Lista de tareas

Instrucciones para los estudiantes

- **Exposición oral**: Cada estudiante tendrá **15 minutos** para presentar su proyecto. Debe enfocarse en los aspectos más relevantes, las decisiones técnicas y las soluciones a los desafíos encontrados.
- **Demostración práctica**: Se espera una demostración en vivo del proyecto, mostrando su funcionalidad y características principales.
- **Repositorio del proyecto**: Se debe proporcionar la **URL del repositorio** (por ejemplo, GitHub) donde se encuentra el código fuente del proyecto. El repositorio debe estar organizado y contener un **README** con instrucciones para ejecutar el proyecto.
- **Comunicación técnica**: Durante la presentación, se valorará el uso correcto de terminología técnica y la capacidad para explicar conceptos complejos de manera clara y precisa.
- **Preparación**: Se recomienda ensayar la presentación y asegurarse de cubrir todos los puntos relevantes dentro del tiempo asignado.

Los estudiantes deben escoger un solo proyecto: Entrega 21 de octubre 2024 a las 4pm.

Proyecto 1: Desarrollo de una librería de validación de datos en TypeScript

Descripción: Crear una librería modular y reusable para la validación de datos de entrada en aplicaciones. La librería debe manejar diferentes tipos de validaciones (números, cadenas, fechas, objetos, etc.).

Requisitos:

- Aplicar principios SOLID y patrones de diseño como Strategy y Factory.

- Utilizar genéricos y utilitarios de tipo en TypeScript para tipar las validaciones.

- Escribir documentación detallada utilizando estándares como JSDoc o TypeDoc.

- Implementar pruebas unitarias exhaustivas con Jest siguiendo TDD.

- Gestionar el proyecto utilizando Git y seguir prácticas de branching.

Rúbrica de evaluación (20 puntos):

1.Implementación técnica (8 puntos)

- Aplicación correcta de principios SOLID. (2 puntos)

- Uso adecuado de patrones de diseño (Strategy, Factory). (2 puntos)

- Implementación de genéricos y tipos avanzados en TypeScript. (2 puntos)

- Modularidad y reutilización del código. (2 puntos)

2.Pruebas y TDD (4 puntos)

- Cobertura de pruebas unitarias con Jest. (2 puntos)
- Evidencia de práctica de TDD durante el desarrollo. (2 puntos)

3.Documentación (2 puntos)

- Calidad y claridad de la documentación técnica. (1 punto)
- Uso de estándares como JSDoc o TypeDoc. (1 punto)

4.Gestión del proyecto (2 puntos)

- Uso efectivo de Git y prácticas de branching. (1 punto)
- Organización y claridad en los commits y mensajes. (1 punto)

5.Presentación y comunicación técnica (4 puntos)

- Uso de terminología técnica precisa durante la exposición. (2 puntos)
- Capacidad para explicar y justificar decisiones de diseño y técnicas utilizadas. (2 puntos)

Proyecto 2: Implementación de un servidor RESTful con Node.js y TypeScript

Descripción: Desarrollar un servidor que exponga una API RESTful para la gestión de recursos (por ejemplo, usuarios, productos).

Requisitos:

- Diseñar la arquitectura siguiendo el patrón MVC o Clean Architecture.
- Utilizar async-await y promesas para operaciones asíncronas.
- Implementar middleware para manejo de errores y autenticación.
- Documentar la API utilizando Swagger o API Blueprint.
- Escribir pruebas unitarias y de integración con Jest y Supertest.

Rúbrica de evaluación (20 puntos):

1.Diseño de arquitectura (6 puntos)

- Implementación del patrón MVC o Clean Architecture. (3 puntos)
- Separación adecuada de capas y responsabilidades. (3 puntos)

2.Funcionalidad y código (6 puntos)

- Uso correcto de async-await y promesas. (2 puntos)

- Implementación de middleware (errores, autenticación). (2 puntos)
- Calidad y legibilidad del código. (2 puntos)

3.Pruebas (3 puntos)

- Pruebas unitarias con Jest. (1.5 puntos)
- Pruebas de integración con Supertest. (1.5 puntos)

4.Documentación de la API (2 puntos)

- Uso de Swagger o API Blueprint. (1 punto)
- Claridad y completitud de la documentación. (1 punto)

5.Gestión del Proyecto (1 punto)

- Uso de Git y manejo de versiones. (1 punto)

6.Presentación y comunicación técnica (2 puntos)

- Explicación clara de la arquitectura y decisiones técnicas. (1 punto)
- Uso adecuado de terminología técnica en la exposición. (1 punto)

Proyecto 3: Desarrollo de una aplicación React con Arquitectura Flux o Redux

Descripción: Crear una aplicación web compleja que gestione estado global utilizando Redux o Context API.

Requisitos:

- Diseñar la arquitectura del front-end siguiendo principios de ingeniería de software.
- Implementar componentes funcionales y Hooks avanzados.
- Aplicar tipado estricto con TypeScript en todo el proyecto.
- Escribir pruebas unitarias para componentes y lógica de negocio con Jest y React Testing Library.
- Documentar decisiones arquitectónicas y patrones utilizados.

Rúbrica de evaluación (20 puntos):

1.Arquitectura y diseño (6 puntos)

- Implementación correcta de Redux o Context API. (3 puntos)
- Organización del proyecto y modularidad. (3 puntos)

2.Implementación técnica (6 puntos)

- Uso de componentes funcionales y Hooks avanzados. (2 puntos)
- Tipado estricto con TypeScript. (2 puntos)
- Calidad del código y buenas prácticas. (2 puntos)

3.Pruebas (3 puntos)

- Pruebas unitarias con Jest y React Testing Library. (3 puntos)

4.Documentación (2 puntos)

- Documentación de decisiones arquitectónicas. (1 punto)
- Uso de comentarios y documentación en el código. (1 punto)

5.Gestión del proyecto (1 punto)

- Uso efectivo de Git y manejo de ramas. (1 punto)

6.Presentación y comunicación técnica (2 puntos)

- Capacidad para explicar el flujo de datos y estado en la aplicación. (1 punto)
- Uso correcto de terminología técnica. (1 punto)

Proyecto 4: Diseño y desarrollo de un módulo de autenticación y autorización

Descripción: Implementar un sistema completo de autenticación y autorización para una aplicación web.

Requisitos:

- Utilizar JWT (JSON Web Tokens) para gestionar sesiones.
- Aplicar patrones de diseño como Facade y Singleton.
- Implementar control de acceso basado en roles (RBAC).

- Escribir documentación técnica y guías de integración.

- Realizar pruebas de seguridad y escribir pruebas unitarias.

Rúbrica de evaluación (20 puntos):

1.Implementación de Autenticación y Autorización (8 puntos)

- Uso correcto de JWT para sesiones. (3 puntos)
- Implementación de RBAC. (3 puntos)
- Aplicación de patrones de diseño (Facade, Singleton). (2 puntos)

2.Pruebas y seguridad (4 puntos)

- Pruebas unitarias y de seguridad realizadas. (2 puntos)
- Manejo adecuado de errores y vulnerabilidades comunes. (2 puntos)

3.Documentación (3 puntos)

- Documentación técnica detallada. (2 puntos)
- Guías de integración claras. (1 punto)

4.Gestión del proyecto (1 punto)

- Uso de Git y organización del repositorio. (1 punto)

5.Presentación y comunicación técnica (4 puntos)

- Explicación de los mecanismos de seguridad implementados. (2 puntos)
- Uso de terminología específica de seguridad y autenticación. (2 puntos)

Proyecto 5: Creación de un algoritmo de planificación de rutas optimizadas

Descripción: Desarrollar un algoritmo que calcule la ruta óptima entre múltiples puntos (similar al problema del viajante).

Requisitos:

- Implementar el algoritmo utilizando técnicas de programación como recursión y backtracking.
- Aplicar principios de POO y patrones de diseño.
- Analizar y documentar la complejidad temporal y espacial del algoritmo.
- Escribir pruebas unitarias para validar el correcto funcionamiento.
- Documentar el diseño y las decisiones tomadas.

Rúbrica de evaluación (20 puntos):

1.Implementación del algoritmo (8 puntos)
  - Correctitud del algoritmo y resultados. (4 puntos)
  - Uso de técnicas como recursión y backtracking. (2 puntos)
  - Aplicación de POO y patrones de diseño. (2 puntos)

2.Análisis de complejidad (4 puntos)
  - Cálculo y explicación de la complejidad temporal. (2 puntos)
  - Cálculo y explicación de la complejidad espacial. (2 puntos)

3.Pruebas (2 puntos)
  - Pruebas unitarias exhaustivas. (2 puntos)

4.Documentación (2 puntos)
  - Documentación del diseño y decisiones tomadas. (2 puntos)

5.Gestión del proyecto (1 punto)
  - Uso adecuado de control de versiones. (1 punto)

6.Presentación y comunicación técnica (3 puntos)
  - Capacidad para explicar el funcionamiento del algoritmo. (1.5 puntos)
  - Uso de terminología técnica precisa en algoritmos y estructuras de datos. (1.5 puntos)

Proyecto 6: Refactorización y mejora de código Legacy

Descripción: Proporcionar a los estudiantes un código base con problemas de mantenibilidad y rendimiento para que lo analicen y mejoren.

Requisitos:

- Identificar code smells y aplicar refactorización utilizando principios SOLID.
- Mejorar el rendimiento aplicando optimizaciones de código.
- Añadir pruebas unitarias donde falten y asegurar que todas pasen.
- Documentar los cambios realizados y justificar las decisiones.
- Utilizar herramientas de análisis estático de código.

Rúbrica de evaluación (20 puntos):

1.Análisis y refactorización (8 puntos)

- Identificación correcta de code smells. (2 puntos)
- Aplicación de principios SOLID en la refactorización. (4 puntos)
- Mejoras en la mantenibilidad del código. (2 puntos)

2.Optimización de rendimiento (4 puntos)

- Implementación de optimizaciones efectivas. (2 puntos)
- Evidencia de mejora en el rendimiento. (2 puntos)

3.Pruebas (2 puntos)

- Adición y corrección de pruebas unitarias. (2 puntos)

4.Documentación (2 puntos)

- Registro detallado de cambios y justificaciones. (2 puntos)

5.Herramientas de análisis (1 punto)

- Uso de herramientas de análisis estático y reporte de resultados. (1 punto)

6.Presentación y comunicación técnica (3 puntos)

- Explicación clara de los problemas encontrados y soluciones aplicadas. (1.5 puntos)
- Uso adecuado de terminología en refactorización y optimización. (1.5 puntos)

Proyecto 7: Desarrollo de un cliente HTTP modular y extensible

Descripción: Crear un cliente HTTP que permita realizar peticiones GET, POST, PUT y DELETE, y que sea fácilmente extensible.

Requisitos:

- Utilizar fetch, promesas y async-await.
- Implementar patrones de diseño como Builder y Adapter.
- Soportar middleware para logging, autenticación y manejo de errores.
- Escribir documentación y ejemplos de uso claros.
- Implementar pruebas unitarias y de integración.

Rúbrica de evaluación (20 puntos):

1.Implementación técnica (8 puntos)

- Soporte completo de métodos HTTP (GET, POST, PUT, DELETE). (2 puntos)
- Uso de fetch, promesas y async-await. (2 puntos)
- Implementación de patrones de diseño (Builder, Adapter). (2 puntos)
- Extensibilidad y modularidad del cliente. (2 puntos)

2.Middleware y funcionalidades adicionales (4 puntos)

- Implementación de middleware para logging, autenticación y errores. (4 puntos)

3.Pruebas (2 puntos)

- Pruebas unitarias y de integración efectivas. (2 puntos)

4.Documentación (2 puntos)

- Calidad de la documentación y claridad en los ejemplos de uso. (2 puntos)

5.Gestión del proyecto (1 punto)

- Uso de Git y organización del código. (1 punto)

6.Presentación y comunicación técnica (3 puntos)

- Capacidad para explicar la arquitectura del cliente HTTP. (1.5 puntos)
- Uso preciso de terminología relacionada con HTTP y patrones de diseño. (1.5 puntos)

Proyecto 8: Implementación de un sistema de Plugins para una aplicación web

Descripción: Desarrollar un sistema que permita añadir funcionalidades a una aplicación web mediante plugins.

Requisitos:

- Aplicar el patrón de diseño Observer y Mediator.
- Permitir la carga dinámica de módulos utilizando import() dinámico.
- Garantizar la seguridad y aislamiento de los plugins.
- Documentar cómo crear e integrar nuevos plugins.
- Escribir pruebas unitarias y realizar pruebas de integración.

Rúbrica de evaluación (20 puntos):

1.Diseño e implementación del sistema de plugins (8 puntos)

- Uso de patrones Observer y Mediator. (4 puntos)
- Implementación de carga dinámica con import(). (2 puntos)

- Seguridad y aislamiento de plugins. (2 puntos)

2.Documentación y extensibilidad (4 puntos)

- Guías claras para crear e integrar nuevos plugins. (2 puntos)
- Documentación de la API y ejemplos. (2 puntos)

3.Pruebas (2 puntos)

- Pruebas unitarias y de integración. (2 puntos)

4.Gestión del proyecto (1 punto)

- Uso efectivo de control de versiones. (1 punto)

5.Presentación y comunicación técnica (5 puntos)

- Explicación de la arquitectura del sistema de plugins. (2 puntos)
- Demostración práctica de añadir y utilizar un plugin. (2 puntos)
- Uso adecuado de terminología técnica en la exposición. (1 punto)

Proyecto 9: Construcción de un analizador de código estático

Descripción: Crear una herramienta que analice el código fuente en busca de patrones específicos o posibles errores.

Requisitos:

- Utilizar técnicas de parsing y AST (Abstract Syntax Tree).
- Implementar reglas personalizadas de análisis.
- Generar reportes detallados de los hallazgos.
- Escribir documentación y guías de uso.
- Implementar pruebas para validar el correcto funcionamiento del analizador.

Rúbrica de evaluación (20 puntos):

1.Implementación técnica (8 puntos)

- Correcta construcción y manipulación del AST. (4 puntos)
- Implementación de reglas de análisis personalizadas. (4 puntos)

2.Funcionalidad y usabilidad (4 puntos)

- Generación de reportes claros y detallados. (2 puntos)
- Facilidad de uso y configuración de la herramienta. (2 puntos)

3.Pruebas (2 puntos)

- Pruebas que validen las reglas y el funcionamiento general. (2 puntos)

4.Documentación (3 puntos)

- Manual de usuario y guías de uso. (2 puntos)
- Documentación técnica interna. (1 punto)

5.Gestión del proyecto (1 punto)

- Organización y uso de control de versiones. (1 punto)

6.Presentación y comunicación técnica (2 puntos)

- Capacidad para explicar las técnicas de parsing y uso de AST. (1 punto)
- Uso de terminología específica en análisis estático de código. (1 punto)

Proyecto 10: Desarrollo de un sistema de cacheo en memoria

Descripción: Implementar un sistema de caché en memoria para optimizar el acceso a datos frecuentes.

Requisitos:

- Aplicar patrones de diseño como Singleton y Proxy.

- Soportar estrategias de expiración como LRU (Least Recently Used).
- Asegurar la integridad y consistencia de los datos cacheados.
- Documentar la API y proporcionar ejemplos de uso.
- Escribir pruebas unitarias y pruebas de rendimiento.

Rúbrica de evaluación (20 puntos):

1.Implementación del sistema de caché (8 puntos)
  - Correcta implementación de almacenamiento en memoria. (2 puntos)
  - Uso de patrones Singleton y Proxy. (2 puntos)
  - Implementación de estrategias de expiración (LRU). (4 puntos)

2.Integridad y consistencia (3 puntos)
  - Manejo adecuado de concurrencia y sincronización. (2 puntos)
  - Asegurar la consistencia de los datos. (1 punto)

3.Pruebas (3 puntos)
  - Pruebas unitarias que cubran casos clave. (1.5 puntos)
  - Pruebas de rendimiento con resultados documentados. (1.5 puntos)

4.Documentación (2 puntos)
  - API clara y ejemplos de uso. (2 puntos)

5.Gestión del proyecto (1 punto)
  - Uso de Git y prácticas de buenas prácticas. (1 punto)

6.Presentación y comunicación técnica (3 puntos)
  - Explicación del funcionamiento interno del caché. (1.5 puntos)
  - Uso de terminología técnica relacionada con sistemas de caché y patrones de diseño. (1.5 puntos)

Proyecto 11: Implementación de integración continua y despliegue continuo (CI/CD)

Descripción: Configurar un pipeline de CI/CD para automatizar pruebas, construcción y despliegue de una aplicación.

Requisitos:

- Utilizar herramientas como Jenkins, GitLab CI/CD o GitHub Actions.
- Implementar scripts para automatizar tareas comunes.
- Asegurar que el pipeline incluya etapas de prueba, análisis estático y despliegue.
- Documentar el proceso y las herramientas utilizadas.
- Realizar una demostración del pipeline en funcionamiento.

Rúbrica de evaluación (20 puntos):

1.Configuración del Pipeline (8 puntos)
  - Implementación correcta de las etapas de CI/CD. (4 puntos)
  - Uso de herramientas adecuadas (Jenkins, GitLab CI/CD, GitHub Actions). (2 puntos)
  - Automatización efectiva de pruebas y despliegue. (2 puntos)

2.Automatización y scripts (4 puntos)
  - Calidad y eficiencia de los scripts implementados. (2 puntos)
  - Reusabilidad y claridad del código de automatización. (2 puntos)

3.Documentación (3 puntos)
  - Documentación detallada del proceso de CI/CD. (2 puntos)
  - Instrucciones para reproducir el entorno. (1 punto)

4.Demostración práctica (2 puntos)

- Presentación efectiva del pipeline en funcionamiento. (2 puntos)

5.Gestión del proyecto (1 punto)

- Uso de control de versiones y buenas prácticas en Git. (1 punto)

6.Presentación y comunicación técnica (2 puntos)

- Capacidad para explicar el flujo de CI/CD y las herramientas utilizadas. (1 punto)
- Uso de terminología técnica precisa en DevOps y automatización. (1 punto)