

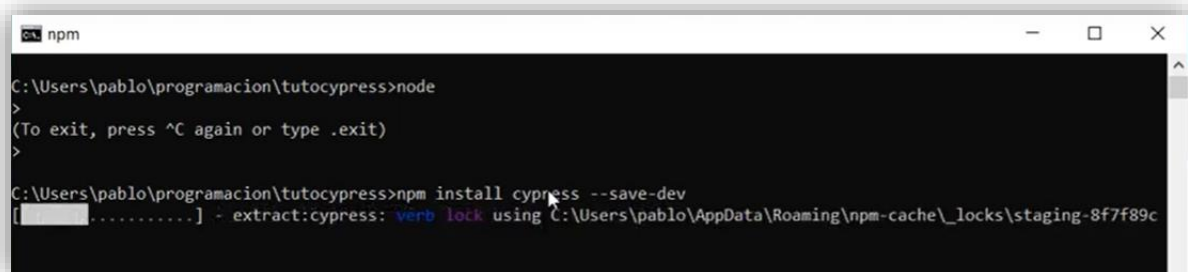
Escenarios automatizados Cypress

Cypress es una plataforma “todo en uno” de instalación simple, que no depende de descargas extra o cambios en el código fuente de la aplicación, así mismo permite tomar *screenshots* o videos de la ejecución de las pruebas a través del tablero de cypress, se ejecuta mediante un proceso de node a través de pruebas escritas en código.

1.Instalación de cypress

Para instalarlo basta descargar desde una instancia abierta de node (en caso de no contar con una instalación previa de node js, esta debe realizarse antes de instalar cypress), el paquete de cypress a través del comando:

```
npm install cypress --save-dev
```



```
npm
C:\Users\pablo\programacion\tutocypress>node
>
(To exit, press ^C again or type .exit)
>
C:\Users\pablo\programacion\tutocypress>npm install cypress --save-dev
[.....] - extract:cypress: verb lock using C:\Users\pablo\AppData\Roaming\npm-cache\_locks\staging-8f7f89c
```

```
Cypress 3.8.2 is installed in C:\Users\pablo\AppData\Local\Cypress\Cache\3.8.2
```

```
npm WARN saveError ENOENT: no such file or directory, open 'C:\Users\pablo\programacion\tutocypress\package.json'
npm WARN enoent ENOENT: no such file or directory, open 'C:\Users\pablo\programacion\tutocypress\package.json'
npm WARN tutocypress No description
npm WARN tutocypress No repository field.
npm WARN tutocypress No README data
npm WARN tutocypress No license field.
```

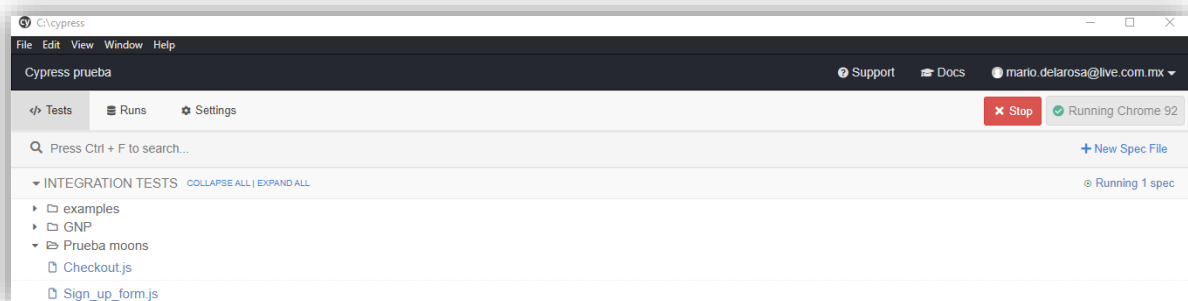
Una vez instalado está listo para usarse

Para abrir la consola gráfica de cypress es necesario teclear el siguiente código dentro de la consola de node:

`.\node_modules\.bin\cypress open`

```
C:\cypress>.\node_modules\.bin\cypress open
GET /__/ 200 229.280 ms - -
GET /__cypress/runner/cypress_runner.css 200 40.968 ms - -
POST /ListAccounts?gpsia=1&source=ChromiumBrowser&json=standard 200 1000.559 ms - -
GET /chrome-variations/seed?osname=win&channel=stable&milestone=92 200 1079.708 ms - -
GET /__cypress/runner/cypress_runner.js 200 44.497 ms - -
```

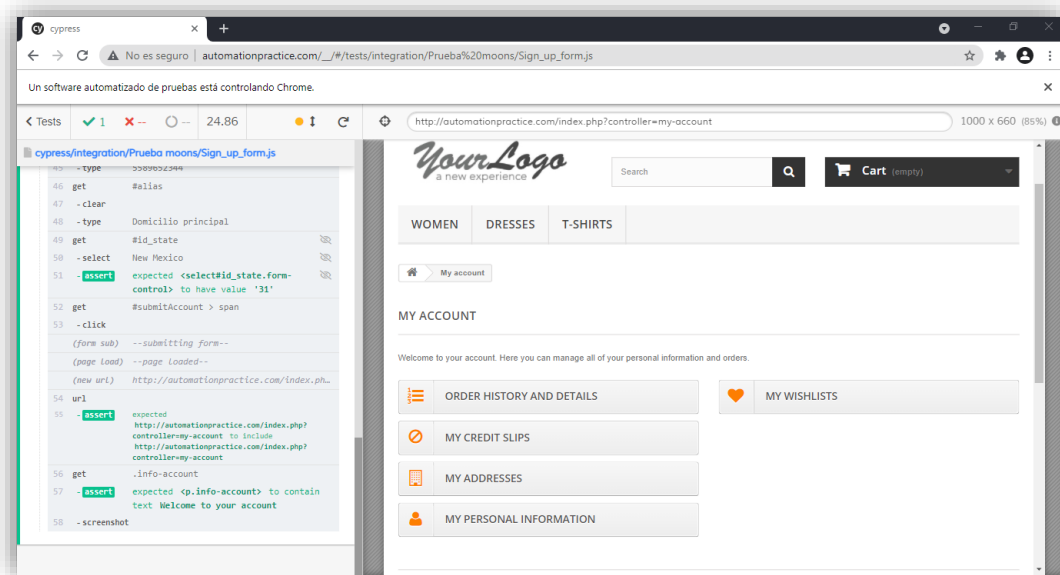
A través de la interfaz gráfica de cypress podemos examinar la librería de pruebas que se encuentre en nuestro directorio raíz, al seleccionar uno comenzará inmediatamente su ejecución:



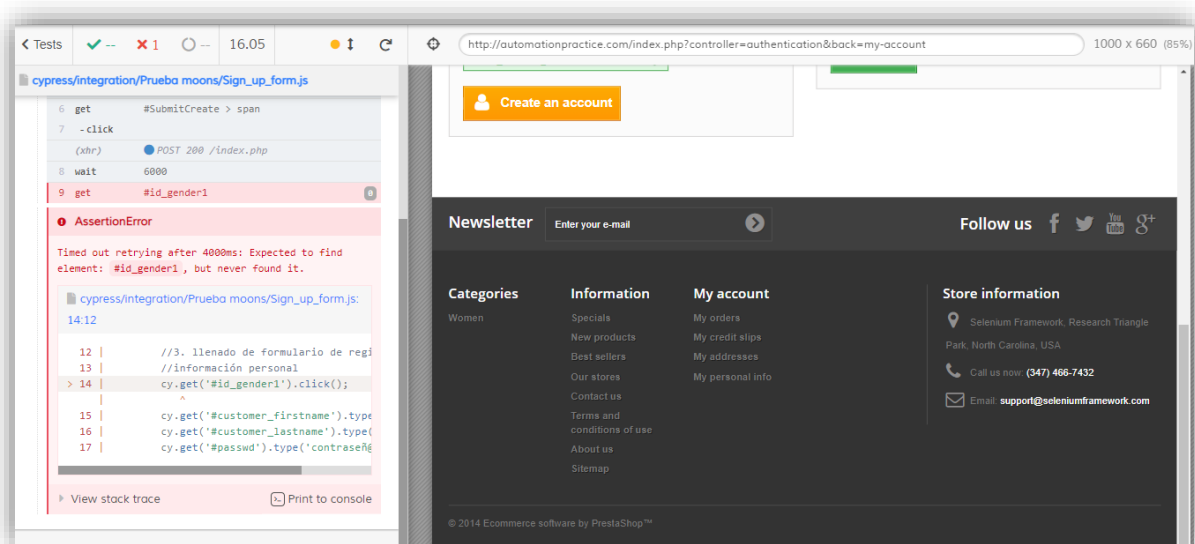
2. Ejecución de un caso de prueba

Para ejecutar un caso de prueba se da click sobre la prueba deseada y esta comenzara a ejecutarse mostrando una instancia nueva de chrome donde se mostrará del lado izquierdo el panel de cypress indicando que pasos del caso se encuentran ejecutando y los errores en caso de encontrarse alguno, del lado derecho veremos la pantalla del explorador que normalmente veríamos en la prueba manual.

Ejemplo de un caso exitoso:



Ejemplo de un caso donde se encuentra un error:



3. Estructura de un caso de prueba

La estructura de un caso de prueba es la siguiente:

```
describe('Flujo de prueba', function() {  
  it('Caso de prueba', function(){  
    Paso A;  
    Paso B;  
    Paso C;  
  })  
})
```

Donde Describe alberga nuestro flujo completo de prueba o escenario a probar, it contendrá nuestro caso de prueba donde se integran los pasos a seguir, validaciones, aserciones, toma de evidencias etc.

4. Ejemplo de registro de un nuevo usuario en una plataforma

Como primer ejemplo tomaremos el código de un *checkout* de compras:

```
1 describe('Registro de usuario', function(){  
2   it('registro', function(){  
3     //1. ingreso al sitio  
4     cy.visit('http://automationpractice.com/index.php')  
5   })  
6 })
```

Comenzamos declarando nuestros elementos describe e it para poder comenzar a cargar los pasos a ejecutar, en este caso usar el comando cy.visit para ingresar a la url que queremos probar

```
//2. ingreso al form de registro  
cy.get('.login').click();  
cy.get('#email_create').type('thanos_1989uth@live.com.mx');  
cy.get('#SubmitCreate > span').click();  
cy.wait(6000);
```

En el siguiente paso correspondiente al *log in* del usuario usamos el comando cy.get para especificar los elementos de la página web con los que vamos a interactuar como botones, campos, listas, etcétera y posteriormente la acción en específico que vamos a realizar en el elemento seleccionado

```
cy.get('#email_create').type('thanos_1989uth@live.com.mx');
```

cy.get para apuntar al campo #email_create en el cual vamos a ingresar el valor 'thanos_1989uth@live.com.mx' mediante el comando *type*

```
cy.get('#SubmitCreate > span').click();
```

cy.get para apuntar al botón #SubmitCreate el cual se presionará con el comando *click*

```
cy.wait(6000);
```

cy.wait para ingresar un periodo de espera medido en milisegundos para esperar la carga de la siguiente página y evitar que se mande un error por *time out*

```
//3. llenado de formulario de registro
//información personal
cy.get('#id_gender1').click();
cy.get('#customer_firstname').type('Mario');
cy.get('#customer_lastname').type('de la Rosa');
cy.get('#passwd').type('contraseña@89');
cy.get('#days').select('29');
cy.get('#months').select('December');
cy.get('#years').select('1989');
cy.get('#newsletter').check();
cy.get('#optin').check();

//dirección
cy.get('#company').type('Moons - Ortodoncia invisible');
cy.get('#address1').type('10880 Malibu Point, 90265, Malibú, California');
cy.get('#address2').type('Colima 220, Roma Nte., Cuauhtémoc, 06700 Ciudad de México, CDMX');
cy.get('#city').type('CDMX');
cy.get('#postcode').clear().type('90265');
cy.get('#id_country').select('United States');
cy.get('#other').type('Frente al oxo');
cy.get('#phone').type('5529044570');
cy.get('#phone_mobile').type('5589652344');
cy.get('#alias').clear().type('Domicilio principal');
cy.get('#id_state').select('New Mexico').should('have.value', '31');
```

Seguimos con el llenado de la información personal del usuario y su dirección, se utilizan las mismas sentencias que en caso anterior con excepción de los casos:

```
cy.get('#newsletter').check();
```

cy.get apuntando al check #newsletter para hacer click sobre el *checkbox*

```
cy.get('#optin').check();
```

cy.get apuntando al check #optin para hacer click sobre el *checkbox*

```
cy.get('#id_country').select('United States');
```

cy.get apuntando a la lista desplegable #id_country para seleccionar el elemento 'United States'

```
cy.get('#postcode').clear().type('90265');
```

cy.get en el campo #postcode ingresando el comando *clear* para borrar el contenido previamente cargado en el campo y posteriormente ingresar el dato 90265 con el comando type

```
cy.get('#alias').clear().type('Domicilio principal');
```

cy.get en el campo #alias ingresando el comando *clear* para borrar el contenido previamente cargado en el campo y posteriormente ingresar el dato Domicilio principal con el comando type

```
cy.get('#id_state').select('New Mexico').should('have.value', '31')
```

cy.get en el campo #id_state seleccionando el valor New Mexico y el comando should (have.value) como aserción para validar que el valor New Mexico corresponda al número 31

```
//Envío de información  
cy.get('#submitAccount > span').click();
```

En el siguiente paso se presiona el botón #submitAccount para enviar toda la información del nuevo usuario y registrarlo en el sistema

```
//Aserción de registro vía url y cadena de texto  
cy.url().should('include', 'http://automationpractice.com/index.php?controller=my-account')  
cy.get('.info-account').should('include.text', 'Welcome to your account')  
cy.screenshot()
```

En el último paso del caso de prueba se generan 2 aserciones con las cuales se comprobará que todo el caso de prueba se ejecuto exitosamente

```
cy.url().should('include', 'http://automationpractice.com/index.php?controller=my-account')
```

cy.url should include para indicar una url que debe contener el texto ingresado, si la url que se genera en este estado de la prueba coincide con la ingresada se debe mostrar como una aserción indicando que el caso fue exitoso

```
cy.get('.info-account').should('include.text', 'Welcome to your account')
```

cy.get en campo info-account should include text, para indicar una cadena de texto en el campo mencionado, si el texto que contiene el campo en este estado de la prueba coincide con el texto ingresado se debe mostrar como una aserción indicando que el caso fue exitoso

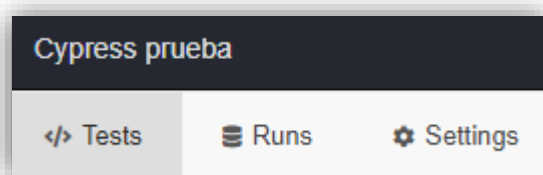
```
cy.screenshot()
```

este comando sirve para tomar una captura de pantalla de nuestro caso de prueba

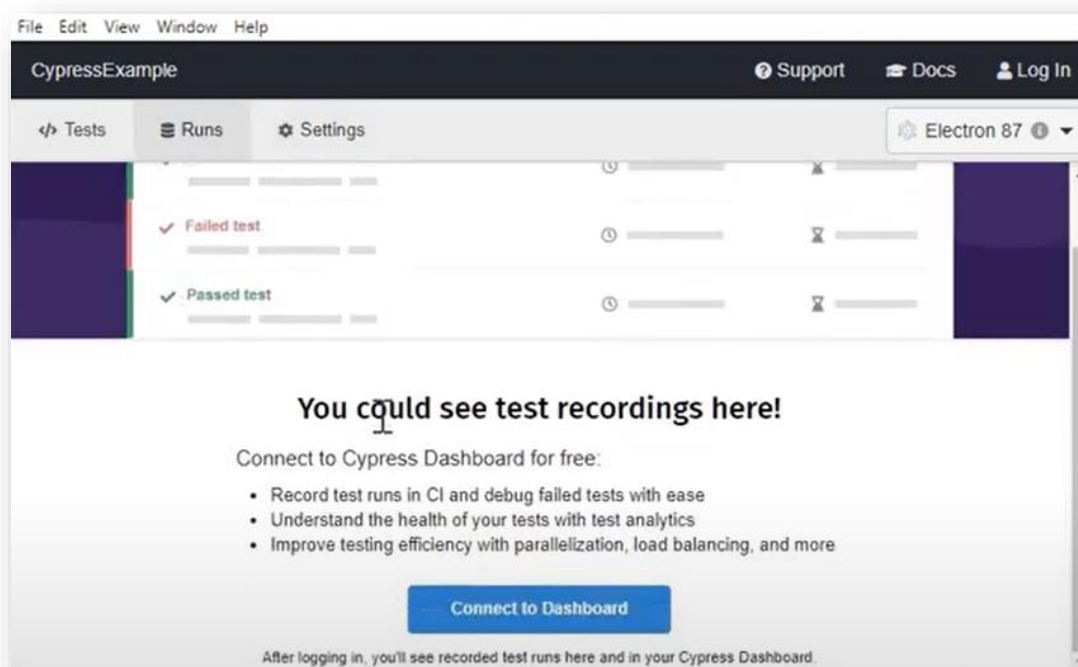
5. Implementación de cypress dashboard

El panel de cypress permite una integración con github para correr *suites* de pruebas, generar evidencias como capturas de pantalla o videos de los casos de prueba y conocimiento a detalle del resultado de cada caso de prueba fallido o exitoso

Se debe ingresar al panel de cypress en la opción *Run*



Se selecciona la opción *Connect to Dashboard* donde será necesario iniciar sesión con alguno de los siguientes servicios para iniciar la configuración



Log in



Log in with GitHub



Log in with Google



Log in with SSO



Log in with Email

Al seleccionar la opción deseada se deberá iniciar sesión y autorizar la conexión, una vez realizado es necesario configurar el proyecto dándole un nombre y asociando una organización, en caso de no contar con una es necesario crearla.

Set up project

What's the name of the project?

(You can change this later)

CypressExample

Who should own this project? ⓘ

[Manage organizations](#)

You don't have any organizations yet.

Organizations can help you manage projects, including billing.

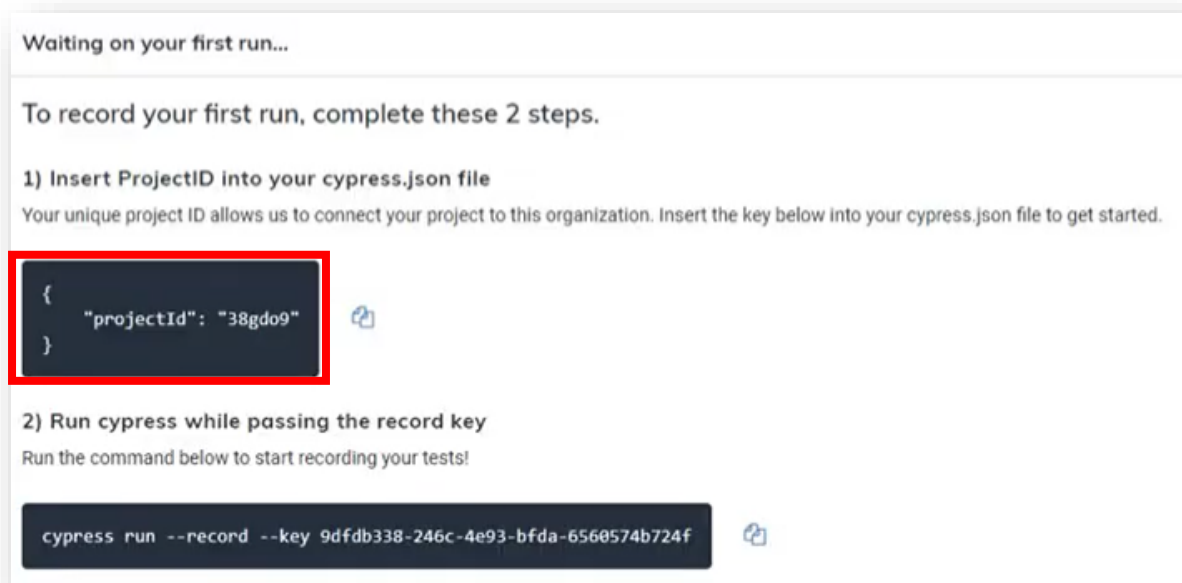
[+ Create organization](#)

Set up project

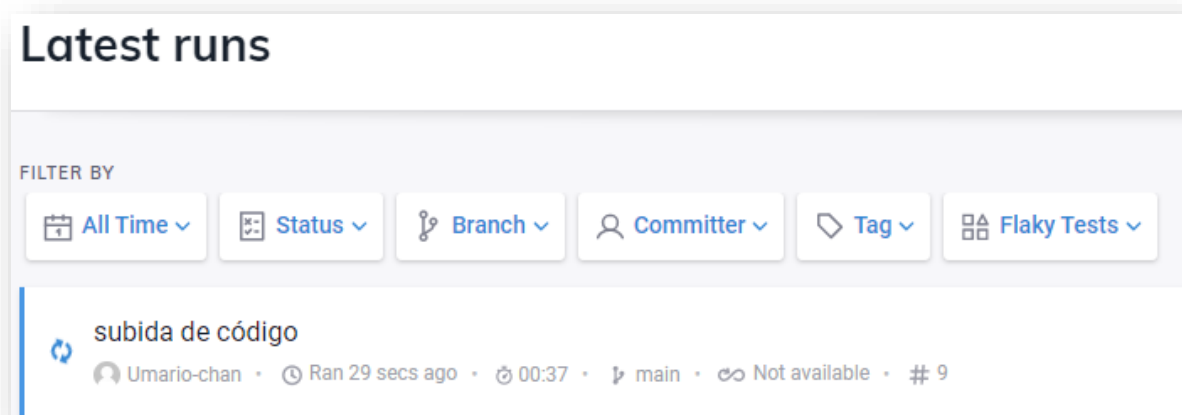
El siguiente paso es la configuración de la conexión del *dashboard* con el proyecto mediante el archivo `cypress.json`

Copiamos el valor `projectId` dentro del archivo `cypress.json` y lo guardamos, posteriormente dentro de la instancia de node, detenemos el ciclo de cypress (en caso de tenerlo corriendo) y ejecutamos la sentencia

`Npx cypress run --record --key b33e88e3-be67-40f7-aa3b-04adff5478ad`



Inmediatamente despues se comenzara a correr toda la suite de pruebas alojada en el proyecto, 1 caso a la vez, indicando si fue exitoso o si hubo un error en algún caso de prueba



Al terminar de ejecutarse los casos de prueba se mostrará el resultado de cada uno, adjuntando capturas de pantalla y video

MD

Mario de la Rosa

Mario de la Rosa

100%

Onboarding complete

Great work! Now invite your teammates to share your test suite analytics.

Copy invite link

Cypress prueba

View all projects

Latest runs

Analytics

Run status

Run duration

Test suite size

Top failures

Slowest tests

Most common errors

Flaky tests

Project settings

subida de código

Run 5 days ago · 03:01 · main · Not available · #6

Overview

Test Results 2

Specs 2

All specs are complete! 2 specs passed

SORT BY

Execution order

FILTER BY

Status

Flaky Tests

Last Modified

Spec File

Run Group

Browser

Operating System

Prueba moons/Checkout.js

1 · 01:05 · Windows 10.0.19042 · Electron 89 · View Output

Checkout de carrito de compras > checkout

Prueba moons/Sign_up_form.js

1 · 00:30 · Windows 10.0.19042 · Electron 89 · View Output

Registro de usuario > registro

registro

Prueba moons/Sign_up_form.js

Create issue

Screenshots

Watch video

Previous runs

Runtime environment

Artifacts

Test code history

First seen 5 days ago

Failure rate

Flaky rate

✓ checkout

Prueba moons/Checkout.js

Create issue

Screenshots

Watch video

6. Implementación de pruebas en Firefox y Edge

A partir de la versión 4.0 de cypress es posible seleccionar un explorador diferente a Chrome para poder ejecutar las pruebas, a través de la interfaz gráfica de cypress se selecciona el explorador en el que se desea correr la prueba antes de ejecutarla

