UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
FICHA TÉCNICA DEL CURSO:

Laboratorio de análisis y diseño de sistemas 2

Código 785	**Créditos** 5	
Escuela Ciencias y Sistemas	**Área a la que pertenece** Área de desarrollo de Software	**Vigencia** Segundo semestre 2020
Horas por semana 2	**Horario laboratorio** Miércoles 10:40-12:20	
Pre-requisitos: 283-Análisis y Diseño de Sistemas 1	**Post-requisitos:** 780-Software avanzado	
Catedrático: William Samuel Guevara Orellana	**Tutor:** Brandon Javier Soto Castañeda	

Sección: A+

I. Descripción del laboratorio

El laboratorio de Análisis y diseño de sistemas 2 es la continuación y el complemento de lo que se aprendió en el curso de Análisis y diseño de sistemas 1, es la aplicación de los conceptos y/o conocimientos adquiridos, pero en la práctica.

Muchas empresas a la hora de desarrollar software no tienen implementada ninguna metodología de desarrollo de software, incluso los departamentos se mantienen aislados unos de otros, esto es una desventaja y conlleva a posibles fracasos en los proyectos o incluso al fracaso de la organización.

El uso correcto de buenas prácticas y metodologías en conjunto con herramientas adecuadas para el desarrollo de un software nos permite tener mayores ventajas y éxito en la elaboración de proyectos.

El curso de análisis y diseño de sistemas 2 permite tener un amplio conocimiento de las buenas prácticas y en el laboratorio se conocerán las herramientas necesarias para que dichas prácticas sean correctamente aplicadas. Una de las practicas que se manejara en dicho curso es **DevOps**.

II. Objetivos

General:

Lograr que el estudiante adquiera los conocimientos necesarios para poder analizar y diseñar un sistema de acuerdo con las tecnologías y herramientas más recientes, adoptando para ello buenas prácticas y metodologías de desarrollo.

Específicos:

- Que el estudiante comprenda el proceso para efectuar la entrega continua de un software.
- Que el estudiante comprenda el proceso para efectuar el despliegue continuo de un software.
- Que el estudiante conozca y ponga en práctica los conceptos de arquitectura de software abarcados durante el curso.
- Que el estudiante comprenda y practique los distintos patrones de diseño que existen.
- Familiarizar al estudiante con las herramientas disponibles para aplicar un completo y correcto desarrollo de software.

III. Habilidades

- Que el estudiante investigue, comprenda y aplique los conocimientos adquiridos durante cursos anteriores, para diseñar y analizar sistemas de forma óptima.

- Analizar los requerimientos de un software para la realización de un sistema que cumpla con todas las etapas del ciclo de vida del software.

- Extraer y representar el conocimiento necesario para construir e implementar una solución para un proyecto de software cumpliendo prácticas de administración de la configuración y el diseño de soluciones informáticas a un problema propuesto.

IV. Competencias

- Que el estudiante pueda aplicar una buena administración de la configuración.
- Capacidad para analizar y determinar requerimientos de software.
- Resolver y reconocer problemas clásicos de la administración de proyectos.
- Capacidad para crear y utilizar herramientas de integración continua, control de versiones y pruebas de software.
- Capacidad de adaptarse a cambios en los requerimientos iniciales de software.

V. Metodología

- Se impartirán exposiciones virtuales en salas meet con temas enfocados a la práctica de los conceptos del curso.

- El laboratorio estará dividido en una parte teórica y en una parte práctica, para que la reunión sea un poco más dinámica y los conceptos puedan ser aplicados.

- Se realizarán exámenes cortos como parte de la evaluación de prácticas y contenido impartido en el laboratorio.

- Se realizarán prácticas donde se busca que el estudiante conozca y experimente con las distintas herramientas mencionadas a lo largo del contenido del laboratorio, aplicándolas sobre problemas reales.

- Se realizarán tareas y hojas de trabajo para que el estudiante pueda complementar y comprender de mejor manera los tema tratados.

VI. Contenido

Tópico	Subtemas	Herramientas
Administración de la configuración	• Control de Versiones • Herramientas de control de versiones • Integración Continua • Ciclo general de la integración continua • Herramienta para integración continua	▪ Git ▪ GitLab ▪ Nodejs ▪ Docker
Entrega continua	• Ambientes del software • Flujo de entrega de software • Componentes de la entrega continúa	▪ GitLab-runner ▪ Docker-compose
Pruebas	• Pruebas funcionales o Pruebas unitarias o Pruebas de aceptación o Pruebas de integración o Pruebas de regresión • Pruebas no funcionales: o Pruebas de carga o Pruebas de estrés o Pruebas de escalabilidad	▪ Selenium ▪ K6 ▪ Mocha

	○ Pruebas de portabilidad	
Arquitectura de software	• Arquitectura por capas • Arquitectura 4 + 1 vistas • MVC	
Arquitecturas orientadas al servicio	• SOA • Web services y SOA • Cloud Computing	▪ Google cloud platform ▪ Kubernetes
Patrones de diseño	• Patrones de creación • Patrones de estructura • Patrones de comportamiento	

VII. Evaluación

El laboratorio se aprueba con una nota mayor o igual a 61 puntos

- Practicas (60 puntos)
  - Practica 1, 15 puntos
  - Practica 2, 20 puntos
  - Practica 3, 25 puntos
- Hoja de trabajo (10 puntos)
- Cortos (10 puntos)
- Tareas (10 puntos)
- Examen final (10 puntos)

VIII. Observaciones:

- Es obligatorio aprobar el laboratorio para tener derecho a examen final, se aprueba con 61/100
- Solo se calificarán las actividades de estudiantes asignados al curso. NO se agregan estudiantes en actas.
- Los equipos para trabajar las practicas son de 3 personas, esto para fomentar el trabajo colaborativo y serán formados por el tutor académico, si algún integrante se queda sin grupo luego de una entrega puede adherirse a otro grupo luego de discutirlo con el tutor y con el grupo al que se va a adherir. NO SE CALIFICARÁN TRABAJOS INDIVIDUALES EN NINGUNA CIRCUNSTANCIA.

Bibliografía	▪ "Version Control with Subversion", Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michal Pilato. ▪ "Essential Software Architecture", Ian Gorton ▪ "Headfirst Design Patterns", Elisabeth Freeman, Eric Freeman, Bert Bates, Kathy Sierra ▪ "Continuous Delivery", Jez Humble, David Farley ▪ "Service Oriented Architecture, Concepts, Technology and Design", Thomas Er
No. De Secciones	2
Director de Escuela	Ing. Carlos Alonzo