MINISTERIO DE EDUCACIÓN

DIRECCIÓN DE FORMACIÓN TÉCNICA SUPERIOR

Instituto de Formación Técnica Superior Nº 29

Materia Metodología de prueba de sistemas	Año 2025 – 2° cuatrimestre
Régimen Cuatrimestral	

FUNDAMENTACIÓN

La asignatura se propone que los estudiantes adquieran conocimientos y habilidades referidos a la gestión y ejecución de las pruebas de software para generar software confiable y de calidad.

Asimismo se abordan las principales técnicas de gestión de pruebas, sus alcances y la aplicabilidad de acuerdo a las características del sistema en el que se esté trabajando. Esto implica planificar las etapas de testing acorde al tamaño y complejidad de los sistemas en los que participa, definir el alcance y tipos de pruebas a ejecutar, determinar la cobertura de los diferentes tipos de pruebas, elegir la metodología a aplicar.

Para la organización de la enseñanza de este módulo los contenidos giran en torno a dos ejes: “Gestión de Pruebas” y “Ejecución de Pruebas”. La diferenciación entre las etapas de gestión de pruebas y ejecución de pruebas en el desarrollo de software es una práctica común para garantizar la calidad y eficacia de un producto de software. Cada una de estas etapas tiene un propósito específico y se realiza en momentos diferentes del ciclo de vida del desarrollo de software.

OBJETIVOS DE LA MATERIA

Se espera que al finalizar el cursado de la materia los/as estudiantes sean capaces de:

- Identificar los diferentes tipos de pruebas involucradas en la construcción del software.

- Identificar los diferentes roles y entregables generados a partir de la planificación y ejecución de las pruebas.
- Conocer y utilizar las técnicas de ejecución de prueba.
- Aplicar metodología diversas al diseñar casos de prueba.
- Diseñar casos de prueba de caja negra y blanca con cobertura integral.
- Identificar y asignar clases de equivalencia para condiciones externas.
- Implementar y automatizar pruebas unitarias con Selenium.

ESTRATEGIAS DE ENSEÑANZA

- *Enfoque narrativo del usuario:*

Centrar la enseñanza en comprender las necesidades y expectativas del usuario final por medio de la presentación de Historias de Usuarios. Los/as estudiantes pueden acceder y crear historias de usuario que describen casos de uso realistas para el software a probar. Esto ayuda a identificar requisitos y escenarios críticos.

- *Diseño de casos de prueba:*

Los/as estudiantes se sumergen en la creación de casos de prueba exhaustivos para diversas funciones del software. Este enfoque implica considerar una amplia gama de condiciones y situaciones, cultivando así la habilidad de identificar y abordar casos críticos. La práctica constante en el diseño detallado de casos de prueba nutre la destreza necesaria para evaluar el rendimiento del software de manera integral.

- *Prácticas formativas autogestionadas: Aprendizaje autónomo en pruebas de software:*

Promovemos la autonomía y la autoeficacia al proporcionar a los/as estudiantes recursos y herramientas para que practiquen y perfeccionen sus habilidades de prueba de software de forma independiente. A través de ejercicios prácticos, los estudiantes tienen la oportunidad de aplicar y consolidar sus conocimientos, recibiendo retroalimentación constructiva para fortalecer sus habilidades y desarrollar confianza en su capacidad para abordar desafíos en el campo de las pruebas de software.

- *Empleo de dominios:*

Los/as estudiantes se centran en comprender la relevancia de elegir cuidadosamente un conjunto representativo de condiciones para la prueba al

trabajar en la identificación y definición de dominios de prueba para diversas secciones del software. Este proceso implica la selección de valores y condiciones válidos y pertinentes, teniendo en cuenta la complejidad y las limitaciones de la prueba. A través de esta estrategia, los estudiantes desarrollan habilidades críticas para garantizar la eficacia y la exhaustividad en la evaluación del software.

- ***Cinco dominios***

La incorporación efectiva de dominios es esencial en nuestra estrategia de enseñanza, y nos enfocamos en cinco dominios específicos: Factura Digital, Taxi Móvil, Sistema GPS, Bravo y Blackjack. En cada uno de estos dominios, exploramos los requisitos del cliente mediante entrevistas con expertos en el dominio y la identificación de historias de usuario. Esta información no solo sirve como base para comprender las funcionalidades esperadas del software, sino que también actúa como un marco sólido para la creación de casos de prueba y, en ocasiones, pruebas unitarias.

Al analizar cada dominio de manera detallada, los estudiantes aprenden a seleccionar cuidadosamente un conjunto representativo de condiciones bajo las cuales el software debe funcionar correctamente. Esto implica considerar la complejidad y la diversidad de posibles escenarios en cada dominio. Además, la conexión entre los requerimientos del cliente, las entrevistas con expertos y las historias de usuario proporciona a los estudiantes una comprensión completa de los contextos específicos en los cuales se llevarán a cabo las pruebas.

Este enfoque no solo les brinda a los estudiantes una visión práctica de cómo aplicar la teoría de dominios en la metodología de prueba de software, sino que también les permite desarrollar habilidades críticas para identificar y evaluar de manera efectiva el comportamiento del software en entornos del mundo real."

Software que se van a usar: NUnit y Selenium

En el ámbito de las pruebas de software, la implementación de tecnologías y herramientas especializadas es crucial para garantizar la calidad y eficacia de los procesos de prueba. Dos herramientas destacadas en este contexto son NUnit para pruebas unitarias y Selenium, en particular Selenium WebDriver, para pruebas automatizadas.

- *NUnit para pruebas unitarias en VSCode:*

NUnit, una poderosa herramienta de prueba unitaria, se integra perfectamente con el entorno de desarrollo Visual Studio Code (VSCode). Este entorno proporciona una plataforma flexible y eficiente para escribir y ejecutar

pruebas unitarias en proyectos de software. NUnit facilita la creación de casos de prueba, la ejecución y el análisis de resultados, permitiendo a los desarrolladores identificar y corregir rápidamente posibles problemas en unidades de código específicas. La integración con VSCode agrega una capa adicional de comodidad y productividad, permitiendo a los equipos de desarrollo realizar pruebas unitarias de manera efectiva dentro de un entorno familiar y versátil.

- *Selenium WebDriver para pruebas automatizadas:*

Cuando se trata de pruebas automatizadas, *Selenium WebDriver* es una herramienta líder que ha demostrado ser invaluable. Se especializa en la automatización de pruebas funcionales, ofreciendo un conjunto robusto de funciones para interactuar con aplicaciones web. Selenium WebDriver permite la creación de scripts de prueba en varios lenguajes de programación, proporcionando flexibilidad a los equipos de desarrollo. Su capacidad para simular las interacciones del usuario, como hacer clic, completar formularios y navegar por páginas, facilita la creación de pruebas exhaustivas y repetibles. La integración de Selenium WebDriver en el proceso de desarrollo agiliza la detección temprana de problemas, mejora la eficiencia y garantiza la consistencia en el rendimiento de las aplicaciones web a lo largo del tiempo.

En conjunto, el uso de *NUnit y Selenium WebDriver* abarca tanto las pruebas unitarias como las pruebas automatizadas, ofreciendo a los equipos de desarrollo herramientas poderosas y complementarias para garantizar la calidad y la confiabilidad de sus productos de software. La combinación de estas tecnologías facilita una estrategia integral de pruebas que aborda tanto la funcionalidad de unidades específicas como la evaluación completa del comportamiento de las aplicaciones en entornos reales.

PRÁCTICAS FORMATIVAS AUTOGESTIONADAS

Continuamos con la presentación de prácticas formativas autogestionadas como ciclo de preparación, de formación, de desarrollo de habilidades y competencias que luego se pondrán al servicio de las diferentes relaciones socioculturales, económicas, productivas, etc. que caracterizan el mundo del trabajo.

Por lo que para adquirir aquellas capacidades es central poner el foco en procesos de aprendizaje centrados en prácticas formativas que nos permitan poner en acción y reflexión a partir de situaciones laborales reales o simuladas.

Así, en la materia "Metodología de pruebas de sistemas", pensamos prácticas formativas que promuevan las capacidades del perfil del/la egresado/a.

¿A qué nos referimos a que son "autogestionadas"?

- No son obligatorias, pero altamente recomendadas para poner en práctica los contenidos de la materia y prepararte para los trabajos prácticos integrados que son calificados por el/la docente.
- Autocorrección y autoevaluación: Se presentan las claves de autocorrección para comparar tus soluciones con otras posibles respuestas.
- Autonomía en el aprendizaje: Permite administrar los tiempos y trabajar en estas actividades de forma autónoma. Favorece a que el estudiante establezca un plan de estudio y mantenga una rutina para mantener un buen ritmo de trabajo.
- Interactuar con otros/as estudiantes: Trabajar con compañeros/as puede enriquecer tu aprendizaje. Se puede formar grupos de estudio o participar en sesiones de estudio en línea para discutir los temas y resolver dudas de manera colaborativa.
- Preguntar al/la docente si es necesario: Aunque estas actividades no sean obligatorias, podés escribirle al docente si tenés dudas o necesitás aclaraciones adicionales sobre las consignas. Ellos/as están allí para ayudarte en tu proceso de aprendizaje.

PLANIFICACIÓN DE CONTENIDOS

Unidad 1: Introducción a casos de pruebas y estrategías
Introducción general a MPS.
Métodos de la estrategia de caja negra.
Métodos de la estrategia de caja blanca.
Métodos para pruebas de errores.

Unidad 2: Desarrollo de casos de pruebas y tipos de estrategías
Métodos de prueba del enfoque estático
Pruebas unitarias con con la herramienta Nunit + Vscode + C#
Ciclo iterativo para escribir pruebas

Unidad 3: Niveles de pruebas
Pruebas de integración, sistemas y aceptación
Beneficios de la automatización con Selenium
Funcionalidades Selenium Web Driver
Introducción a Cypress [Opcional]

A continuación se presenta una distribución de los contenidos por semana:

Semana 1 Introducción general MPS	1	Introducción a las pruebas de software. Principios y procesos del testing. Procesos de comprobación: verificación y validación. Los enfoques en MPS (estático y dinámico). Ciclo de desarrollo y despliegue de software. Niveles y tipos de pruebas.
Semana 2 Estrategia de caja negra	1	Testeo de caja negra y de caja blanca Generación de casos de pruebas usando estrategias de caja negra: partición de clases de equivalencia, análisis de valores límites, gráficos de causa y efecto, adivinación de errores. Pruebas funcionales: generación de casos o datos de prueba, clases de equivalencia.
Semana 3 Estrategia de caja blanca	1	Repaso de estrategias de testing y estrategia de caja negra. Estrategia de caja blanca. Diagramas de flujo. Generación de casos de pruebas usando estrategias de caja blanca. Pruebas funcionales: generación de casos o datos de prueba. Registro de fallas y seguimiento de fallas.
Semana 4 Métodos para reporte de errores	1	Repaso de estrategias de caja negra y caja blanca. Introducción a Bug Advocacy. Objeciones del equipo de desarrollo. Registro de fallas e informes técnicos. Seguimiento de defectos.
Semana 5 Métodos de prueba del enfoque estático	2	Repaso de enfoque estático de pruebas. Pruebas estructurales: Pruebas estáticas. Inspecciones de código: Procedimiento, objetivos y beneficios. Errores comunes encontrados en inspecciones. Walkthroughs. Pruebas de escritorio.Calificaciones de Pares - **Entrega TP 1**
Semana 6 Pruebas unitarias (Parte 1)	2	Repaso de niveles de pruebas. Pruebas unitarias. Frameworks de pruebas unitarias en C#: NUnit. Escritura de pruebas unitarias. Pruebas unitarias con parámetros. Automatización de pruebas: Introducción y tipos de frameworks de. automatización.
Semana 7 Pruebas unitarias (Parte 2)	2	Continuación de pruebas unitarias. Introducción a los stubs. Introducción a los mocks. Frameworks de pruebas unitarias: Refactorización de código
Semana 8	1 y 2	**Primer parcial**
Semana 9 Ciclo iterativo para escribir pruebas	3	Test-Driven Development. Acceptance Test-Driven Development. Behavior-Driven Development. Story Test-Driven Development. Need-Driven Development.
Semana 10 Pruebas de integración, sistemas y aceptación	3	Pruebas de Integración. Estrategias de Integración. Pruebas de regresión. Tipos de pruebas de sistema.
Semana 11 Pruebas de integración, sistemas y aceptación	3	Pruebas de aceptación. Tipos de pruebas de aceptación. **Entrega TP 2**

Semana 12 Beneficios de la automatización	3	Beneficios y desafíos de las pruebas automatizadas Selenium como herramienta de automatización y seguimiento de ejecución de pruebas. Instalación y configuración de Selenium WebDriver. Componentes básicos de Selenium
Semana 13 Funcionalidades Selenium Web Driver	3	TSelenium WebDriver Waits. Esperas explícitas, implícitas y fluidas. Selenium WebDriver Elements. File Uploads. Locators Finders. Interactions. Information
Semana 14 Interacciones en aplicación web	3	Selenium Web Driver Interactions: Mecanismos de obtención de información del navegador. Navegación en la página web. Alertas, Prompts y Confirmaciones. Cookies. Windows y Tabs Autenticador Virtual.
Semana 15		**Segundo parcial**
Semana 16		**Recuperatorios**

Material opcional: Introducción a Cypress. Diferencias entre Selenium y Cypress. Cypress Open. Cypress Launchpad. Configuración de Cypress. Consulta de elementos con Cypress. Cadenas de comandos. Aserciones con Cypress. Implementación de pruebas End-to-end

Condiciones de regularización de la cursada

Haber aprobado con un mínimo de nota 4 (cuatro) los dos exámenes parciales.

• **Recuperatorios de exámenes parciales**:

- Se recupera en caso de nota menor a 4, ausencia o no conectarse al Meet propuesto.
- Se podrá recuperar únicamente uno de los dos parciales, en ambos casos el recuperatorio a rendir será el correspondiente del parcial a recuperar.
- La nota del recuperatorio reemplaza, sin excepción, a la nota del examen que se recupera.

<u>Condiciones de aprobación de la materia</u>

Promoción:

• Calificación de 7 (siete) o más en cada una de las 2 (dos) evaluaciones parciales.

• Entregar los dos Trabajos Prácticos.

• Aprobar los dos Trabajos Prácticos.

Final: aquellos que posean una calificación menor a 7 (siete) en alguna de las evaluaciones parciales, hayan requerido presentarse en instancia de recuperatorio y/o no hayan cumplimentado los requisitos establecidos de los Trabajos Prácticos.

BIBLIOGRAFÍA

- Hamilton, T. (2022, August 27). Integration Testing: What is, Types with Example. Guru99. Retrieved October 23, 2022, from https://www.guru99.com/integration-testing.html
- Integration Testing - javatpoint. (n.d.). www.javatpoint.com. Retrieved October 23, 2022, from https://www.javatpoint.com/integration-testing
- Jacobson, Ivar. (1994). Object Oriented Software Engineering. Estados Unidos: Addison Wesley
- Kaner, C. (2000). How to Win Friends, and SToMp BUGs. (Not necessarily in that order.) (1.1 ed.).
- Kniberg, H. (2007). Scrum and XP from the Trenches (Enterprise Software Development) (1st ed.). Lulu.com.
- Lucidchart (2022) Qué es un diagrama de flujo. Recuperado 4 de julio de 2022, de https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo
- Myers, Glenford (1983). El arte de Probar el Software. Editorial El Ateneo.
- Myers, Glenford (2011). The art of software testing- 3rd Edition. Editorial Wiley.
- Osherove, R. (2014). The Art of Unit Testing (2nd ed.). Manning Publications Co.
- ¿Qué es la prueba de regresión? Definición, herramientas, método y ejemplo. (n.d.). Pruebas De Automatización. Retrieved October 23, 2022, from https://spa.myservername.com/what-is-regression-testing
- Salazar Martínez, E. S. M. (2015, 1 enero). Procedimiento para realizar pruebas de Caja Blanca. Informatica Jurídica. Recuperado 4 de julio de 2022, de https://www.informatica-juridica.com
- Vargas, J. R. (2020, December 18). .Net 5: Creando una prueba unitaria NUnit con Visual Code.
- Vargas, J. R. Retrieved July 17, 2022, from https://jarvars.github.io/Creando-una-prueba-unitaria-con-Visual-Code/
- System Testing - javatpoint. (n.d.). www.javatpoint.com. Retrieved October 23, 2022, from https://www.javatpoint.com/system-testing

Material multimedia:

- Introducción a Selenium Testing. (n.d.). QAlified. Retrieved November 1, 2022, from https://qalified.com/es/introduccion-a-selenium-testing/
- Selenium Overview. (n.d.). Selenium. https://www.selenium.dev/documentation/overview/
- NUNIT https://docs.nunit.org/articles/nunit/intro.html
- TestingEducation. (2018, June 24). BBST Bug Advocacy Lecture 1: Basic Concept BBST Bug Advocacy [Video]. YouTube. https://www.youtube.com/watch?v=qf6xLGIsaoM