



TEST AUTOMATION ENGINEER – FORMACIÓN INTEGRAL

{desafío}
latam_

CURSO:

TEST AUTOMATION

ENGINEER – FORMACIÓN

INTEGRAL

- Módulo 1: FUNDAMENTOS DEL TESTING DE SOFTWARE
- Módulo 2: CONTROL DE VERSIONES Y ENTORNOS DE DESARROLLO
- Módulo 3: FUNDAMENTOS DE PROGRAMACIÓN APLICADOS AL TESTING (JAVASCRIPT)
- Módulo 4: CARACTERISTICAS DE CYPRESS – AUTOMATIZACIÓN WEB I
- Módulo 5: CARACTERISTICAS DE CYPRESS – AUTOMATIZACIÓN WEB II
- Módulo 6: PRUEBAS DE APIS CON POSTMAN Y SUPERTEST



Te encuentras aquí

CURSO:

TEST AUTOMATION

ENGINEER – FORMACIÓN

INTEGRAL

- Módulo 7: HERRAMIENTAS DE PLAYWRIGHT Y PRUEBAS CON MÚLTIPLES NAVEGADORES
- Módulo 8: DESARROLLO GUIADO EN EL COMPORTAMIENTO (BDD) CON CUCUMBER.JS
- Módulo 9: HERRAMIENTAS DE AUTOMATIZACIÓN MÓVIL CON APPIUM
- Módulo 10: HERRAMIENTAS DE INTEGRACIÓN DE PRUEBAS EN CI/CD
- Módulo 11: HERRAMIENTAS DE DOCKER, ENTORNOS VIRTUALIZADOS Y PRUEBAS EN LA NUBE
- HERRAMIENTAS DE AUTOMATIZACIÓN DE UN FLUJO COMPLETO WEB + API + CI/CD

Módulo 4: CARACTERISTICAS DE CYPRESS – AUTOMATIZACIÓN WEB I.



OBJETIVO ESPECÍFICO DEL MÓDULO

- CONOCER CARACTERISTICAS DE CYPRESS AUTOMATIZACIÓN WEB I, DE ACUERDO A LAS APLICACIONES WEB, MÓVILES Y APIs.



¿Qué expectativas tiene sobre el uso de herramientas como Cypress para automatizar pruebas, y qué desafíos cree que podrían surgir al validar el comportamiento de una aplicación web?



INSTALACIÓN Y ESTRUCTURA INICIAL DE CYPRESS

- **Instalación del entorno:**
- Requiere tener Node.js y un proyecto con npm init
- **Comando de instalación:**

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

- Iniciar Cypress:

```
npx cypress open
```

- Estructura del proyecto Cypress:
- **/cypress/**: contiene el código de las pruebas
- **/e2e/**: pruebas automatizadas
- **/fixtures/**: datos de prueba (JSON)



- **/support/**: comandos personalizados y configuración global
- **cypress.config.js**: archivo de configuración del proyecto
- **Primer test automatizado:**
- Cypress genera ejemplos por defecto para explorar.



COMANDOS BÁSICOS DE CYPRESS Y VALIDACIONES

- Comandos frecuentes:
- `cy.visit(url)`: abre una página
- `cy.get(selector)`: selecciona un elemento
- `cy.contains(texto)`: busca un texto en la pantalla
- `cy.click()`, `cy.type()`, `cy.select()`
- **Validaciones (assertions):**
- `should('be.visible')`, `should('have.text', 'Texto')`
- Uso de `expect()` con valores personalizados



- **Esperas y sincronización automática:**
- Cypress espera que los elementos estén listos antes de continuar (no requiere sleep() como otras herramientas).
- Cypress destaca por su legibilidad y por incluir todas las herramientas necesarias para testing end-to-end en un solo entorno.



DATOS DINÁMICOS Y BUENAS PRÁCTICAS

- **Uso de fixtures:**
- Archivos .json con datos simulados o reales de prueba
- Se cargan con:

```
cy.fixture('usuario.json').then((user) => {
  cy.get('#nombre').type(user.nombre)
})
```

- **Buenas prácticas:**
- Separar lógica de negocio del test
- Usar `beforeEach()` para setup común
- Nombrar las pruebas de forma clara (`it('debería registrar un usuario con éxito')`)
- Mantener pruebas independientes entre sí
- No depender de estado anterior o de datos persistentes en la app

El uso de fixtures mejora la mantenibilidad, evita repetir datos y permite simular múltiples escenarios con facilidad.

EVIDENCIAS EN CYPRESS: CAPTURAS, VIDEOS Y REPORTES

- **Captura automática:**
- Cypress toma screenshots en fallas por defecto.
- **Se puede forzar una captura con:**

```
cy.screenshot('nombre-opcional')
```

- **Videos:**

Se graba automáticamente la ejecución de las pruebas en modo headless (npx cypress run)

Se guarda un .mp4 por cada test ejecutado.

- **Reportes:**

- Se pueden integrar herramientas como mochawesome para generar reportes HTML
- Cypress también se puede integrar con CI (GitHub Actions, GitLab, Jenkins)
- Esta evidencia es crucial para validar, comunicar errores y generar documentación auditable en entornos de QA profesional.





No olvide desarrollar los ejercicios que contiene el Módulo...

¿De qué manera la correcta instalación, estructura del proyecto, uso de fixtures y captura de evidencia pueden contribuir a la calidad, trazabilidad y reproducibilidad de una suite de pruebas automatizadas?



**Éxito en la evaluación parcial y
en la Prueba Final...**

{desafío}
latam_

*Academia de
talentos digitales*

