«Talento Tech»

Automation Testing

Clase 1

Buenos Aires Ciudad

Clase N°1 | Introducción al Automation Testing

Temario:

- ¿Qué es Automation Testing?
- Rol del Automation Tester en el ciclo de desarrollo
- Herramientas más utilizadas en Automation Testing
- Habilidades necesarias para un Automation Tester
- Mercado laboral y oportunidades
- Instalación del entorno básico:
    - Python
    - Visual Studio Code
    - Configuración inicial

Objetivos de clase:

En esta clase vamos a introducirnos en el mundo del Automation Testing. Vamos a entender qué es, para qué sirve, el rol del Automation Tester, qué herramientas se usan frecuentemente, las habilidades necesarias para tener éxito en este rol, las oportunidades laborales y cómo configurar tu entorno de trabajo básico para empezar a automatizar pruebas. Además, vamos a repasar brevemente los temas que veremos a lo largo del curso.

¿Qué aprenderemos en la cursada?

Durante las próximas clases, cubriremos en profundidad los siguientes temas:

- **Módulo 1: Introducción y Fundamentos**
    - Introducción al Automation Testing y configuración inicial
    - Fundamentos de Python
    - Control de versiones con Git
    - Pytest y automatización básica
    - HTML y estructura web básica
- **Módulo 2: Selenium y Automatización Web**
    - DOM y selectores CSS/XPath
    - Introducción a Selenium WebDriver
    - Localización de elementos y acciones en Selenium
    - Page Object Model (POM)
- **Módulo 3: Técnicas Avanzadas de Automatización**
    - Manejo de datos de prueba
    - Automatización de pruebas API
    - Reportes y logging
- **Módulo 4: BDD, CI/CD y Proyecto Final**
    - Behavior-Driven Development (BDD) con Behave
    - Integración continua (CI/CD)
    - Proyecto final y cierre del curso

Al finalizar el curso:

Serán capaces de automatizar flujos clave de una aplicación web, utilizando herramientas como Selenium, Pytest y Git. Aplicarán buenas prácticas de testing automatizado y documentarán su trabajo de forma profesional, desarrollando un proyecto completo desde la planificación hasta la ejecución y validación de pruebas.

¿Qué hace un Tester?

Recordemos que un **tester**, también conocido como **analista de pruebas** o **QA (Quality Assurance) tester**, es el profesional responsable de verificar que un producto de software funcione correctamente antes de que se libere al usuario final. Su tarea principal es **detectar errores, fallos o comportamientos inesperados** en las aplicaciones para garantizar que cumplan con los requisitos funcionales y de calidad establecidos.

Principales funciones de un tester:

1. **Analizar requerimientos**: Revisa los documentos funcionales y técnicos para entender qué debe hacer el software.
2. **Diseñar casos de prueba**: Crea escenarios específicos que permitan comprobar que cada funcionalidad del sistema funciona correctamente.
3. **Ejecutar pruebas**:
    - **Pruebas manuales**: Interactuar directamente con el sistema como lo haría un usuario.
    - **Pruebas automatizadas**: Usar herramientas para ejecutar pruebas repetitivas o complejas de forma automática (por ejemplo, con Selenium o JUnit).
4. **Registrar defectos**: Documenta los errores encontrados (bugs), cómo reproducirlos y su impacto, generalmente en herramientas como Jira o TestRail.
5. **Realizar pruebas de regresión**: Verifica que una nueva funcionalidad o corrección no haya afectado otras partes del sistema.
6. **Colaborar con el equipo de desarrollo**: Trabaja junto a programadores, analistas y otros roles para resolver problemas, proponer mejoras y garantizar la calidad del producto.
7. **Validar correcciones**: Una vez que los errores son corregidos por los desarrolladores, el tester los vuelve a probar para confirmar que están solucionados.

¿Qué es Automation Testing?

El **Automation Testing** es el proceso mediante el cual se emplean herramientas y scripts especializados para ejecutar automáticamente casos de prueba sobre una aplicación o sistema. Esta automatización permite validar funcionalidades de forma más **rápida, precisa y repetible**, reduciendo significativamente el margen de error humano y los tiempos de ejecución en comparación con las pruebas manuales.

Rol del Automation Tester en el ciclo de desarrollo:

El **Automation Tester** es el profesional encargado de **diseñar, desarrollar y mantener scripts de prueba automatizados**, alineados con los requisitos funcionales y técnicos del software. Su rol es clave en la detección temprana de errores, ya que permite validar continuamente nuevas versiones del sistema mediante pruebas de regresión, integración y rendimiento. Al integrarse en flujos de desarrollo ágil y DevOps, contribuye a mejorar la **calidad, estabilidad y eficiencia** del proceso de entrega de software.

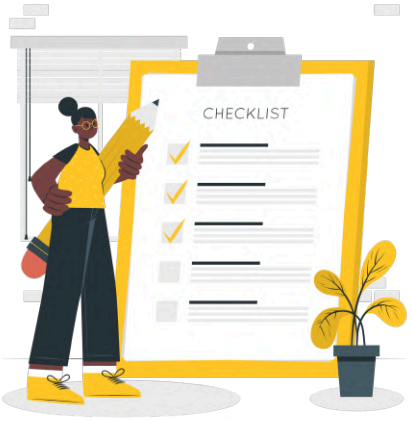
Herramientas más utilizadas en Automation Testing:

- **Selenium WebDriver:** Herramienta para automatizar pruebas de navegadores web, permitiendo simular acciones del usuario real.
- **Pytest:** Framework de pruebas en Python que facilita la escritura y ejecución de casos de prueba automáticos.
- **Postman:** Herramienta para automatizar pruebas de APIs, facilitando la creación y ejecución de peticiones HTTP.
- **Jenkins/GitHub Actions:** Plataformas de Integración Continua que permiten ejecutar pruebas automatizadas cada vez que hay cambios en el código.

Habilidades necesarias para un Automation Tester:

- **Programación básica**

Es necesario manejar al menos un lenguaje como Python o Java para escribir y mantener scripts de prueba automatizados.



- **Lógica y análisis crítico**

Permite entender el comportamiento del sistema y crear pruebas que cubran diferentes escenarios, incluso los menos evidentes.

- **Atención al detalle**

Detectar errores sutiles en los resultados de las pruebas requiere observar cuidadosamente el funcionamiento del software.

- **Comunicación efectiva:**

Es clave para documentar fallos con claridad y colaborar con desarrolladores, testers y otros miembros del equipo.

Mercado laboral y oportunidades:

El mercado demanda cada vez más Automation Testers debido a la necesidad creciente de lanzar software de calidad rápidamente. Los Automation Testers son muy solicitados en sectores como tecnología, banca, comercio electrónico, salud, telecomunicaciones y otros donde la calidad y rapidez son fundamentales.

La paradoja del Automation Testing:

Curiosamente, una de las paradojas del Automation Testing es que usamos software para verificar otros sistemas de software. Es decir, necesitamos asegurarnos que nuestras herramientas y scripts sean altamente confiables, ya que cualquier error en estos podría generar falsos positivos o negativos en las pruebas automatizadas.

Automatización de Pruebas: Enfoque y Alcance

Una vez entendido el **rol del tester automation**, es clave entender el **ciclo de vida de las pruebas automatizadas** y **qué tipo de pruebas se pueden o no automatizar**.

La automatización no reemplaza todas las pruebas, pero sí permite optimizar tiempo y recursos en aquellas que son repetitivas, críticas o de alto impacto. Esto requiere no solo saber programar scripts, sino también **comprender qué se está probando** y cuál es el objetivo de cada tipo de prueba dentro del flujo de aseguramiento de calidad.

Automatizar sin una estrategia clara puede generar scripts inútiles, poco mantenibles o incluso contraproducentes. Por eso, el conocimiento de los distintos tipos de pruebas y su función es clave para tomar buenas decisiones desde el rol técnico.

Ciclo de Vida de las Pruebas Automatizadas

Son las etapas que se siguen para planificar, diseñar, implementar, ejecutar y mantener pruebas automatizadas dentro del proceso de aseguramiento de calidad del software.

Las etapas son:

1. **Análisis:** Identificar qué pruebas son candidatas para automatización. No todo debe automatizarse.
2. **Diseño:** Planificar la estructura de los scripts y seleccionar herramientas adecuadas.
3. **Implementación:** Codificar los scripts de prueba con buenas prácticas de programación.
4. **Ejecución:** Correr las pruebas de forma periódica, preferentemente en un entorno de CI/CD.
5. **Mantenimiento:** Actualizar los scripts cuando cambia la aplicación o se detectan falsos positivos.

Tipos de Pruebas Automatizadas

En el proceso de asegurar la calidad de un software, existen distintos **tipos de pruebas automatizadas**, cada una enfocada en validar aspectos específicos del sistema. La automatización no se aplica de forma única, sino que se adapta a las necesidades del proyecto: desde comprobar funciones pequeñas de código hasta evaluar el rendimiento de toda una aplicación.

A continuación, se describen los principales tipos de pruebas que se pueden automatizar, junto con su objetivo y algunas herramientas comúnmente utilizadas para llevarlas a cabo:

Tipo de Prueba	Descripción	Herramientas
Pruebas Unitarias	Verifican componentes individuales	Pytest, JUnit
Pruebas de Integración	Validan interacciones entre componentes	Selenium, RestAssured
Pruebas de UI	Comprueban la interfaz de usuario	Selenium, Cypress
Pruebas de API	Verifican servicios web	Postman, RestSharp
Pruebas de Rendimiento	Analizan velocidad y estabilidad	JMeter, Locust

Configuración Inicial del Entorno

Instalar Python

1. Accede a la página oficial de descargas de Python: https://www.python.org/downloads/
2. Haz clic en el botón **Download Python (última versión)**.
3. **Importante:** Durante la instalación, marca la opción:
   ✅ **Add Python to PATH**
   Esto permite ejecutar Python desde cualquier terminal.
4. Completa la instalación con las opciones por defecto.
5. **Verificar instalación:**
   - Abre la **Consola** (CMD en Windows, Terminal en macOS/Linux).

   Escribe:
   ```
   python --version
   ```

Deberías ver algo como:
`Python 3.xx.x`

Instalar Visual Studio Code

1. Descarga VS Code desde:
   👉 https://code.visualstudio.com/download

2. Ejecuta el instalador y sigue los pasos por defecto.

3. (Opcional pero recomendado) Marca:
   ✅ **Add to PATH** y **Register Code as editor** durante la instalación.

Configurar el Proyecto en VS Code

1. Abre Visual Studio Code.
2. Crea una carpeta para el curso:
   - **Ve a Archivo → Abrir carpeta.**

Nómbrala:
`curso-automation-testing`

3. Abre la carpeta recién creada en VS Code.

Instalar Extensiones en VS Code

1. Haz clic en el ícono de Extensiones (barra lateral izquierda).
2. Busca e instala:
   - Python (desarrollado por Microsoft).
   - Pytest (para pruebas automatizadas).
3. Reinicia VS Code para aplicar los cambios.

Crear el Primer Script

Dentro de la carpeta del proyecto, crea un archivo llamado: `test.py`
Escribe el siguiente código: `print("¡Hola Automation Tester!")`

Ejecutar el Script

1. Abre la Terminal integrada en VS Code:
   Ver → Terminal (o `Ctrl + Ñ` en Windows / `Ctrl + Shift +` en macOS).

Ejecuta el archivo con: `python test.py`
Si todo está correcto, deberías ver: `¡Hola Automation Tester!`

✅¡Listo! Ahora tienes tu entorno configurado para comenzar con Python y Automatización de Pruebas.

🗨️**Storytelling**

¡Bienvenido a TalentoLab!

Nos complace darte la bienvenida nuevamente a nuestro equipo. En Talento Lab, una consultora líder especializada en soluciones tecnológicas, estamos iniciando un proyecto clave para automatizar las pruebas del nuevo sistema interno de gestión de talentos. Tu rol como Automation Tester Junior será esencial para asegurar la calidad y eficiencia de nuestro sistema, contribuyendo significativamente al éxito del proyecto y la satisfacción de nuestros clientes internos.

Nuestra forma de trabajar: Metodología Agile

En Talento Lab, trabajamos con la metodología Agile, un enfoque iterativo e incremental que prioriza la colaboración, la flexibilidad y la entrega continua de valor. Esto significa que trabajamos en ciclos cortos llamados Sprints, adaptándonos a los cambios y entregando resultados tangibles en cada iteración.

Roles clave en nuestro equipo Agile:

- Product Owner (PO): Representa al cliente y define la visión del producto. Prioriza las funcionalidades y se asegura de que el equipo trabaje en lo más valioso.
- Scrum Master: Facilita el trabajo del equipo, eliminando obstáculos y asegurando que se sigan las prácticas Agile.
- Equipo de Desarrollo: Diseña, construye y prueba el producto. Incluye testers, desarrolladores y otros roles necesarios.

Objetivo General

El objetivo principal de este proyecto es garantizar el correcto funcionamiento del sitio web de Talento Lab, brindando una experiencia de usuario óptima y eficiente. Para ello, contamos con tu expertise para llevar a cabo pruebas exhaustivas y detalladas, que permitan identificar y documentar cualquier error, problema de usabilidad o aspecto susceptible de mejora.

¿Cómo trabajaremos?

Silvia: Product Owner. Silvia definirá la prioridad de los defectos.

Matias: Automation Tester Senior. Matias te ayudará en tu rol como QA Automation tester junior.

Ejercicio Práctico

Configuración del entorno de trabajo.

Se te solicita que prepares tu entorno de trabajo para empezar lo antes posible. Para ello, sigue estos pasos:

- Instala Python desde https://www.python.org/downloads/. Selecciona la opción "Add Python to PATH" al instalar y verifica la instalación ejecutando en consola: `python --version`.
- Instala Visual Studio Code desde https://code.visualstudio.com/download. Sigue los pasos por defecto durante la instalación.
- Abre VS Code, crea una carpeta llamada "curso-automation-testing".
- Instala las extensiones "Python" y "Pytest" en VS Code.
- Dentro de la carpeta, crea un archivo `test.py` con el código:

  ```
  print("¡Hola Automation Tester!")
  ```

- Ejecuta el archivo desde la terminal integrada para verificar la correcta configuración.

Matías espera que tengas tu entorno listo para comenzar las primeras tareas en la siguiente sesión.

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad