# Trabajo Práctico Integrador l



**ASIGNATURA:**

**Ingeniería de Software II**

**Introducción al Trabajo Práctico**

El desarrollo de software es una disciplina compleja y en constante evolución que requiere una comprensión profunda de una variedad de conceptos y enfoques. En este Trabajo Práctico, exploraremos varios aspectos esenciales relacionados con el diseño de software, la arquitectura de software, los conceptos clave en diseño de software y las metodologías ágiles. Cada uno de estos temas desempeña un papel crucial en la creación de sistemas de software de alta calidad y en la gestión eficiente de proyectos de desarrollo.

A lo largo de este trabajo, nos sumergiremos en escenarios prácticos y desafiantes que requieren la aplicación de conocimientos teóricos en situaciones del mundo real. Cada sección del trabajo abordará un tema específico y presentará preguntas y ejercicios diseñados para fomentar el pensamiento crítico, la resolución de problemas y la aplicación práctica de los conceptos aprendidos.

**Recomendaciones para la Realización del Trabajo Práctico**

**Claridad de Objetivos:** Antes de iniciar la ejecución del Trabajo Práctico, es imperativo que se comprendan de manera clara y precisa los objetivos establecidos. Esta comprensión facilitará la alineación de esfuerzos y garantizará la satisfacción de las expectativas planteadas.

**Exhaustiva Lectura y Comprensión:** Se recomienda efectuar una minuciosa lectura y comprensión de todas las directrices, interrogantes y escenarios presentados en el Trabajo Práctico. La comprensión cabal de las tareas a realizar es fundamental para cumplir adecuadamente con las expectativas del proyecto.

**Investigación Preliminar:** Es esencial llevar a cabo una investigación preliminar sobre los temas abordados en el Trabajo Práctico, ya sea en relación con el Diseño de Software, la Arquitectura de Software, los Conceptos de Diseño de Software o las Metodologías Ágiles. Este proceso de investigación permitirá una mejor preparación para enfrentar las interrogantes y ejercicios propuestos.

**Planificación y Organización**: Se aconseja la planificación meticulosa del tiempo disponible. Dividir el trabajo en secciones manejables y establecer plazos para cada una de ellas es una estrategia efectiva que contribuye a evitar la sensación de abrumo y asegura un avance constante.

**Aplicación Práctica:** Abordar cada pregunta y ejercicio desde una perspectiva eminentemente práctica es altamente recomendable. La inclusión de ejemplos concretos y situaciones reales, cuando sea pertinente, enriquecerá las respuestas y las fundamentará adecuadamente.

**Efectividad en la Comunicación:** La comunicación debe ser clara y concisa en todas las respuestas. La estructuración adecuada de las ideas, el uso de viñetas y, si es necesario, la incorporación de gráficos y diagramas, contribuirán a la comprensión de los puntos planteados.

**Referencias y Citas:** En caso de recurrir a fuentes externas, ya sean libros o recursos en línea, es imprescindible citarlas de forma apropiada y proporcionar referencias precisas. Esta práctica respalda la integridad académica del trabajo.

**Revisión y Edición:** Una revisión exhaustiva y una posterior edición del trabajo son etapas fundamentales antes de su entrega final. Durante este proceso, se deben verificar aspectos como la ortografía, la gramática y la coherencia, así como la completitud de todas las respuestas.

**UNIDAD 1 - Introducción a la Ingeniería de Software**

**Descripción:**

El propósito de este ejercicio es aplicar sus habilidades de pensamiento crítico y resolución de problemas en el contexto de la Ingeniería de Software. A través de un escenario ficticio, tendrá la oportunidad de analizar un proyecto de desarrollo de software que enfrenta desafíos y proponer soluciones para optimizar el proceso de desarrollo, con el fin de garantizar la calidad y la eficiencia.

**Escenario:** Imaginemos que usted es un profesional de Ingeniería de Software en una empresa líder en el desarrollo de software. Se le ha asignado un proyecto de gran importancia, pero el proceso de desarrollo actual está experimentando retrasos y problemas de calidad que amenazan el éxito del proyecto.

**Identificación de Problemas:** Realice un análisis detallado del proceso de desarrollo actual y enumere de manera sistemática los problemas que identifique. Estos problemas pueden incluir aspectos como el flujo de trabajo, la comunicación, la gestión de requerimientos, las prácticas de pruebas y otros.

**Propuesta de Soluciones:** Desarrolle sus propias ideas para abordar cada uno de los problemas que haya identificado. Debe ser específico en sus propuestas y fundamentarlas con argumentos sólidos. Las soluciones pueden involucrar cambios en el proceso, la adopción de nuevas herramientas, la redefinición de roles o cualquier mejora que considere necesaria.

**Justificación:** Para cada solución que proponga, proporcione una justificación sólida que explique por qué cree que esta solución específica resolverá el problema que identificó. Debe destacar cómo espera que cada solución contribuya a mejorar la calidad y la eficiencia del proceso de desarrollo.

**Plan de Implementación:** Diseñe un plan detallado para llevar a cabo las soluciones que ha propuesto. Este plan debe incluir un cronograma claro, las responsabilidades de cada miembro del equipo y los recursos necesarios para implementar las mejoras.

**Punto de Reflexión:**

¿Cómo considera usted que la optimización del proceso de desarrollo de software, según las soluciones propuestas, puede tener un impacto positivo en el éxito de un proyecto y en la calidad del producto final?

**UNIDAD 2 – Arquitectura del Software**

**Optimización de la Arquitectura de Software:** Aplicación de Estilos Arquitectónicos y Patrones de Arquitectura"

**Descripción:**

El objetivo de este ejercicio es aplicar sus conocimientos de Arquitectura de Software, específicamente en relación con los Estilos Arquitectónicos y los Patrones de Arquitectura. A través de un escenario ficticio, tendrá la oportunidad de analizar una arquitectura de software con desafíos y proponer soluciones para optimizarla, mejorando así la calidad y la eficiencia del sistema.

**Escenario:** En el rol de arquitecto de software, se le ha encomendado un proyecto crítico. Sin embargo, la arquitectura de software actual presenta problemas de rendimiento, escalabilidad y modularidad. Su tarea es identificar estos problemas y proponer soluciones utilizando Estilos Arquitectónicos y Patrones de Arquitectura.

**Análisis de la Arquitectura Actual:** Realice un análisis exhaustivo de la arquitectura de software actual y enumere de manera sistemática los problemas identificados. Estos problemas pueden estar relacionados con la escalabilidad, el rendimiento, la modularidad u otros aspectos clave de la arquitectura. Acompañe su análisis con un diagrama que ilustre la arquitectura actual.

**Propuesta de Mejoras:** Desarrolle propuestas concretas y detalladas para optimizar la arquitectura de software. Cada propuesta debe abordar uno o más de los problemas identificados en el paso anterior. Utilice Estilos Arquitectónicos y Patrones de Arquitectura relevantes para respaldar sus propuestas. Acompañe cada propuesta con un diagrama que ilustre cómo se implementaría la mejora en la arquitectura.

**Justificación:** Para cada mejora propuesta, proporcione una justificación sólida que explique por qué cree que esta solución específica mejorará la arquitectura y cómo abordará los problemas identificados. Destaque cómo se aplicarán Estilos Arquitectónicos y Patrones de Arquitectura en cada caso.

**Plan de Implementación:** Diseñe un plan de implementación detallado para llevar a cabo las mejoras propuestas en la arquitectura de software. Este plan debe incluir un cronograma claro, las responsabilidades de cada miembro del equipo y los recursos necesarios para implementar las mejoras. Acompañe este plan con un gráfico de Gantt que muestre las fases y los plazos de implementación.

**Punto de Reflexión:**

¿Cómo considera usted que la aplicación de Estilos Arquitectónicos y Patrones de Arquitectura puede contribuir al éxito del proyecto y mejorar la calidad del sistema?

**UNIDAD 3 – Diseño de Software**

**Exploración de Conceptos Clave en Diseño de Software**

**Descripción:**

Este ejercicio tiene como objetivo desafiarlo a profundizar en los conceptos fundamentales de diseño de software y a aplicar su comprensión de estos conceptos en un contexto práctico. A través de una serie de escenarios y preguntas, explorará conceptos clave y considerará cómo se aplican en situaciones de diseño de software.

**Escenario 1:** Abstracción: Imagine que está diseñando una aplicación de gestión de tareas. ¿Cómo aplicaría el concepto de "abstracción" en el diseño de la interfaz de usuario y la estructura de datos para asegurar que la aplicación sea fácil de entender y usar?

**Escenario 2:** Modularidad: Suponga que está trabajando en un proyecto de desarrollo de software en equipo. ¿Cómo podría aplicar el principio de "modularidad" en el diseño del código fuente para permitir una colaboración eficiente y facilitar futuras actualizaciones?

**Escenario 3:** Cohesión y Acoplamiento: Imagine que está diseñando un sistema de comercio electrónico. ¿Cómo abordaría la "cohesión" y el "acoplamiento" en la estructura de sus componentes para garantizar que el sistema sea mantenible y escalable?

**Escenario 4:** Herencia y Polimorfismo: Suponga que está diseñando una aplicación de procesamiento de documentos. ¿Cómo aplicaría los conceptos de "herencia" y "polimorfismo" en la definición de clases y métodos para gestionar varios tipos de documentos de manera eficiente?

**Escenario 5:** Patrones de Diseño: Piense en un proyecto de desarrollo de software en el que haya participado o del que tenga conocimiento. Identifique un problema de diseño específico que se haya enfrentado y explique cómo podría haber aplicado un patrón de diseño, como el patrón Singleton o el patrón Observer, para abordar ese problema de manera más efectiva.

**Punto de Reflexión:**

¿Por qué considera que comprender y aplicar estos conceptos clave en el diseño de software es esencial para la creación de sistemas de software de alta calidad y mantenibles?

**UNIDAD 4 – Metodologías Ágiles**

**"Exploración de Metodologías Ágiles en el Desarrollo de Software"**

**Descripción:**

Este ejercicio tiene como objetivo profundizar en los conceptos fundamentales de las metodologías ágiles en el desarrollo de software y a aplicar su comprensión de estos conceptos en un contexto práctico. A través de una serie de escenarios y preguntas, explorará cómo se aplican las metodologías ágiles en

**Escenario 1:** Scrum y Planificación: Suponga que está liderando un equipo de desarrollo de software y necesita entregar un producto en un plazo ajustado. ¿Cómo aplicaría los principios de Scrum para la planificación del proyecto, incluyendo la creación de sprints y la asignación de tareas?

**Escenario 2:** Kanban y Visualización del Flujo de Trabajo: Imagine que está trabajando en un proyecto de desarrollo continuo con múltiples equipos. ¿Cómo implementaría un tablero Kanban para visualizar el flujo de trabajo y gestionar las prioridades en este entorno?

**Escenario 3:** Extreme Programming (XP) y Pruebas: Suponga que está trabajando en un proyecto en el que se requiere una alta calidad y un enfoque en las pruebas. ¿Cómo aplicaría las prácticas de Extreme Programming (XP), como las pruebas unitarias y la integración continua, para garantizar la calidad del código?

**Escenario 4:** Imagine que está en un proyecto donde se necesita optimizar los recursos y eliminar el desperdicio. ¿Cómo aplicaría los principios de ***Lean Software Development*** para identificar y reducir actividades que no agregan valor al proceso de desarrollo?

**Escenario 5:** Retrospectivas y Mejora Continua: Piense en un proyecto de desarrollo ágil en el que haya participado o del que tenga conocimiento. ¿Cómo podría haber utilizado las reuniones de retrospectiva para identificar áreas de mejora y promover la mejora continua en el equipo?

Punto de Reflexión:

¿Por qué considera que la adopción de metodologías ágiles es beneficiosa en el desarrollo de software y cómo pueden contribuir a la entrega de productos de alta calidad y a la satisfacción del cliente?

* **Para realizar los diagramas utilizar la siguiente herramienta: https://app.diagrams.net/, también conocida como Draw.io.**
* **El trabajo practico debe ser entregado en un presentación archivo fomato ppt**