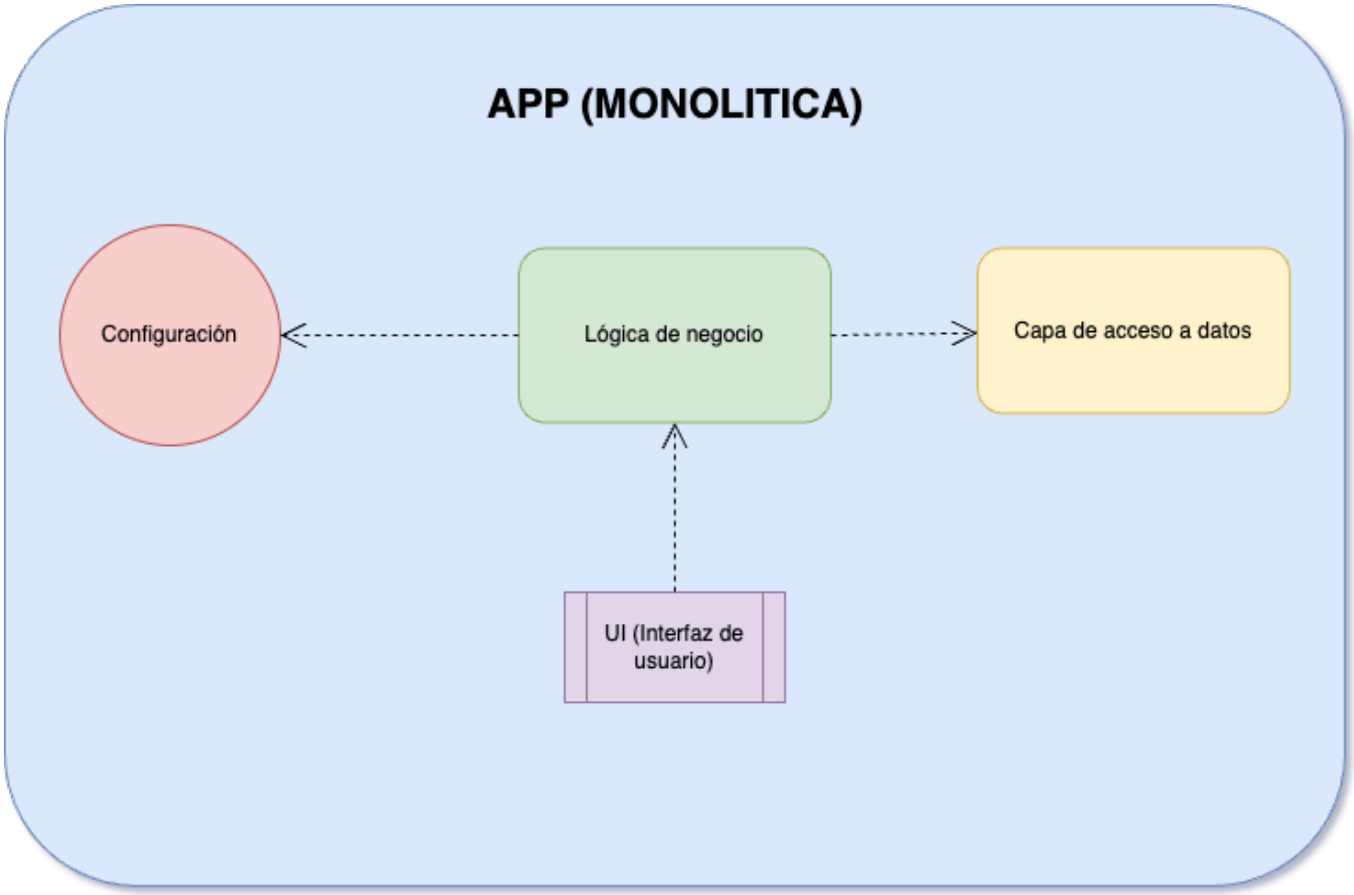
Explorando el Patrón de Arquitectura Monolítica

La arquitectura monolítica o monolito es un patrón de arquitectura bastante popular e importante, a pesar de ser bastante antiguo aún sigue siendo vigente, por su flexibilidad es común combinarlo con otros patrones de arquitectura como los microservicios.

Definir este patrón de arquitectura es simple, en un monolito todas las funcionalidades del sistema se desarrollan y despliegan en una misma unidad, todo lo contrario a lo que se hace en los microservicios.

Por ejemplo si nos vamos a http://start.spring.io y generamos un proyecto siguiendo la configuración por defecto al descargarlo nos encontraremos con un monolito.

¿Cómo se ve un monolito?



La imagen anterior muestra lo que es un monolito tradicional que utiliza una arquitectura en capas (muy común tanto en monolitos como en microservicios), lo crucial es que podemos observar como todo está contenido dentro de la misma aplicación, lo que genera una dependencia mutua, esta dependencia puede ser un problema para garantizar el funcionamiento de partes críticas de la aplicación.

Sin embargo en aplicaciones de tamaño pequeño-mediano puede ser el modelo ya que es fácil de mantener y fácil de desplegar.

Ventajas de un monolito

1. Simplicidad

Como lo mencioné en puntos anteriores, la arquitectura monolítica es más fácil de desarrollar y desplegar debido a su simplicidad. Todos los componentes se encuentran en un solo lugar, lo que facilita la gestión del código.

2. Rendimiento

Al estar todos los componentes en un solo proceso, las llamadas entre ellos son rápidas y eficientes, sin la sobrecarga de las comunicaciones entre servicios, de nuevo quiero insistir que esto es lo ideal para la mayoría de aplicaciones en el mercado.

3. Desarrollo Rápido

Para proyectos pequeños, medianos o con equipos reducidos, el desarrollo monolítico permite un progreso rápido sin la necesidad de gestionar múltiples servicios, además de la reducción de costos en la nube comparado con otros patrones de arquitectura.

4. Facilidad de Pruebas

Las pruebas integradas son más sencillas, ya que todos los componentes están disponibles en el mismo entorno.

Desventajas de un monolito

1. Escalabilidad Limitada

La arquitectura monolítica puede ser difícil de escalar. A medida que el sistema crece, es

posible que todo el sistema necesite ser escalado en lugar de solo los componentes que lo requieren.

Debemos ser cuidadosos con este punto ya que si bien puede ser una tarea compleja escalar un monolito, muchas veces no se necesita, muchísimos proyectos sobretodo aquellos que son para uso interno de una compañía no es necesario escalarlos más allá de la definición inicial.

2. Complejidad de Mantenimiento

Cuando un monolito alcanza un tamaño demasiado grande puede convertirse en un dolor de cabeza mantenerlo, sobretodo si el diseño de clases en su interior es cuestionable, como resultado podríamos tener un proyecto con demasiado código, difícil de entender y que se convierte en un dolor de cabeza para cualquier que lo quiera modificar.

3. Despliegue Riesgoso

Dependiendo del sistema y del lenguaje utilizado, el cambiar una sola línea de código puede desembocar en un montón de tareas consecuencia de un nuevo despliegue y todo nuevo despliegue tiene aunque sea un riesgo mínimo de salir mal y tumbar la aplicación por completo.

4. Falta de Flexibilidad Tecnológica

Aunque crear módulos parcialmente independientes en un monolito es posible, no es una muy buena práctica, por ende implementar nuevas tecnologías o empezar a hacer migraciones parciales hacía nuevos frameworks puede resultar difícil o hasta imposible dependiendo del caso.

Cuándo Utilizar una Arquitectura Monolítica

1. Proyectos pequeños

Para aplicaciones con funcionalidades limitadas y un equipo pequeño, la simplicidad de una arquitectura monolítica puede ser beneficiosa o incluso ideal.

2. Proyectos con escasez de tiempo (Bajo Time-to-Market)

Si necesitas lanzar un producto rápidamente y con menos recursos, una arquitectura

monolítica permite un desarrollo y despliegue más rápido, cuando digo menos recursos me refiero en todos los sentidos, pocos recursos en desarrollo y pocos recursos para mantener la aplicación operativa quizás en la nube.

3. Fases inicial de un proyecto grande

En las etapas tempranas de un proyecto, cuando las funcionalidades y requisitos no están completamente definidos, una arquitectura monolítica permite iterar rápidamente y realizar cambios frecuentes, además es importante tener en cuenta que un monolito después puede ser parte de un conjunto de monolitos que son microservicios bellamente orquestados por algún orquestador.

Buenas prácticas en la Arquitectura Monolítica

1. Modularización

Tal y como dije antes aunque modular un monolito pueda sonar extraño, es posible hacerlo. Aunque siempre existirá un grado de dependencia entre las partes, es recomendable intentar modular las partes de un monolito, ya sea por funcionalidad, rol, o cualquier otro esquema que se defina.

2. Pruebas Automatizadas

Implementar pruebas automáticas puede ser un salvavidas que reduzca casi a cero el riesgo de cometer un error grave en un despliegue.

3. Documentación útil

Crear una documentación que resulte verdaderamente útil para un desarrollador, no basta con diagramas, pueden acompañarlo con alguna wiki, videos, etc.

4. Monitoreo y Logging

Estás dos son vitales, es necesario tener logs bien estructurados e informativos además de que debe ser salir comprobar el estado de la aplicación ambas cosas dependen al 100% de un correcto manejo de excepciones y yo tengo un curso gratuito en YouTube que les puede servir.

Curso manejo de excepciones