Node.js trebuie sa fie instalat pe masina (node -v sau npm -v)

npm init -y 🡪 create package.json in proiect asemanator cu pom.xml

install cypress on local folder ( se poate instala pentru fiecare director)

1.install cypress

npm -g install cypress

npm cypress -v

2.check cypress version : npm cypress -v

3.install project structure : npx cypress open

Structura de baza pentru un proiect

Graphical user interface

Description automatically generated

Plugins 🡪 index.js

Support 🡪command.js si index.js

4. testele se adaga in folderul : integration. Fisier .js

Se adauga in fiser : // <reference types = “cypress” />

Avem nevoie de un test runner : mocha (build-in cypress)

it('search on google',function(){

    cy.get('www.google.com');

})

it('search on google',()=>{

    cy.get('www.google.com');

})

Run basic cypress : npx cypress open

5. configuration : cypress.json

- “watchForFileChanges”:true (save and run)

-“defaultCommandTimeout”: 6000 (time out la nivel global)

AUTO COMPLEATE :

/// <reference types="Cypress" />

Sau :

jsconfig.json/ [tsconfig.json](http://www.typescriptlang.org/docs/handbook/tsconfig-json.html) on root project

{

"include": ["./node\_modules/cypress", "cypress/\*\*/\*.js"]

}

{

"compilerOptions": {

"allowJs": true,

"types": ["cypress"]

},

"include": ["\*\*/\*.\*"]

}

**Pentru a rula code TypeScript**

1. npm install typescript
2. Se instaleaza plug-in pentru TypeScript 🡪 code runner

A screenshot of a computer

Description automatically generated with medium confidence

3. npx tsc --init --types cypress --lib dom,es6

6. Command line :

cy.get(‘object.locator,{timeout:5000}) sau cy.contains(object.locator)

Default se incerca de doua ori

pentru a gasi elementul in pagina apoi pica testul

hard wait : cy.wait(5000)

cy.visit(‘www.test.com’)

cy.get(‘objext.locator’).click()

cy.get(‘object.locator’).type(‘text{enter}’) SAU .type(‘{enter}’)

cy.contains(‘text’).click() (find element base on text)

cy.wait(4000)

cy.get(‘object.locator’).find(descendent.object.locator).click()

7. run only a test from a suite : it.only(‘ ’, function(){})

8. use xpath :

npm install -D cypress-xpath

Then include in your project's cypress/support/index.js

require('cypress-xpath')

Ex : cy.xpath('//h1/text()')

9. Situatie cand cypress nu poate identifica obiectul , dar gaseste o zona care continue obiectul

Cy.get(‘pana unde gasete el’).find(‘#ID-ul obictului in cauza’).click()

10. Assertion

Implicit – build assertion

cy.get(‘object.locator’).

should(‘contain’,text’) ;

should(‘have.class’,’numele clasei’)

should(‘have.text’,’’);

should(‘have.class’,’’);

should(‘have.html’,’’);

should(‘be.disabled/enabled/empty/selected/focused’

invoke(‘attr’,id).should(‘equal’,’idValue’)

should-equal

in cazul in care vream sa verificam ca exista un obiect si il identificam dupa ID

    cy.get('#resetBtn')

                .invoke('attr','id')

                         .should('equal','resetBtn');

and(); folosit pentru inlantuire de aserturi

    cy.get('#searchBtn')

                .should('have.class','searchbutton')

                .and

     cy.get('#resetBtn')

                .invoke('attr','id')

                         .should('equal','resetBtn');

Daca vream sa facem, assert cu timeout : cy.get(‘obj.indentificator’**,{timeout:6000}).**should…

Explicit – assertion commands

Expect() ;

Expect(true).to.be.true

Let x = ‘test’;

expect(name).to.be.equal(‘test’)

to.not.equal()

to.be.a(‘string’)

to.be.true

to.be.false

to.be.null

to.exist

assert.equal / notEqual /strictEqual/isAbove/exist/notExist

**Obs:**

🡪DESCRIBE 🡪beforeEach🡨

describe('test', function () {

    this.beforeEach(function () {

        loginPage.autentication();

    })

it()…..it()…..it()…..it()…..it()……

})

POM

Objects and Methods separate from test script

REPORTER

**mochawesome reporter :**

npm i mochawesome-report-generator

npm i mochawesome-merge

npm install mochawesome

va apare in package json :

    "mochawesome": "^7.0.1",

    "mochawesome-merge": "^4.2.1",

    "mochawesome-report-generator": "^6.0.1"

npm install mochawesome --save-dev

npm install mochawesome-merge --save-dev

**configure reporter**

se adauga in cypress.json :

        "reporter": "mochawesome",

        "reporterOptions": {

          "charts": true,

          "overwrite": false,

          "html": false,

          "json": true,

          "reportDir": "cypress/report/mochawesome-report"

         }

npx cypress run --reporter mochawesome

se genereaza un folder reports cu cate 1 json file pentru fiecare fisier de test

addition : npm install rimraf 🡪 delete rm on windows

**Merge multiple json report file**

npx mochawesome-merge cypress/report/mochawesome-report/\*.json ＞ cypress/report/output.json

---LOCATIE-----SURSA DESTINATIE

**npx mochawesome-merge .\cypress\report\mochawesome-report\\*.json > report1.json**

in caz de probleme…

npx mochawesome-merge cypress/report/mochawesome-report/\*.json | out-file -encoding ascii cypress/report/output.json

**npx mochawesome-merge .\cypress\report\mochawesome-report\\*.json | out-file -encoding ascii report1.json**

**se genereaza correct fisiereul**

**Generate single html report**

npx marge cypress/report/output.json --reportDir ./ --inline

**npx marge .\report1.json -- reportDir .\cypress\reports**

genereaza un fisier de raport html

**Add report generation command in package.json scripts :**

**Se generaza automat raportul fara a mai rula manual comenzile**

**In package.json**

Obs : npm run test 🡪 "echo \"Error: no test specified\" && exit 1"

Se adauga :

“merge-reports”:”” comezile excutate de mai sus **fara npx**

“generate-htmlreport”:””

**. Run test : npx cypress run**

**Va rula proiectul cu tot cu comenzile de generare raport 🡪 npm run test**

**POSIBIL DE INSTALAT :**

**Cypress Helper**

**----------------------------------------------------**

* **Citit din ecran si facut assert (3 implementari)**
* **Selectare valoare din dropdown ( 2 implemetari) ( 1 functions/commads)**
* **Validare valoare din dropdown**
* **Validare dynamica pentru numar de valori din lista**
* **Mouse hover**
* **Validare campuri/butoane…**
* **Citim datele pentru mediu : link, user/pass , db config dintr-un fisier cypress.jos**
* **Acelasi test sa se execute cu date multiple**
* **Un test sa foloseasca date dintr-un fisier extern pentru a nu avea date**

**direct in fisier**

**---------------------------------------------------------------------------------------------**

* **Selectare valoare din dropdown si assert (foma clasica de implemtare)—in work**

[**https://www.youtube.com/watch?v=mUyl07TOca8**](https://www.youtube.com/watch?v=mUyl07TOca8)

[**https://www.youtube.com/watch?v=y\_L88Yzd4-Y**](https://www.youtube.com/watch?v=y_L88Yzd4-Y)

* **Select din baza si folosit in test DONE**
* **Click print si inchidere fereastra noua deschisa**

**---------------------------------------------------------------------------------------------**

**---------------------------------------------**

**baza**

**https://www.youtube.com/watch?v=CVjd6emyoIA&list=PLhW3qG5bs-L9LTfxZ5LEBiM1WFfvX3dJo&index=10**

**https://www.youtube.com/watch?v=OIAzwr-\_jhY**

**https://www.youtube.com/watch?v=avb-VDa3ZG4**

[**https://www.youtube.com/c/TheTestingAcademy/search?query=cypress**](https://www.youtube.com/c/TheTestingAcademy/search?query=cypress)

**!!! citire din ecran si assert !!!**

**https://www.youtube.com/watch?v=SInWGlUNxbY&t=576s**

**-------------**

**https://www.youtube.com/watch?v=l5eaLPHGAeM**

**https://www.youtube.com/watch?v=8UsyREnqwEA**

[**https://www.youtube.com/watch?v=C2DjGl5a\_-Y**](https://www.youtube.com/watch?v=C2DjGl5a_-Y)

* **De discutat**

**Read value :**

   cy.xpath(cautareVehicul.inputSerieSasiu).invoke('val').then((myValue:any)=>{

        cy.log(myValue);

**Citim valoarea unei proprietati al unui obiect si citiva valoarea text a unui obiect din pagina**

    it('se citeste valoarea unei proprietati a unui obiect', function(){

        cy.xpath("XXXXXXXXXX").invoke('prop','className').then((myValue: any)=>{

              cy.log("afiseaza valoare : " + myValue);

  // Explicatie : se selecteaza obiectul elements-->proprieties-->selctam prop care ne intereseaza(ex: numele clasei)

        })

        cy.xpath("XXXXXXXXXX").click().invoke('prop','className').then((myValue: any)=>{

            cy.log("afiseaza valoare : " + myValue);

// Explicatie : se poate da click !! si apoi logica este la fel -->se selecteaza obiectul elements-->proprieties-->selctam prop care ne intereseaza(ex: numele clasei)

      })

    })

    it('se citeste valoarea unui obiect din pagina' , function(){

        cy.xpath("CCCCCCCCCCCCC").invoke('text').then((text1) => {

            //cy.log(text1);

            // text1.charAt(text1.length - 1)

            cy.log(text1.charAt(text1.length - 1));

            let x = parseInt(text1.charAt(text1.length - 1));

        })

    })

**Comparam doua stringuri : (unul este expected si celalat este citit din ecran**

        cy.xpath(element).invoke('text').then((actualMessage: any) => {

            //cy.log(text1);

            // text1.charAt(text1.length - 1)

            cy.log(actualMessage);

            let result = actualMessage.localeCompare(expectedMessage);

            if (result == 0) {

                result = true;

                expect(result).to.be.true;

            } else {

                result = false;

                expect(result).to.be.true;

            }

})

**Parcurgem un table in cypress :**

                for (let i = 1; i <= 4; i++) {

                    cy.xpath(`//div[2]/table/tbody/tr[1]/td[${i + 1}]`).invoke('text').then((valoareSearch) => {

                        searchValues[i] = valoareSearch;

                        cy.log(searchValues[i]);

                        cy.xpath(pentruTiparirePage.getInputSearch()).type(searchValues[i]);

                        cy.xpath(`//div[2]/table/tbody/tr[1]/td[${i + 1}]`).should('have.text', searchValues[i])

                        cy.xpath(pentruTiparirePage.getInputSearch()).clear();

                    })

                }

**Pentru accesarea unui vector definit in exteriorul unei functii : este problema de asincron**

It is because JS works asynchronously your log statement is running before the value of minPrice is updated. To make sure that the log has the updated minprice, we can use then to make sure the log statement is run only after each has finished executing.

                .then(() => {

                    cy.log("dimensiune vector varianta asincron:" + searchValues.length)

                    for (let i = 0; i < searchValues.length; i++) {

                        cy.log("afisam valorile din vector: valoarea "+i+" : " + searchValues[i])

                    }

                  })