PROGRAMA DE LABORATORIO

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS

SOFTWARE AVANZADO

CÓDIGO:	**0780**	PONDERACIÓN:	6
ESCUELA DE INGENIERÍA EN:	**CIENCIAS Y SISTEMAS**	ÁREA A LA QUE PERTENECE:	**DESARROLLO DE SOFTWARE**
PRE REQUISITO:	**785 - ANÁLISIS Y DISEÑO DE SISTEMAS 2**	POST REQUISITO:	**N/A**
CATEGORÍA:	**OBLIGATORIO**	VIGENCIA:	**SEGUNDO SEMESTRE 2025**
HORAS POR SEMANA DEL CURSO:	**4**	HORAS POR SEMANA DEL LABORATORIO:	**1.7**
HORAS DE AUTOAPRENDIZAJE:	X	TOTAL DE HORAS DE APRENDIZAJE:	4
CATEDRÁTICO (A):	Everest Medinilla	AUXILIAR:	Julio Roberto Vasquez Santiago
EDIFICIO:	Virtual	SECCIÓN:	A
SALÓN DEL CURSO:	Virtual	SALON DEL LABORATORIO:	A
DIAS QUE SE IMPARTE EL CURSO:	Martes y Jueves	DIAS QUE SE IMPARTE EL LABORATORIO:	Jueves
HORARIO DEL CURSO:	7:10 - 8:50	HORARIO DEL LABORATORIO:	17:20 - 19:00

Breve descripción del Laboratorio

Software Avanzado es un curso profesional que pertenece al área de software de la carrera de Ingeniería en Ciencias y Sistemas, el cual trata sobre conceptos fundamentales de ingeniería de software, se tiene especial énfasis en tecnologías modernas en la nube, devops y metodologías ágiles.

Índice

Competencias Vinculadas al Perfil del Egresado

Competencias Específicas

No.	Competencia
1	Toma decisiones profesionales con base en fundamentos teóricos, datos e información pertinente, válida y confiable.
2	Demuestra destreza y habilidad en la selección, uso y adaptación de herramientas metodológicas, tecnológicas, equipos especializados y en la lectura e interpretación de datos, pertinentes al contexto de su ejercicio profesional.
3	Identifica sus necesidades de actualización, capacitación y formación, durante su proceso formativo y en el ejercicio profesional, y busca los medios para cubrirlas por medios formales e informales, nacionales e internacionales, presenciales y en línea.
4	Lidera y colabora proactivamente en equipos de trabajo y en comunidades profesionales para el logro de objetivos y mejoramiento de la calidad de vida.
5	Identifica oportunidades y riesgos para la innovación y adaptación de conocimientos y tecnologías para resolver problemas.

Competencias Generales

No.	Competencia
1	Aplica estándares de calidad, eficiencia y seguridad en la implementación adecuada de soluciones de software, hardware y TIC en general.
2	Maneja e Interpreta adecuadamente datos masivos, sean estos estructurados o no estructurados, facilitando su visualización e interpretación de forma eficaz en apoyo a la toma de decisiones.
3	Construye soluciones integrales trabajando en forma colaborativa y propositiva en equipos interdisciplinarios, en forma presencial o utilizando plataformas virtuales.
4	Actualiza permanente sus conocimientos relacionados con TIC en general, apoyándose en las estrategias de aprendizaje apropiadas.
5	Aplica principios básicos de ingeniería, ciencias de computación y sistemas de información y comunicación, en la formulación y resolución adecuada de problemas complejos.

Competencias del Laboratorio

Competencia(s) Específica(s)

No.	Competencia	Nivel de Aprendizaje
1	Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Crear
2	Mejora continuamente el rendimiento, la seguridad y la mantenibilidad de sistemas de software existentes, mediante el análisis de métricas, refactorización de código, aplicación de principios SOLID y adopción de herramientas de monitoreo y automatización, asegurando soluciones más eficientes y alineadas con los estándares de calidad del sector.	Evaluar
3	Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Aplicar
4	Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	Evaluar

Competencia(s) General(es)

No.	Competencia	Nivel de Aprendizaje
1	Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas. Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los sistemas informáticos.	Crear
2	Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	Define
3	Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	Recordar

Diseño Didáctico por Competencias

Esta sección organiza las sesiones del laboratorio en función de las competencias que el estudiante debe desarrollar. Cada clase incluye valores (saber ser), contenidos teóricos (saber) y habilidades prácticas (saber hacer), permitiendo un aprendizaje integral y aplicado. Las actividades están alineadas con los objetivos del curso y el perfil del egresado.

Sesión de Diagnóstico

Evaluación de conocimientos previos

Se aplicará una actividad diagnóstica con el objetivo de identificar el nivel de conocimientos y habilidades que los estudiantes poseen al inicio del curso. No influye en la nota final, pero es obligatoria para todos los estudiantes.

Tipo de Actividad	Descripción
(puede ser un cuestionario, una dinámica participativa o un ejercicio práctico breve)	

Presentación del tutor

El tutor se presenta formalmente al grupo, compartiendo su formación académica, experiencia profesional y educativa, así como sus expectativas sobre el curso. También se abordan aspectos como normas de convivencia, canales de comunicación, disponibilidad para consultas y métodos de acompañamiento.

Presentación de los estudiantes

Se escogen un grupo de estudiantes al azar. En su presentación, se les pedirá que compartan información básica como su nombre, intereses personales o profesionales, experiencias previas relacionadas con el curso y sus expectativas. Esta actividad busca promover la interacción, el reconocimiento entre pares y la construcción de un entorno participativo y respetuoso.

Presentación del programa del curso

Se presenta el contenido del programa del curso, se aclaran dudas y se fomenta el compromiso del estudiante con su aprendizaje.

Evaluación de conocimientos del laboratorio actual

Se realiza una evaluación o práctica que permite conocer el grado de familiaridad de los estudiantes con las herramientas, entornos o competencias técnicas necesarias para el laboratorio actual.

Tipo de Actividad	Descripción
por ejemplo, uso de simuladores, entornos de desarrollo, hardware específico, etc. Puede	

incluir ejercicios prácticos, pruebas técnicas o autoevaluaciones guiadas.	

Sesión No. 1, Unidad No. 1 - Fundamentos de Desarrollo

Nombre: Puntualidad y Honestidad

Conocimiento (Saber)

Competencia(s)	
Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas. Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los sistemas informáticos.	
Tema	Subtema
Introducción a herramientas de frontend y backend	Frontend: HTML, CSS, JavaScript
	Frameworks de frontend
	Backend: Node.js
	Bases de datos
	APIs REST
	Autenticación y autorización (OAuth, JWT)
	Session-based Auth y SAML

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Diseñar algoritmos aplicando diagramas de flujo y pseudocódigo para resolver problemas básicos de lógica computacional Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida	Práctica	0

del software y colaborar de forma efectiva en equipos de desarrollo ágil.		
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	Actividad	0

Sesión No. 2, Unidad No. 1 - Fundamentos de Desarrollo

Valor de la semana (Saber ser)

Nombre: Honestidad

Conocimiento (Saber)

Competencia(s)	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	
Tema	Subtema
Metodología Ágil	Principios de la metodología ágil
	Scrum vs. Kanban
	Roles en Scrum (Product Owner, Scrum Master, Development Team)
	Ceremonias de Scrum (Sprint Planning, Daily Standup, Sprint Review, Retrospective)
	Artefactos (Product Backlog, Sprint Backlog, Burndown charts)
	Historias de usuario y estimación ágil (Story Points)
	Otras metodologías Ágiles

	Principios de la metodología ágil

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas.Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los sistemas informáticos.	N/A	N/A
	N/A	N/A
	N/A	N/A

Sesión No. 3, Unidad No. 1 - Fundamentos de Desarrollo

Valor de la semana (Saber ser)

Nombre: Lealtad y responsabilidad

Conocimiento (Saber)

Competencia(s)	
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	
Tema	Subtema

Nubes (AWS, GCE, Azure)	Computación en la nube: Modelos de servicio (IaaS, PaaS, SaaS)

Competencia(s)	
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	
Tema	**Subtema**
Nubes (AWS, GCE, Azure)	Introducción a AWS (EC2, S3, RDS, Lambda)
	Introducción a GCP (Compute Engine, Cloud Storage, Cloud Functions)
	Introducción a Azure (Virtual Machines, Blob Storage, App Services)
	Costos y escalabilidad en la nube
	Configuración de entornos de desarrollo en la nube

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas.Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los sistemas informáticos.	Práctica	N/A
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos	Práctica	N/A

organizacionales.		

Sesión No. 4, Unidad No. 1 - Fundamentos de Desarrollo

Valor de la semana (Saber ser)

Nombre: Curiosidad y perseverancia

Conocimiento (Saber)

Competencia(s)	
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	
Tema	Subtema
Docker	Conceptos básicos de contenedores
	Instalación de Docker
	Imágenes y contenedores
	Dockerfile y construcción de imágenes personalizadas
	Volúmenes y redes en Docker
	Docker Compose para gestión de múltiples contenedores
	Registro de imágenes (DockerHub, ECR, GCR)

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
N/A	N/A	N/A

Sesión No. 5, Unidad No. 1 - Fundamentos de Desarrollo

Valor de la semana (Saber ser)

Nombre: Disciplina

Conocimiento (Saber)

Competencia	
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	
Tema	Subtema
Microservicios	Introducción a la arquitectura de microservicios
	Comparación entre monolitos y microservicios
	Diseño de microservicios (Principios SOLID)
	Comunicación entre microservicios (HTTP, gRPC, eventos)
	Patrones de diseño en microservicios (Saga, Circuit Breaker)
	Implementación de APIs RESTful con microservicios
	Despliegue de microservicios con Docker

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	N/A	N/A
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	Práctica	2.5
	Corto/Cuestionario	
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Práctica	
	Corto/Cuestionario	

Sesión No. 6, Unidad No. 2 - Arquitectura de Software, Orquestación

Valor de la semana (Saber ser)

Nombre: Honestidad

Conocimiento (Saber)

Competencia
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías,

patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	
Tema	Subtema
Kubernetes	Conceptos básicos de Kubernetes
	Instalación de Kubernetes (Minikube, K3s, Kubernetes en la nube)
	Pods, Servicios y Deployments
	ConfigMaps y Secrets
	Gestión de almacenamiento en Kubernetes (Persistent Volumes)

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	N/A	N/A
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	N/A	N/A

Sesión No. 7, Unidad No. 3 - DevOps

Valor de la semana (Saber ser)

Nombre: Paciencia y empatía

Conocimiento (Saber)

Competencia	
Mejora continuamente el rendimiento, la seguridad y la mantenibilidad de sistemas de software existentes, mediante el análisis de métricas, refactorización de código, aplicación de principios SOLID y adopción de herramientas de monitoreo y automatización, asegurando soluciones más eficientes y alineadas con los estándares de calidad del sector.	
Tema	**Subtema**
Introducción a DevOps	Qué es DevOps y por qué es importante
	Cultura DevOps (colaboración y automatización)
	Ciclo de vida de DevOps
	Principales herramientas DevOps
	DevOps vs. otros enfoques de desarrollo (Agile, Waterfall)

Habilidades (Saber Hacer)

Competencia	**Tipo de Actividad**	**Ponderación**
Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas.Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los sistemas informáticos.	N/A	N/A

Sesión No. 8, Unidad No. 3 - DevOps

Valor de la semana (Saber ser)

Nombre: Honestidad y Veracidad

Conocimiento (Saber)

Competencia	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	
Tema	**Subtema**
DevOps: Planificación y Codificación	Gestión de versiones con Git y GitFlow
	Backlog de producto y planificación ágil
	Pipeline de desarrollo: Continuous Integration (CI)
	Automatización de builds con Jenkins/Travis/GitLab CI
	Automatización de dependencias (npm, pip, Maven)

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Ejercicio	N/A
Mejora continuamente el rendimiento, la seguridad y la mantenibilidad de sistemas de software existentes, mediante el análisis de métricas, refactorización de código, aplicación de principios SOLID y adopción de herramientas de monitoreo y automatización, asegurando soluciones más eficientes y alineadas con los estándares de calidad del sector.	Ejercicio Actividad	N/A
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	Ejercicio	N/A
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Ejercicio	N/A
Mejora continuamente el rendimiento, la seguridad y la mantenibilidad de sistemas de software existentes, mediante el análisis de métricas, refactorización de código, aplicación de principios SOLID y adopción de herramientas de monitoreo y automatización, asegurando soluciones más eficientes y alineadas con los estándares de calidad del sector.	Práctica	N/A
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	N/A	N/A

Sesión No. 9, Unidad No. 3 - DevOps

Valor de la semana (Saber ser)

Nombre: Sacrificio

Conocimiento (Saber)

Competencia	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	
Tema	**Subtema**
DevOps: Building & Testing	Conceptos de CI/CD
	Pipelines CI/CD y Automatización de despliegues
	Gestión de versiones, release tagging y Principales herramientas CI/CD (Jenkins, CircleCI, GitLab CI)
	Compilación automatizada de aplicaciones
	Creación de artefactos de compilación (paquetes, binarios, imágenes) y Almacenamiento de artefactos (Artefactory, Nexus)
	Tipos de pruebas en DevOps (unitarias, integración, end-to-end), Pruebas de carga y rendimiento (JMeter, Gatling)
	Automatización de pruebas con frameworks (JUnit, Mocha, Jest) e Integración de pruebas unitarias en el pipeline de construcción

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Ejercicio	2.5
	Práctica	
	Cuestionario /Corto	
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Ejercicio	0
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	Práctica	0
Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	Práctica	0

Sesión No. 10, Unidad No. 3 - DevOps

Valor de la semana (Saber ser)

Nombre: Disciplina

Conocimiento (Saber)

Competencia	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	
Tema	**Subtema**
DevOps: Release & Deploy	Automatización de despliegues
	Estrategias de despliegue (Blue-Green, Canary, Rolling Updates)
	Despliegues seguros y reversión
	Despliegue en entornos de producción (AWS, Azure, GCP)
	Infraestructura como código (Terraform, Ansible)

Habilidades (Saber Hacer)

Competencia	**Tipo de Actividad**	**Ponderación**
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Ejercicio	0

Mejora continuamente el rendimiento, la seguridad y la mantenibilidad de sistemas de software existentes, mediante el análisis de métricas, refactorización de código, aplicación de principios SOLID y adopción de herramientas de monitoreo y automatización, asegurando soluciones más eficientes y alineadas con los estándares de calidad del sector.	Ejercicio / Practica	0
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	Ejercicio	0
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Ejercicio	0
Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	Ejercicio	0
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Ejercicio	0
Integra componentes, servicios y tecnologías de software mediante el uso de APIs, servicios web y herramientas de integración continua, asegurando la interoperabilidad, cohesión funcional y eficiencia en la entrega de soluciones informáticas.	Ejercicio	0

Sesión No. 11, Unidad No. 3 - Devops

Valor de la semana (Saber ser)

Nombre: Gratitud

Conocimiento (Saber)

Competencia	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar de forma efectiva en equipos de desarrollo ágil.	
Tema	**Subtema**
DevOps: Operate & Monitor	Principales herramientas de monitoreo (Prometheus, Grafana, ELK Stack)
	Alertas y métricas (Alertmanager, Grafana)
	Centralización de logs
	Monitoreo de aplicaciones en la nube
	Tracing distribuido (Jaeger, OpenTelemetry)
	Logging en contenedores y microservicios

Habilidades (Saber Hacer)

Competencia	**Tipo de Actividad**	**Ponderación**
Planifica de forma técnica , profunda y estratégica el desarrollo de software utilizando herramientas especializadas para el diseño y modelado de arquitecturas escalables y robustas.Aplica metodologías ágiles como Scrum o Kanban, combinadas con prácticas DevOps y diseño orientado a servicios (SOA o microservicios), para asegurar una implementación óptima, continua y de alta calidad de los	Ejercicio	5

sistemas informáticos.		
Diseña arquitecturas de software modulares y escalables, seleccionando tecnologías, patrones de diseño y estilos arquitectónicos adecuados (como microservicios, cliente-servidor o arquitectura en capas), utilizando herramientas de modelado como UML, ArchiMate o Draw.io, para asegurar la sostenibilidad, mantenibilidad y evolución del sistema.	Ejercicio	
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	Ejercicio	
Define de manera estratégica y técnica los lineamientos, estándares y buenas prácticas para el desarrollo de software, considerando los requerimientos del negocio, la arquitectura del sistema y las metodologías ágiles, con el fin de garantizar soluciones eficientes, escalables y alineadas a los objetivos organizacionales.	Ejercicio	
Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	Ejercicio	
Identificar de manera crítica y sistemática los requerimientos funcionales y no funcionales de los sistemas informáticos, así como los riesgos técnicos y necesidades del usuario utilizando técnicas de análisis y herramientas especializadas, para asegurar un diseño y desarrollo de software alineado a los objetivos del proyecto y del negocio.	Ejercicio Práctica	
Utiliza herramientas de control de versiones, entornos de desarrollo integrados (IDE) y sistemas de gestión de incidencias para gestionar eficientemente el código fuente, automatizar tareas del ciclo de vida del software y colaborar	Ejercicio	

de forma efectiva en equipos de desarrollo ágil.		

Tiempo de Auto-aprendizaje

Tipo	Horas de Auto-aprendizaje
Proyectos	105
Prácticas	105
Tareas	0
Total	**210**

Rúbrica de Evaluación

Cada una de las actividades del laboratorio (proyectos, prácticas, tareas y otras) cuenta con una rúbrica de evaluación específica, la cual está detallada en el documento que se entrega al estudiante al momento de asignar la actividad. Estas rúbricas describen los criterios de evaluación, niveles de desempeño esperados y la ponderación correspondiente de cada aspecto evaluado.

Es **responsabilidad del estudiante** leer detenidamente la rúbrica asignada antes de iniciar el desarrollo de la actividad. Comprender los criterios de evaluación no solo permite orientar adecuadamente el trabajo, sino también mejorar el desempeño académico y fomentar la autorregulación del aprendizaje.

En caso de no recibir la rúbrica al momento de la asignación, el estudiante **debe solicitarla directamente al tutor académico**, ya que constituye una herramienta esencial para el cumplimiento de los objetivos de aprendizaje y la evaluación transparente.

Resumen de Ponderaciones

Tipo	Valor
Actividades en Clase	5
Proyectos (3 fases)	60
Prácticas (8 practicas)	30
Tareas	0

Examen Final	5
Total	**100**

Normativa Académica y Ética del Curso

En concordancia con el perfil del estudiante de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, se espera un alto nivel de compromiso con la excelencia académica y la ética profesional. Por ello, que se establece los siguientes lineamientos de carácter obligatorio que regulan el comportamiento académico del estudiante:

Plagio y copias

- Todo proyecto será sometido a verificación para confirmar su autoría y originalidad, con la finalidad de evitar cualquier plagio, copia o que la actividad no haya sido realizada por el estudiante.
- Cualquier evidencia de lo antes descrito en las distintas actividades será sancionada con una calificación de 0 (cero) y el caso será reportado al Docente quien a su vez informará a la Escuela de Ciencias y Sistemas para su seguimiento institucional.

Prórrogas y reposiciones

- No se otorgarán prórrogas para entregas de actividades.
- No se permitirá la reposición de proyectos bajo ninguna circunstancia.

Requisitos para evaluación final del curso

- Es obligatorio aprobar el laboratorio para tener derecho a la evaluación final del curso.
- La calificación de prácticas, proyectos y otras actividades que se indique será asignada de forma presencial, en la fecha y hora establecidas por el tutor académico.

Asistencia

- Para obtener la nota del laboratorio, se requiere un mínimo del 80% de asistencia a las sesiones de laboratorio.
- En caso de inasistencia, sólo se aceptarán justificaciones válidas respaldadas por constancia oficial.

Entregas

- No se aceptarán entregas tardías de tareas, prácticas, exámenes cortos, exámenes finales o proyectos sin justificación.

Medio oficial de entrega

- La plataforma UEDI de la Facultad será el único medio oficial para la entrega de actividades del curso.

Equipo Académico

Coordinador del Área

Nombre: **Marlon Francisco Orellana Lopez**	Correo electrónico:

Sección B

Docente

Nombre del Docente **Everest Medinilla**	Correo electrónico **emedin@gmail.com**

	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Día		x		x		
Horario		7:10 - 8:50		7:10 - 8:50		
Lugar		Virtual		Virtual		

Tutor(es)

Nombre del Tutor	**Julio Roberto Vasquez Santiago**	
Correo electrónico institucional	**2176801782211@ingenieria.usac.edu.gt**	

Tipo		Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Clase	Día				x		
	Horario				17:20 - 19:00		

	Lugar				Virtual		
Atención al Estudiante	Día	x	x	x	x	x	
	Horario	19:00 - 21:00	19:00 - 21:00	19:00 - 21:00	19:00 - 21:00	19:00 - 21:00	
	Lugar	Foros UEDI y medios virtuales	Foros UEDI y medios virtuales	Foros UEDI y medios virtuales	Foros UEDI y medios virtuales	Foros UEDI y medios virtuales	

Bibliografía

- [1] Roger S. Presman. Ingeniería de Software. Un enfoque práctico. McGraw Hill,
- Quinta Edición.E.U.A., 2007.
- [2] Documentos elaborados por el Catedrático del Curso. Ian Sommerville. Ingeniería
- de Software. Prentice Hall. 7ma edición.
- [3] Adair, J., Decision Making and Problem Solving Strategies, 2nd Ed., Kogan Page, E.U.A. 2007.
- [4] Gerald Kontoya and Ian Sommerville, Requirements Engineering - Process and Techniques
- [5] Erl, Thomas. SOA Principles of Service Desing. Prentice Hall. 2008.
- [6] HashiCorp. Terraform.
- https://www.terraform.io Docker docs.
- https://docs.docker.com