



دانشگاه اصفهان
دانشکده مهندسی کامپیوتر

آزمون واسط کاربری

UI Testing

پدیدآورنده:

محمد امین کیانی

4003613052

دانشجوی کارشناسی، دانشکده‌ی کامپیوتر، دانشگاه اصفهان، اصفهان،
Aminkianiworkeng@gmail.com

استاد درس: جناب آقای دکتر شعرباف

نیمسال اول تحصیلی 1403-04

فهرست مطالب

3	مستندات
3	لینک ویدیو
3	بررسی خطاهای متداول کلی
4	تکلیف سوم
4	بخش اول:
9	بخش دوم:

مستندات

لینک ویدیو

<https://drive.google.com/file/d/1UCS4HVLLe9EOMyuH27Vo2tuQaaMe6cURE/view?usp=sharing>

بررسی خطاهای متداول کلی

1- در بعضی از موارد بعضی از سایت ها، تست ها ارور مربوط به **کش** دارند (مثال هنگامی که سایت لود شد و صفحه privacy&Cookies میاد که باید ریجت یا اکسپت کرد و سپس ادامه تست را اجرا کرد):

The page keeping the extension port is moved into back/forward cache, so the message channel is closed.

چون به اطلاعات آن دکمه نمیتوان دسترسی پیدا کرد و تنها بار اول که سایت را باز میکنیم، پیدا می شود پس دیگر نخواهد بود مگر با پاک کردن کوکی ها شدنی است.

2- در بعضی از موارد همه ی سایت ها، تست ها ارور مربوط به **منقضی شدن جستجو** دارند (مثال هنگامی که سایت لود شد و دکمه ی مورد نظر را می خواهد پیدا کند اما نمیتواند):

Trying to find css=.order-number > strong... Failed:16:55:41 Implicit Wait timed out after 30000ms

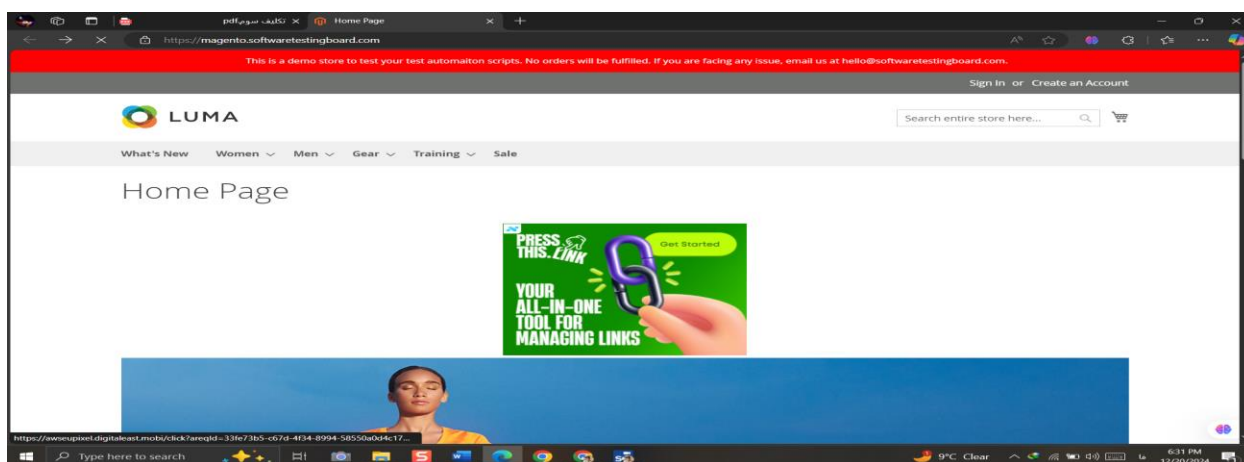
- **'Buy a new stuff testing' ended with 1 error(s)**

گاهی با تغییر هدف در کامند مربوطه می توان جهت گیری دکمه یابی را تغییر داده و دسترسی به آن پیدا کرد و حتی می توان در برخی موارد از دابل کلیک فیدبک درستی از موقعیت مکانی آن ایجاد کرد اما در حالت کلی مشکل پیدا شدن دکمه ها به تعداد دفعات زیادی وجود دارد.

تکلیف سوم

بخش اول:

در ابتدا [سایت شماره 7](#) را انتخاب کرده و 9 تست مهم از عملکرد کلی آن رکورد کردم که 5 مورد را در ادامه شرح می‌دهم:



Selenium IDE - UI Test2 _ Amin Kiani

Project: UI Test2 _ Amin Kiani

Tests

- ✓ Adv Search testing
- ✓ Buy a new stuff testing
- ✗ Can't Buy without Login
- ✓ Pages tester
- ✗ Search testing
- ✗ Shop Checkout testing
- ✓ Sign In testing
- ✓ Sign Up testing
- ✓ comment testing

Search tests...

https://magento.softwaretestingboard.com

	Command	Target	Value
16	✓ double click	css=.base	
17	✓ click	css=.checkout > span	
18	✓ click	id=customer-email	
19	✓ type	id=customer-email	aj@gmail
20	✓ click	css=#shipping-new-address-form > .field:nth-child(1)	

Command: assert

Target: [input field]

Value: not exist email

Description: [input field]

Log

Log	Reference
9. click on id=option-label-color-93-item-53 OK	16:54:54
10. click on id=option-label-size-143-item-170 OK	16:54:54
11. click on id=product-addtocart-button OK	16:54:55
12. runScript on window.scrollTo(0,552) OK	16:54:55
13. Trying to find linkText=shopping cart... OK	16:54:56
14. click on css=.item > .primary OK	16:54:57
15. runScript on window.scrollTo(0,42.400001525878906) OK	16:55:00

سناریو 1: بررسی کلی صفحات هدر

عملکرد دکمه‌های هدر سایت و صفحات کلی آن تست شده تا صحت لینک بودن آن‌ها تست شود.

The screenshot displays the Selenium IDE interface for a project named 'UI Test2_Amin Kiani'. The browser window shows the URL 'https://magento.softwaretestingboard.com'. The test suite 'Pages tester' is selected, and the test 'Buy a new stuff testing' is highlighted. The test results show a failure at step 28, 'Trying to find css= order-number > strong', with an error message: 'Warning Element found with secondary locator xpath=//div[4]/div/button/span. To use it by default, update the test step to use it as the primary locator. Implicit Wait timed out after 30000ms'. The test step details show the command 'verify text', target 'css= page-title-wrapper', and value 'Men'.

Command	Target	Value
open	/	
set window size	1056x808	
click	linkText=Sign In	
verify text	css= base	Customer Login
click	linkText=Create an Account	
verify text	css= base	Create New Customer Account
click	css=img	
verify text	css= page-title-wrapper	Home Page
click	css=#ui-id-3 > span	
verify text	css= page-title-wrapper	What's New

Log Reference

- 25. Trying to find css= button > span... OK
- 26. runScript on window.scrollTo(0,0) OK
- 27. Trying to find css= checkout > span... OK
- Warning Element found with secondary locator xpath=//div[4]/div/button/span. To use it by default, update the test step to use it as the primary locator.
- 28. Trying to find css= order-number > strong... Failed: Implicit Wait timed out after 30000ms
- 'Buy a new stuff testing' ended with 1 error(s)

Running 'Buy a new stuff testing'

Command	Target	Value
verify text	css= page-title-wrapper	What's New
click	css=#ui-id-4 > span:nth-child(2)	
verify text	css= page-title-wrapper	Women
click	css=#ui-id-5 > span:nth-child(2)	
verify text	css= page-title-wrapper	Men
click	css=#ui-id-6 > span:nth-child(2)	
verify text	css= page-title-wrapper	Gear
click	css= nav-5 > .level-top > span	
verify text	css= page-title-wrapper	Training
click	id=ui-id-8	
verify text	css= page-title-wrapper	Sale
click	css= logo > img	
verify text	css= page-title-wrapper	Home Page
run script	window.scrollTo(0.5599999904632568)	

Command verify text

Target css= page-title-wrapper

Value Men

Description

سناریو 2: ثبت نام در سایت

عملکرد دکمه‌های این بخش و صحت تایپ‌ها و اعتبارسنجی بررسی شده و تا یک کاربر ثبت شود و در صورت وجود این اطلاعات از قبل به درستی ارور تکراری بودن اطلاعات را با توجه به کوکی‌ها می‌دهد.

The screenshot shows the Selenium IDE interface for a project named 'UI Test2_Amin Kiani'. The URL is 'https://magento.softwaretestingboard.com'. The test script consists of 16 steps:

Step	Command	Target	Value
1	open	/	
2	set window size	1104x824	
3	click	linkText=Create an Account	
4	click	id=firstname	
5	verify text	css=field-name-firstname span	First Name
6	type	id=firstname	amin
7	click	id=lastname	
8	verify text	css=field-name-lastname span	Last Name
9	type	id=lastname	ksani
10	click	id=email_address	
11	verify text	css=field-nth-child(3) span	Email
12	type	id=email_address	amin@gmail.com
13	click	id=password	
14	verify text	css=field-nth-child(4) > label > span	Password
15	type	id=password	123456789
16	click	id=password-confirmation	

Below the table, there is a form to add a new step with fields for Command, Target, Value, and Description.

This screenshot shows the continuation of the Selenium IDE test script, starting from step 14:

Step	Command	Target	Value
14	verify text	css=field-nth-child(4) > label > span	Password
15	type	id=password	123456789
16	click	id=password-confirmation	
17	type	id=password-confirmation	123456789
18	verify element present	id=password	123456789
19	click	css=submit > span	
20	click	id=password	
21	verify text	css=field-nth-child(4) > label > span	Password
22	type	id=password	AminKi@ni123456789
23	click	id=password-confirmation	
24	click	id=password-confirmation	
25	type	id=password-confirmation	AminKi@ni123456789
26	verify element present	id=password	AminKi@ni123456789
27	click	css=submit	

Similar to the first screenshot, there is a form at the bottom to add a new step.

سناریو 3: خرید یک کالای جدید

در این بخش صفر تا صد مراحل خرید یک کالا همراه با لاگین کردن برای توان ثبت کالا و دریافت کد معتبر سفارش انجام می‌شود که یک تست اضافی و کمکی نیز در فایل سلنیوم وارد کردم تا عدم توان خرید توسط ثبت‌نام نکرده‌ها نیز بررسی شود.

The screenshot displays the Selenium IDE interface for a project named 'UI Test2_Amin Kiani'. The URL bar shows 'https://magento.softwaretestingboard.com'. The test script is organized into a table with columns for Command, Target, and Value.

Command	Target	Value
open	/	
set window size	1061x811	
click	css=#ui-id-5 > span nth-child(2)	
verify text	css=.page-title-wrapper	Men
click	css=.mens-category-hoodies .title	
click	linkText=Marco Lightweight Active Hoodie	
verify text	css=.page-title	Marco Lightweight Active Hoodie
run script	window.scrollTo(0,24)	
click	id=option-label-color-93-item-53	
click	id=option-label-size-143-item-170	
click	id=product-addtocart-button	
run script	window.scrollTo(0,552)	
click	linkText=shopping cart	
click	css=.item > .primary	
run script	window.scrollTo(0,42.400001525878906)	
click	css=.action-auth-fonode > span	

Below the table, there are input fields for Command, Target, Value, and Description. The Command field is set to 'open', and the Target field is set to '/'. The Value and Description fields are empty.

The interface also includes a 'Log' tab and a 'Reference' tab at the bottom. The Windows taskbar at the bottom shows the time as 6:17 PM on 12/20/2024.

سناریو 4: جستجوی کالا

در این بخش به بررسی جستجوی پیشرفته برای یافتن کالای مورد نظر پرداخته شده که در کنار این بخش نیز یک تست سرچ معمولی توسط نوار منو انجام شده که هر دو به درستی انجام می‌شوند.

The screenshot displays the Selenium IDE interface for a project named 'UI Test2_Amin Kiani'. The URL bar shows 'https://magento.softwaretestingboard.com'. The test suite 'Adv Search testing' is selected, and the test 'Buy a new stuff testing' is active. The test steps are as follows:

Step	Command	Target	Value
1	open	/	
2	set window size	1058x808	
3	click	linkText=Advanced Search	
4	click	id=name	
5	type	id=name	hoodie
6	click	id=sku	
7	type	id=sku	82
8	click	id=description	
9	type	id=description	just use
10	click	id=short_description	
11	type	id=short_description	jwo
12	click	id=price	
13	type	id=price	10012
14	click	id=price_to	
15	type	id=price_to	110001
16	click	css=primary > .search > span	

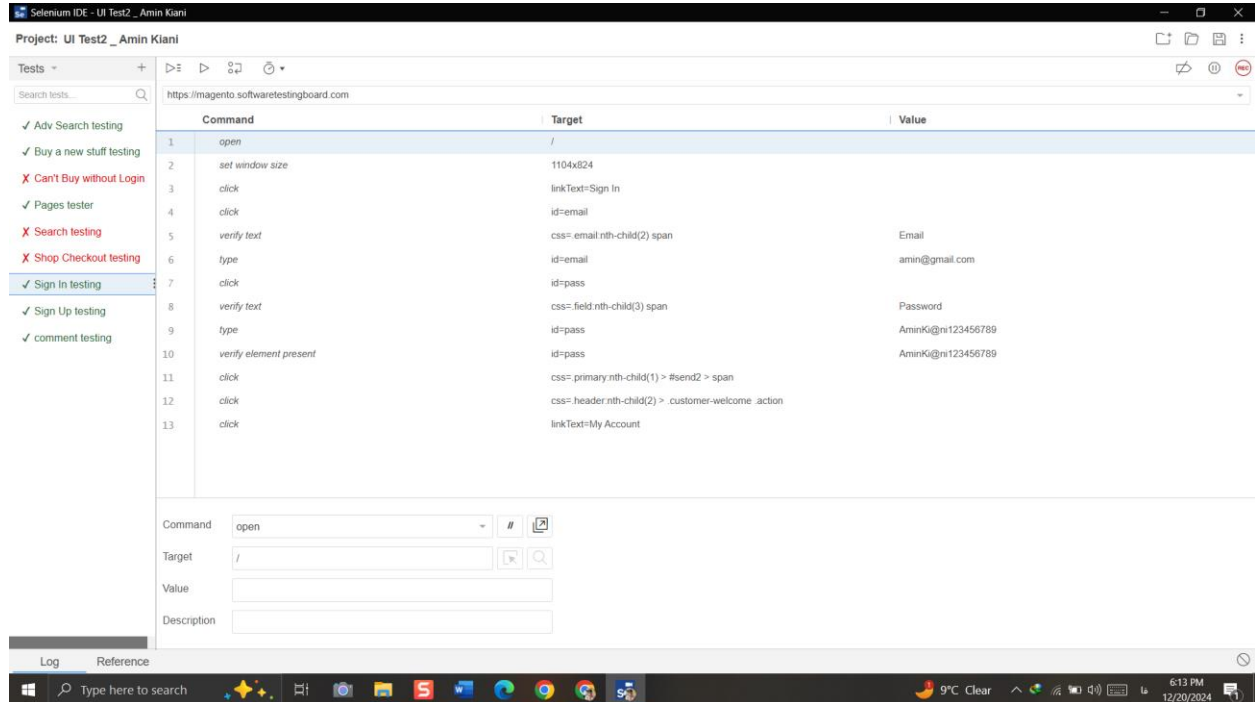
The bottom section of the screenshot shows the command editor with the following details:

- Command: open
- Target: /
- Value: (empty)
- Description: (empty)

The Windows taskbar at the bottom indicates the system time is 6:18 PM on 12/20/2024, with a temperature of 9°C and clear weather.

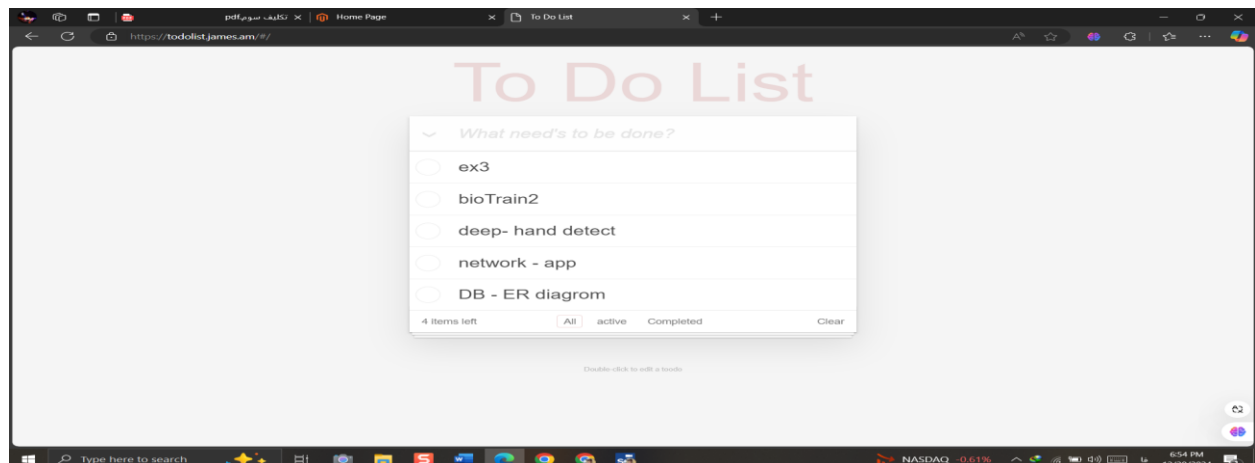
سناریو 5: ورود به اکانت شخصی

در این بخش نیز با وارد کردن اطلاعات سعی در لاگین شدن دارد که در صورت رجیستری بودن به درستی وارد شده و عدم وجود چنین اکانتی سبب جلوگیری از ورود و ارور تست می‌شود.



بخش دوم:

در ابتدا [سایت شماره 8](#) را انتخاب کرده و 5 تست مهم از عملکرد کلی آن رکورد کردم که در ادامه شرح می‌دهم:



ابزارها:

- **Unittest**: چارچوب تست استاندارد پایتون.
- **Selenium و WebDriverWait**: ابزار خودکارسازی مرورگر.
- **EC.presence_of_element_located**: برای اطمینان از این که عنصر در DOM بارگذاری شده است.
- **time.sleep**: برای ایجاد تأخیرهای زمانی و مشاهده‌ی تغییرات UI.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import unittest
import time

class TodoAppTests(unittest.TestCase):
    URL = "https://todolist.james.am/"

    def setUp(self):
        """Set up the browser before each test."""
        self.driver = webdriver.Chrome()
        self.driver.maximize_window()
        self.driver.get(self.URL)
        self.wait = WebDriverWait(self.driver, 10)

    def tearDown(self):
        """Close the browser after each test."""
        self.driver.quit()

    def add_tasks(self, tasks):
        """Helper function to add multiple tasks."""
        input_box = self.wait.until(
            EC.presence_of_element_located((By.CSS_SELECTOR,
"input.new-todo")))
        for task in tasks:
            input_box.send_keys(task + Keys.RETURN)
            time.sleep(1) # Delay after adding each task

    def test_add_to_do(self):
        """Test adding tasks and verifying their count."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
```

```

        self.add_tasks(tasks)

        # Verify the number of tasks added
        time.sleep(2) # Delay before verification
        list_items = driver.find_elements(By.CSS_SELECTOR,
"ul.todo-list li")
        self.assertEqual(len(list_items), len(tasks),
                          f"Expected {len(tasks)} tasks, but found
{len(list_items)}")

    def test_check_items_left(self):
        """Test the remaining items count."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
        self.add_tasks(tasks)

        # Complete one task
        first_task_checkbox = driver.find_element(By.CSS_SELECTOR,
"ul.todo-list li:first-child .toggle")
        first_task_checkbox.click()
        time.sleep(2) # Delay after clicking checkbox

        # Verify the remaining tasks count
        remaining_count = driver.find_element(By.CSS_SELECTOR,
"footer.footer .todo-count strong").text
        self.assertEqual(int(remaining_count), len(tasks) - 1,
                          f"Expected remaining count to be
{len(tasks) - 1}, but found {remaining_count}")

    def test_check_filter(self):
        """Test filtering tasks by 'active' and 'completed'."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
        self.add_tasks(tasks)

        # Complete some tasks
        driver.find_element(By.CSS_SELECTOR, "ul.todo-list
li:first-child .toggle").click()
        time.sleep(1) # Delay after completing a task
        driver.find_element(By.CSS_SELECTOR, "ul.todo-list li:last-
child .toggle").click()
        time.sleep(1)

        # Check active filter
        driver.find_element(By.CSS_SELECTOR, 'ul.filters li
a[href="#/active"]').click()
        time.sleep(2) # Delay after switching to active filter
        active_todos = driver.find_elements(By.CSS_SELECTOR,
"ul.todo-list li")

```

```

        active_texts = [todo.text for todo in active_todos]
        self.assertEqual(active_texts, ["task 2"], "Active filter
did not return correct tasks.")

        # Check completed filter
        driver.find_element(By.CSS_SELECTOR, 'ul.filters li
a[href="#/completed"]').click()
        time.sleep(2) # Delay after switching to completed filter
        completed_todos = driver.find_elements(By.CSS_SELECTOR,
"ul.todo-list li")
        completed_texts = [todo.text for todo in completed_todos]
        self.assertEqual(set(completed_texts), {"task 1", "task
3"},
                                "Completed filter did not return correct
tasks.")

    def test_delete_item(self):
        """Test deleting a task."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
        self.add_tasks(tasks)

        # Delete the first task
        first_task = driver.find_element(By.CSS_SELECTOR, "ul.todo-
list li:first-child")
        delete_button = first_task.find_element(By.CSS_SELECTOR,
"button.destroy")
        driver.execute_script("arguments[0].click();",
delete_button)
        time.sleep(2) # Delay after deleting task

        # Verify the task is deleted
        remaining_tasks = driver.find_elements(By.CSS_SELECTOR,
"ul.todo-list li label")
        remaining_texts = [task.text for task in remaining_tasks]
        self.assertNotIn("task 1", remaining_texts, "Task 'task 1'
was not deleted.")

    def test_clear_completed(self):
        """Test clearing all completed tasks."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
        self.add_tasks(tasks)

        # Complete all tasks
        toggles = driver.find_elements(By.CSS_SELECTOR, "ul.todo-
list li .toggle")
        for toggle in toggles:
            toggle.click()

```

```

        time.sleep(1) # Delay after completing each task

        # Clear completed tasks
        driver.find_element(By.CSS_SELECTOR, "button.clear-completed").click()
        time.sleep(2) # Delay after clearing tasks

        # Verify all tasks are cleared
        remaining_tasks = driver.find_elements(By.CSS_SELECTOR, "ul.todo-list li")
        self.assertEqual(len(remaining_tasks), 0, "Completed tasks were not cleared.")

    def test_toggle_all(self):
        """Test toggling all tasks as completed."""
        driver = self.driver
        tasks = ["task 1", "task 2", "task 3"]
        self.add_tasks(tasks)

        # Use JavaScript to click the toggle-all checkbox to avoid interception
        toggle_all = driver.find_element(By.CSS_SELECTOR, "input#toggle-all")
        driver.execute_script("arguments[0].click();", toggle_all)

        # Wait a moment to allow UI updates
        import time
        time.sleep(2)

        # Verify all tasks are marked as completed
        completed_todos = driver.find_elements(By.CSS_SELECTOR, "ul.todo-list li.completed")
        self.assertEqual(len(completed_todos), len(tasks), "Not all tasks were marked as completed.")

if __name__ == "__main__":
    unittest.main()

```

کلاس بالا، شامل چند تست برای بررسی قابلیت‌های یک برنامه‌ی مدیریت وظایف (To-Do List) است. که سناریوهای آن به شرح زیر اند:

test add to do

- **هدف:** تست اضافه کردن وظایف به لیست.
- **کارکرد:**
 - چند وظیفه به لیست اضافه می‌شود.
 - بررسی می‌کند که تعداد وظایف موجود در لیست برابر با تعداد وظایف اضافه‌شده باشد.
- **موفقیت:** زمانی که تعداد وظایف موجود در DOM با تعداد وظایف اضافه‌شده برابر باشد.

test check items left

- **هدف:** بررسی تعداد وظایف باقی‌مانده (وظایفی که هنوز تکمیل نشده‌اند).
- **کارکرد:**
 - چند وظیفه به لیست اضافه می‌شود.
 - یکی از وظایف به‌عنوان تکمیل‌شده علامت‌گذاری می‌شود.
 - تعداد وظایف باقی‌مانده بررسی می‌شود و با مقدار مورد انتظار مقایسه می‌شود.
- **موفقیت:** زمانی که شمارنده‌ی وظایف باقی‌مانده مقدار درست را نمایش دهد.

test check filter

- **هدف:** بررسی فیلتر کردن وظایف بر اساس وضعیت ("فعال" یا "تکمیل‌شده").
- **کارکرد:**
 - وظایفی اضافه می‌شوند.
 - تعدادی از وظایف به‌عنوان تکمیل‌شده علامت‌گذاری می‌شوند.
 - فیلتر "فعال" انتخاب می‌شود و بررسی می‌کند که فقط وظایف فعال نمایش داده شوند.
 - فیلتر "تکمیل‌شده" انتخاب می‌شود و بررسی می‌کند که فقط وظایف تکمیل‌شده نمایش داده شوند.
- **موفقیت:** زمانی که هر فیلتر، وظایف درست را نمایش دهد.

test delete item

- **هدف:** تست حذف یک وظیفه از لیست.
- **کارکرد:**
 - چند وظیفه به لیست اضافه می‌شود.
 - اولین وظیفه حذف می‌شود.
 - بررسی می‌کند که وظیفه‌ی حذف‌شده دیگر در لیست وجود نداشته باشد.
- **موفقیت:** زمانی که وظیفه‌ی حذف‌شده دیگر در DOM وجود نداشته باشد.

test clear completed

- **هدف:** تست پاک کردن تمام وظایف تکمیل‌شده.
- **کارکرد:**
 - چند وظیفه به لیست اضافه می‌شود.
 - تمام وظایف به‌عنوان تکمیل‌شده علامت‌گذاری می‌شوند.
 - با استفاده از دکمه‌ی "پاک کردن تکمیل‌شده‌ها"، تمام وظایف پاک می‌شوند.
 - بررسی می‌کند که لیست وظایف خالی باشد.
- **موفقیت:** زمانی که تمام وظایف تکمیل‌شده از DOM حذف شوند.

test toggle all

- **هدف:** تست علامت‌گذاری همه‌ی وظایف به‌عنوان تکمیل‌شده.
- **کارکرد:**
 - چند وظیفه به لیست اضافه می‌شود.
 - از دکمه‌ی "انتخاب همه" استفاده می‌شود تا همه‌ی وظایف به‌عنوان تکمیل‌شده علامت‌گذاری شوند.
 - بررسی می‌کند که تمام وظایف در DOM دارای کلاس completed باشند.
- **موفقیت:** زمانی که همه‌ی وظایف به‌عنوان تکمیل‌شده در DOM نمایش داده شوند.