

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

COORDINACIÓN GENERAL DE FORMACIÓN PROFESIONAL

PROGRAMA DE UNIDAD DE APRENDIZAJE

I. DATOS DE IDENTIFICACIÓN

- 1. Unidad Académica:** Facultad de Ciencias Administrativas y Sociales, Ensenada; Facultad de Ciencias Administrativas, Mexicali; Facultad de Ciencias de la Ingeniería, Administrativas y Sociales, Tecate; y Facultad de Contaduría y Administración, Tijuana.
- 2. Programa Educativo:** Licenciado en Inteligencia de Negocios
- 3. Plan de Estudios:** 2021-2
- 4. Nombre de la Unidad de Aprendizaje:** Calidad e Implantación del Software
- 5. Clave:** 39080
- 6. HC:** 02 **HT:** 00 **HL:** 02 **HPC:** 00 **HCL:** 00 **HE:** 02 **CR:** 06
- 7. Etapa de Formación a la que Pertenece:** Terminal
- 8. Carácter de la Unidad de Aprendizaje:** Optativa
- 9. Requisitos para Cursar la Unidad de Aprendizaje:** Ninguno



Equipo de diseño de PUA

Claudia Viviana Álvarez Vega
Karina Caro Corrales
Sandra Julieta Saldivar González
Emma Sofia Castillejos Caballero

Vo.Bo. de subdirectores de las Unidades Académicas

Adelaida Figueroa Villanueva
Angélica Reyes Mendoza
Esperanza Manrique Rojas
Jesús Antonio Padilla Sánchez

Fecha: 12 de noviembre de 2020

II. PROPÓSITO DE LA UNIDAD DE APRENDIZAJE

La unidad de aprendizaje brinda el uso de estándares, técnicas y herramientas para validar y verificar la calidad de los sistemas de inteligencia de negocios lo que permite elaborar el plan que contenga el diseño y ejecución de pruebas, asegurando la calidad del software dentro de las organizaciones.

Se ubica en la etapa terminal, es de carácter optativa y forma parte del área de Ciencia de Datos.

III. COMPETENCIA GENERAL DE LA UNIDAD DE APRENDIZAJE

Diseñar un plan de pruebas de sistemas de inteligencia de negocios, a través del uso de estándares, técnicas y herramientas vanguardistas, para asegurar la calidad del software y apoyar la operación de la organización, desarrollando una actitud proactiva, organizada y confidencial.

IV. EVIDENCIA(S) DE APRENDIZAJE

Proyecto final que abarque las inspecciones de software, plan de pruebas, reporte de ejecución de pruebas así como el informe de evaluación del aseguramiento de la calidad del software.

V. DESARROLLO POR UNIDADES
UNIDAD I. Fundamentos de Ingeniería de Software

Competencia:

Comparar la definición de requerimientos e interfaces de frontend desarrolladas, mediante el diseño de casos de prueba, para identificar si el resultado esperado y obtenido es el mismo, con creatividad, responsabilidad y actitud analítica.

Contenido:

Duración: 4 horas

- 1.1. Análisis de frontend y backend
 - 1.1.1. Análisis de requerimientos
 - 1.1.2. Análisis de arquitectura

UNIDAD II. Proceso de verificación y validación

Competencia:

Examinar el proceso de verificación y validación, mediante el análisis de los objetivos, métricas, estándares del proyecto, producto y proceso, de acuerdo a la importancia y los beneficios, para el aseguramiento de la calidad acorde a la especificación de requerimientos y diseño definido, con actitud objetiva, metódica y organizada.

Contenido:

Duración: 8 horas

- 2.1. Terminología y fundamentos de la V&V
- 2.2. Objetivos y limitaciones de V&V
- 2.3. Métricas para la medición del proyecto, producto y proceso
- 2.4. Participación de V&V en diferentes puntos del ciclo de vida
- 2.5. Ventajas de V&V con respecto al aseguramiento de la calidad de software
- 2.6. Estándares y modelos de procesos para V&V
 - 2.6.1. Estándar IEEE 1012 V&V
 - 2.6.2. Niveles de integridad
 - 2.6.3. Recomendaciones V&V para requerimientos de software
- 2.7. El estándar ISO/IEC/IEEE 1220 -Verificación y Validación
 - 2.7.1. Procesos de verificación
 - 2.7.2. Proceso de validación
- 2.8. V&V en otros modelos de procesos
 - 2.8.1. Modelo CMMI Model: Verificación y Validación
 - 2.8.2. ISO/IEC 29110 and V&V: Verificación y Validación
 - 2.8.3. V&V en metodologías ágiles: Scrum
- 2.9. Tipos de pruebas (estáticas y dinámicas)

UNIDAD III. Técnicas de verificación

Competencia:

Probar el software, mediante la aplicación de técnicas de verificación automatizadas y no automatizadas, para llevar a cabo las buenas prácticas de codificación, de manera lógica, sistemática y asertiva.

Contenido:

Duración: 8 horas

3.1. Tipos de técnicas de verificación automatizadas y no automatizadas

- 3.1.1. Inspecciones: de diseño, de código
- 3.1.2. Revisiones
- 3.1.3. "Walkthroughs"
- 3.1.4. "Desk-checking", "Checklists"
- 3.1.5. Auditorías

3.2. Buenas prácticas de codificación

- 3.2.1. Esquema de nombramiento de variables y comentarios
- 3.2.2. Principio DRY (Don't Repeat Yourself)
- 3.2.3. Vulnerabilidades en el código
 - 3.2.3.1. SQL Injection
 - 3.2.3.2. Code injection
 - 3.2.3.3. Cross-Site Scripting
 - 3.2.3.4. Buffer Overflow
 - 3.2.3.5. Sensitive data exposure
 - 3.2.3.6. Credential management
 - 3.2.3.7. Otros
- 3.2.4. Refactorización
- 3.2.5. Guías de estilo de codificación
- 3.2.6. Modelo Vista Controlador (MVC)
- 3.2.7. Otros

UNIDAD IV. Técnicas de validación

Competencia:

Probar el software, mediante la aplicación de técnicas de validación, para asegurar que el software cumple la funcionalidad requerida de acuerdo a la naturaleza del proyecto de software, de manera lógica, sistemática y asertiva.

Contenido:

Duración: 8 horas

- 4.1. Plan de Pruebas
- 4.2. Diseño de casos de pruebas
- 4.3. Ejecución de pruebas
- 4.4. Métodos de pruebas
 - 4.4.1. Caja Negra
 - 4.4.2. Caja Blanca
 - 4.4.3. Caja Gris
- 4.5. Tipos de pruebas
 - 4.5.1. Pruebas unitarias
 - 4.5.2. Pruebas de integración
 - 4.5.3. Pruebas del sistema
 - 4.5.4. Pruebas de aceptación
 - 4.5.5. Pruebas de regresión
- 4.6. Técnicas y Herramientas para realizar pruebas (usando datos de prueba)
- 4.7. Automatización de las pruebas
 - 4.7.1. Herramientas para la automatización

UNIDAD V. Aseguramiento de la calidad

Competencia:

Distinguir los conceptos, componentes, atributos y características de la calidad del software, mediante el estudio de modelos y normas actuales y vigentes, para su aplicación en el software desarrollado, con actitud analítica, eficiente y responsable.

Contenido:

Duración: 4 horas

- 5.1. Conceptos básicos en la calidad del software
- 5.2. Componentes y atributos
- 5.3. Características que aseguran la calidad del software
- 5.4. Aplicación de modelos y normas de calidad en el software (actuales y vigentes)

VI. ESTRUCTURA DE LAS PRÁCTICAS DE LABORATORIO

No.	Nombre de la Práctica	Procedimiento	Recursos de Apoyo	Duración
UNIDAD III				
1	Diseña una prueba estática de verificación	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para elaborar el diseño de una prueba estática para verificación 2. Selecciona el artefacto de software para realizar la prueba estática (requerimientos, diseño, código, etc.). 3. Selecciona la prueba estática a aplicar. 4. Establece el procedimiento a realizar con base a la prueba seleccionada. 5. Realiza la prueba seleccionada. 6. Analiza resultados de la prueba. 7. Elabora un reporte con las actividades realizadas 8. Entrega el reporte al docente para su evaluación y retroalimentación. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Artefacto de software ● Entorno de ejecución ● Software de aplicación procesador para elaborar el reporte. 	4 horas
2	Herramienta de análisis estático automatizado	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para utilizar una herramienta de análisis estático automatizado 2. El profesor brinda el caso práctico sobre el código fuente del software que se va a analizar. 3. Descarga o accede (en caso de ser herramienta en línea) a la herramienta de análisis estático 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Código fuente ● Herramienta de análisis estático automatizado. ● Software de aplicación procesador para elaborar el reporte. 	4 horas

		<p>automatizado.</p> <ol style="list-style-type: none"> 4. Realiza el análisis estático del código fuente utilizando la herramienta de análisis. 5. Analiza los resultados del análisis. 6. Elabora un reporte con las actividades realizadas. 7. Entrega el reporte al docente para su evaluación y retroalimentación. 		
UNIDAD IV				
3	Diseña una prueba dinámica de validación	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para elaborar el diseño de una prueba dinámica para validación. 2. Selecciona la prueba dinámica a aplicar. 3. Establece el procedimiento a realizar con base a la prueba seleccionada. 4. Ejecuta la prueba. 5. Analiza resultados de la prueba. 6. Elabora un reporte con las actividades realizadas 7. Entrega el reporte al docente para su evaluación y retroalimentación. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Código fuente ● Entorno de ejecución ● Software de aplicación procesador para elaborar el reporte. 	4 horas
4	Métodos de pruebas de caja blanca y negra	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para elaborar el diseño de una prueba dinámica para validación. 2. Realiza una revisión bibliográfica del método de pruebas de caja negra y blanca. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Código fuente ● Herramienta de análisis estático automatizado. ● Software de aplicación procesador de palabras y presentaciones. 	4 horas

		<ol style="list-style-type: none"> 3. Analiza una área de oportunidad y selecciona el método de prueba más adecuado de los siguientes: <ol style="list-style-type: none"> a. Caja Negra. b. Caja Blanca. c. Caja Gris. 4. Expone los resultados de la prueba para su retroalimentación. 		
5	Casos de prueba	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para elaborar el diseño de casos de prueba. 2. Identifica los elementos para el caso de prueba en función de la especificación de requerimientos 3. Diseña los casos de prueba (componente, conjunto de entradas, salidas esperadas) 4. Expone los resultados de la prueba para su retroalimentación. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Computadora. ● Internet. ● Apoyo de libros físicos o virtuales. ● Software para elaborar el caso de prueba ● Software de aplicación procesador de palabras y presentaciones. ● Equipo de videoproyección para exponer resultados 	4 horas
6	Elaboración y ejecución del plan de pruebas	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para la elaboración y ejecución de los casos del plan de pruebas 2. Realiza el plan de pruebas 3. Selecciona el conjunto de datos para la ejecución. 4. Ejecuta caso de prueba. 5. Documenta y registra fallos encontrados. 6. Corrige los defectos encontrados. 7. Ejecuta pruebas de regresión que garantice proceso de 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Código fuente ● Entorno de ejecución ● Herramienta de gestión de defectos ● Software de aplicación procesador de palabras. 	4 horas

		<p>depuración.</p> <p>8. Entrega el reporte de actividad al docente para su evaluación y retroalimentación.</p>		
7	Herramientas para diseñar prueba automatizadas	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para elaborar el diseño de una prueba automatizada. 2. Selecciona la prueba automatizada a aplicar. 3. Establece el procedimiento a realizar con base a la prueba seleccionada. 4. Ejecuta la prueba. 5. Analiza resultados de la prueba. 6. Elabora un reporte con las actividades realizadas 7. Entrega el reporte al docente para su evaluación y retroalimentación. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Computadora ● Internet ● Apoyo de libros físicos o virtuales ● Software para elaborar el caso de prueba ● Software de aplicación procesador de palabras y presentaciones ● Equipo de videoproyección para exponer resultados 	4 horas
UNIDAD V				
8	Modelo de calidad de software	<ol style="list-style-type: none"> 1. Atiende las orientaciones del profesor para aplicar el modelo de calidad. 2. Define aspectos que lleven a la calidad del producto por cada nivel de madurez. 3. Entrega reporte de actividad al docente. 	<ul style="list-style-type: none"> ● Computadora ● Internet ● Apoyo de libros físicos o virtuales. ● Software de aplicación procesador de palabras y presentaciones. ● Proyector para exponer resultados 	4 horas

VII. MÉTODO DE TRABAJO

Encuadre: El primer día de clase el docente debe establecer la forma de trabajo, criterios de evaluación, calidad de los trabajos académicos, derechos y obligaciones docente-alumno.

Estrategia de enseñanza (docente):

- Estudio de caso
- Método de proyectos
- Aprendizaje basado en problemas
- Técnica expositiva
- Presenta ejercicios prácticos relacionados con las temáticas
- Selección y proyección de material audiovisual
- Proporciona información para las prácticas de laboratorio
- Propicia la participación activa de los estudiantes
- Revisa y evalúa reportes de prácticas y actividades
- Diseña y aplica evaluaciones

Estrategia de aprendizaje (alumno):

- Investigación
- Estudio de caso
- Trabajo en equipo
- Exposiciones
- Realiza prácticas de laboratorio
- Participa activamente en clase
- Trabaja de manera individual y en equipo
- Elabora y entrega actividades y reportes de prácticas en tiempo y forma
- Atiende puntualmente las indicaciones del docente
- Realiza proyecto final

VIII. CRITERIOS DE EVALUACIÓN

La evaluación será llevada a cabo de forma permanente durante el desarrollo de la unidad de aprendizaje de la siguiente manera:

Criterios de acreditación

- Para tener derecho a examen ordinario y extraordinario, el estudiante debe cumplir con los porcentajes de asistencia que establece el Estatuto Escolar vigente.
- Calificación en escala del 0 al 100, con un mínimo aprobatorio de 60.

Criterios de evaluación

- Evaluaciones parciales.....	20%
- Tareas	10%
- Prácticas de laboratorio.....	30%
- Proyecto final	40%
Total.....	100%

IX. REFERENCIAS

Básicas

- Cervantes, M. H., Velasco-Elizondo, P., y Castro, C. L. (2016). *Arquitectura de software: Conceptos y ciclo de desarrollo*. Recuperado de <https://ebookcentral.proquest.com>
- Haenkon, K., y Roger, L. (2021). *Software Engineering in IoT, Big Data, Cloud and Mobile Computing*. Springer.
- Lewis, W. E. (2017). *Software testing and continuous quality improvement (3^a ed.)*. CRC press.
- Pantaleo, G. (2016). *Calidad en el desarrollo de Software*. Alfaomega.
- Pressman, R. (2010). *Ingeniería de Software: un enfoque práctico (10th ed)*. McGRAW Hill. [clásica]
- Sommerville, I. (2016). *Software engineering (10th ed)*. Boston: Pearson.

Complementarias

- Ammann, P., y Offutt, J. (2017). *Introduction to Software Testing (2^a ed)*. Cambridge University Press.
- Chopra, R. (2018). *Software Testing: A Self-Teaching Introduction*. Mercury Learning and Information.
- Hernández, C. (2018). *Entendiendo la Norma ISO 9001 2015: Sistemas de Gestión*. CreateSpace Independent Publishing Platform.
- Kenett, R. S., Ruggeri, F., y Faltin, F. W. (Eds.). (2018). *Analytic methods in systems and software testing*. John Wiley & Sons.
- Winters, T., Manshreck, T., Wright, H. (2020). *Software Engineering at Google: Lessons Learned from Programming Over Time*. O'Reilly Media.

X. PERFIL DEL DOCENTE

El docente que imparta la unidad de aprendizaje debe contar con título de Licenciado en Inteligencia de Negocios, Licenciado en Informática o área afín, maestría en área afín, con tres años de experiencia laboral preferentemente en el área de calidad y pruebas de software. Además debe ser proactivo, analítico, organizado, que promueva la investigación, el trabajo en equipo y tener facilidad de palabra.