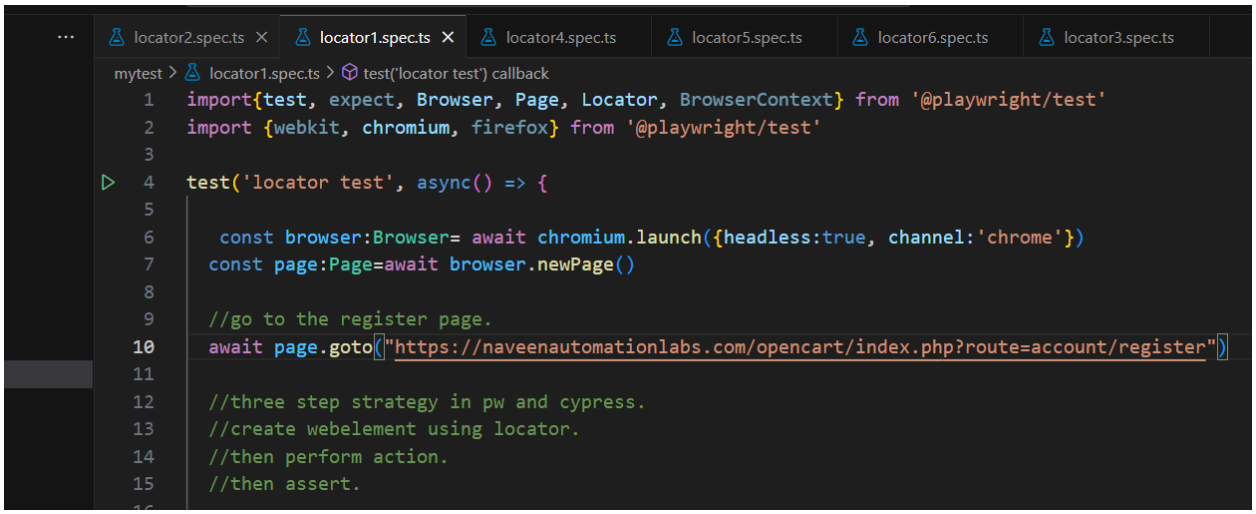
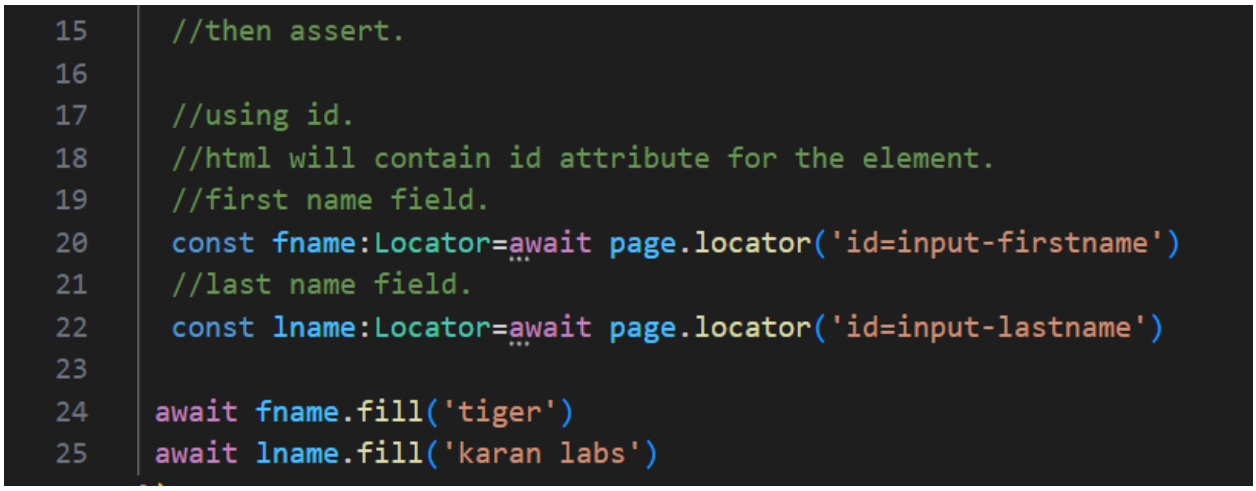
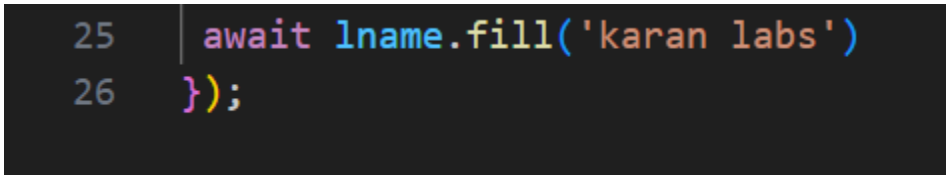
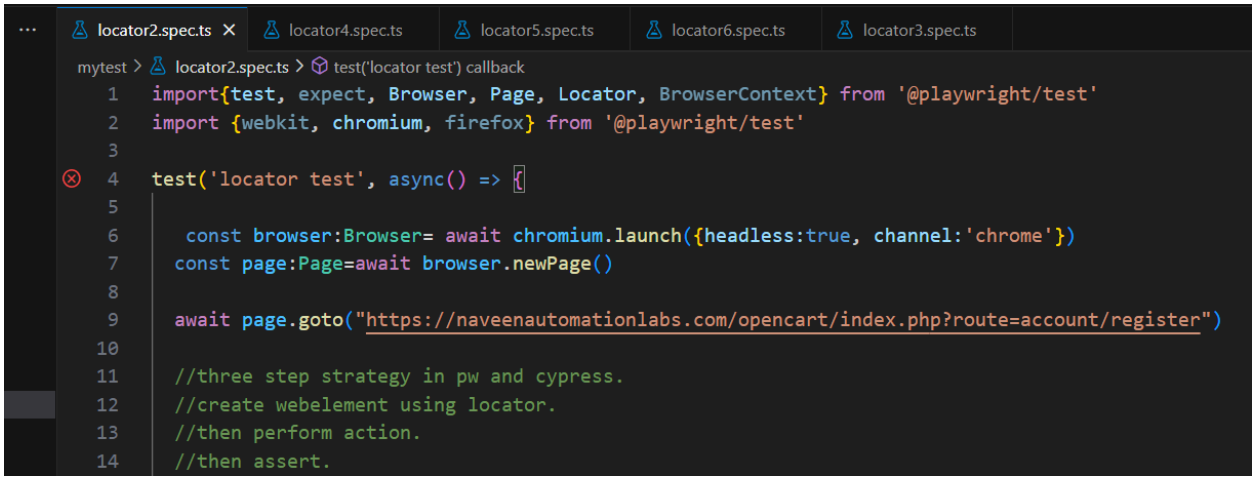
Locator -
When id attribute is present.

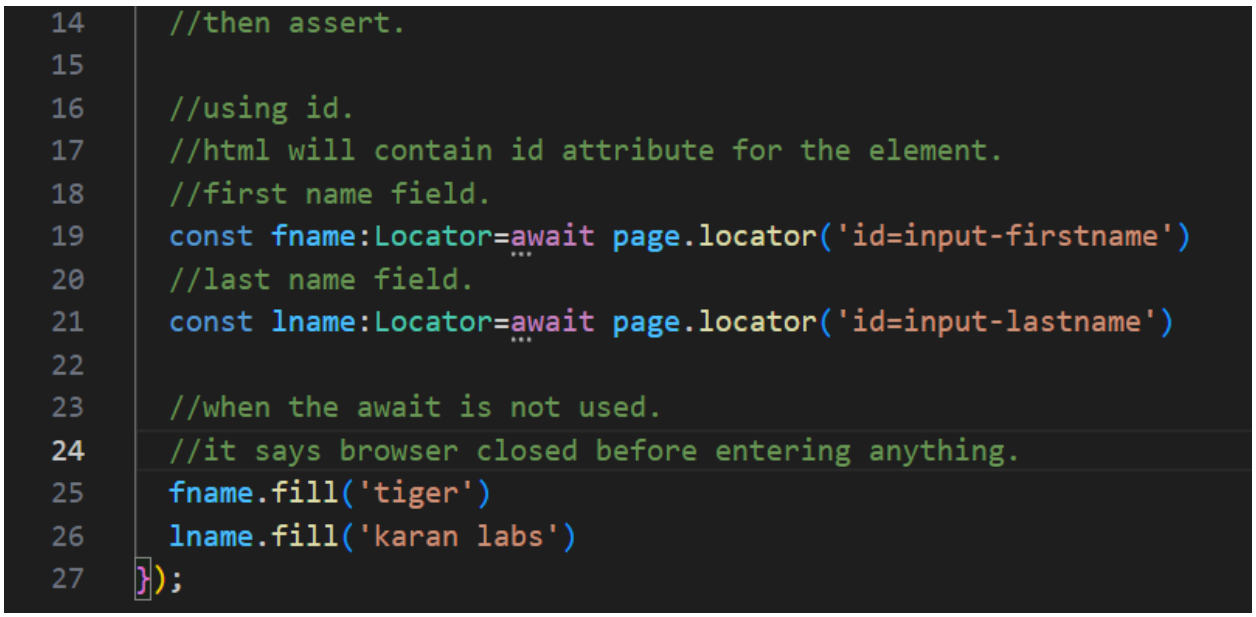




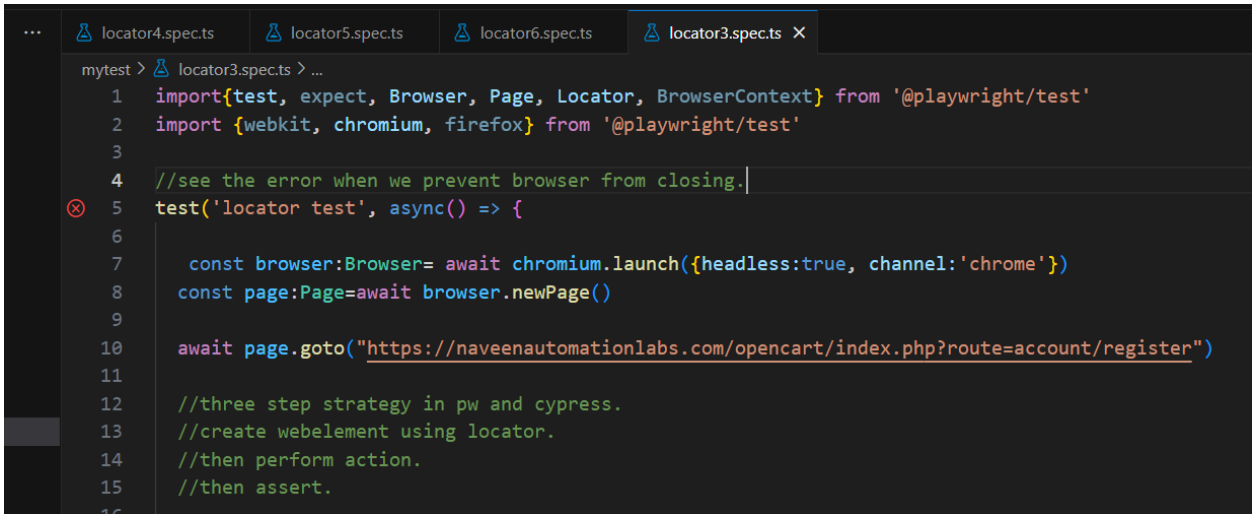


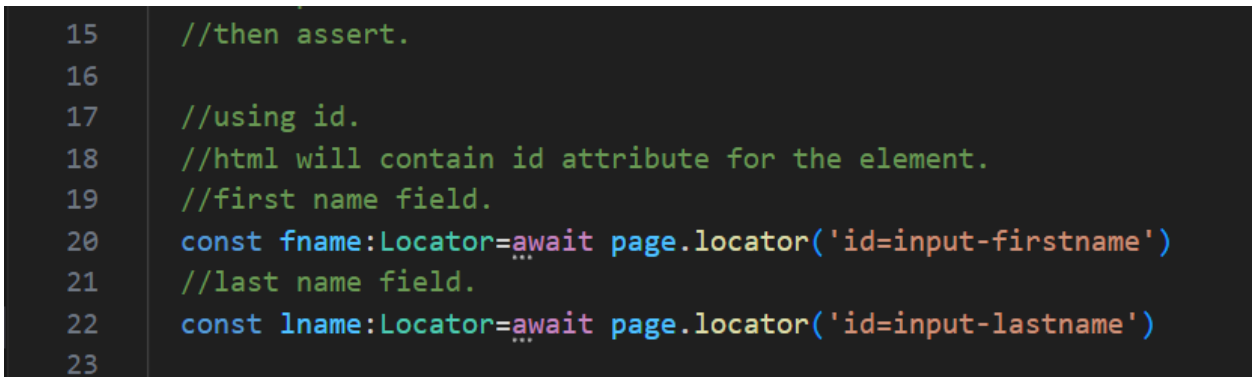
Use await else error





To avoid closing browser after run automatically


locator4.spec.ts
locator5.spec.ts
locator6.spec.ts
locator3.spec.ts
mytest > locator3.spec.ts > ...
import{test, expect, Browser, Page, Locator, BrowserContext} from '@playwright/test'
import {webkit, chromium, firefox} from '@playwright/test'
//see the error when we prevent browser from closing.
test('locator test', async() => {
const browser:Browser= await chromium.launch({headless:true, channel:'chrome'})
const page:Page=await browser.newPage()
await page.goto("https://naveenautomationlabs.com/opencart/index.php?route=account/register")
//three step strategy in pw and cypress.
//create webelement using locator.
//then perform action.
//then assert.


//then assert.
//using id.
//html will contain id attribute for the element.
//first name field.
const fname:Locator=await page.locator('id=input-firstname')
//last name field.
const lname:Locator=await page.locator('id=input-lastname')


await fname.fill('tiger')
await lname.fill('karan labs')
//to avouid closing browser immediately after running.
await new Promise(()=>{})
});

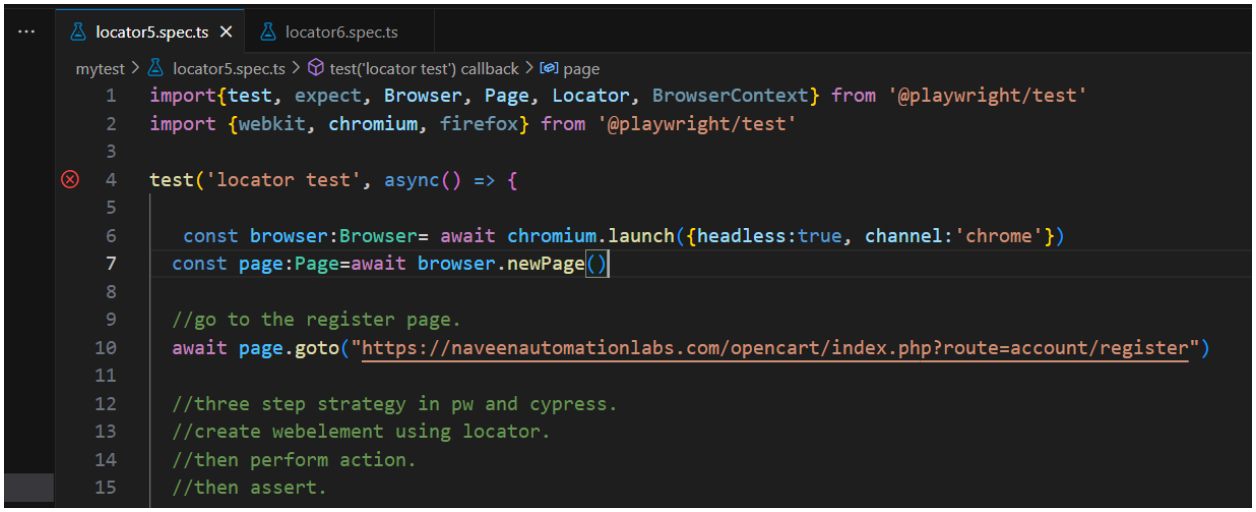
```
17    //using id
PROBLEMS  OUTPUT  DEBUG CONSOLE  TEST RESULTS  TERMINAL  PORTS  PLAYWRIGHT

Running 1 test using 1 worker
  1) [chromium] › mytest\locator3.spec.ts:4:5 › locator test

    Test timeout of 30000ms exceeded.

    Error Context: test-results\locator3-locator-test-chromium\error-context.md

  1 failed
    [chromium] › mytest\locator3.spec.ts:4:5 › locator test
```

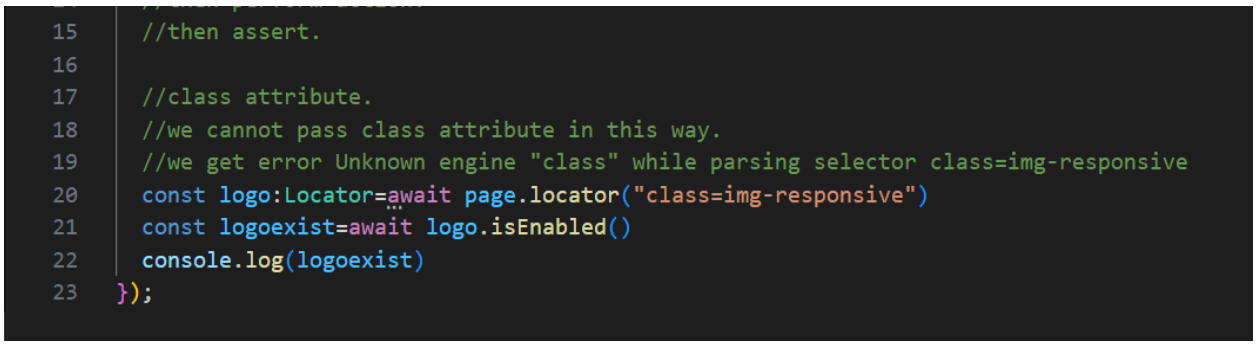
When class name attribute is present-

```
locator4.spec.ts  locator5.spec.ts  locator6.spec.ts
mytest > locator4.spec.ts > test('locator test') callback
1  import{test, expect, Browser, Page, Locator, BrowserContext} from '@playwright/test'
2  import {webkit, chromium, firefox} from '@playwright/test'
3
4  test('locator test', async() => {
5
6     const browser:Browser= await chromium.launch({headless:true, channel:'chrome'})
7    const page:Page=await browser.newPage()
8
9    //go to the register page.
10   await page.goto("https://naveenautomationlabs.com/opencart/index.php?route=account/register")
11
12   //three step strategy in pw and cypress.
13   //create webelement using locator.
14   //then perform action.
15   //then assert.
```

```
15   //then assert.
16
17   //class attribute.
18   //class name attribute is present.
19   //check if the logo is enabled.
20   const logo:Locator=await page.locator(".img-responsive")
21   const logoexist=await logo.isEnabled()
22   console.log(logoexist)
23 });
```

When we pass class like this then error-





Use the visible text on the browser-
Check if text works for button and links also-

locator6.spec.ts

mytest > locator6.spec.ts > ...

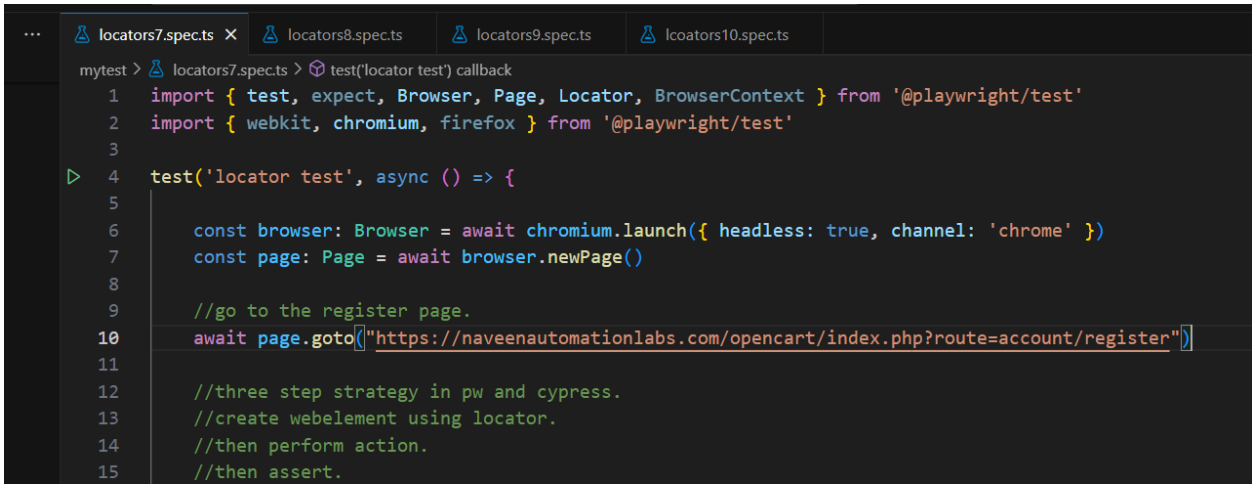
```ts
import{test, expect, Browser, Page, Locator, BrowserContext} from '@playwright/test'
import {webkit, chromium, firefox} from '@playwright/test'

test('locator test', async() => {

   const browser:Browser= await chromium.launch({headless:true, channel:'chrome'})
  const page:Page=await browser.newPage()

  //go to the register page.
  await page.goto("https://naveenautomationlabs.com/opencart/index.php?route=account/register")

  //three step strategy in pw and cypress.
  //create webelement using locator.
  //then perform action.
  //then assert.
```

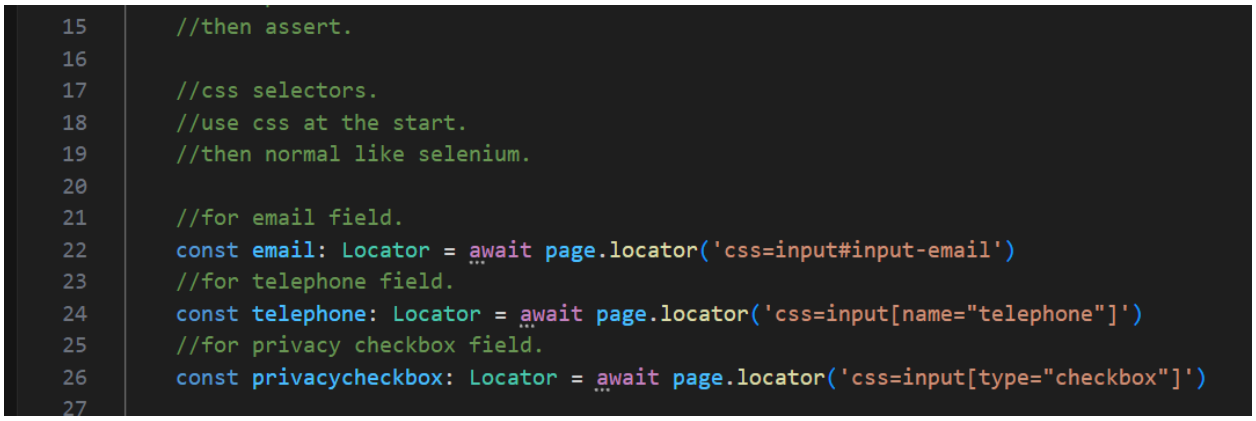
```ts
  //then assert.

  //text attribute.
//we capture using the visible text on browser.
//check register account seen.
const header:Locator=await page.locator('text=Register Account')
const headerenabled=await header.isEnabled()
console.log(headerenabled)
```

```ts


//text works on any visible element including buttons, links etc.

//check continue button seen.
const continuebutton:Locator=await page.locator('text=Continue')
const conitnueisenabled=await continuebutton.isEnabled()
console.log(conitnueisenabled)
```

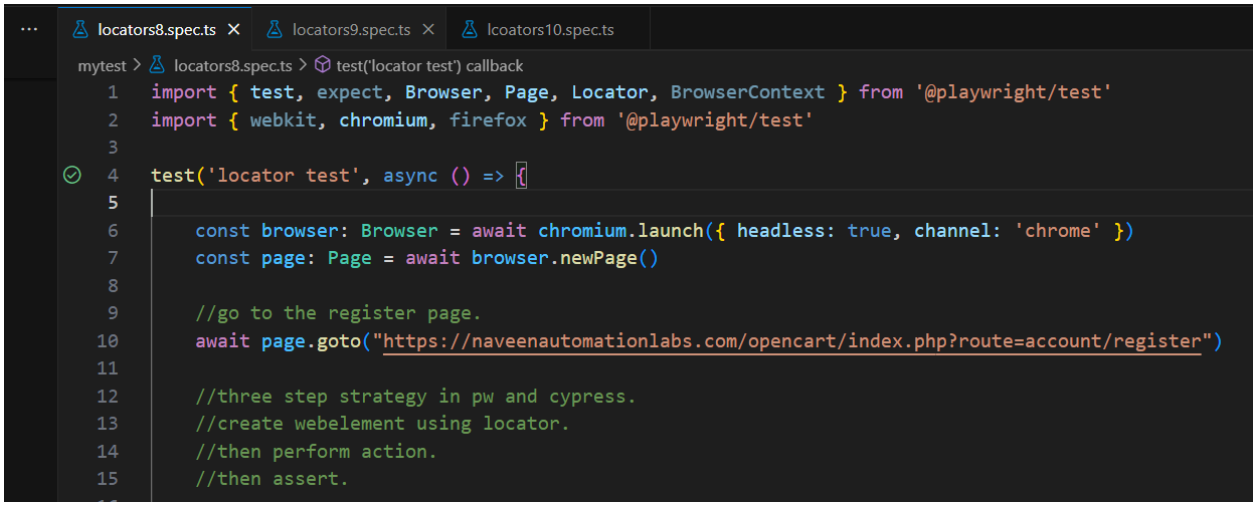
```ts

 //check forgotten password link seen.
const forgotpasswordlink:Locator=await page.locator('text=Forgotten Password')
const forgotPasswordEnabled=await forgotpasswordlink.isEnabled()
console.log(forgotPasswordEnabled)
});
```

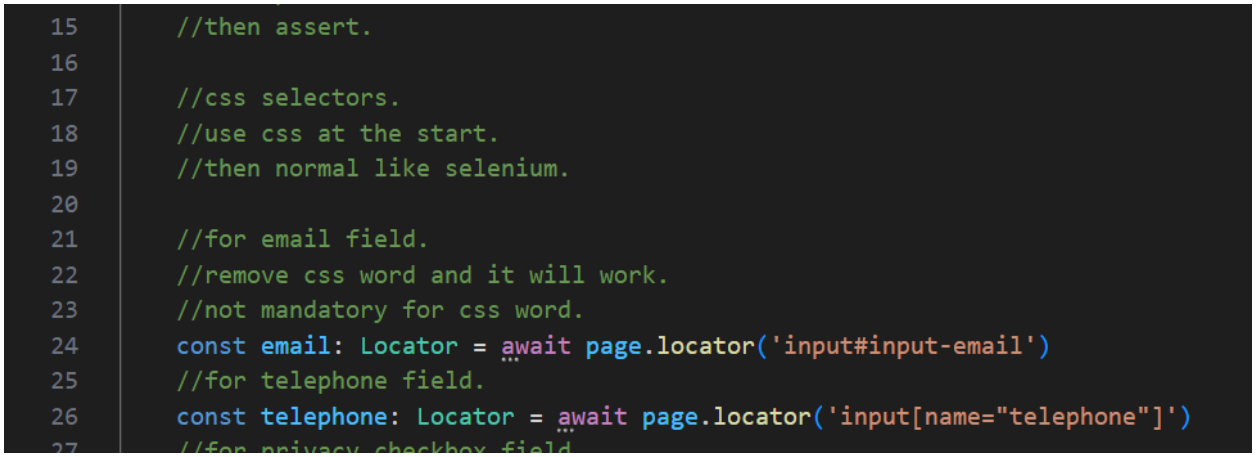
Css
Use css as the key.
Then use normal way as selenium.



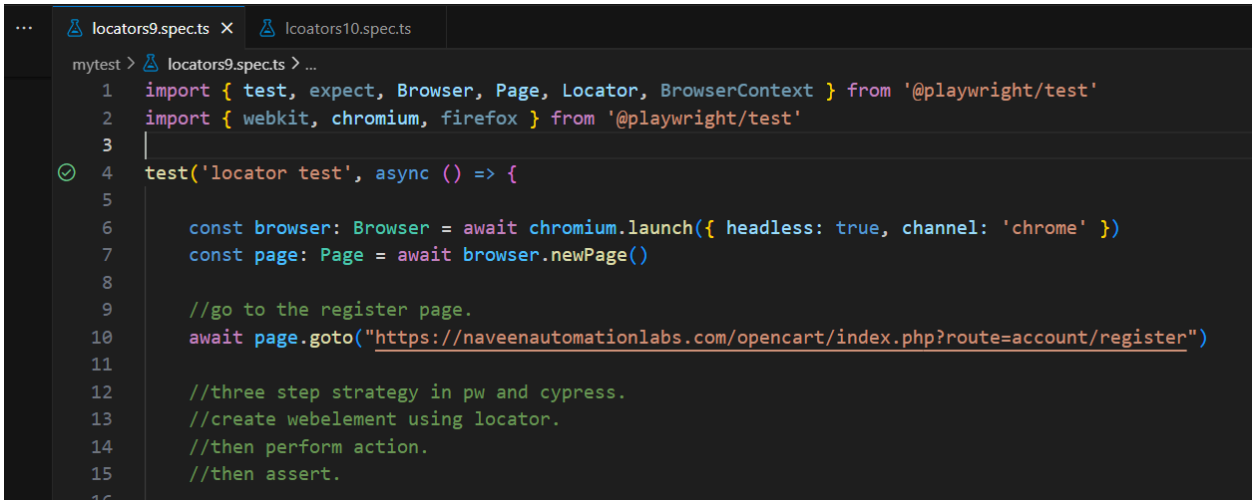


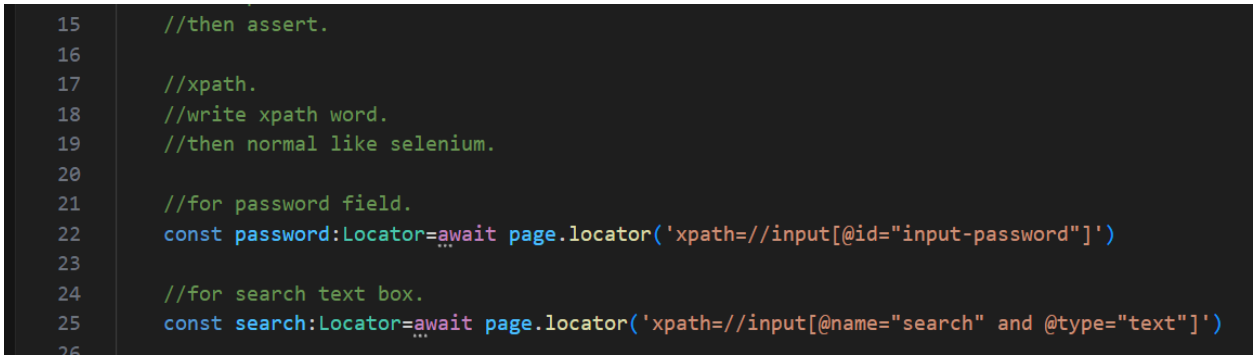
By default all locators in pw is css.
Even if we remove css it works.

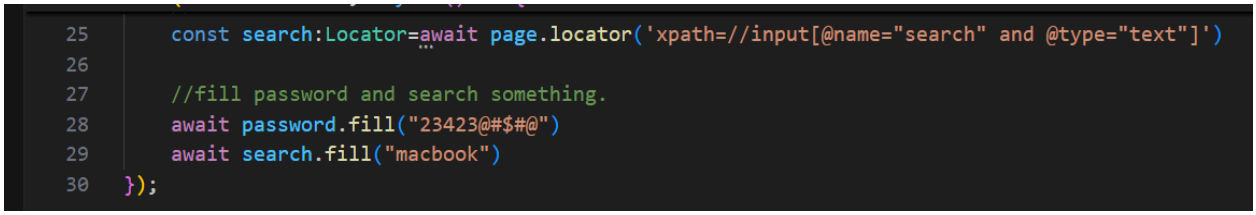




Xpath-
Initial write xpath.
Rest same as selenium.







Without xpath word also it works fine-

lcoators10.spec.ts

mytest > lcoators10.spec.ts > test('locator test') callback

```ts
import { test, expect, Browser, Page, Locator, BrowserContext } from '@playwright/test'
import { webkit, chromium, firefox } from '@playwright/test'

test('locator test', async () => {

    const browser: Browser = await chromium.launch({ headless: true, channel: 'chrome' })
    const page: Page = await browser.newPage()

    //go to the register page.
    await page.goto("https://naveenautomationlabs.com/opencart/index.php?route=account/register")

    //three step strategy in pw and cypress.
    //create webelement using locator.
    //then perform action.
    //then assert.
```

```ts
    //then assert.

    //xpath.
    //write xpath word.
    //then normal like selenium.

    //for password field.
    //without xpath word also it works.
    const password:Locator=await page.locator('//input[@id="input-password"]')
```

```ts
    const password:Locator=await page.locator('//input[@id="input-password"]')

    //for search text box.
    const search:Locator=await page.locator('//input[@name="search" and @type="text"]')

    //fill password and search something.
    await password.fill("23423@#$#@")
    await search.fill("macbook")
});
```