G100 Desarrollos, Plaza de la Paz,

Puerto Interior #102, Silao, Gto.

**Descripción**

**Nombre de la Plataforma:** TaskFlow.

TaskFlow es una plataforma de gestión de tareas y proyectos diseñada para equipos que buscan optimizar su flujo de trabajo y colaboración en tiempo real. Ofrece una interfaz intuitiva que permite a los usuarios organizar proyectos, establecer plazos y seguir el progreso de las tareas. TaskFlow se adapta a una variedad de metodologías de trabajo y está diseñada para escalar conforme crecen los equipos y sus necesidades.

**Especificación**

**Usuarios Esperados:**

1. **Profesionales de Proyecto:** Gerentes y líderes de equipo que necesitan una herramienta para asignar tareas y monitorizar el progreso del equipo.
2. **Miembros del Equipo:** Individuos que colaboran en proyectos y necesitan gestionar sus tareas diarias y marcar tareas como completadas.

**Funcionalidades y Servicios:**

1. **Creación y asignación de tareas** con fechas de vencimiento y prioridades. (del líder a los miembros).
2. **Funciones de seguimiento del tiempo** para monitorear las horas dedicadas a cada tarea. (permite calcular la fecha de inicio con la fecha fin para calcular el tiempo invertido).
3. **Análisis de productividad** y generación de informes para medir la eficiencia del equipo. (ver quien tiene más tareas como completadas).

**Transacciones**: Aunque TaskFlow no maneja transacciones monetarias directas, procesa transacciones de datos como la creación de tareas, actualizaciones de estado, y registros de tiempo. Cada acción dentro de la plataforma se registra y almacena de forma segura para mantener un historial de la actividad del proyecto.

**Requerimientos**

1. **Normalización y Estructura de Bases de Datos:**

En el modelo de la BD se deben aplicar las reglas de normalización de bases de datos, hasta 3NF para minimizar la redundancia. También se debe considerar la integridad referencial, el uso correcto de claves primarias y foráneas, así como la implementación de índices para optimizar la recuperación de datos.

1. **Principios de Programación Orientada a Objetos:**

Se debe considerar la aplicación de los cuatro principios fundamentales de POO: encapsulamiento, abstracción, herencia y polimorfismo. Contar con una adecuada estructura de las clases y objetos para confirmar la correcta aplicación de estos principios, el uso de interfaces y clases abstractas, y la implementación de patrones de diseño para resolver problemas comunes, así como empaquetamiento de las clases en las capas adecuadas.

1. **Implementación de Spring Boot:**

Se debe considerar una adecuada configuración y estructura del proyecto Spring Boot, incluyendo el uso de anotaciones como @RestController, @Service, @Repository, y la separación de responsabilidades. Así como, la configuración de los beans y el uso de diferentes capas como controladores, servicios, y acceso a datos.

1. **Manejo de Excepciones y Validación de Datos:**

Es necesario considerar una estrategia de manejo de excepciones globales con @ControllerAdvice o @RestControllerAdvice y la implementación de validaciones de entrada usando @Valid y la anotación @Validator. Así como, la cobertura de casos de error y la claridad de los mensajes de error proporcionados al cliente.

1. **Cobertura de Pruebas Unitarias:**

Se deben generar las pruebas unitarias con JUnit para garantizar que cubran una amplia gama de casos de uso. Comprobar la implementación de Mockito para simular dependencias. Las pruebas deben ser capaces de ejecutarse de manera independiente y en cualquier entorno.

1. **Seguridad con Spring Security y JWT:**

Considerar la configuración de Spring Security para la autenticación y autorización, y cómo se generan, distribuyen y validan los JWT. Verificar la seguridad de las rutas, el cifrado de contraseñas, y la gestión de permisos para diferentes roles de usuarios.

1. **Uso y Manejo de Git:**

Considerar el uso efectivo de Git, incluyendo la estructura de ramas, estrategias de merge, y resolución de conflictos. Comprobar el uso de commits semánticos, mensajes descriptivos, y la aplicación de pull requests para revisión de código antes de la integración en la rama principal.

**Evaluación**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descriptor** | **Porcentaje** | **Cumplimiento** |
| **1** | Estructura de la BD | 10% |  |
| **2** | POO | 10% |  |
| **3** | SpringBoot | 40% |  |
| **4** | Excepciones | 10% |  |
| **5** | Pruebas Unitarias | 10% |  |
| **6** | Seguridad | 10% |  |
| **7** | GIT | 10% |  |
| **TOTAL** | | **100%** |  |