_create\_private\_room(self): |
| pass |
|  |
| def test\_file\_upload(self): |
| pass |
|  |
| class TestSocketIO(unittest.TestCase): |
|  |
| def test\_global\_namespace\_REST\_endpoint(self): |
| res = requests.get('http://localhost:3000/users') |
| self.assertTrue( res.status\_code == 200 ) |
|  |
| def test\_global\_namespace\_connect(self): |
| with SocketIO('localhost', 3000) as socket: |
| global\_ns = socket.define(BaseNamespace, '/') |
| global\_ns.emit('user', {'username':'test', 'action':'connect'}) |
|  |
| res = requests.get('http://localhost:3000/users') |
| self.assertTrue('test' in res.json() ) |
|  |
|  |
| class SeleniumTests(unittest.TestCase): |
|  |
| def test\_something(self): |
| pass |
|  |
| if \_\_name\_\_ == '\_\_main\_\_': |
| unittest.main() |

UI functional testing cases:

a.GroupMessageSending:

\_\_author\_\_ = **'yuyan'**

**from** selenium **import** webdriver

**from** selenium.webdriver.support.ui **import** WebDriverWait

**from** selenium.webdriver.support **import** expected\_conditions **as** EC

**from** selenium.webdriver.common.by **import** By

*# user 1*

driver1 = webdriver.Firefox()

driver1.get(**"http://52.4.50.51/"**)

driver1.set\_window\_size(700, 1000)

driver1.set\_window\_position(0, 0)

*#user 2*

driver2 = webdriver.Firefox()

driver2.get(**"http://52.4.50.51/"**)

driver2.set\_window\_size(700, 1000)

driver2.set\_window\_position(700, 0)

*#user 1 and user 2 talking in team one*

driver1.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver1.find\_element\_by\_id(**"room-6"**).click()

driver2.find\_element\_by\_id(**"room-6"**).click()

*#communicate*

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 11 typing"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 22 typing"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 11 typing"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 22 typing"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

driver1.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"menu\_button"**).click()

*#check if general team can see message in group one*

*#error can not get the text content from messagecontent div*

*#check user1*

driver1.find\_element\_by\_id(**"room-5"**).click()

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-5.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-5.messagecontent"**),**"user 22 typing"**))

driver1.find\_element\_by\_id(**"menu\_button"**).click()

driver1.find\_element\_by\_id(**"room-7"**).click()

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-7.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-7.messagecontent"**),**"user 22 typing"**))

driver1.find\_element\_by\_id(**"menu\_button"**).click()

driver1.find\_element\_by\_id(**"room-8"**).click()

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-8.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-8.messagecontent"**),**"user 22 typing"**))

driver1.find\_element\_by\_id(**"menu\_button"**).click()

driver1.find\_element\_by\_id(**"room-9"**).click()

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-9.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver1,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-9.messagecontent"**),**"user 22 typing"**))

driver1.find\_element\_by\_id(**"menu\_button"**).click()

*#check user2*

driver2.find\_element\_by\_id(**"room-5"**).click()

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-5.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-5.messagecontent"**),**"user 22 typing"**))

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"room-7"**).click()

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-7.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-7.messagecontent"**),**"user 22 typing"**))

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"room-8"**).click()

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-8.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-8.messagecontent"**),**"user 22 typing"**))

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"room-9"**).click()

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-9.messagecontent"**),**"user 11 typing"**))

wait=WebDriverWait(driver2,10)

wait.until\_not(EC.text\_to\_be\_present\_in\_element((By.CSS\_SELECTOR,**"#room-9.messagecontent"**),**"user 22 typing"**))

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver1.close()

driver2.close()

b.GeneralMessageSending

\_\_author\_\_ = **'yuyan'**

**from** selenium **import** webdriver

**from** selenium.webdriver.support.ui **import** WebDriverWait

**from** selenium.webdriver.support **import** expected\_conditions **as** EC

**from** selenium.webdriver.common.by **import** By

**from** selenium.webdriver.support.ui **import** Select

*#user1 window*

driver1=webdriver.Firefox()

driver1.get(**"http://52.4.50.51/"**)

driver1.maximize\_window()

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 1"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 1"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

wait=WebDriverWait(driver1,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,**"messagecontent"**),**"user 1"**))

*#user 2 window*

driver2=webdriver.Firefox()

driver2.get(**"http://52.4.50.51/"**)

driver2.set\_window\_size(800,1000)

driver2.set\_window\_position(500,0)

driver2.find\_element\_by\_id(**"menu\_button"**).click()

driver2.find\_element\_by\_id(**"room-5"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 2"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 2"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

driver2.find\_element\_by\_id(**"menu\_button"**).click()

*#user1 type*

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 1"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

driver1.find\_element\_by\_id(**"text"**).send\_keys(**"user 1"**)

driver1.find\_element\_by\_id(**"sendcleartext"**).click()

wait=WebDriverWait(driver1,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,**"messagecontent"**),**"user 1"**))

wait=WebDriverWait(driver1,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,**"messagecontent"**),**"user 2"**))

*#user2 type*

driver2.find\_element\_by\_id(**"room-5"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 2"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

driver2.find\_element\_by\_id(**"text"**).send\_keys(**"user 2"**)

driver2.find\_element\_by\_id(**"sendcleartext"**).click()

wait=WebDriverWait(driver2,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,**"messagecontent"**),**"user 1"**))

wait=WebDriverWait(driver2,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,**"messagecontent"**),**"user 2"**))

driver1.close()

driver2.close()

c.Button testing & responsive UI interface testing

\_\_author\_\_ = 'yuyan'

import unittest

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from selenium.webdriver.common.by import By

class TestOne(unittest.TestCase):

def setUp(self):

self.driver = webdriver.Firefox()

def test\_mobile\_cp\_switch(self):

driver = self.driver

#test the cp size

driver.maximize\_window()

driver.get("http://52.4.50.51/")

driver.find\_element\_by\_id("text").send\_keys("cp size test")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("cp size test")

driver.find\_element\_by\_id("sendcleartext").click()

#test the mobile size

driver.set\_window\_size(800,1000)

#if go to the home page when change size

wait=WebDriverWait(driver,10)

wait.until(EC.invisibility\_of\_element\_located((By.ID,"sidebar")))

driver.find\_element\_by\_id("menu\_button").click()

#go to general room

driver.find\_element\_by\_id("room-5").click()

wait=WebDriverWait(driver,10)

wait.until(EC.invisibility\_of\_element\_located((By.ID,"sidebar")))

driver.find\_element\_by\_id("text").send\_keys("mobile size test")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("mobile size test")

driver.find\_element\_by\_id("sendcleartext").click()

#go to cp again, sidebar shows

driver.maximize\_window()

wait=WebDriverWait(driver,10)

wait.until(EC.visibility\_of\_element\_located((By.ID,"sidebar")))

def test\_button\_function(self):

driver = self.driver

driver.maximize\_window()

driver.get("http://52.4.50.51/")

#test the send button and return for message sending

#check if there have message through sending button

driver.find\_element\_by\_id("text").send\_keys("sendcleartext button test")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("sendcleartext button test")

driver.find\_element\_by\_id("sendcleartext").click()

wait=WebDriverWait(driver,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,"messagecontent"),"sendcleartext button test"))

#check if there have message through sending button

driver.find\_element\_by\_id("text").send\_keys("return key press test")

driver.find\_element\_by\_id("text").send\_keys(Keys.RETURN)

driver.find\_element\_by\_id("text").send\_keys("return key press test")

driver.find\_element\_by\_id("text").send\_keys(Keys.RETURN)

wait=WebDriverWait(driver,10)

wait.until(EC.text\_to\_be\_present\_in\_element((By.CLASS\_NAME,"messagecontent"),"return key press test"))

#room and home change buttons

driver.set\_window\_size(800,1000)

#general room

driver.find\_element\_by\_id("menu\_button").click()

driver.find\_element\_by\_id("room-5").click()

driver.find\_element\_by\_id("text").send\_keys("general room")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("general room")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("menu\_button").click()

#team one

driver.find\_element\_by\_id("room-6").click()

driver.find\_element\_by\_id("text").send\_keys("team one")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("team one")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("menu\_button").click()

#team two

driver.find\_element\_by\_id("room-7").click()

driver.find\_element\_by\_id("text").send\_keys("team two")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("team two")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("menu\_button").click()

#team three

driver.find\_element\_by\_id("room-8").click()

driver.find\_element\_by\_id("text").send\_keys("team three")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("team three")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("menu\_button").click()

#team four

driver.find\_element\_by\_id("room-9").click()

driver.find\_element\_by\_id("text").send\_keys("team four")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("text").send\_keys("team four")

driver.find\_element\_by\_id("sendcleartext").click()

driver.find\_element\_by\_id("menu\_button").click()

def tearDown(self):

self.driver.close()

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

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