Programación Avanzada

Planificación Ciclo lectivo 2026

1. Datos administrativos de la asignatura

Departamento:	Sistemas de Información	Carrera	Ing. en Sistemas de Información
Asignatura:	**Programación avanzada**		
Nivel de la carrera	4to Año	Duración	Cuatrimestral
Plan	2023 Ord. 1877	Modalidad	Presencial
Bloque Curricular:	Tecnologías Básicas		
Área	Desarrollo de Software	RTF	-
Horas cátedra semanal:	2	Horas reloj anual total:	48
Carga horaria no presencial semanal	----------	% horas no presenciales	----------
Profesor/es Titular/Asociado/Adjunto:	Mg. Vanzetti Juan José	Dedicación:	Simple
Auxiliar/es de 1°/JTP		Dedicación:	
Competencias	Específicas	Genéricas	
	CE1.1	CG5 – CG6	

2. Presentación, Fundamentación

Las habilidades específicas en el desarrollo de sistemas de información han desarrollado una creciente importancia en la industria del Software. El mercado actual y el perfil de la carrera, demanda en nuestros egresados conocimientos sobre programación y construcción de sistemas. En las asignaturas obligatorias de los dos primeros años del Área Programación, se abordan gran cantidad de conocimientos y actitudes que son puestos en práctica en la construcción de sistemas de información. A la vez, no se tratan en profundidad muchos otros aspectos específicos y particulares, los cuales necesitan ser tratados en forma planificada y detenida, para poder desarrollar en los alumnos las destrezas necesarias para afrontar la participación adecuada en la dirección o en la ejecución del desarrollo de sistema de información.

Relación de la asignatura con el perfil de egreso.

Brinda habilidades técnicas y blandas indispensables al diseño, desarrollo, testing e implementación e integración de aplicaciones.

Relación de la asignatura con los alcances del título.

Brinda las herramientas necesarias para contemplar desarrollos de software desde los distintos roles y perspectivas que puede abordar un profesional de sistemas.

3. Relación de la asignatura con las competencias de egreso de la carrera

Se detalla en la siguiente tabla, la relación de la asignatura con las competencias de egreso específicas, genéricas tecnológicas, sociales, políticas y actitudinales de la carrera. Indicando a cuáles competencias de egreso tributa y en qué nivel (No Tributa, 1=bajo, 2=medio, 3=alto).

Competencias	Nivel
Competencias genéricas tecnológicas (CG):	
CG. 1. Identificar, formular y resolver problemas de ingeniería.,	No Tributa
CG. 2. Concebir, diseñar y desarrollar proyectos de ingeniería.	No Tributa
CG. 3. Gestionar, planificar, ejecutar y controlar proyectos de ingeniería.	No Tributa
CG. 4. Utilizar de manera efectiva las técnicas y herramientas de aplicación en la ingeniería.	No Tributa
CG. 5. Contribuir a la generación de desarrollos tecnológicos y/o innovaciones tecnológicas.	2
Competencias genéricas sociales, políticas y actitudinales (CG):	

CG. 6. Desempeñarse de manera efectiva en equipos de trabajo.	2
CG. 7. Comunicarse con efectividad	No Tributa
CG. 8. Actuar con ética, responsabilidad profesional y compromiso social, considerando el impacto económico, social y ambiental de su actividad en el contexto local y global.	No Tributa
CG. 9. Aprender en forma continua y autónoma.	No Tributa
CG. 10. Actuar con espíritu emprendedor	No Tributa
Competencias Específicas (CE):	
CE. 1.1. Especificar, proyectar y desarrollar sistemas de información para concebir soluciones tecnológicas que permitan resolver situaciones en las organizaciones mediante el empleo de metodologías de sistemas y tecnologías asociadas a los sistemas de información	3
CE. 1.2. Especificar, proyectar y desarrollar sistemas de comunicación de datos, evaluando posibles soluciones tecnológicas disponibles para dar soporte a los sistemas de información en lo referido al procesamiento y comunicación de datos.	No Tributa
CE. 1.3. Especificar, proyectar y desarrollar software para la elaboración de soluciones informáticas con el propósito de resolver problemas estratégicos y operativos, así como de servicios y de negocios, en el marco de una actividad económica que sea social y ambientalmente sustentable.	No Tributa
CE. 2.1. Proyectar y dirigir lo referido a seguridad informática para seleccionar y aplicar técnicas, herramientas, métodos y normas, garantizando la seguridad y privacidad de la información procesada y generada por los sistemas de información.	No Tributa
CE. 3.1. Establecer métricas y normas de calidad de software para medir, evaluar, controlar y monitorear el rendimiento, impulsando mejoras de acuerdo a técnicas y normas vigentes definidas por los organismos de estandarización.	No Tributa
CE. 4.1. Certificar el funcionamiento, condición de uso o estado de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software para asegurar la generación de los resultados deseados en función de restricciones de tiempo y recursos establecidos.	No Tributa
CE. 5.1. Dirigir y controlar la implementación, operación y mantenimiento de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software, a los fines de alcanzar los objetivos fijados por la organización.	No Tributa

CE. 6.1. Asesorar y capacitar a organizaciones, empresas, organismos públicos o privados en la adquisición, instalación y uso, en lo que respecta a sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software, a los fines de un uso correcto de los sistemas intervinientes.	No Tributa
CE. 7.1. Realizar pericias, tasaciones y arbitrajes relacionados con su actividad profesional, respetando marcos normativos y jurídicos con el objeto de asesorar a las partes o a los tribunales de Justicia.	No Tributa

4. Propósito

El propósito de esta materia es que el alumno/a construya programas en un entorno visual orientado a objetos. Además, que le permita gestionar datos de una base de datos relacional de manera consistente, desarrolle listados y reportes estadísticos que colaboren en la toma de decisiones del sistema de información, fortaleciendo el trabajo en equipo mediante una aproximación a una experiencia profesional concreta.

5. Objetivos establecidos en el Diseño Curricular

- Introducir y capacitar al alumno en la construcción y programación avanzada de sistemas de información.
- Realizar prácticas avanzadas en herramientas de programación.
- Adquirir conocimientos que le permitan construir programas en forma manual o con uso de asistentes en entornos visuales.
- Comprender los roles de los integrantes de un equipo en un proyecto de desarrollo de software.

6. Contenidos Mínimos establecidos en el Diseño Curricular

No Aplica

7. Resultados de aprendizaje

Se describen los Resultados de aprendizaje (RA) a promover en el desarrollo de la asignatura.

Identificador del RA	Redacción

RA1	Aplica componentes y conceptos fundamentales de desarrollo frontend utilizando React, JavaScript y TypeScript, creando interfaces de usuario dinámicas y escalables que integran APIs y cumplan con las mejores prácticas de testing automatizado para garantizar la calidad del software.
RA2	Desarrolla servicios backend utilizando Node.js y Spring Boot, aplicando principios de diseño y patrones de arquitectura que integran bases de datos SQL y NoSQL, para construir APIs eficientes y escalables que satisfacen los requerimientos de sistemas modernos.
RA3	Implementa estrategias de testing automatizado en aplicaciones frontend y backend, realizando pruebas unitarias, de integración y de extremo a extremo, configurando entornos aislados para garantizar la calidad y estabilidad del software en diferentes entornos de desarrollo.
RA4	Aplica técnicas de mantenimiento y refactorización en proyectos de software, mejorando la calidad del código y gestionando la deuda técnica para asegurar la sostenibilidad y escalabilidad de las aplicaciones a lo largo del tiempo**.**
RA5	Configura herramienta de inteligencia empresarial para gestionar y visualizar datos provenientes de bases de datos NoSQL, aplicando análisis en tiempo real que faciliten la toma de decisiones estratégicas en entornos empresariales**.**
RA6	Gestiona proyectos de desarrollo de software mediante metodologías ágiles, utilizando tableros de gestión y prácticas de colaboración que incluyan Pull Requests y revisiones de código para asegurar el cumplimiento de los objetivos establecidos**.**

8. Asignaturas correlativas previas

Para cursar y rendir

Cursada:

- Desarrollo de Software

Aprobada:

- No aplica

9. Asignaturas correlativas posteriores

- Calidad en Desarrollo de Software

10. Programa analítico, Contenidos temáticos

Ejes temáticos	Contenidos
Fundamentos de React	Componentes, estado, props y ciclos de vida. Introducción a TypeScript: Tipos básicos y avanzados, interfaces y tipos genéricos. Consumo de APIs: Uso de axios y hooks para llamadas asíncronas. Optimización de rendimiento y optimización de renderizados.
Creación de API RESTful en Node.js y Express	Rutas, middlewares y manejo de errores. Uso de Spring Boot: Configuración, controladores y servicios, junto con la Inyección de Dependencias Comunicación entre servicios: Implementación de microservicios y API Gateway. Integración de bases de datos: Conexión y manipulación de datos en MongoDB y PostgreSQL.
Testing en backend	Pruebas unitarias y de integración utilizando Jest y JUnit. Automatización de pruebas de API: Configuración y ejecución de pruebas con Postman Testing end-to-end. Herramientas de análisis: Evaluación de cobertura de código y calidad. Introducción a los contenedores y su rol en el testing: Conceptos básicos de contenedores.Ventajas de los entornos aislados en el desarrollo y testing. Comparación con entornos tradicionales. Creación de entornos aislados para pruebas automatizadas. Configuración de entornos para frontend y backend. Definición de dependencias específicas para uso de imágenes base optimizadas para testing.
Principios de refactorización	Identificación de code smells y aplicación de técnicas de mejora. Mejores prácticas de refactorización en frontend y backend: Ajustes en React y Node/Spring Boot. Evaluación y priorización de deuda técnica: Métodos para gestionar y reducir la deuda en proyectos.
Configuración de Metabase	Creación de dashboards y consultas personalizadas. Fundamentos de bases de datos NoSQL: Diseño e integración de MongoDB.

Uso de tableros de gestión (Jira, Trello)	Seguimiento y control de tareas en sprints. Metodologías ágiles: Aplicación de roles, ceremonias y métricas de calidad. Establecimiento de OKRs: Definición y seguimiento de objetivos claros. Prácticas de colaboración: Implementación de Pull Requests y revisiones de código efectivas.

Cuadro de Carga Horaria de Formación Práctica

	Carga horaria	
	Presencial	**A Distancia**
Instancias supervisadas de formación práctica (prácticas en diferentes ámbitos tales como aula, laboratorios, campo u otros)	32	
Proyecto Integrador		
Práctica profesional supervisada		
Carga horaria total	32	

11. Metodología de enseñanza

La materia aborda cada uno de los temas mediante la realización de diferentes trabajos, el cual contempla diferentes entregables La programación de estos trabajos se realiza utilizando el lenguaje distintos lenguajes para el front y el back y de manera grupal. Se podrá emplear cualquier IDE y base de datos. El trabajo consiste en el diseño, construcción y prueba de una aplicación Windows (formularios) que permita gestionar los requerimientos funcionales de una situación problema.

En términos generales se desarrolla desde cero un sistema y de manera iterativa e incremental se va incorporando al desarrollo los requerimientos y tecnologías necesarias para llevarlo a cabo.

El sistema contempla:

ABM de tablas del sistema.

Autenticación básica.

Manejo de transacciones

Manejo de errores.

Gestión de menúes.

Programación en capas.

Listados y reportes gráficos parametrizados con información estadística.

Uso de patrones de programación.
Utilización de API
Generación de casos de prueba automáticos

En cada entregable se detallan los requerimientos funcionales (El alumno podrá utilizar casos de usos/historia del usuario) y las herramientas tecnologías mínimas con la que se debe desarrollar.
A partir del segundo entregable será obligatorio utilizar Git, Maven y JUnit
Los trabajos grupales y su documentación deben almacenarse en el repositorio Git creado a tal efecto. Serán considerados los siguientes aspectos al evaluar:
Puntualidad en los tiempos de entrega.
Dominio técnico del tema evaluado.
Capacidad de fundamentar lo entregado.
Dominio de terminología técnica.
Expresión clara y técnicamente correcta.
Evidencias de ceremonias ágiles (Retro - Review - Planning).

Resumen de Competencias - Resultados de Aprendizaje - Metodologías de Enseñanza de la asignatura.

COMPETENCIA - NIVEL DE TRIBUTACIÓN	**RESULTADO DE APRENDIZAJE (RA)**
CE1.1(3) - CG5(2) – CG6(2)	RA1 - RA2 - RA3- RA4 – RA5 – RA6
EJES TEMÁTICOS - CONTENIDOS	
Fundamentos de React - Creación de API RESTful en Node.js y Express - Testing en backend - Principios de refactorización - Configuración de Metabase - Uso de tableros de gestión (Jira, Trello)	
ACTIVIDADES DEL ESTUDIANTE (FORMATIVAS)	
El **estudiante** deberá realizar las siguientes actividades para alcanzar los **RA** propuestos: - Lectura y análisis de material bibliográfico, diapositivas y videos. - Participar activamente en clase. - Resolver ejercicios prácticos - Interpretar y debatir resultados obtenidos. - Utilizar software de programación como herramienta para la validación. - Realizar el Trabajo Práctico Final. - Elaborar una exposición oral para presentar el TPF.	

COMPETENCIA - NIVEL DE TRIBUTACIÓN	RESULTADO DE APRENDIZAJE (RA)
CE1.1(3) - CG5(2) – CG6(2)	RA1 - RA2 - RA3- RA4 – RA5 – RA6

EJES TEMÁTICOS - CONTENIDOS

Fundamentos de React - Creación de API RESTful en Node.js y Express - Testing en backend - Principios de refactorización - Configuración de Metabase - Uso de tableros de gestión (Jira, Trello)

ACTIVIDADES DEL ESTUDIANTE (FORMATIVAS)

ESTRATEGIAS DE ENSEÑANZA

El **docente** diseñará las siguientes **actividades pedagógicas**:

- Clase expositiva dialogada
- Acompañamiento continuo.
- Diseño de casos de estudios/ guías de trabajos prácticos
- Diseñar el Trabajo Práctico Final.
- Diseñar Instancia de evaluación individual de integración de saberes.
- Diseñar instrumentos de evaluación del TPF.
- Retroalimentación formativa.
- Coordinación de trabajo colaborativo.

12. Recomendaciones para el estudio

- Tener presente el cronograma de la materia.
- Aprovechar las clases del docente.
- En las clases, tener siempre disponible el material de lectura y el software sugerido
- Sé constante, lo ideal es todos los días dedicarle tiempo al estudio.
- Lee el material y mira los videos recomendados por los docentes.
- Lee los temas abordados en la bibliografía sugerida, para tener un conocimiento más profundo de la materia.
- Dedicar tiempo individual de estudio.

13. Metodología de evaluación

Para llevar a cabo la evaluación desde el enfoque formativo es necesario que el docente incorpore en el aula estrategias de evaluación congruentes con las características y necesidades individuales de cada alumno y las colectivas del grupo. El cumplimento en la entrega en tiempo y forma de las actividades prácticas. En caso que las actividades contengan errores, los alumnos tendrán la posibilidad de ir corrigiéndolos a media que transcurre el cuatrimestre en entregas

sucesivas. Al finalizar el cuatrimestre, los alumnos deberán haber concluido correctamente todas las actividades prácticas. Se evaluarán además a los alumnos en instancias parciales

Momentos: Evaluación Diagnóstica, Evaluación formativa o continua

Instrumentos:

- 3 Trabajos práctico y un recuperatorio
- Resolución de guías de trabajos prácticos.

Criterios de Regularidad:

Para obtener la regularidad de la materia el alumno deberá:

- Tener el 75% de asistencias
- Tener completas y entregadas las guías de trabajos prácticos.
- Tener aprobados las instancias de prácticos y teóricos con nota promedio 6 o superior.
- Podrá recuperar una instancia de evaluación práctico - teórico.

Criterios de Promoción:

Para obtener la promoción directa de la materia el alumno deberá:

- Tener el 75% de asistencias
- Tener completas y entregadas las guías de trabajos prácticos.
- Tener aprobados las instancias de evaluación prácticos y teóricos con nota 8 o superior.
- Podrá recuperar una instancia de evaluación práctico - teórico

14. Cronograma de clases año 2026

Semana	Contenido / Actividad
Semana 1	Fundamentos de React: Componentes, estado, props y ciclos de vida.
Semana 2	Introducción a TypeScript: Tipos básicos y avanzados, interfaces y tipos genéricos.
Semana 3	Consumo de APIs: Uso de axios y hooks para llamadas asíncronas. Optimización de rendimiento y optimización de renderizados
Semana 4	Creación de API RESTful en Node.js y Express: Rutas, middlewares y manejo de errores. Uso de Spring Boot: Configuración, controladores y servicios, junto con la Inyección de Dependencias (DI).

Semana 5	Comunicación entre servicios: Implementación de microservicios y API Gateway. Integración de bases de datos: Conexión y manipulación de datos en MongoDB y PostgreSQL.
Semana 6	Realización de Trabajos Prácticos
Semana 7	Uso de tableros de gestión (Jira, Trello): Seguimiento y control de tareas en sprints. Metodologías ágiles: Aplicación de roles, ceremonias y métricas de calidad. Establecimiento de OKRs: Definición y seguimiento de objetivos claros. Prácticas de colaboración: Implementación de Pull Requests y revisiones de código efectivas.
Semana 8	Prácticas pull request
Semana 9	Testing en backend: Pruebas unitarias y de integración utilizando Jest y JUnit. Automatización de pruebas de API: Configuración y ejecución de pruebas con Postman
Semana 10	Testing end-to-end. Herramientas de análisis: Evaluación de cobertura de código y calidad.
Semana 11	Introducción a los contenedores y su rol en el testing: Conceptos básicos de contenedores.Ventajas de los entornos aislados en el desarrollo y testing. Comparación con entornos tradicionales. Creación de entornos aislados para pruebas automatizadas. Configuración de entornos para frontend y backend. Definición de dependencias específicas para uso de imágenes base optimizadas para testing.
Semana 12	Realización de Trabajos Prácticos
Semana 13	Principios de refactorización: Identificación de code smells y aplicación de técnicas de mejora. Mejores prácticas de refactorización en frontend y backend: Ajustes en React y Node/Spring Boot. Evaluación y priorización de deuda técnica: Métodos para gestionar y reducir la deuda en proyectos.
Semana 14	Realización de Trabajos Prácticos.
Semana 15	Realización de Trabajos Prácticos.
Semana 16	Realización de Trabajos Prácticos.

15. Recursos necesarios

- Recopilación de apuntes de la Cátedra
- Revistas y publicaciones científicas específicas de Programación Avanzada
- Ejemplos de código fuentes de los diferentes temas tratados en la cátedra

16. Referencias bibliográficas (citadas según Normas APA)

Bibliografía Obligatoria

- **Documentación Oficial de Spring Boot (En español)** Disponible en Spring.io
- *Aprende React: Desarrollo de interfaces con JavaScript.* **Santana Roldán, Carlos** Marcombo, 2020.
- *React.js: La Guía Completa, Incluye Hooks y Redux.* **De la Torre Valdez, Juan Pablo** Independently Published, 2021.
- *Spring Boot: Desarrollo rápido de aplicaciones Java empresariales.* **Fernández, Luis Miguel** Independently Published, 2021.

17. Función Docencia

- Dictado de las clases
- Elaboración de plan de trabajos de situaciones prácticas
- Actualización de material bibliográfico
- Aprobación de instrumentos de evaluación.
- Dictado de las clases colaborativas en resolución de problemas.

18. Reuniones de asignatura y área

Reuniones periódicas de asignatura y se prevén dos reuniones de área en el año.

19. Atención y orientación a las y los estudiantes

Se establecerán canales y horarios para la atención virtual, dependiendo de las necesidades de los estudiantes.