**Decisiones técnicas**

* **Angular:** Se seleccionó por su capacidad para construir interfaces de usuario dinámicas y modulares. Su estructura basada en componentes permite una fácil reutilización y escalabilidad del código, ideal para aplicaciones de gran tamaño o con necesidades de actualización frecuente.
* **.NET:** Se eligió debido a su robustez y soporte para la creación de APIs RESTful eficientes y seguras. Su ecosistema de herramientas permite una integración fluida con bases de datos, sistemas de autenticación y servicios externos, facilitando el desarrollo ágil y escalable.
* **SQL Server:** Fue elegido por su capacidad para manejar grandes volúmenes de datos de manera eficiente. SQL Server no solo es altamente confiable, sino que también cuenta con soporte avanzado para la gestión de transacciones, recuperación de desastres y escalabilidad, lo cual es crucial para el almacenamiento de documentos como PDFs.
* **Librerías:**
  + **SweetAlert:** Se integró para mejorar la experiencia de usuario, proporcionando notificaciones visualmente atractivas y fáciles de personalizar.
  + **iText7 y BouncyCastle:** Elegidas específicamente para el manejo avanzado de PDFs y la implementación de firmas electrónicas. iText7 permite manipular documentos PDF de manera eficiente, mientras que BouncyCastle ofrece las herramientas criptográficas necesarias para garantizar la seguridad y autenticidad de los documentos firmados.

**Patrones de Diseño y Técnicas Utilizadas**

* **Arquitectura Hexagonal:** Esta arquitectura se adoptó para desacoplar las capas internas del sistema (dominio) de las interfaces externas (por ejemplo, bases de datos, servicios web). Esto facilita la adaptabilidad y permite que los cambios en la tecnología de almacenamiento o el frontend no afecten la lógica central de la aplicación.
* **Inyección de dependencias:** Utilizada para gestionar de manera eficiente las dependencias entre los diferentes componentes del sistema, permitiendo la creación de componentes más fácilmente testeables y modulares. Esta técnica también promueve la reutilización de código y facilita las pruebas unitarias.
* **Patrón Repositorio:** Ayuda a encapsular la lógica de acceso a los datos, promoviendo un código más limpio y organizado. Además, este patrón mejora la capacidad de cambiar el origen de datos sin afectar la lógica del negocio, permitiendo que la aplicación sea más mantenible y flexible.

**Requisitos adicionales**

* **Autenticación:** Un sistema de autenticación permitiría restringir el acceso a las operaciones sensibles como la carga y descarga de documentos, asegurando que solo usuarios autorizados puedan realizarlas.
* **Versionado de documentos:** En muchos casos, los documentos PDF pueden necesitar diferentes versiones a lo largo del tiempo, por lo que implementar un sistema de versionado permitiría a los usuarios rastrear los cambios realizados en un documento específico.
* **Sistema de Notificaciones**: Para mejorar la experiencia del usuario, un sistema de notificaciones podría alertar a los usuarios cuando se cargue, firme o actualice un documento, proporcionando actualizaciones en tiempo real sobre el estado de los archivos.

Estos requisitos adicionales podrían adaptarse mediante la incorporación de microservicios para la autenticación, almacenamiento de archivos en la nube para mayor escalabilidad y un sistema de mensajería para manejar las notificaciones de estado.