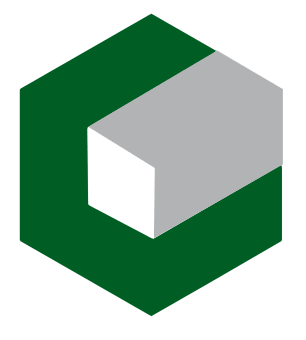
Universidad Tecnológica de La Habana “José Antonio Echeverría”

Facultad de Ingeniería Informática.



Arquitectura candidata del módulo de gestion de usuarios de Noswork Workspace

**Autor**: Carlos Daniel Vilaseca Illnait

**Tutores**: Dra. C. Raisa Socorro Llanes

Dra. C. Lisandra Bravo Ilisastigui

La Habana, Cuba

Enero,2024

Resumen

El trabajo a realizar tiene como objetivo principal definir la arquitectura candidata de un software para la gestión de usuarios y grupos de un Directorio Activo en la empresa Avangenio. Para ello se realizarán comparaciones en cuanto a las posibles arquitecturas presentes, además de distintas comparaciones que justifican la selección de las tecnologías.

**Palabras clave:** Noswork, arquitectura, arquitectura candidata

Abstract

The main objective of the work to be carried out is to define the candidate architecture for software for managing users and groups of an Active Directory in the Avangenio company. To do this, comparisons will be made of the possible architectures present, as well as different comparisons that justify the selection of technologies.

**Palabras clave:** Noswork, architecture, candidate architecture

Índice

[Introducción 1](#_Toc1)

[Definición de la arquitectura 3](#_Toc2)

[Stack tecnológico 4](#_Toc3)

[Directorio Activo y LDAP 4](#_Toc4)

[¿Por qué no simplemente usar una base de datos como Postgres? 4](#_Toc5)

[Framework de frontend: Sveltekit 5](#_Toc6)

[¿Por qué no otro framework como Next.js? 5](#_Toc7)

[Framework de backend: FastAPI 7](#_Toc8)

[¿Por qué no otro framework como Django? 7](#_Toc9)

[Conclusiones 10](#_Toc10)

[Referencias bibliográficas 11](#_Toc11)

[Anexos 13](#_Toc12)

Tablas y figuras

[Figura 1. Tamaño minificado de cada framework en el registro de npm 13](#_Toc1)

[Figura 2. Rendimiento editando un elemento entre 10000 elementos presentes en el DOM 13](#_Toc2)

# Introducción

Un software colaborativo es una herramienta que permite a los miembros de un equipo trabajar juntos de manera más eficiente y productiva, independientemente de su ubicación física. Estas herramientas proporcionan una plataforma en línea para compartir información, comunicarse y colaborar en tiempo real. Los softwares colaborativos suelen incluir funciones como chat, videoconferencia, edición de documentos en tiempo real y gestión de proyectos, lo que facilita el trabajo en equipo y la coordinación entre los miembros del equipo [1, 2].

Los softwares colaborativos surgen como una respuesta a la necesidad de mejorar la comunicación, el intercambio de información y el trabajo en equipo dentro de las empresas. Al tener toda la información y los recursos ubicados en un solo sitio y con cada miembro teniendo el acceso correspondiente, se aumenta la transparencia entre departamentos y se facilita el trabajo en equipo. Ejemplos de software colaborativos incluyen *Google Workspace*, *Microsoft 365* y *Zoho* [2, 3].

*NosWork Workspaces* es un ejemplo de software colaborativo que permite crear espacios de trabajo virtuales colaborativos, donde los usuarios pueden compartir archivos, comunicarse por chat o videoconferencia, y acceder a diversas aplicaciones web integradas. Esta plataforma facilita el trabajo remoto y la gestión de proyectos de forma ágil y segura [7].

Las aplicaciones colaborativas generalmente tienen un módulo integrado para la gestión de usuarios, facilitando la administración de los miembros del equipo y sus permisos dentro de la plataforma [55, 56, 57]. Esto permite a los administradores agregar, eliminar o modificar usuarios y controlar su acceso a diferentes áreas y funciones de la plataforma. Al tener una herramienta de gestión integrada, los administradores pueden asegurarse de que cada miembro del equipo tenga acceso a las herramientas y recursos necesarios para realizar su trabajo de manera efectiva. Al mismo tiempo que se protege la información confidencial y se mantiene la seguridad de la plataforma [3].

Comúnmente estas herramientas utilizan un procedimiento de autenticación que permite a los usuarios acceder a múltiples sistemas, recursos, o aplicaciones con una sola identificación base (dígase usuario y contraseña), denominado *Single Sign-On* (SSO). El SSO se suele utilizar en un contexto empresarial, cuando las aplicaciones de los usuarios las asigna y gestiona un equipo interno de tecnologías de la información (TI) [4, 5, 6]. Existen varios sistemas que implementan SSO como lo son el *Central Authentication Service* (CAS), el *Security Assertion Markup Language* (SAML) y OAuth [8, 9, 10, 13].

El Protocolo de Acceso Ligero a Directorios (LDAP por sus siglas en inglés) es un protocolo utilizado por muchos sistemas de SSO como proveedor de identidad, incluido Directorio Activode Microsoft, para acceder a la información del directorio y gestionarla [10, 11, 12, 16].

Los servicios de Directorio Activo, actúan como una base de datos centralizada para almacenar y gestionar información sobre usuarios, computadoras y otros recursos en una organización. Los administradores pueden utilizar esta base de datos para controlar el acceso a los recursos y redes, verificando la identidad de los usuarios y asignándoles permisos adecuados. [12, 16].Existen varios sistemas que implementan Directorio Activo, como *Azure Active Directory* de Microsoft, *Apache Directory Studio* y *Oracle Directory Server Enterprise Edition* [12, 14, 15]*.* Sin embargo, no todos son gratuitos, o sencillos de configurar.

Samba 4 es una de las alternativas gratuitas que implementan Directorio Activo y actúa como controlador de dominio en sistemas Unix. Es una implementación libre del protocolo de archivos compartidos de Microsoft para sistemas de tipo Unix [16, 17]. En Noswork se utiliza un servidor de Samba 4 como controlador de dominio en el directorio. Para la administración remota de este servidor se desarrolló un módulo propio, dado que las herramientas existentes como Remote Server Administration Tools (RSAT) o Webmin, no cumplen con las necesidades específicas del producto. El módulo está construido en dos partes, un *backend* llamado *AD-manager-API* y un *frontend* llamado *Admin Console*.

# Definición de la arquitectura

Dado que el flujo del sistema esta basado en capas evidentemente se presenta una **arquitectura de n capas**:

Es un patrón de diseño de software que divide un sistema en capas horizontales, donde cada capa tiene un rol específico y solo se comunica con las capas adyacentes.

La arquitectura de n capas está presente en un sistema cuando este se divide en capas lógicas independientes, cada una con una responsabilidad específica. Las capas se comunican entre sí a través de interfaces bien definidas.

Ejemplos de areas en las que se usa:

* **Aplicaciones web**: Interfaz de usuario, lógica de negocio, acceso a datos.
* Aplicaciones móviles: Presentación, lógica de negocio, acceso a datos, almacenamiento local.
* otros

Capas comunes:

* Presentación: Interfaz de usuario, interacción con el usuario.
* Lógica de negocio: Reglas y procesos del sistema.
* Acceso a datos: Interacción con la base de datos o almacenamiento.

¿Por qué no otra arquitectura?

No existe una arquitectura de software que sea perfecta para todos los sistemas. La arquitectura adecuada para un sistema depende de sus requisitos específicos. En el caso de las aplicaciones web, la arquitectura de N capas es una opción común ya que resuelve la mayor parte de los problemas comunes, especialmente los relacionados a seguridad, escalabilidad y mantenibilidad. Sin embargo, también se puede combinar con una arquitectura de microservicios para mejorar el rendimiento y/o la escalabilidad (si así se requiere y el presupuesto lo permite).

Por ejemplo si se compara con una arquitectura hexagonal la respuesta obvia es que la arquitectura hexagonal es muy compleja de implementar y llegar a ese punto.

Complejidad: La arquitectura hexagonal puede ser más compleja de implementar que otras arquitecturas, como la arquitectura de N capas. Esto se debe a que requiere un mayor conocimiento de los patrones de diseño y de las técnicas de inyección de dependencias.

Costo: La arquitectura hexagonal puede ser más costosa de implementar que otras arquitecturas. Esto se debe a que requiere un mayor esfuerzo de desarrollo y mantenimiento.

# Stack tecnológico

En el contexto del desarrollo de software, un stack tecnológico es el conjunto de tecnologías, herramientas y lenguajes de programación que se utilizan para crear un sistema o aplicación. Incluye la base de datos, los lenguajes de programación, los frameworks, las bibliotecas y otros componentes. Esta sección está centrada en la justificación de tecnologías como los frameworks y el directorio.

## Directorio Activo y LDAP

Un directorio activo es un servicio de directorio que proporciona un sistema de administración de identidades y acceso para redes informáticas. Un directorio activo puede almacenar información sobre usuarios, equipos, dispositivos y otros recursos de red.

Un directorio activo proporciona un entorno seguro para almacenar datos de identidad y acceso. El protocolo LDAPS, que es una versión segura de LDAP, utiliza el cifrado para proteger la comunicación entre el directorio activo y la aplicación web (similar a HTTP vs HTTPS).

### ¿Por qué no simplemente usar una base de datos como Postgres?

* **Requisitos de integración con otros sistemas**: Un directorio puede actuar como un proveedor de identidad (Identity Provider, IdP) para proporcionar SSO a otros sistemas empresariales.
* Requisitos de seguridad: Un directorio proporciona un entorno seguro para almacenar datos de identidad y acceso. El protocolo LDAPS, que es una versión segura de LDAP, utiliza el cifrado para proteger la comunicación entre el directorio y la aplicación web.
* Requisitos de escalabilidad: Un directorio está diseñado para ser escalable. Puede admitir un gran número de usuarios y recursos.
* Requisitos de facilidad de administración: Un directorio es fácil de administrar. Proporciona herramientas y servicios para administrar usuarios, equipos y otros recursos de red.

En particular, un directorio es la elección de preferencia para las empresas porque:

* Es un producto de terceros ampliamente utilizado y probado.
* Ofrece un conjunto completo de características de seguridad y administración.
* Es compatible con una amplia gama de sistemas empresariales.

El hecho de que un directorio actúe como IdP (*Identity Provider*) permite la integración con SSO a otros sistemas empresariales, como Gitlab, que también tienen soporte para el protocolo LDAP. Esto simplifica la administración de usuarios y accesos para los usuarios de la aplicación web.

## Framework de frontend: Sveltekit

SvelteKit es un framework de desarrollo web de código abierto que utiliza el lenguaje de programación Svelte. SvelteKit ofrece una serie de ventajas, como:

* Eficiencia: SvelteKit es un framework muy eficiente, ya que genera código JavaScript mínimo y optimizado.
* Flexibilidad: SvelteKit es un framework flexible que permite a los desarrolladores crear aplicaciones web complejas y personalizables.
* Escalabilidad: SvelteKit es un framework escalable que puede admitir aplicaciones web con un gran número de usuarios.

### ¿Por qué no otro framework como Next.js?

Sveltekit:

* Beneficios:
  + Eficiencia: SvelteKit genera código JavaScript mínimo y optimizado, lo que puede traducirse en una mejora del rendimiento de la aplicación web.
  + Curva de aprendizaje: SvelteKit tiene una curva de aprendizaje relativamente baja, lo que significa que los desarrolladores pueden aprender a usarlo rápidamente y comenzar a crear aplicaciones web complejas.
  + SSR: SvelteKit ofrece soporte integrado para SSR, lo que significa que el contenido de la página se puede renderizar en el servidor antes de que se envíe al navegador del usuario. Esto puede mejorar el rendimiento de las aplicaciones web en las que el contenido está disponible de inmediato.
  + SEO: SvelteKit también ofrece soporte para SEO, lo que puede ayudar a que las aplicaciones web sean más visibles en los motores de búsqueda. SvelteKit genera metaetiquetas y otros elementos de encabezado de SEO de forma automática. Además, SvelteKit admite el uso de archivos de mapas de sitio XML y archivos robots.txt.
* Desventajas:
* Ecosistema: SvelteKit tiene un ecosistema más pequeño que Next.js, lo que puede dificultar la búsqueda de herramientas y bibliotecas compatibles.

Next.js

* Beneficios:
  + Ecosistema: Next.js tiene un ecosistema más grande que SvelteKit, lo que facilita la búsqueda de herramientas y bibliotecas compatibles.
  + Soporte para SEO: Next.js ofrece soporte integrado para SEO, lo que puede ayudar a que las aplicaciones web sean más visibles en los motores de búsqueda.
  + Soporte para SSR: Next.js ofrece soporte integrado para SSR, lo que puede mejorar el rendimiento de las aplicaciones web en las que el contenido está disponible de inmediato.
* Desventajas:
  + Curva de aprendizaje: Next.js tiene una curva de aprendizaje más pronunciada que SvelteKit, lo que puede dificultar el aprendizaje para los desarrolladores principiantes.
  + Rendimiento: Next.js puede no ser la mejor opción para aplicaciones web que requieren un rendimiento muy alto.

Es muy complicado realizar una comparación de cada aspecto de todos los frameworks por las siguientes razones:

* Hay muchos frameworks disponibles. En la actualidad, hay cientos de frameworks de desarrollo web disponibles. Cada framework tiene sus propias características y ventajas.
* Los frameworks son complejos. Los frameworks de desarrollo web son complejos y ofrecen una amplia gama de características. Es difícil comparar todas estas características de forma exhaustiva.
* Los requisitos de las aplicaciones varían. Las aplicaciones web tienen diferentes requisitos. Un framework que es ideal para una aplicación web puede no serlo para otra.

Por estas razones, es difícil decir que un framework es mejor que otro. El mejor framework para una aplicación web específica depende de los requisitos de la aplicación y de las preferencias del equipo de desarrollo.

En última instancia, todo se reduce a si puedes resolver el problema que se tiene entre manos con ese framework. Si un framework ofrece las características que se necesita para crear la aplicación que se desea, entonces ese framework es el adecuado.

## Framework de backend: FastAPI

FastAPI es un framework de backend de Python de código abierto que se centra en la velocidad, la facilidad de uso y la escalabilidad. FastAPI ofrece una serie de ventajas, como:

Velocidad: FastAPI está diseñado para ser rápido. Utiliza un enfoque minimalista para el diseño y la implementación, lo que reduce la sobrecarga y mejora el rendimiento.

Facilidad de uso: FastAPI es fácil de aprender y usar. El código es conciso y fácil de leer, y la documentación es completa y exhaustiva.

Escalabilidad: FastAPI está diseñado para ser escalable. Puede manejar un gran número de solicitudes simultáneas sin comprometer el rendimiento.

### ¿Por qué no otro framework como Django?

FastAPI vs Django

FastAPI y Django son dos frameworks de backend de Python de código abierto que ofrecen una serie de ventajas. Sin embargo, hay algunas diferencias clave entre los dos frameworks que pueden hacer que uno sea una mejor opción que el otro para una aplicación web específica.

En el caso de una aplicación web con un directorio como capa de datos, la elección entre FastAPI y Django depende de los siguientes factores:

Rendimiento

* FastAPI: FastAPI está diseñado para ser rápido. Utiliza un enfoque minimalista para el diseño y la implementación, lo que reduce la sobrecarga y mejora el rendimiento.
* Django: Django puede ser lento, ya que utiliza una base de datos relacional y un ORM.

Facilidad de uso

* FastAPI: FastAPI es fácil de aprender y usar. El código es conciso y fácil de leer, y la documentación es completa y exhaustiva.
* Django: Django puede ser difícil de aprender y usar, ya que tiene una curva de aprendizaje pronunciada.

Escalabilidad

* FastAPI: FastAPI está diseñado para ser escalable. Puede manejar un gran número de solicitudes simultáneas sin comprometer el rendimiento.
* Django: Django puede ser escalable, pero requiere más esfuerzo que FastAPI.

Compatibilidad

* FastAPI: FastAPI es compatible con una amplia gama de herramientas y bibliotecas.
* Django: Django también es compatible con una amplia gama de herramientas y bibliotecas.

Un factor tambien que se puede tener en cuenta es que la capa de datos es un Directorio, o sea, no se estaría usando todo el potencial de Django dado que este reside en su ORM y base de datos relacional integradas.

En este contexto, FastAPI es una mejor opción dado que provee un alto rendimiento y facilidad de uso. Django puede ser una mejor opción para aplicaciones web más complejas que requieren una base de datos relacional y un ORM.

# Conclusiones

La arquitectura de n capas es una buena opción para la aplicación web propuesta, ya que ofrece una serie de ventajas, como la separación de responsabilidades, la flexibilidad y la seguridad.

El uso de un directorio como capa de datos ofrece una serie de ventajas, como la seguridad, la escalabilidad y la facilidad de administración.

El uso de SvelteKit como framework de frontend ofrece un alto rendimiento y flexibilidad.

El uso de FastAPI como framework de backend ofrece un alto rendimiento y facilidad de uso.

# Referencias bibliográficas

[1] R. Mercurio and B. Merrill, “Beginning Microsoft 365 collaboration apps: working in the Microsoft cloud”. Berkeley, CA: Apress, 2021.

[2] J. B. Keegan, “An Overview Of Collaboration Software" Sep. 18, 2019. https://www.forbes.com/sites/forbestechcouncil/2019/09/18/an-overview-of-collaboration-software/?sh=3ac85b342cee (accessed Jun. 11, 2023).

[3] Google, “Manage Workspace with Admin Dashboard - Google Workspace.” https://workspace.google.com/intl/en/products/admin/ (accessed Jun. 22, 2023).

[4] Microsoft, “Single Sign-On (SSO): Secure App Access Solutions | Microsoft Security.” https://www.microsoft.com/en-us/security/business/identity-access/azure-active-directory-single-sign-on (accessed Jun. 21, 2023).

[5] V. Beltran, “Characterization of web single sign-on protocols" IEEE Commun. Mag., vol. 54, no. 7, pp. 24–30, Jul. 2016, doi: 10.1109/MCOM.2016.7514160.

[6] E. C. Dias and R. de F. Ribeiro, “Single sign-on: an information security approach" in Proceedings of the 13th CONTECSI International Conference on Information Systems and Technology Management, 2016, vol. 13, pp. 2874–2896, doi: 10.5748/9788599693124-13CONTECSI/PS-4061.

[7] Avangenio, “Productos propios" 2023. https://avangenio.com/productos-propios/ (accessed Jun. 21, 2023).

[8] Usmanu Danfodiyo University, M. K. Hamza, H. Abubakar, Y. M. Danlami, “Identity and Access Management System: a Web-Based Approach for an Enterprise" PoS, vol. 4, no. 11, pp. 2001–2011, Nov. 2018, doi: 10.22178/pos.40-1.

[9] P. Sharma, V. K. Sihag, “Hybrid Single Sign-On Protocol for Lightweight Devices" in 2016 IEEE 6th International Conference on Advanced Computing (IACC), 2016, pp. 679–684, doi: 10.1109/IACC.2016.131.

[10] N. Naik, P. Jenkins, “A secure mobile cloud identity: criteria for effective identity and access management standards" presented at the 2016 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2016, pp. 89–90, doi: 10.1109/MobileCloud.2016.22.

[11] RedHat, “What is lightweight directory access protocol (LDAP) authentication?" Jun. 03, 2022. https://www.redhat.com/en/topics/security/what-is-ldap-authentication (accessed Jun. 21, 2023).

[12] Microsoft, “LDAP authentication with Azure Active Directory" Oct. 01, 2023. https://learn.microsoft.com/en-us/azure/active-directory/fundamentals/auth-ldap (accessed Jun. 22, 2023).

[13] J. Schwenk, “Web security and Single Sign-On protocols" in *Guide to Internet Cryptography: Security Protocols and Real-World Attack Implications*, Cham: Springer International Publishing, 2022, pp. 467–503.

[14] Apache, “Welcome to Apache Directory Studio" Jul. 24, 2021. https://directory.apache.org/studio/ (accessed Jun. 22, 2023).

[15] Oracle, “Oracle Directory Server Enterprise Edition Directory.” https://docs.oracle.com/en/cloud/paas/identity-cloud/idcsc/odsee.html (accessed Jun. 22, 2023).

[16] A. Bartlett, “Samba 4-Active Directory”, *Samba. Org., 78*, vol. 78, 2005.

[17] Samba, “What is Samba?” https://www.samba.org/samba/what\_is\_samba.html (accessed Jun. 22, 2023).

[18] T. Carpenter, “Microsoft Windows Server Administration Essentials”, 1st ed. Indianapolis, Ind: Sybex, 2011, p. 400.

[19] J. Cooper, “The Book of Webmin: Or How I Learned to Stop Worrying and Love UNIX*”*, 1st ed. San Francisco: No Starch Press, 2003, p. 312.

[21] V. S. Garófalo Jerez, “VicentGJ/AD-webmanager: A web interface for administration of Active Directory Domains, made in Python, with focus on ease of use and simplicity.” https://github.com/VicentGJ/AD-webmanager/ (accessed Jun. 22, 2023).

[22] L. Mattias , “DOM benchmark comparison of the front-end JavaScript frameworks React, Angular, Vue, and Svelte”, Åbo Akademi University, 2020.

[23] V. Fernández Palacio , “Desarrollo de gateway de seguridad en python con FastAPI”, UNIVERSIDAD DE CANTABRIA, Mayo 2022.

# Anexos

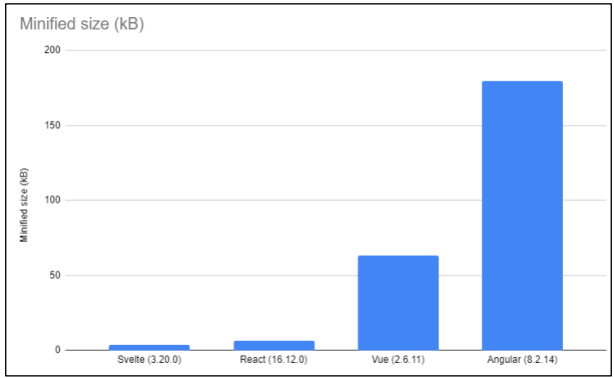


Figura 1. Tamaño minificado de cada framework en el registro de npm

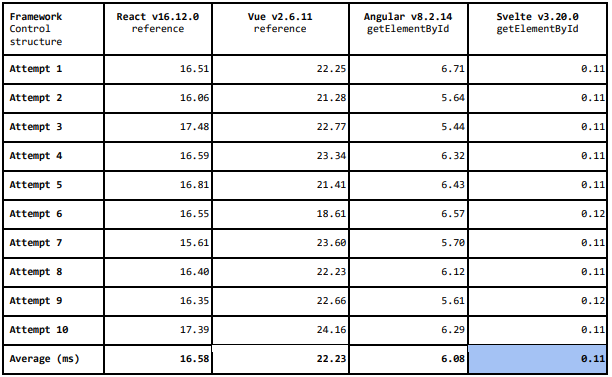


Figura 2. Rendimiento editando un elemento entre 10000 elementos presentes en el DOM