Desarrollo de Software

Planificación Ciclo lectivo 2026

1. Datos administrativos de la asignatura			
Departamento:	Sistemas de Información	Carrera	Ing. en Sistemas de Información
Asignatura:	**Desarrollo de Software**		
Nivel de la carrera	3er Año	Duración	Cuatrimestral
Plan	2023 Ord. 1877	Modalidad	Presencial
Bloque Curricular:	Tecnologías Aplicadas		
Área	Desarrollo de Software	RTF	8
Horas cátedra semanal:	4	Horas reloj anual total:	96
Carga horaria no presencial semanal	----------	% horas no presenciales	----------
Profesor/es Titular/Asociado/Adjunto:	Ing. Matías Cassani	Dedicación:	Exclusiva
Auxiliar/es de 1°/JTP		Dedicación:	
Competencias	Específicas	Genéricas	
	CE1.3 - CE2.1 - CE4.1 - CE5.1	CG2 - CG5 - CG10	

2. Presentación, Fundamentación

Relación de la asignatura con el perfil de egreso.

La asignatura contribuye a la interpretación y resolución de problemas vinculados a implementación de tecnologías de procesamiento de la información en lo que respecta a los lenguajes de programación.

Relación de la asignatura con los alcances del título.

Con el aprendizaje el alumno obtendrá los conocimientos para poder identificar y discernir entre diferentes herramientas y técnicas de programación de acuerdo con necesidades de implementación propuestos.

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.	2
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.	1
Competencias genéricas sociales, políticas y actitudinales (CG):	
CG. 6. Desempeñarse de manera efectiva en equipos de trabajo.	No Tributa
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	1
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	No Tributa
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.	2
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.	2
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.	2
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.	2
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

Proporcionar a los estudiantes los conocimientos, habilidades y herramientas esenciales para desarrollar software de manera efectiva y segura, centrándose en aspectos relevantes como lo son las arquitecturas y herramientas, el desarrollo de interfaces de usuario, aplicaciones orientadas a servicios, el desarrollo seguro y la realización de pruebas unitarias. Brindar a los estudiantes pautas que le permitan desarrollar software de calidad cumpliendo las mejores prácticas para enfrentar los desafíos diarios en la industria fomentando la creación de soluciones tecnológicas seguras y eficientes.

5. Objetivos establecidos en el Diseño Curricular

- Conocer las arquitecturas, herramientas y patrones para el desarrollo de software.
- Desarrollar interfaces de usuario.
- Crear soluciones de software que den respuestas a necesidades reales.
- Aplicar buenas prácticas y tecnologías en el desarrollo seguro.

6. Contenidos Mínimos establecidos en el Diseño Curricular

- Arquitectura de aplicaciones multicapa.
- Herramientas de soporte al proceso de desarrollo.
- Programación de la interfaz de usuario de una aplicación.
- Aplicaciones orientadas a servicios.
- Desarrollo Seguro.
- Pruebas unitarias.

7. Resultados de aprendizaje

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

Identificador del RA	Redacción
RA 1	[Justifica] la elección de una [arquitectura multicapa] aplicando [ventajas y desventajas] según el modelo [para resolver problemas en un escenario específico]
RA 2	[Utiliza] las herramientas de [control de versiones] aplicando [complejidad según escenario analizado] [para gestionar eficientemente el ciclo de vida del software.]

RA 3	[Diseña] [interfaces de usuario interactivas] utilizando [tecnologías web estándar] [para validar y verificar requerimientos de software.]
RA 4	[Identifica] posibles [vulnerabilidades en el proceso de desarrollo] aplicando [prácticas de desarrollo seguro en la creación de software] [para desarrollar planes de contingencias].
RA 5	[Diseña] [pruebas unitarias] efectivas utilizando [frameworks específicos] para [garantizar la calidad y funcionalidad del código.]

8. Asignaturas correlativas previas

Para cursar debe tener cursada:

- Paradigmas de Programación
- Análisis de Sistemas de Información

Para cursar debe tener aprobada:

- Ninguna

Para rendir debe tener aprobada:

- Lógica y Estructuras Discretas
- Algoritmos y Estructuras de Datos

9. Asignaturas correlativas posteriores

- Ingeniería y Calidad de Software
- Seguridad en los Sistemas de Información
- Proyecto Final

10. Programa analítico, Contenidos temáticos

Ejes temáticos	Contenidos
Introducción a TypeScript y Nest.js	**1.1 Introducción a TypeScript** Introducción a TypeScript y su relación con JavaScript. Tipos de datos, variables y funciones en TypeScript. Ventajas de TypeScript en comparación con JavaScript. **1.2 Fundamentos de Nest.js** Presentación de Nest.js como framework para el desarrollo de Backend. Estructura de directorios y convenciones en Nest.js. Creación de controladores y servicios en Nest.js.

Desarrollo de Backend con Nest.js	**2.1 Arquitectura Multicapa con Nest.js** Implementación de arquitecturas multicapa utilizando Nest.js. Uso de decoradores y módulos para organizar el código. Interacción con bases de datos en Nest.js. **2.2 Desarrollo Seguro en el Backend** Principios de desarrollo seguro aplicados al Backend. Validación de datos y prevención de vulnerabilidades comunes. Manejo seguro de contraseñas y autenticación.
Arquitecturas Multicapa	**3.1 Conceptos Fundamentales de Arquitecturas Multicapa** Definición y ventajas de las arquitecturas multicapa. Capas típicas en una arquitectura multicapa. Escenarios de aplicación de arquitecturas multicapa. **3.2 Implementación de Arquitecturas Multicapa en Nest.js** Adaptación de Nest.js para implementar una arquitectura multicapa. Comunicación entre capas en un entorno Backend. Consideraciones de escalabilidad y rendimiento.
Introducción a Angular	**4.1 Fundamentos de Angular** Presentación de Angular y su arquitectura. Creación de componentes, módulos y servicios en Angular. Enlace de datos y manipulación del DOM en Angular. **4.2 Desarrollo Frontend con Angular** Integración de Angular con servicios Backend (Nest.js). Enrutamiento en Angular para construir aplicaciones de una sola página (SPA). Implementación de formularios y manejo de eventos en Angular.
Arquitecturas Orientadas a Servicios	**5.1 Principios de Arquitecturas Orientadas a Servicios (SOA)** Definición y ventajas de SOA. Servicios web como componentes de una arquitectura orientada a servicios. Casos de uso y beneficios en entornos empresariales. **5.2 Implementación de SOA en Nest.js** Uso de servicios web (REST, SOAP) en Nest.js. Creación y consumo de servicios en un contexto Backend. Integración de servicios en una arquitectura multicapa.
Pruebas con Jest en TypeScript	**6.1 Introducción a Jest** Conceptos básicos de pruebas unitarias con Jest.

	Configuración de Jest para proyectos TypeScript. Creación y ejecución de pruebas unitarias. **6.2 Pruebas en el Desarrollo de Bachead y Frontend** Aplicación de pruebas en el código del Backend (Nest.js). Integración de Jest en proyectos de Frontend (Angular). Cobertura de código y generación de informes de pruebas.
Arquitecturas de Microservicios	**7.1 Principios de Arquitecturas de Microservicios** Definición y ventajas de las arquitecturas de microservicios. Comparación con arquitecturas monolíticas y orientadas a servicios. Desafíos y beneficios en la implementación de microservicios. **7.2 Implementación de Microservicios en Nest.js y Angular** Diseño y desarrollo de microservicios utilizando Nest.js. Comunicación entre microservicios. Integración de microservicios en el Frontend con Angular.
Desarrollo Seguro en Angular	**8.1 Conceptos de Desarrollo Seguro en el Frontend** Principios de desarrollo seguro aplicados al Frontend. Prevención de vulnerabilidades en aplicaciones Angular. Consideraciones de seguridad en la comunicación con el Backend.
Gestión de Configuración	**9.1 Principios de Gestión de Configuración** Definición de gestión de configuración y su importancia. Herramientas y prácticas para la gestión de configuración en proyectos TypeScript. Control de versiones y ramificación en entornos de desarrollo. **9.2 Integración de Gestión de Configuración en el Desarrollo** Integración de sistemas de control de versiones (Git) en proyectos TypeScript. Implementación de flujos de trabajo y buenas prácticas en la gestión de configuración. Resolución de conflictos y colaboración en equipo.
Integración Continua y Entrega Continua/Implementación Continua (CI/CD)	**10.1 Principios de CI/CD** Definición de Integración Continua (CI) y Entrega Continua/Implementación Continua (CD/CI). Beneficios y desafíos en la implementación de CI/CD. Herramientas y prácticas asociadas a CI/CD en proyectos TypeScript. **10.2 Configuración de Pipelines de CI/CD** Configuración de pipelines de CI/CD para proyectos TypeScript.

	Automatización de pruebas, construcción y despliegue. Monitoreo y gestión de versiones en un entorno CI/CD. **10.2 Implementación de CI/CD en Proyectos con Nest.js y Angular** Integración de CI/CD en el desarrollo Backend con Nest.js. Despliegue automático de aplicaciones Angular en entornos de producción. Estrategias de rollbacks y manejo de versiones en CI/CD. **10.3 Evaluación y Mejora Continua en CI/CD** Monitoreo y evaluación de los procesos de CI/CD. Mejoras continuas basadas en métricas y retroalimentación. Implementación de estrategias de despliegue canario y blue-green. **10.4 Presentación y Evaluación del Proyecto Final con CI/CD** Integración de CI/CD en los proyectos finales presentados por los estudiantes. Evaluación específica de la implementación de CI/CD en los proyectos. Retroalimentación final y discusión sobre las experiencias de CI/CD.

Cuadro de Carga Horaria Práctica

Tipo 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 hs	
Proyecto Integrador		
Práctica profesional supervisada		
Carga horaria total	32 hs	

11. Metodología de enseñanza

Clases Teóricas: se lleva a cabo la presentación y el desarrollo de los temas teóricos.
Actividades prácticas en aula: destinadas al análisis y resolución de escenarios propuestos e implementados.

Actividades prácticas en laboratorio: que comprenden la resolución de problemas de ingeniería y formación experimental, y en las que se abordan e integran los ejes temáticos.
Consultas: Se trabaja con consultas en diferentes momentos, antes de la clase, en las actividades prácticas, clases previas a los parciales y/o exámenes.

Durante el desarrollo de los Trabajos Prácticos se contará con la presencia permanente del docente de la cátedra para el apoyo a los alumnos y de los becarios que el responsable del Laboratorio designe para asegurar el funcionamiento del equipamiento.

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

COMPETENCIA - NIVEL DE TRIBUTACIÓN	RESULTADO DE APRENDIZAJE (RA)
CE1.3(2) - CE2.1(2) - CE4.1(2) - CE5.1(2) - CG2(2) - CG5(1) - CG10(1)	RA1 - RA2 - RA3- RA4 - RA5

EJES TEMÁTICOS - CONTENIDOS

Introducción a TypeScript y Nest.js - Desarrollo de Backend con Nest.js - Arquitecturas Multicapa - Introducción a Angular - Arquitecturas Orientadas a Servicios - Pruebas con Jest en TypeScript - Arquitecturas de Microservicios - Desarrollo Seguro en Angular - Gestión de Configuración - Integración Continua y Entrega Continua/Implementación Continua (CI/CD)

ACTIVIDADES DEL ESTUDIANTE (FORMATIVAS)

El **estudiante** deberá realizar las siguientes actividades para alcanzar los **RA** propuestos:

- Lectura y análisis de material bibliográfico.
- Participar activamente de la clase.
- Resolver las actividades conceptuales y prácticas en clase. s.
- Utilizar software en el laboratorio, a efectos de adquirir la solvencia básica necesaria, en la resolución de los problemas propuestos en la cátedra.
- Realizar Trabajos Prácticos Grupales.
- Realizar cuestionarios de autoevaluación.
- Realizar Instancia de Evaluación Individual

ESTRATEGIAS DE ENSEÑANZA

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

- Clase expositiva dialogada
- Taller / Seminario.

- Acompañamiento continuo.
- Diseño de problemas/ejercicios.
- Diseño Instancia de Evaluación.
- Retroalimentación formativa.
- Coordinación de trabajo colaborativo.

12. Recomendaciones para el estudio

Lectura del material bibliográfico con anticipación. Utilización de trabajos prácticos semanales como instancias de práctica y aprendizaje.

13. Metodología de evaluación

EVALUACIÓN "La evaluación de los alumnos debe ser congruente con los objetivos y metodologías de enseñanza previamente establecidos. Las evaluaciones deben contemplar de manera integrada la adquisición de conocimientos, la formación de actitudes, el desarrollo de la capacidad de análisis, habilidades para encontrar la información y para resolver problemas reales". (estándar II.14 anexo IV – RM 1232).

Durante el cursado de la asignatura se implementará la evaluación formativa, llevando el registro y control de los trabajos prácticos realizados por los alumnos durante las clases prácticas. Esto permite conocer de alguna manera, el proceso de aprendizaje por parte del alumno y su participación en la asignatura.

Escala de Notas (Evaluaciones Parciales y Final)

Notas	**Porcentajes**	
1	0-10%	**Insuficiente**
2	11-20%	
3	21-30%	
4	31-40%	
5	41-59%	
6	60-69%	**Aprobado**
7	70-79%	**Bueno**
8	80-89%	**Muy Bueno**
9	90-95%	**Distinguido**
10	96-100%	**Sobresaliente**

Para APROBAR LA CURSADA el alumno deberá: tener el 75% de asistencia y aprobar todos los prácticos realizados además deberá aprobar los 3 exámenes parciales que se contemplan en

la presente planificación. Se contempla además la existencia de un parcial recuperatorio que integrará todos los contenidos evaluados en los parciales anteriores.

Para la PROMOCIÓN DIRECTA el alumno deberá: tener el 75% de asistencia y aprobar todos los prácticos realizados además deberá aprobar los 3 exámenes parciales que se contemplan en la presente planificación con nota mayor o igual a 8 (>= 8) en los 3 parciales. Se contempla además la existencia de un parcial recuperatorio que integrará todos los contenidos evaluados en los parciales anteriores. Para poder obtener la promoción directa el alumno que deba asistir a la instancia de recuperación deberá obtener una nota (>= 8) mayor o igual a 8.

EXAMEN FINAL

Constará de una evaluación teórica práctica ante un tribunal que incluye los contenidos dictados por el docente durante el ciclo lectivo. Aquí se evaluarán los conceptos estudiados durante el cursado de la materia. Inicial o diagnóstica, formativa o continua, sumativa o final (Directa o con examen final. Indicar modalidad de examen: oral, escrito, mixto). Tener en cuenta la Ordenanza 908: Reglamento de Estudio, en particular Ptos. 8-1 y 8-3.

14. Cronograma de clases año 2026

Semana	Contenido / Actividad
Semana 1	Introducción a TypeScript.Fundamentos de Nest.js.
Semana 2	Fundamentos de Nest.js.Desarrollo Seguro en el Backend.
Semana 3	Conceptos Fundamentales. Implementación en Nest.j.
Semana 4	Fundamentos de Angular. Desarrollo Frontend con Angular
Semana 5	Principios de SOA. Implementación en Nest.js
Semana 6	Introducción a Jest. Pruebas en Backend y Frontend
Semana 7	Principios y comparación. Implementación en Nest.js y Angular
Semana 8	Conceptos de Desarrollo Seguro. Principios y Herramientas
Semana 9	Integración en Proyectos Type Script. Principios y Beneficios
Semana 10	Configuración de Pipelines. Integración en Backend y Frontend
Semana 11	Estrategias de Despliegue. Monitoreo y Mejoras Continuas
Semana 12	Implementación de Estrategias. Integración de CI/CD en Proyectos

Semana 13	Desarrollo de Proyecto Integrador.
Semana 14	Desarrollo de Proyecto Integrador.
Semana 15	Presentación y Evaluación del Proyecto Final.
Semana 16	Presentación y Evaluación del Proyecto Final.

15. Recursos necesarios

Ámbito de realización: Aula o laboratorio de computación.

Disponibilidad de infraestructura y equipamiento. Los espacios físicos y equipamiento están disponibles para el desarrollo de las actividades prácticas. Eventualmente, los alumnos pueden realizar las actividades prácticas a través del uso de máquinas virtuales en sus notebooks personales dentro del aula sin necesidad de la utilización de laboratorios.

Actividades para desarrollar: Desarrollo de trabajos prácticos planteados por el docente

Tiempo: Durante el dictado de las clases prácticas. (carga horaria, período que abarca).

Evaluación (de seguimiento y final): Evaluación de seguimiento por medio de entregas pautadas cuya aprobación es requerida para la regularización de la materia.

16. Referencias bibliográficas (citadas según Normas APA)

Bibliografía Obligatoria

- Design Thinking en Español - https://www.designthinking.es/inicio/index.php
- Sitio Oficial Nodejs – https://nodejs.org/es • Sitio Oficial de Json – https://www.json.org/json-en.html
- gitbook- Creación de una API – https://juanda.gitbooks.io/webapps/content/api/arquitectura-api-rest.htm
- OWASP - https://owasp.org/www-project-top-ten/
- Serrano, Carlos (2020), TypeScript. Curso práctico-1era -RA-MA Editorial.

17. Función Docencia

Distribución de tareas del equipo docente:

Dictado de las clases

Elaboración de plan de trabajos de situaciones prácticas

Disponibilidad en tiempo y forma de recursos materiales (infraestructura y equipamiento).

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

Horario de consulta: Los alumnos podrán realizar las consultas sobre los contenidos de la materia ya sean estos teóricos o prácticos en el horario de clases. Además, se establece como horario de consulta los lunes de 16 a 18 hs. en la facultad.