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

{desafío} latam_

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

{desafío} latam_

Módulo 1:
FUNDAMENTOS DEL TESTING DE SOFTWARE.

{desafío}
latam_

OBJETIVO ESPECÍFICO DEL MÓDULO

- IDENTIFICAR LOS FUNDAMENTOS DEL TESTING DE SOFTWARE, DE ACUERDO A LAS APLICACIONES WEB, MÓVILES Y APIS.

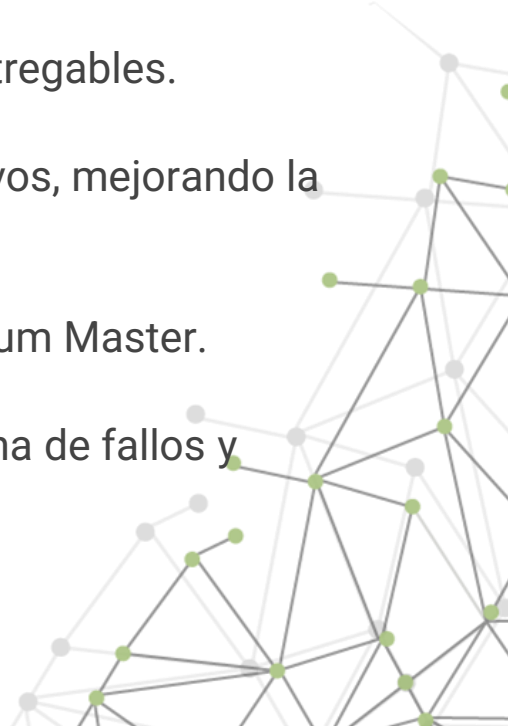
{desafío} latam_

¿Qué cree que implica el trabajo de un profesional de QA Automation dentro de un equipo ágil, y por qué cree que su rol puede marcar la diferencia en la calidad de un producto digital?

{desafío} latam_

EL ROL DEL QA AUTOMATION EN EQUIPOS ÁGILES

- El QA Automation es un facilitador clave en el aseguramiento de calidad continua dentro de ciclos iterativos y cortos.
- Participa desde la planificación del sprint hasta la validación de entregables.
- Su enfoque está en automatizar casos de prueba críticos y repetitivos, mejorando la eficiencia y reduciendo errores humanos.
- Colabora estrechamente con desarrolladores, Product Owner y Scrum Master.
- Aporta a la definición de criterios de aceptación, detección temprana de fallos y entrega de productos con calidad integrada.



{desafío} latam_

TIPOS DE PRUEBAS: UNITARIAS E INTEGRACIÓN

- **Pruebas Unitarias:**
- Se ejecutan sobre componentes individuales del código (métodos, funciones).
- Su objetivo es validar que cada unidad haga lo que debe.
- Usadas por los desarrolladores (ej. JUnit, NUnit, Mocha).

- **Pruebas de Integración:**
- Verifican que módulos integrados interactúen correctamente.
- Se realizan después de las unitarias.
- **Ejemplo:** validación de conexión entre base de datos y backend.

TIPOS DE PRUEBAS: REGRESIÓN, SMOKE Y SANITY

- **Regresión:**
- Validan que funcionalidades existentes no se vean afectadas por nuevos cambios.
- Son esenciales en equipos ágiles con cambios frecuentes.

- **Smoke Testing (“Prueba de humo”):**
- Verifica que el sistema enciende y sus funcionalidades básicas operan correctamente.
- Se realiza antes de pruebas más profundas.

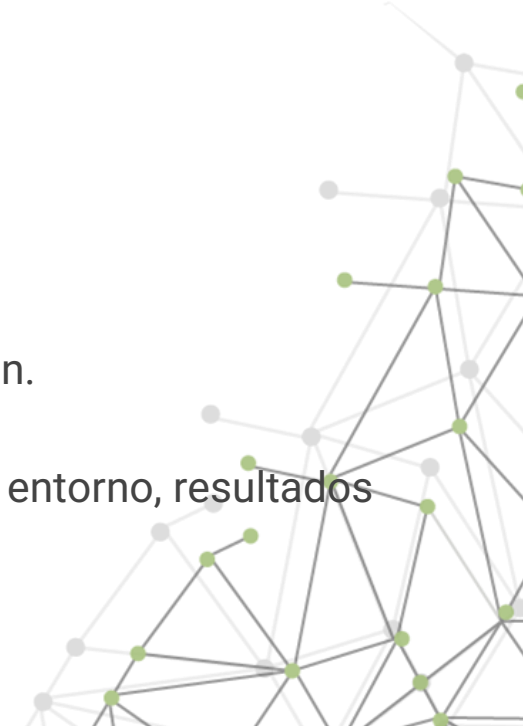
- **Sanity Testing:**
- Se enfoca en validar funciones específicas después de pequeños cambios o correcciones.
- Es más limitado y rápido que un test de regresión.

CICLO DE VIDA DEL BUG Y REPORTE DE INCIDENCIAS

- Un bug pasa por varias etapas desde su detección hasta su resolución:

1. **Nuevo (New):** Se reporta el defecto.
2. **Asignado (Assigned):** Se asigna a un desarrollador.
3. **En progreso (In Progress):** Se trabaja en resolverlo.
4. **Resuelto (Resolved):** Se entrega la solución.
5. **Verificado (Verified):** El QA valida la corrección.
6. **Cerrado (Closed):** Se finaliza si está correctamente resuelto.
7. **Reabierto (Reopened):** Si el error persiste tras la supuesta solución.

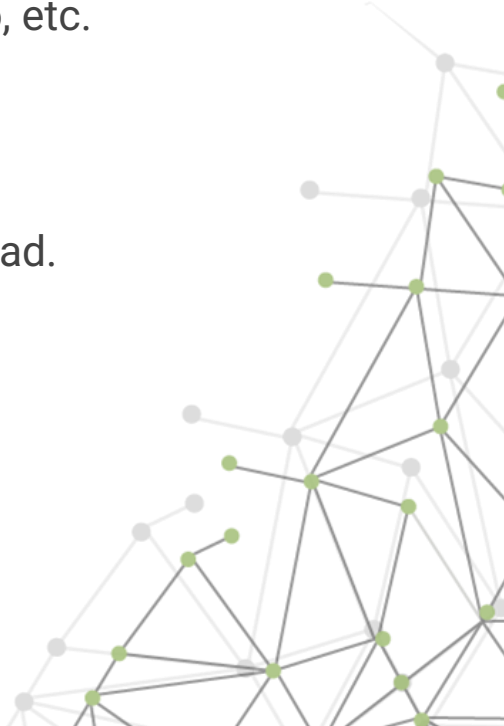
- **Reporte efectivo:** Debe incluir título claro, pasos para reproducirlo, entorno, resultados esperados vs reales, y evidencia (capturas/logs).



QA FUNCIONAL VS NO FUNCIONAL

- **QA Funcional:**
- Evalúa si el sistema hace lo que debe hacer según los requisitos funcionales.
- Incluye validación de flujos, botones, formularios, reglas de negocio, etc.

- **QA No Funcional:**
- Mide cómo el sistema ejecuta sus funciones.
- Evalúa rendimiento, usabilidad, escalabilidad, seguridad, accesibilidad.
- **Ejemplo:** pruebas de carga, estrés, compatibilidad con dispositivos.



METODOLOGÍAS ÁGILES: SCRUM Y KANBAN

- Las metodologías ágiles se enfocan en iteraciones cortas, entregas incrementales y mejora continua.

- **Scrum:**
- Ciclos de trabajo llamados sprints (1-4 semanas).

- **Roles definidos:** Product Owner, Scrum Master, Equipo de Desarrollo.

- **Eventos:** planificación, daily standup, revisión, retrospectiva.
- Permite adaptarse rápidamente a cambios de requerimientos.

{desafío} latam_

- **Kanban:**
- Flujo continuo visualizado en un tablero (To Do, Doing, Done).
- Prioriza la entrega constante y el control de trabajo en curso (WIP).
- Sin necesidad de sprints.

No olvide desarrollar los ejercicios que contiene el Módulo…

{desafío} latam_

¿De qué manera los distintos tipos de pruebas y el ciclo de vida de los bugs pueden mejorar la eficiencia y calidad en proyectos ágiles y, cómo aplicaría estos conocimientos en un entorno real?

{desafío} latam_

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

{desafío}
latam_

Academia de
talentos digitales