Ministerio de Capital Humano
Universidad Tecnológica Nacional
Facultad Regional Rosario

ROSARIO, 17 DIC 2024

VISTO: La solicitud presentada por las Coordinadoras de la Tecnicatura Universitaria en Programación, relacionada con la aprobación de los programas analíticos de asignaturas pertenecientes a la mencionada carrera (Exp. ID N° 8169133), y

CONSIDERANDO:

Que los objetivos y contenidos de estos se ajustan a la reglamentación vigente.

Que la Comisión de Enseñanza evaluó y aconsejó su aprobación.

Por ello y atento a las atribuciones otorgadas por el artículo Nº 85 del Estatuto de la Universidad Tecnológica Nacional.

EL CONSEJO DIRECTIVO DE LA FACULTAD REGIONAL ROSARIO

DE LA UNIVERSIDAD TECNOLÓGICA NACIONAL

RESUELVE:

ARTÍCULO 1°.- Aprobar los programas analíticos de las asignaturas que se detallan a continuación de la carrera Tecnicatura Universitaria en Programación – Plan 2024 y que se agregan como anexo de la presente Resolución:

- Metodología de Sistemas I
- Programación III
- Base de Datos II

ARTÍCULO 2°.- Regístrese. Comuníquese. Cumplido, archívese.

RESOLUCIÓN Nº 847

UTN FRRo
LR

Ing. RUBEN FERNANDO CICCARELLI
DECANO

Ing. ANTONIO LUIS MUIÑOS
Secretario Académico

Ministerio de Educación,
Universidad Tecnológica Nacional
Facultad Regional Rosario

RESOLUCION N° 847

ANEXO N° I

I. Datos Generales de la Actividad Curricular

Carrera: TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN	
Asignatura: Metodología de Sistemas I	
Plan de estudio: Ordenanza Nº 2018	Código: 11
Nivel de Implementación: 2° Año	Régimen: Presencial/Virtual
Cuatrimestre: 1° Cuatrimestre	Horas reloj/semana: 4
Área: Disciplinas Tecnológicas	Horas reloj/cuatrimestre: 64

II. Objetivos

Objetivos Generales:

- Distinguir y comprender enfoques y prácticas de modelos, técnicas y lenguajes del proceso de desarrollo de software.
- Interpretar los documentos generados por las herramientas de programación.
- Implementar acciones correctivas a los resultados generados por las operaciones de pruebas.
- Valorar el trabajo en equipo.
- Identificar el rol del programador en la gestión de un proyecto de software.

Objetivos Específicos:

- Comprender los principales componentes del proceso de desarrollo de software, desde la identificación de requisitos hasta el mantenimiento, para estructurar proyectos que respondan a las necesidades del usuario y garanticen su funcionalidad.
- Seleccionar y utilizar herramientas adecuadas para cada etapa del desarrollo de software, promoviendo la eficiencia en la codificación, el diseño de sistemas, el control de versiones y la automatización de pruebas.
- Aplicar técnicas y herramientas de gestión de proyectos de software que integren metodologías ágiles y tradicionales, asegurando la planificación, organización y ejecución efectiva de proyectos en equipos colaborativos.
- Diseñar y realizar pruebas de software, analizando los resultados mediante métricas clave para garantizar la calidad, el rendimiento y la satisfacción de los objetivos del proyecto en el contexto del ciclo de vida del desarrollo de software.

III. Contenidos

UNIDAD N° 1: Proceso de desarrollo de software:

Temas: Introducción al desarrollo de software: definiciones y conceptos básicos. Tipos de procesos de desarrollo. Ciclo de vida del desarrollo de software (SDLC): análisis, diseño, implementación, pruebas y mantenimiento. Modelos clásicos (Cascada, Espiral) y modelos modernos (Iterativo, Ágil, DevOps). Principales componentes del diseño del sistema y arquitectura: diseño de bases de datos, interfaces de usuario y lógica del negocio. Buenas prácticas para asegurar la calidad en el desarrollo. Análisis de casos reales de procesos exitosos y fallidos en la industria del software.

UNIDAD N° 2: Herramientas para el desarrollo de software:

Temas: Herramientas fundamentales para el desarrollo de software, abarcando entornos de desarrollo, gestión de versiones, diseño de interfaces, pruebas y despliegue. Características y funcionalidades esenciales de los entornos de desarrollo integrados (IDEs). Sistemas de control de versiones: conceptos básicos y aplicaciones para rastreo de cambios y colaboración. Herramientas para el diseño y prototipado de interfaces centradas en el usuario. Automatización de pruebas en diferentes etapas del desarrollo. Introducción a herramientas para la integración y entrega continua (CI/CD): características principales y beneficios. Aplicaciones prácticas en proyectos de software utilizando herramientas que se adapten al contexto del equipo o empresa.

UNIDAD N° 3: Gestión de proyectos de software:

Temas: Principios fundamentales de la gestión de proyectos de software. Introducción a metodologías tradicionales y ágiles. Metodologías ágiles: SCRUM, Lean y Kanban. Roles, ceremonias, artefactos y ciclo de vida en SCRUM. Combinación de métodos ágiles con diseño centrado en el usuario: ventajas, desafíos y enfoques prácticos. Estimaciones y mediciones de esfuerzo en proyectos de software. Gestión de riesgos y cambios en proyectos de software. Actividades prácticas para planificar y gestionar proyectos utilizando tableros Kanban.

UNIDAD N° 4: Pruebas de software y análisis de resultados

Temas: Introducción a las pruebas de software: importancia y objetivos. Tipos de pruebas: unitarias, de integración, funcionales, de aceptación y de rendimiento. Herramientas para la ejecución de pruebas. Diseño de estrategias de prueba: cobertura, gestión de casos de prueba, planificación. Análisis de resultados de pruebas: métricas clave como cobertura, tasa de defectos y rendimiento. Uso de herramientas de automatización en la validación del software. Pruebas de aceptación del usuario (UAT). Prácticas recomendadas para la implementación de pruebas en el ciclo de vida del desarrollo.

IV. Bibliografía

Bibliografía sugerida:

UNIDAD N° 1: Proceso de desarrollo de software

Pressman, R. S. (2006). Ingeniería del software: Un enfoque práctico. McGraw-Hill.

Sommerville, I. (2011). Ingeniería del software. Addison-Wesley.

Pfleeger, S. L., & Atlee, J. M. (2006). Análisis y diseño del software: Enfoque práctico. Pearson.

Villalobos, J. (2012). Ciclo de vida del software y metodologías de desarrollo. Alfaomega.

Jiménez, J. (2019). Introducción a la ingeniería del software. Marcombo.

UNIDAD N° 2: Herramientas para el desarrollo de software

Fuggetta, A. (2000). Tecnologías y herramientas para el desarrollo de software. Addison-Wesley.

Molina, A. (2008). Automatización del desarrollo de software: Enfoque y herramientas. Alfaomega.

Prieto, G. (2015). Herramientas modernas en el desarrollo de software. Paraninfo.

Blanchard, S. (2017). Introducción a herramientas de desarrollo: De lo básico a lo avanzado. Ediciones Díaz de Santos.

Valverde, G. (2020). Técnicas y herramientas para la gestión de software. Ecoe Ediciones.

UNIDAD N° 3: Gestión de proyectos de software

Schwaber, K., & Beedle, M. (2002). Desarrollo ágil con Scrum. Addison-Wesley.

Larman, C. (2004). Desarrollo ágil y programación iterativa: Una guía para los gestores. Pearson.

Highsmith, J. (2009). Gestión de proyectos ágiles: Creando productos innovadores. Addison-Wesley.

Rodríguez, L. (2016). Metodologías ágiles en la gestión de proyectos de software. Ediciones Pirámide.

Serrano, M. (2018). Gestión de proyectos de software: Métodos ágiles y tradicionales. Alfaomega.

UNIDAD N° 4: Pruebas de software y análisis de resultados

Myers, G. J. (2011). El arte de probar software. McGraw-Hill.

Black, R. (2009). Pruebas avanzadas de software: Guía para testers y equipos de calidad. Pearson.

Kaner, C., Bach, J., & Pettichord, B. (2002). Lecciones aprendidas en pruebas de software. Wiley.

Muñoz, R. (2015). Técnicas de pruebas de software: Conceptos, estrategias y herramientas. Marcombo.

Pérez, F. (2019). Automatización y análisis en pruebas de software. Alfaomega.

Ministerio de Capital Humano
Universidad Tecnológica Nacional
Facultad Regional Rosario

RESOLUCION N° 847
ANEXO N° I

I. Datos Generales de la Actividad Curricular

Carrera: TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN	
Asignatura: Programación III	
Plan de estudio: Ordenanza Nº 2018	Código materia: 9
Nivel de Implementación: 2° Año	Régimen: Presencial
Cuatrimestre: 1° Cuatrimestre	Horas reloj/semana: 8
Área: Disciplinas Tecnológicas	Horas reloj/cuatrimestre: 128

II. Objetivos

Objetivos Generales: Trabajar y mejorar los conocimientos adquiridos en desarrollo web dados por la materia Laboratorio de Computación II, desarrollando las habilidades modernas requeridas para la entrada al ámbito laboral para creación de aplicaciones web, en conjunto con el perfeccionamiento de buenas prácticas y aprendizaje de librerías complementarias.

Objetivos Específicos:

- Entender qué es React y qué ventajas aporta a la creación de interfaces de usuario modernas.
- Comprender los conceptos básicos de React (lógica de componentes, *state*, manejo de eventos, listas, entre otros)
- Asimilar los conocimientos requeridos sobre la teoría funcional detrás de React y el ciclo de vida de componentes.
- Crear y comprender la conexión entre el lado cliente y el lado servidor mediante la utilización de Node js y Express.
- Ser capaz de realizar formularios modernos y de establecer comunicación con el lado servidor mediante el conocimiento del funcionamiento de promesas en JavaScript.
- Agregar autenticación de usuario y enrutado interno para la navegación del sitio web.
- Interiorizarse en el desarrollo *mobile* mediante la librería React Native

III. Contenidos

UNIDAD N° 1: Introducción a React

React: ¿Qué es React? ¿Por qué lo utilizamos? Diferencias con otros frameworks / librerías. Uso de *let* y *const*. *Arrow functions*. Operadores *spread* y *rest*. Utilización de *destructuring*. Funciones arreglo claves: *map*, *filter*, *reduce*.

Primeros pasos en React: Lógica de componentes. ¿Cómo iniciamos un nuevo proyecto en React? Análisis de este. Teoría y práctica referida a JSX. NodeJS y la creación de nuestra primera react app. Crear nuestro primer componente, contenido estático y dinámico. Concepto de *props*. Composición de componentes, *two way data binding* (comparación con otros Frameworks). Virtual DOM.

UNIDAD N° 2: Funcionamiento de React.

Manejo de state en React: *event handling*, conceptos relacionados al *state*: teoría y propósito, concepto del *hook useState()*, ejemplificación mediante un formulario simple y t*wo way data binding*, mover el *state* a través del árbol de componentes.

Listas y renderizado condicional: renderizado condicional de componentes, conformación de listas de componentes.

UNIDAD N°3: Creación de UI/UX

Código de lado servidor: requests / responses, controladores, modelos, llamadas a la base de datos, RESTFUL API.

Formularios y pedidos al servidor: *useEffect()* y *useRef()*, autenticación de usuarios, validación en formularios, diferencia entre componentes controlados y no controlados, pedidos al servidor, *axios/fetch*, promesas (*then/catch*) y *async-await*.

UNIDAD N°4: Conceptos avanzados

Componentes de orden alto (HOC): teoría detrás, ventajas, funcionamiento, utilización (autenticación, ruteo, temas), explicación del concepto de *Context* y *useContext()*.

Ampliación de conocimientos en hooks: *useCallback(), useMemo(), custom hooks*

UNIDAD N°5: Desarrollo *mobile*

React Native: Introducción a React Native, Expo, componentes específicos de React Native, navegación en celulares.

IV. Bibliografía

Obligatoria:

- Apuntes de la cátedra.
- Desarrollo web en entorno cliente (https://elibro.net/es/lc/elibrocom/titulos/62488)
- Documentación oficial de React (https://es.reactjs.org/docs/getting-started.html)
- Documentación de ayuda en JavaScript (https://developer.mozilla.org/es/docs/Web/JavaScript)

Optativa:

- *Learning React, 2nd Edition – Banks, Porcello (https://www.oreilly.com/library/view/learning-react-2nd/9781492051718/)*

Ministerio de Capital Humano
Universidad Tecnológica Nacional
Facultad Regional Rosario

RESOLUCION N° 847

ANEXO N° I

I. Datos Generales de la Actividad Curricular

Carrera: TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN	
Asignatura: Base de Datos II	
Plan de estudio: Ordenanza Nº 2018	Código materia: 10
Nivel de Implementación: 2º año	Régimen: Presencial
Cuatrimestre: 1º Cuatrimestre	Horas reloj/semana: 4
Área: Disciplinas Tecnológicas	Horas reloj/cuatrimestre: 64

II. Objetivos

- Analizar la consistencia e integridad de los datos.
- Programar acciones rutinarias de mantenimiento de la base de datos.
- Diseñar consultas para obtener información resumen.
- Formular subconsultas dentro de una consulta principal.
- Utilizar un Sistema de Gestión de Base de datos para crear objetos.
- Identificar los diversos modelos conceptuales de base de datos no relacionales.
- Emplear metodologías de diseño de Base de Datos no relacionales.

III. Contenidos

UNIDAD N° 1: Consistencia e integridad en los datos de una Base de Datos.

Definición, conceptos, historia y evolución de la consistencia e integridad de los datos en una Base de Datos. Tipos de integridad en una Base de Datos. Coherencia de los datos en una Base de Datos.

UNIDAD N° 2: Rutinas en Bases de Datos

Utilizar funciones definidas por el usuario. Preparación para desarrollar rutinas en las Bases de Datos. Desplegar y ejecutar rutinas Comparar y modificar rutinas.

UNIDAD N° 3: Subconsultas en Bases de Datos

¿Qué son las subconsultas?. Análisis de distintos tipos de subconsultas. Aplicación e implementación de subconsultas en MySql utilizando MySqlWorbench.

UNIDAD N° 4: Implementación ORM

Definición, conceptos, historia y evolución de la persistencia en los datos de un sistema. ¿Qué son los ORM? Análisis de ORM en la actualidad.

UNIDAD N° 5: Implementación de bases de datos NO relacionales

¿Qué son las Bases de Datos NO relacionales? Surgimiento y conceptualización de Bases de Datos NoSQL. Comparativa de las Bases de Datos NO relaciones con las Relacionales. Implementar diseños de Bases de Datos NO relacionales.

Sistemas no SQL, sistema key value, motores de búsqueda de texto complete.

IV. Bibliografía

Obligatoria:

Autor	Título	Editorial
ELMASRI / NAVATHE	Fundamentos de sistemas de bases de datos	ADISON WESLEY LONGMAN

Optativa:

Autor	Título	Editorial
CONNOLLY THOMAS M.	Sistemas de Bases de Datos: un enfoque Práctico para Diseño, Implementación y gestión	PEARSON EDUCATION
Eric Redmond, Jim Wilson.	Seven databases in seven weeks: a guide to Modern Databases and the NoSQL Movement.	Pragmatic Programmers (2012).
