San Francisco, 28 de mayo de 2025

VISTO la Resolución de Consejo Directivo Nº 557/2016, la Ordenanza N° 1622 y la Ordenanza Nº 2018, y

CONSIDERANDO:

Que la Resolución C.D. Nº 557/2016 aprueba el modelo de planificación y programa analítico utilizado por la Facultad Regional San Francisco.

Que la Ordenanza 1622 Reglamento de Estudio para todas las Tecnicaturas en la Universidad Tecnológica Nacional, en su artículo 6.2 establece "El programa sobre el cual versará la instancia de evaluación final será el programa analítico completo de la asignatura, aprobado por el Consejo Directivo y vigente al momento de rendir."

Que la Ordenanza Nº 2018 establece el nuevo Diseño Curricular y los programas sintéticos de las asignaturas de la carrera Tecnicatura Universitaria en Programación - Plan 2024.

Que la Comisión de Enseñanza del Consejo Directivo de la Facultad Regional San Francisco, analiza los antecedentes y recomienda avalar la solicitud.

Que el dictado de la medida se efectúa en uso de las atribuciones otorgadas por el Estatuto Universitario.

Por ello,

EL CONSEJO DIRECTIVO DE LA FACULTAD REGIONAL SAN FRANCISCO
DE LA UNIVERSIDAD TECNOLÓGICA NACIONAL
RESUELVE:

ARTÍCULO 1°.- Aprobar el Programa Analítico de la asignatura Metodología de Sistemas II, correspondiente al 2º nivel de la carrera Tecnicatura Universitaria en Programación (Plan 2024, Ordenanza Nº 2018 del Diseño Curricular), con una carga horaria de 2 horas anuales y régimen de dictado cuatrimestral (2º cuatrimestre), conforme al ANEXO I que se adjunta a la presente.
ARTÍCULO 2°.- Regístrese. Comuníquese. Cumplido, archívese.

RESOLUCIÓN CD N°: 384/2025

Ing. JUAN C. CALLONI
Secretario
Académico

Ing. Alberto R. TOLOZA
Decano

ANEXO I
RESOLUCIÓN CD N° 384/2025

UTN
UNIVERSIDAD TECNOLOGICA NACIONAL

FACULTAD REGIONAL SAN FRANCISCO

Carrera:

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN

Asignatura

Metodología de Sistemas II

PROGRAMA ANALÍTICO

PLAN 2024

Contenido

1. Datos administrativos de la asignatura 2
2. Programa analítico eje/unidad 3

1. DATOS ADMINISTRATIVOS DE LA ASIGNATURA

Carrera:	Tecnicatura Universitaria en programación
Asignatura:	Metodología de Sistemas II
Nivel de la carrera	2°
Duración	2hs anuales
Régimen:	Cuatrimestral
Área:	Disciplinas Tecnológicas

2. PROGRAMA ANALÍTICO EJE/UNIDAD

Eje Temático Nº 1: Herramientas de Versionado de Código Avanzado

Unidad Nº 1: Git Avanzado

Objetivo: Dominar funcionalidades avanzadas de Git como rebase, cherry-pick, stash y manejo de conflictos. Estas herramientas son esenciales para flujos de trabajo colaborativos y para mantener un historial limpio y comprensible en proyectos reales.

Unidad Nº 2: Gitflow

Objetivo: Comprender y aplicar el flujo de trabajo Gitflow, ampliamente utilizado en equipos profesionales para organizar el desarrollo de funcionalidades, corrección de errores y releases en proyectos complejos.

Unidad N° 3: Revisión de código

Objetivo: Incorporar la revisión de código como una práctica fundamental para mejorar la calidad del software, compartir conocimiento entre pares y detectar errores antes de llegar a producción.

Unidad N° 4: Integración continua, despliegue continuo

Objetivo: Implementar procesos de CI/CD automatizados que permitan validar, construir, probar y desplegar aplicaciones de forma segura, ágil y frecuente, siguiendo las prácticas modernas de DevOps.

Eje Temático Nº 2: Buenas prácticas de desarrollo

Unidad Nº 3: Refactoring

Objetivo: Aprender a mejorar el diseño interno del código sin alterar su comportamiento, manteniendo sistemas sostenibles a largo plazo y facilitando su mantenimiento.

Unidad Nº 4: Clean Code

Objetivo: Aplicar principios de código limpio para escribir software legible, mantenible y con menor propensión a errores, tal como se espera en ambientes profesionales.

Unidad N° 5: Patrones de diseño

Objetivo: Utilizar patrones de diseño para resolver problemas comunes de arquitectura y diseño de software, favoreciendo la reutilización de soluciones comprobadas.

Unidad N° 6: Principios de buenas prácticas de desarrollo

Objetivo: Incorporar principios como SOLID, DRY y YAGNI, claves para el desarrollo de software escalable y entendible, y muy valorados en entornos laborales exigentes.

Eje Temático Nº 3: *Testing/*Pruebas de Software

Unidad Nº 7: Pirámide de pruebas

Objetivo: Comprender la estrategia de pruebas recomendada que prioriza las pruebas unitarias, minimiza las pruebas e2e y distribuye correctamente los tipos de testing para lograr eficiencia y cobertura.

Unidad Nº 8: Pruebas unitarias

Objetivo: Comprender la importancia de validar de forma aislada cada unidad funcional del código. Esta unidad busca que el estudiante adquiera la capacidad de construir pruebas unitarias efectivas para garantizar la correcta lógica interna y detectar errores en etapas tempranas del desarrollo.

Unidad Nº 9: Pruebas de Integración

Objetivo: Entender el rol de las pruebas de integración dentro de la estrategia global de testing. Se busca que el estudiante pueda construir pruebas que verifiquen el correcto funcionamiento entre módulos y componentes de un sistema, desarrollando habilidades para detectar problemas en la interconexión de partes.

Unidad N° 10: Pruebas de e2e

Objetivo: Simular experiencias completas del usuario final sobre una aplicación. Esta unidad apunta a que el estudiante sea capaz de automatizar flujos de uso reales, comprendiendo el valor de este tipo de pruebas para la detección de errores funcionales antes del despliegue.

Unidad N° 11: TDD, *Mocking y Stubbing* y Cobertura de Testing

Objetivo: Adoptar el enfoque de desarrollo guiado por pruebas (TDD), junto con técnicas de mocking y stubbing para aislar dependencias y lograr alta cobertura con pruebas significativas.

Eje Temático N° 4: Arquitectura de Software

Unidad Nº 12: Introducción a las Arquitecturas de Software

Objetivo: Comprender el rol de la arquitectura en la construcción de sistemas robustos, y su impacto en la escalabilidad, mantenibilidad y calidad del software.

Unidad Nº 13: Tipos de Arquitecturas

Objetivo: Explorar los diferentes tipos de arquitecturas utilizadas en la industria (en capas, hexagonal, basada en eventos, etc.), y sus ventajas según el contexto.

Unidad Nº 14: Arquitecturas Monolíticas

Objetivo: Analizar los beneficios y limitaciones de las arquitecturas monolíticas, su aplicabilidad en sistemas simples o en etapas iniciales de un proyecto.

Unidad N° 15: Arquitecturas Distribuidas

Objetivo: Estudiar las arquitecturas distribuidas (como microservicios), sus desafíos y ventajas para escalar aplicaciones en entornos modernos con alta demanda y disponibilidad.