Two test folder – speciallocators.spec.js and calendar.spec.js for the entire section\_7

32. Understand how GetByLabel & Playwright UI Runner works with an example

getBylabel - if any text is written inside the label tag<label> </label>, then you can actually go to that point to perform actions. You just need to select that label text and give it here.

 when there is a scope of typing in the field, then getbyLabel is not that efficient.

Check method – ensure that checkbox or radio element are checked.

Click and check almost same.

If u want to select the options form the dropdown. There is a direct method which will tell you that “selectoption”. So this will be valid only when you are selecting options within this select tag. <select> </select>

npx playwright test –ui 🡪playwirght test terminal will open ( test runner) ( double hyphen)

33. Filtering elements with GetByRole,GetByText and perform chaining methods in step

If you have a placeholder attribute for any of the field you can use direct “getbyplaceholder” method directly it can pointout to the field and type/fill the values.

Getbyrole - playwright will filter all the buttons what it see on the page. It gets the information about all the buttons in the page out of all the buttons, which button you need that is

important so that you have to write in your second argument with curly braces ( button name is submit )

If the tag name is “a” then it’s a link

34. code

const {test, expect} = require('@playwright/test');

test('Playwright special locator', async ({page})=>    {

    await  page.goto("https://rahulshettyacademy.com/angularpractice/");

    await page.getByLabel("Check me out if you Love IceCreams!").click();

    await page.getByLabel("Employed").check();

    await page.getByLabel("Gender").selectOption("Female");

    await page.getByPlaceholder("Password").fill("abc1234");

    await page.getByRole("button", {name : 'Submit'}).click();

    await page.getByText("Success! The Form has been submitted successfully!.").isVisible();

    await page.getByRole("link",{name : "Shop"}).click();

    await page.locator("app-card").filter({hasText : 'Nokia Edge'}).getByRole("button").click();

});

35. Strategy on handling Calendars automation using Playwright

Note : 1. when there are multiple classes separated by space, you can choose only one class.

You cannot take all the classes.

2. this is a string(6) you cannot apply minus to string. You need to convert that into number. ( monthNumber-1)

So in JavaScript you can convert a string to number by simply wrapping that in the number of

month number minus one. (Number(monthNumber)-1).

expose a test variable And then you also need expect for assertions from where you are doing. We are exporting it from our core playwright @playwright test.

This takes two arguments.

First argument we can say calendar validations, and in the second argument is a place where it takes the actual code. And here I am sending page property.

So that's how you can access page inside the code.

36. End to end example of validating calendars with assertion in playwright

37. code

const {test,expect} = require("@playwright/test");

test("Calendar validations",async({page})=>

{

    const monthNumber = "6";

    const date = "15";

    const year = "2027";

    const expectedList = [monthNumber,date,year];

    await page.goto("https://rahulshettyacademy.com/seleniumPractise/#/offers");

    await page.locator(".react-date-picker\_\_inputGroup").click();

    await page.locator(".react-calendar\_\_navigation\_\_label").click(); // double click on the same button

    await page.locator(".react-calendar\_\_navigation\_\_label").click();

    await page.getByText(year).click();

    await page.locator(".react-calendar\_\_year-view\_\_months\_\_month").nth(Number(monthNumber)-1).click();

    await page.locator("//abbr[text()='"+date+"']").click(); //xpath

//assertions

    const inputs = await page.locator(".react-date-picker\_\_inputGroup input");

    for (let index = 0; index <inputs.length; index++)

    {

        const value =inputs[index].getAttribute("value");

        expect(value).toEqual(expectedList[index]);

    }

});