Ministerio de Cultura y Educación **(Programa del año 2026)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area IV: Pr. y Met. de Des. del Soft.

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
() OP.SEGURIDAD DE SISTEMAS DE SOFTWARE	ING. EN COMPUT.	28/12	2026	2° cuatrimestre
() OP.SEGURIDAD DE SISTEMAS DE SOFTWARE	ING. INFORM.	026/12-08/15	2026	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
BERON, MARIO MARCELO	Prof. Responsable	P.Adj Exc	40 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	2 Hs	Hs	3 Hs	5 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	2º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
03/08/2026	13/11/2026	15	75

IV - Fundamentación

La seguridad de los sistemas informáticos constituye un pilar esencial en la formación de un ingeniero en informática, ya que de ella depende la confiabilidad y continuidad de los procesos tecnológicos que sostienen a las organizaciones modernas. En un entorno global cada vez más digitalizado, los sistemas informáticos administran información crítica: datos personales de millones de usuarios, operaciones financieras, procesos industriales, servicios de salud y hasta infraestructuras estratégicas de los estados. La ausencia de medidas de seguridad adecuadas expone a estos sistemas a vulnerabilidades que pueden ser explotadas por atacantes, generando pérdidas económicas, daños reputacionales y, en casos extremos, afectaciones sociales de gran escala.

Para los estudiantes de ingeniería, comprender la importancia de la seguridad informática significa reconocer que no basta con diseñar sistemas funcionales; es imprescindible que sean seguros, resilientes y confiables. La seguridad se traduce en la aplicación de principios como la confidencialidad, integridad y disponibilidad de la información, que garantizan que los datos estén protegidos contra accesos no autorizados, modificaciones indebidas y caídas del servicio. Estos principios no son abstractos: se materializan en prácticas concretas como el uso de criptografía, la gestión de identidades y accesos, la implementación de firewalls, auditorías periódicas y pruebas de penetración, análisis de programa, entre otras posbilidades.

Además, la seguridad informática es un campo en constante evolución. Las amenazas cambian y se sofistican día a día, lo que exige que los futuros ingenieros desarrollen una mentalidad crítica y proactiva. No se trata solo de reaccionar ante incidentes,

sino de anticipar riesgos mediante el análisis de vulnerabilidades, el modelado de amenazas y la adopción de metodologías de desarrollo seguro como DevSecOps o el Secure SDLC, entre otras. De esta manera, los profesionales pueden integrar la seguridad desde las primeras etapas del diseño, evitando que se convierta en un “parche” tardío y costoso.
Otro aspecto relevante es el cumplimiento normativo. Diversas legislaciones y estándares internacionales —como la Ley de Protección de Datos Personales en Argentina, el Reglamento General PD en Europa o las normas ISO/IEC— obligan a las organizaciones a garantizar la seguridad de la información. Un ingeniero que domina estos marcos regulatorios no solo protege a los usuarios, sino que también asegura la competitividad y legitimidad de la empresa en el mercado.
En definitiva, la seguridad de los sistemas informáticos no es un tema accesorio, sino un requisito indispensable para el ejercicio profesional. Para los estudiantes de Ingeniería en Informática, representa la oportunidad de adquirir competencias que los preparen para enfrentar los desafíos reales de la industria, aportando soluciones tecnológicas que sean no solo innovadoras, sino también seguras y confiables. Un sistema sin seguridad es una puerta abierta al riesgo; un sistema seguro, en cambio, es la base sobre la cual se construye la confianza y el progreso tecnológico.

V - Objetivos / Resultados de Aprendizaje

Al finalizar el curso se espera que el estudiante sea capaz de:
*Comprender los fundamentos de la seguridad informática y su importancia en el ciclo de vida del software.
*Desarrollar competencias para aplicar metodologías, estándares y buenas prácticas de seguridad en cada etapa del desarrollo.
*Formar profesionales capaces de identificar, analizar y mitigar riesgos de seguridad en sistemas y aplicaciones.
*Promover una cultura de responsabilidad, aprendizaje continuo y adaptación a nuevas tecnologías y amenazas.

Durante el dictado de la asignatura se abordan los siguientes ejes transversales:
1. Proyecto y dirección en lo referido a seguridad informática.
2. Establecimiento de métricas y normas de calidad de software.
3. Procedimientos y certificaciones del funcionamiento, condición de uso o estado de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software.
4. Dirección y control de la implementación, operación y mantenimiento de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software.
5. Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.
6. Concepción, diseño y desarrollo de proyectos de ingeniería en sistemas de información / informática.
7. Gestión, planificación, ejecución y control de proyectos de ingeniería en sistemas de información / informática.
8. Generación de desarrollos tecnológicos y/o innovaciones tecnológicas.
9. Desempeño en equipos de trabajo.
10. Comunicación efectiva.
11. Actuación profesional ética y responsable.
12. Evaluación y actuación en relación con el impacto social de su actividad profesional en el contexto global y local.
13. Aprendizaje continuo.

VI - Contenidos

Contenidos Mínimos

Vulnerabilidad. Ataque. Defensa. Distintos Roles del Equipo de Desarrollo en el Contexto de Seguridad. Metodologías de Desarrollo Seguro. Codificación Segura. Estándares de Codificación Segura. Desarrollo de Software Seguro. Protección de Software. Análisis de Programa. Herramientas de Generación Automática de los Componentes de una Herramienta de Protección. Diseño y Construcción de Herramientas de Verificación de la Seguridad de los Sistemas Informáticos. Seguridad de la Información.

Unidad I: Seguridad de los Sistemas Informáticos

Introducción. Conceptos Básicos de Seguridad: Vulnerabilidad, Ataque, Defensa, Roles en el Equipo de Desarrollo. Metodologías de Desarrollo Seguro de Software. SDLC Seguro (SSDLC). DevSecOps. Modelos de Madurez: OWASP SAMM (Software Assurance Maturity Model), NIST SSDF (Secure Software Development Framework). Seguridad en Metodologías Ágiles: Scrum/Kanban + seguridad, Agile Security. Análisis de Amenazas y Modelado (TAM). Técnicas: STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege), MITRE ATT&CK. Normativas y Estándares Internacionales: ISO/IEC 27002:2022 (Seguridad de la Información), ISO/IEC 29148

(Ingeniería de Requerimientos), entre otros.

Unidad II: Seguridad en la Etapa de Captura de Requisitos
Ingeniería de Requerimientos. Concepción. Objetivos. Principales Actividades. Requerimientos Funcionales. Requerimientos No Funcionales. Requerimiento de Dominio. Interesados. Requisitos de Seguridad. Programación y Planificación del Proyecto. Requerimientos de Seguridad. Requerimientos Seguridad de Aplicaciones Web. Requerimientos de Seguridad de APIs. Requerimentos de Seguridad de Base de Datos.

Unidad III: Seguridad en la Etapa de Diseño
Introducción. Modelación del Peligro. Patrones y Conceptos de Diseño Seguro. Arquitecturas Seguras. Diseño de Controles de Seguridad. Análisis de Flujo de Datos. Metodologías Ágiles: Seguridad de Historias de Usuario.

Unidad IV: Seguridad en la Etapa de Codificación
Introducción. Entrenamiento. Organizaciones. Individuos. Codificación Segura. Revisiones de Código. Herramientas de Análisis Estático de Primer y Segunda Generación. Barandillas de seguridad. Complementos IDE y otras guías. Verificación de la seguridad de sus dependencias (SCA). Selección de las Dependencias que se deben Actualizar o Modificar. Cómo encontrar y gestionar secretos. Testing Dinámico (DAST). Estándares de Codificación Segura. Protección de Software. Estrategias de Protección de Software. ANTLR4: Gramáticas de Atributos. Problemas en las Gramáticas. Lexer, Parser, Atributos Heredados, Sintetizados, Oyentes, Visitantes, Construcción de Herramientas Orientadas a la Detección de Vulnerabilidades y Protección de Software.

Unidad V: Seguridad en la Etapa de Pruebas
Introducción. Cobertura y Sincronización de las Pruebas. Profundidad vs Cobertura. Escaneo de Infraestructura. Entornos de producción o de nivel inferior. Alcance. Sincronización. Pruebas Manuales. Pruebas Automatizadas. Fuzzing. Pruebas interactivas de seguridad de aplicaciones (IAST). Programas de recompensas por detección de errores. Resultados de las Pruebas.

Unidad VI: Seguridad en la Etapa de Lanzamiento / Despliegue
Eventos de seguridad dentro del CI/CD. Desmontado de la construcción. Escaneado de Secretos. Análisis Estático. Análisis Dinámico. Análisis de Composición de Software. Linting. Infraestructuras como Escaners de Código. Seguridad del pipeline de CI/CD. Integridad del Lanzamiento. Aprobación de la divulgación de seguridad.

Unidad VII: Seguridad en la Etapa de Mantenimiento
Monitoreo, alerta y observabilidad. Bloqueo/Protección. Cortafuegos de aplicaciones web (WAF). Redes de distribución de contenido (CDN). Autoprotección de aplicaciones en tiempo de ejecución (RASP). Parcheo virtual. Puertas de enlace API. Pruebas Continuas. Incidentes de Seguridad. Planificación de la continuidad del negocio y la recuperación ante desastres.

Unidad VIII: Consideraciones Finales
Buenos Hábitos. Responsabilidad. Uso seguro de la inteligencia artificial en cada una de las Etapas del Proceso de Desarrollo de Software. Aprendizaje Continuo. Transición al equipo de seguridad. Cómo solicitar puestos de trabajo en seguridad fuera de su organización.

VII - Plan de Trabajos Prácticos

Práctico Nro. 1: Seguridad de los Sistemas Informáticos
Objetivo: en este práctico se pretende que el estudiante:
*Comprender los conceptos básicos de seguridad en software y su importancia en el ciclo de vida del desarrollo.
*Reconocer vulnerabilidades, ataques y defensas, así como los roles clave dentro de un equipo de desarrollo seguro.
*Introducir metodologías, marcos y estándares internacionales que guían la práctica profesional en seguridad de software.

El práctico incluye ejercicios donde se pretende que el estudiante:
Construya ontologías donde queden explícitamente identificados y conceptualizados los diferentes roles de un equipo de desarrollo seguro de software y discutan sus responsabilidades. Así mismo se presentarán casos simples de vulnerabilidades

de forma tal que la puedan identificar y subsanar.

Práctico Nro. 2: Seguridad en la Etapa Captura de Requisitos
Objetivo:
- Comprender el proceso de ingeniería de requerimientos y su rol en la concepción y planificación de proyectos de software.
- Diferenciar y redactar requerimientos funcionales, no funcionales y de dominio, integrando la perspectiva de los interesados.
- Incorporar requisitos de seguridad como parte esencial del ciclo de vida del software, aplicados a aplicaciones web, APIs y bases de datos.

El práctico incluye ejercicios donde se pretende que el estudiante:
A partir de un ejemplo de aplicación (en dominios web, base de datos, APIs), redacten los objetivos generales y específicos incluyendo objetivos de seguridad. Se identifiquen los interesados, sus roles y expectativas. Se lleve adelante procesos sencillos de elicitación y validación de requisitos.

Práctico Nro. 3: Seguridad en la Etapa de Diseño
Objetivo:
- Comprender la importancia de incorporar la seguridad desde la etapa de diseño del software.
- Conocer técnicas y patrones de diseño seguro para anticipar y mitigar vulnerabilidades.
- Integrar la seguridad en metodologías ágiles mediante historias de usuario y análisis de flujo de datos.

El práctico incluye ejercicios donde se pretende que el estudiante:
El práctico incluye ejercicios donde se pretende que el estudiante modele amenazas en arquitecturas simples aplicando las técnicas destinadas para tal fin, analice y corrija diseños inseguros, redacte requisitos de seguridad para controles específicos como autenticación, autorización y cifrado, represente flujos de datos identificando puntos críticos de exposición, e integre requisitos de seguridad en historias de usuario dentro de metodologías ágiles. De esta manera, se busca que el alumno desarrolle competencias aplicadas en el diseño de soluciones seguras y resilientes desde la etapa inicial del desarrollo de software.

Práctico Nro. 4: Seguridad en la Etapa de Codificación
Objetivo:
- Comprender la importancia de aplicar prácticas de codificación segura en el desarrollo de software.
- Incorporar herramientas, estándares y estrategias que permitan detectar y mitigar vulnerabilidades durante la implementación.
- Desarrollar competencias para proteger el software y gestionar de forma segura sus dependencias y secretos.

El práctico incluye ejercicios donde se pretende que el estudiante: Aplique principios de codificación segura en fragmentos de código, realice revisiones de código utilizando checklist de seguridad y herramientas de análisis estático de primera y segunda generación, configure complementos IDE para prevenir errores comunes, verifique la seguridad de dependencias mediante análisis de composición de software (SCA) y decida cuáles deben actualizarse o reemplazarse, identifique y gestione secretos como claves y credenciales de manera adecuada, ejecute pruebas dinámicas (DAST) sobre aplicaciones en ejecución para detectar vulnerabilidades, y redacte requisitos alineados con estándares de codificación segura. Asimismo, se busca que el alumno explore estrategias de protección de software frente a ataques de manipulación o ingeniería inversa, integrando buenas prácticas y herramientas en su entorno de desarrollo cotidiano. Implemente herramientas sencillas de análisis estático/dinámico utilizando tecnologías de compilación.

Esta unidad consta de los siguientes laboratorios:

Laboratorio 1: Instalación y utilización con ejemplos sencillos de herramientas que generen analizadores lexicográficos y sintácticos.

Laboratorio 2: Implementación y ejecución de reconocedores de lenguajes de programación de la vida real: Java, Python, C, entre otros.

Laboratorio 3: Extracción de la información utilizando atributos heredados y sintetizados.

Laboratorio 4: Extracción de la información utilizando oyentes/ visitantes.

Laboratorio 5: Detección de fallas en la codificación.

Laboratorio 6: Implementación de una técnica de protección. Desarrollo de un sistema de detección de vulnerabilidades/protección que se base en una metodología de desarrollo de software seguro.

Práctico Nro. 5: Seguridad en la Etapa de Pruebas
Objetivo:

- Comprender la importancia de la etapa de pruebas en la detección y mitigación de vulnerabilidades de seguridad.
- Aplicar técnicas manuales y automatizadas para evaluar la robustez del software frente a amenazas.
- Integrar herramientas y metodologías de prueba que permitan garantizar la calidad y seguridad en entornos de producción y preproducción.

El práctico incluye ejercicios donde se pretende que el estudiante: planifique y ejecute pruebas de seguridad considerando la cobertura y sincronización en distintos entornos, realice escaneos de infraestructura para detectar configuraciones inseguras, y compare la profundidad frente a la cobertura de las pruebas. Se busca que el alumno lleve a cabo tanto pruebas manuales como automatizadas para descubrir vulnerabilidades y errores en aplicaciones. Además, se propone la simulación de un programa de recompensas por detección de errores (bug bounty) para comprender su valor en la seguridad colaborativa. Finalmente, el estudiante deberá interpretar los resultados de las pruebas, incluyendo pruebas de actuación y desempeño bajo condiciones adversas, con el fin de generar conclusiones que fortalezcan la seguridad del software en escenarios reales.

Práctico Nro. 6: Seguridad en la Etapa de Lanzamiento / Despliegue
Objetivo:

- Comprender la importancia de garantizar la seguridad en la etapa de lanzamiento y despliegue de software.
- Aplicar controles y herramientas que aseguren la integridad del pipeline de CI/CD y del producto final.
- Desarrollar competencias para gestionar secretos, dependencias y divulgación de seguridad en entornos de despliegue.

El práctico incluye ejercicios donde se pretende que el estudiante: configure un pipeline de CI/CD con controles básicos de seguridad, practique el escaneo de secretos y la verificación de dependencias, y utilice herramientas de análisis estático y dinámico para detectar vulnerabilidades/problemas de codificación/errores en el código antes del despliegue. También se busca que aplique técnicas de linting y escaneo de composición de software, garantice la seguridad del propio pipeline mediante controles de acceso y auditorías, y compruebe la integridad del lanzamiento. Finalmente, el alumno deberá simular un proceso sencillo de divulgación de seguridad, aprendiendo a gestionar reportes de vulnerabilidades de manera responsable.

Práctico Nro. 7: Seguridad en la Etapa de Mantenimiento
Objetivo:

- Comprender la importancia del mantenimiento continuo para garantizar la seguridad del software en producción.
- Aplicar herramientas y técnicas de monitoreo, protección y respuesta ante incidentes.
- Desarrollar competencias para asegurar la continuidad del negocio y la recuperación ante desastres.

El práctico incluye ejercicios donde se pretende que el estudiante: Discuta la configure sistemas básicos de monitoreo y alertas, utilice herramientas de protección, y practique técnicas de autoprotección en tiempo de ejecución. También se busca que realice pruebas continuas para validar la seguridad en producción. Finalmente, el alumno deberá analizar incidentes de seguridad y elaborar un plan sencillo de continuidad del negocio y recuperación ante desastres, asegurando la resiliencia del sistema en la etapa de mantenimiento.

Práctico Nro. 8: Consideraciones Finales
Objetivo:

- Reflexionar sobre los contenidos vistos en la materia.
- Analizar las ventajas/desventajas que presenta el uso de la inteligencia artificial en cada una de las etapas del proceso de desarrollo de software.

El práctico incluye ejercicios donde se pretende que el estudiante: Analice las ventajas de utilizar la IA para resolver/detectar problemas de seguridad de los sistemas informáticos. Además se valorará la discusión de posibles implementaciones de aquellos enfoques y de la factibilidad de la realización de las mismas.
Práctico Final Integrador
Durante el cursado de la materia se realizará un Práctico Final Integrador el cual consistirá en la implementación de una herramienta que permita detectar problemas de seguridad en los sistemas informáticos. La misma puede ser construida

combinando herramientas o utilizando tecnologías de compilación. En este práctico se valorará la calidad de la aplicación del proceso de desarrollo de seguro de software seleccionado por el estudiante.

A continuación se explica como se abordan y evalúan los ejes transversales empleados en la materia:

1. Proyecto y dirección en lo referido a seguridad informática.
    1. ¿Cómo se aborda? Mediante el análisis y desarrollo de herramientas básicas de seguridad en donde se debe aplicar el proceso de desarrollo de software seguro.
    2. ¿Cómo se evalúa? Analizando la calidad de las producciones realizadas por los estudiantes en las cuales se tiene que constatar la aplicación correcta de los conceptos vistos en la materia.

2. Establecimiento de métricas y normas de calidad de software.
    1. ¿Cómo se aborda? A través del cálculo de métricas definidas en la literatura o por el estudiante y el seguimiento de las normas de calidad de software establecidas por los organismos internacionales de estandarización.
    2. ¿Cómo se evalúa? A través de la explicación/demostración de proceso utilizado para seguir los estándares de calidad y el cálculo de métricas utilizadas para evaluar la seguridad.

3. Procedimientos y certificaciones del funcionamiento, condición de uso o estado de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software.
    1. ¿Cómo se aborda? Se plantearán proyectos/ejercicios donde los estudiantes deban aplicar procedimientos de certificación y validación del funcionamiento de sistemas de información, comunicación de datos y software. Se los guiará revisen estándares internacionales (como ISO/IEC 27001 para seguridad de la información o ISO/IEC 25010 para calidad de software) y elaboren informes que documenten el estado de los sistemas de ejemplo que se darán en la materoa, considerando tanto el cumplimiento técnico como las implicancias sociales y profesionales.
    2. ¿Cómo se evalúa? La evaluación contempla la comprensión/aplicación de procedimientos de certificación/normas y estándares, la claridad en la documentación del estado de los sistemas de ejemplo, y la reflexión crítica sobre la importancia de las certificaciones en el ejercicio profesional.

4. Dirección y control de la implementación, operación y mantenimiento de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software.
    1. ¿Cómo se aborda? Se afronta mediante proyectos donde los estudiantes asuman la la responsabilidad en cada etapa del desarrollo de sistemas de información. Se los guiará para que diseñen planes de despliegue seguro, gestionen la operación diaria con monitoreo y alertas, y definan estrategias de mantenimiento que incluyan pruebas continuas y respuesta a incidentes. La idea es que comprendan que la seguridad no termina en el desarrollo, sino que requiere liderazgo y control permanente en todo el ciclo de vida del sistema.
    2. ¿Cómo se evalúa? La evaluación contemplará la capacidad de planificar e implementar medidas de seguridad, la eficacia en la operación y monitoreo de sistemas, la pertinencia de las estrategias de mantenimiento aplicadas, y la claridad en la documentación y control de procesos. También se tendrá en cuenta la habilidad del estudiante para dirigir equipos, tomar decisiones frente a incidentes y garantizar la calidad del software y la seguridad en entornos reales, integrando aspectos técnicos y de gestión profesional.

5. Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.
    1. ¿Cómo se aborda? A través de la presentación de sistemas que violan las características de seguridad.
    2. ¿Cómo se evalúa? Analizando las soluciones propuesta y su relación con los conceptos de seguridad vistos en la materia.

6. Concepción, diseño y desarrollo de proyectos de ingeniería en sistemas de información / informática.
    1. ¿Cómo se aborda? A través del desarrollo de herramientas de análisis de código que tengan por objetivo la detección de problemas de seguridad en los sistemas de software.

2. ¿Cómo se evalúa? Chequeando que el proceso de desarrollo aplicado para la construcción de las herramientas sea seguro y que los resultados producidos por la misma sean correctos.

7. Gestión, planificación, ejecución y control de proyectos de ingeniería en sistemas de información / informática.

1. ¿Cómo se aborda? Se concretizará con proyectos donde los estudiantes deban planificar, ejecutar y controlar iniciativas de seguridad en sistemas de información. Se los puede guiar para que definan objetivos claros, asignen roles dentro del equipo, establezcan cronogramas y gestionen recursos, aplicando metodologías de gestión de proyectos adaptadas a la seguridad. Además, se promoverá que documenten riesgos, medidas de mitigación y resultados, conectando la práctica técnica con la gestión profesional.

2. ¿Cómo se evalúa? Se realizará mediante una estrategia que contemple la claridad en la planificación, la ejecución ordenada de tareas de seguridad, el control de avances y riesgos, y la calidad de la documentación entregada. También se valorará la capacidad del equipo para adaptarse a imprevistos, la pertinencia de las decisiones tomadas y la integración de buenas prácticas de gestión en el proyecto. De esta forma, se mide tanto la competencia técnica en seguridad como la habilidad de gestionar proyectos de manera profesional y responsable.

8. Generación de desarrollos tecnológicos y/o innovaciones tecnológicas.

1. ¿Cómo se aborda? Mediante la discusión de soluciones a las vulnerabilidades tradicionales y nuevas que se van mencionando en la literatura.

2. ¿Cómo se evalúa? A través del análisis de la calidad de las soluciones presentadas. En este análisis se tiene en cuenta las aplicación correcta de los conceptos vistos en la materia.

9. Desempeño en equipos de trabajo.

1. ¿Cómo se aborda? Como en todo proyecto de desarrollo de software el trabajo en equipo es fundamental. Por esta razón, los prácticos y laboratorios están pensados para que se resuelvan en grupos de dos o tres personas.

2. ¿Cómo se evalúa? La evaluación se llevará a cabo a través de la participación de los integrantes del grupo, el cumplimiento de los objetivos y plazos en cada actividad, la calidad del producto final, en la forma en que se toman las decisiones.

10. Comunicación efectiva.

1. ¿Cómo se aborda? A través de presentaciones de ejercicios y avances de los proyectos desarrollados en la materia.

2. ¿Cómo se evalúa? Teniendo en cuenta la calidad de la presentación en cuanto a estructura y contenido. Considerando la forma de expresar el contenido y resultado por parte de los integrantes del grupo para los distintos interesados.

11. Actuación profesional ética y responsable.

1. ¿Cómo se aborda? En los proyectos de desarrollo de software es fundamental incorporar la ética y la seguridad desde el inicio, asegurando el respeto por la privacidad de los usuarios, la transparencia en las decisiones técnicas y el uso responsable de los datos, al mismo tiempo que se aplican prácticas de codificación segura, gestión adecuada de dependencias y secretos, y pruebas continuas que garanticen la confiabilidad y resiliencia del sistema frente a amenazas.

2. ¿Cómo se evalúa? Se evaluará observando que el estudiante demuestre la aplicación de prácticas de codificación segura, gestión responsable de datos y dependencias, pruebas de seguridad continuas, y al mismo tiempo reflexione sobre dilemas éticos vinculados a la privacidad, la transparencia y el impacto social del software, integrando estos criterios en el trabajo en equipo y en la documentación del proyecto.

12. Evaluación y actuación en relación con el impacto social de su actividad profesional en el contexto global y local.

1. ¿Cómo se aborda? Se trabajará mediante actividades donde los estudiantes analicen el impacto social de vulnerabilidades, ataques o medidas de protección en sistemas reales. Se los invitará a reflexionar sobre cómo la seguridad afecta la confianza de los usuarios, la protección de datos sensibles, la continuidad de servicios críticos y la responsabilidad profesional frente a incidentes en ámbitos locales y globales.

2. ¿Cómo se evalúa? Se realizará realizarse con una estrategia que contemple la capacidad de identificar riesgos y sus

consecuencias sociales, la pertinencia de las soluciones de seguridad propuestas y la reflexión crítica sobre el rol del ingeniero en la protección de la información. También se valorará la integración de estas consideraciones en informes técnicos, simulaciones de incidentes y debates, asegurando que el estudiante demuestre conciencia ética y compromiso con la seguridad en su práctica profesional.

13. Aprendizaje continuo.

1. ¿Cómo se aborda? Se trabajará promoviendo que los estudiantes se mantengan actualizados frente a nuevas vulnerabilidades, herramientas y normativas. Se los incentivará a investigar incidentes recientes, explorar buenas prácticas emergentes y reflexionar sobre la necesidad de seguir aprendiendo más allá de la cursada, entendiendo que la seguridad es un campo dinámico que exige actualización permanente.

2. ¿Cómo se evalúa? Se realizará mediante actividades donde el estudiante demuestre su capacidad de búsqueda y análisis de información actualizada, por ejemplo, presentando un breve informe sobre una vulnerabilidad reciente y sus implicancias, o integrando nuevas prácticas de seguridad en sus proyectos. Se puede usar una estratégia que valore la pertinencia de la información encontrada, la capacidad de relacionarla con el contexto local y global, y la reflexión crítica sobre la importancia del aprendizaje continuo en su desarrollo profesional.

VIII - Regimen de Aprobación

1. Condiciones para REGULARIZAR la asignatura:
   1.1. Haber asistido al menos al 70% de las clases teóricas y prácticas de la asignatura.
   1.2. Haber aprobado el Trabajo Práctico Integrador (TPI) solicitado por la cátedra con nota mayor o igual a 7.
   1.3. Haber aprobado dos exámenes parciales o una de sus respectivas dos recuperaciones con nota mayor o igual a 6.

2. Condiciones para PROMOCIONAR la asignatura:
   2.1. Haber asistido al menos al 80% de las clases teóricas y prácticas de la asignatura.
   2.2. Haber aprobado el TPI solicitado por la cátedra con nota mayor o igual a 7.
   2.3. Haber aprobado dos exámenes parciales o una de sus respectivas dos recuperaciones con nota mayor o igual a 7.
   2.4. Haber aprobado una Evaluación Global Integradora (EGI) con nota mayor o igual a 7.
   2.5. En caso de obtener nota de promoción, el alumno aprobará la materia con una nota que surgirá del promedio de las notas obtenidas de cada una de las instancias de evaluación.

3. Trabajo Práctico Integrador (TPI):
   3.1. El TPI es grupal, con grupos de 3 estudiantes; excepcionalmente se aceptarán grupos de 2 o de 4 estudiantes. Al finalizar, el grupo deberá presentar un informe escrito del mismo; además, todos los miembros del grupo deberán defender el trabajo frente a sus pares en una exposición oral.

4. Recuperaciones:
   4.1. Cada parcial cuenta con dos recuperaciones, quedando la última nota obtenida. El TPI y la EGI no se recuperan.

5. Examen Final:
   5.1. En caso de regularizar la materia, el alumno deberá rendir un examen final, el cual podrá ser oral o escrito, en cualquiera de los turnos de examen establecidos en el Calendario Académico. En dicho examen se evalúan conceptos teóricos en los cuales se basa la práctica regularizada.

6. Exámenes Libres:
   6.1. Dada las características de la práctica de la asignatura, no se admitirán exámenes de alumnos libres.

IX - Bibliografía Básica

[1] Janca, Tanya. Bob and Alice learn Secure Coding. Wiley. 2025.
[2] Hoffman, Andrew. Web Application Security Exploitation and Contermeasures for Modern Web Application . O’Reilly. 2024.
[3] Richards, Mark; Ford Neal. Fundamentals of Software Architecture. O’Reilly. 2020.
[4] Heather Adkins, Betsy Beyer, Paul Blankinship, Piotr Lewandowski, Ana Oprea & Adam Stubblefi eld. Building Secure & Reliable Systems Best Practices for Designing, Implementing and Maintaining Systems. 2020.

[5] Michael Howard and David LeBlanc. WRITING SECURE CODE Best Practices. Microsoft. 2003.
[6] Erickson, Jon. Hacking: The art of exploitation. No Starch Press. 2008.
[7] Georgia Weidman. Penetration testing A Hands-On Introduction to Hacking. No Starch Press. 2014.
[8] Chuck Easttom. Computer Security Fundamentals, Third Edition. Perason. 2016,
[9] Eilam, Eldad. Reversing: Secrets of Reverse Engineering.Wiley Publishing Inc. 2005.
[10] Axel van Lamsweerde. Requirements Engineering from System Goals to UML Models to Software Specifications. Wiley. 2012.
[11] Parr, Terence. The Definitive ANTLR4 Reference. The Pragmatic Bookshelf. 2012.

X - Bibliografia Complementaria

[1] Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman. Compilers: Principles, Techniques, and Tools. Addison-Wesley / Pearson Education. 1986.
[2] Roger S. Pressman y Bruce R. Maxim. Software Engineering: A Practitioner’s Approach. McGraw-Hill Education.2019.
[3] Martin Fowler (con colaboración de Rebecca Parsons). Domain-Specific Languages. Addison-Wesley Professional (Signature Series). 2010.
[4] https://docs.python.org/3/tutorial/ .2026
[5] Stuart Russell y Peter Norvig. Artificial Intelligence: A Modern Approach 4ta Edition. 2021.

XI - Resumen de Objetivos

- Comprender los fundamentos de la seguridad informática y su importancia en el ciclo de vida del software.
- Desarrollar competencias para aplicar metodologías, estándares y buenas prácticas de seguridad en cada etapa del desarrollo.
- Formar profesionales capaces de identificar, analizar y mitigar riesgos de seguridad en sistemas y aplicaciones.
- Promover una cultura de responsabilidad, aprendizaje continuo y adaptación a nuevas tecnologías y amenazas.

XII - Resumen del Programa

Unidad I: Seguridad de los Sistemas Informáticos
Unidad II: Seguridad en la Etapa Captura de Requisitos
Unidad III: Seguridad en la Etapa de Diseño
Unidad IV: Seguridad en la Etapa de Codificación
Unidad V: Seguridad en la Etapa de Pruebas
Unidad VI: Seguridad en la Etapa de Lanzamiento / Despliegue
Unidad VII: Seguridad en la Etapa de Mantenimiento
Unidad VIII: Consideraciones Finales

XIII - Imprevistos

Los imprevistos serán resueltos por la cátedra en la medida que aparezcan.
Correo de Contacto: Mario Berón -- mberon@email.unsl.edu.ar

XIV - Otros

Las vías de comunicación con los estudiantes son las siguientes:

- Correo electrónico del responsable de la materia: mberon@email.unsl.edu.ar
- Oficina: 3 Bloque II - Primer Piso
- Teléfono: +54 (266) 4520300 - Interno: 2103