Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

TALLER 2 – MIGRACIÓN A MICROSERVICIOS
Arquitectura de Sistemas

OBJETIVO

1. Aplicar conceptos teóricos de microservicios en una implementación práctica.
2. Desarrollar habilidades de trabajo en equipo mediante la colaboración en un sistema distribuido.

CONTEXTO

La plataforma de streaming StreamFlow fue un éxito total superando toda expectativa de uso teniendo millones de visitas por minuto. Debido a esto, la Corporación Nacional del Software (CODELSOFT) ha decidido migrar el sistema a una arquitectura de microservicios y añadir nuevas funcionalidades a StreamFlow. Para ello, se enfocará inicialmente en migrar la lógica de negocio del sistema, hacia una arquitectura de microservicios, que integrará microservicios especializados para la gestión de usuarios, autenticación, facturación, administración de contenido audiovisual. Además, se añadirán nuevos microservicios de monitoreo de servicios, creación de listas de reproducción, de interacciones sociales, envío de correos y una API Gateway para la comunicación.

Según el Chief Technology Officer (CTO) de CODELSOFT, la aplicación debe permitir a los usuarios crear y administrar sus cuentas, autenticarse de manera segura mediante tokens JWT y acceder a un catálogo de videos. Los administradores podrán gestionar el contenido audiovisual —realizando operaciones como la subida, actualización y eliminación de videos—, así como administrar las suscripciones y pagos de los usuarios.

Esta etapa inicial se centrará únicamente en el back-end y la lógica de negocio. No obstante, CODELSOFT ha proporcionado su logo corporativo (Figura 1) para que los desarrolladores puedan familiarizarse con la identidad de la organización.

Figura 1: Logo ficticio de la empresa CODELSOFT.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

ÍNDICE

NUEVA ARQUITECTURA

Se migrará cada módulo de la aplicación anterior a un nuevo microservicio, además de incluir los nuevos microservicios solicitados por CODELSOFT cada microservicio tendrá su propia base de datos independiente y utilizarán **RabbitMQ** como cola de mensajería. Los microservicios con los que contará la aplicación son los siguientes:

1. **Autenticación**: Control de la autenticación, permitirá a los usuarios autenticarse de manera segura en la aplicación, se debe utilizar la técnica de **tokens blacklist** para el mantener un listado de tokens no admitidos. El servicio de autenticación utilizará una base de datos **PostgreSQL.**
2. **Facturación**: Manejo de pagos, permitirá a los administradores gestionar los pagos de los clientes en el sistema. El módulo de facturación utilizará una base de datos **MariaDB.**
3. **Videos**: Administración del contenido audiovisual, permitirá a los clientes acceder al contenido audiovisual, y a los administradores gestionarlo. El servicio de videos utilizará una base de datos **MongoDB.**
4. **Usuarios**: Gestión de cuentas, permitirá a los clientes registrarse en la aplicación y a los administradores gestionar a estos. El servicio de usuarios utilizará una base de datos a **libre elección, que no sea PostgreSQL.**
5. **Monitoreo**: Registra todas acciones y errores realizados en la aplicación, y entrega el listado de estas a los administradores de esta. El servicio de monitoreo utilizará una base de datos **MongoDB.**
6. **Listas de reproducción**: Manejo de listas de reproducción de vídeos, permitirá a los clientes crear y gestionar listas de múltiples videos. El servicio de listas de reproducción utilizará una base de datos **PostgreSQL.**
7. **Interacciones sociales**: Control de comentarios y “likes”, permitirá a los usuarios darle like a videos y dejar comentarios. El servicio de interacciones sociales utilizará una base de datos **MongoDB.**
8. **Envío de correos**: Envío de correos electrónicos de registro de usuarios y facturas, permitirá a los clientes y administradores recibir correos electrónicos. **El servicio de envío de correos no utilizará base de datos.**
9. **API Gateway**: Permitirá gestionar el acceso a los múltiples microservicios de la aplicación, actuando como único punto de entrada y verificando la autenticación en las peticiones. **La API Gateway no utilizará base de datos.**

El protocolo de comunicación entre la **API Gateway y los distintos microservicios será gRPC**, con excepción del microservicio de autenticación, el cual utilizará HTTP.

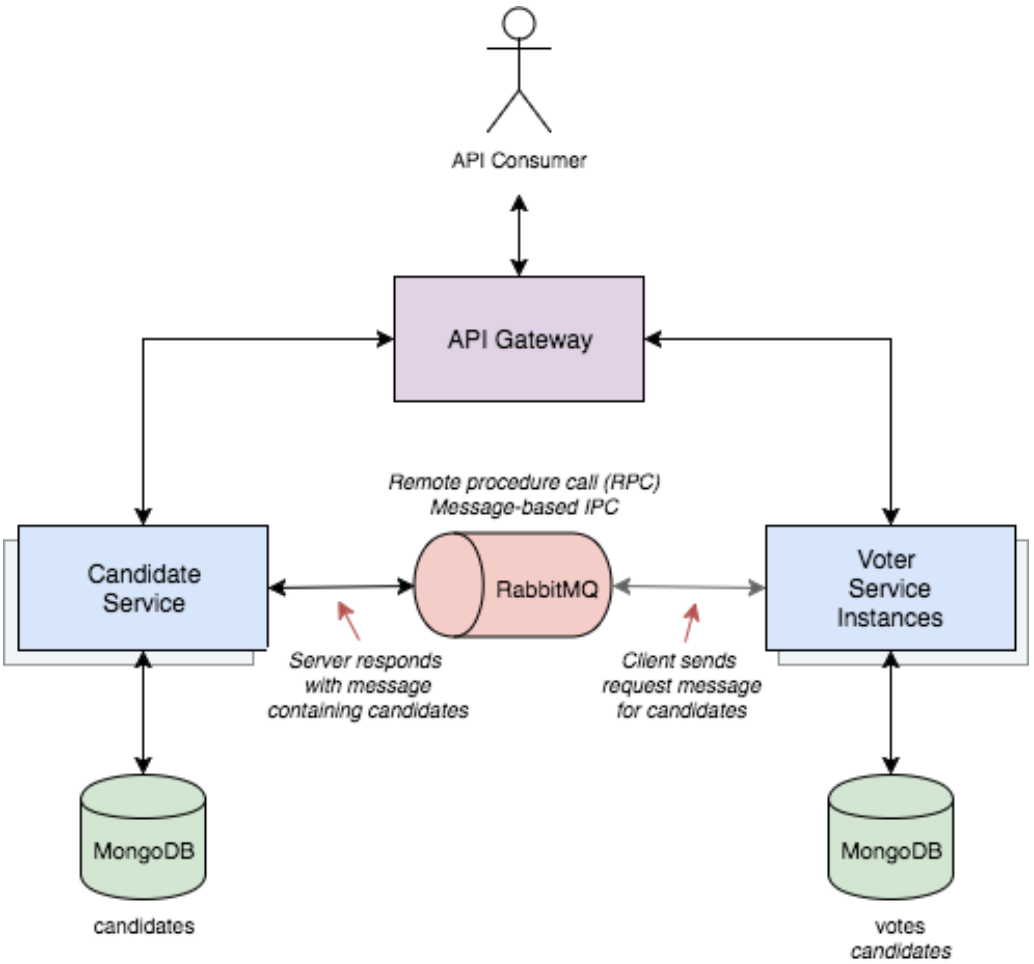
CODELSOFT no posee ninguna restricción respecto a los Frameworks que se utilicen en la aplicación.

Por lo anterior mencionado, es necesario que el Stack tecnológico que se utilizará para el desarrollo de la aplicación, sea compatible con el protocolo HTTP (para comunicación externa), gRPC e incluya algun SDK que permita trabajar con RabbitMQ.

Para dividir el trabajo de forma equitativa, el equipo deberá decidir quién será identificado como desarrollador "A" y "B", una vez realizado, se deberá cumplir obligatoriamente las responsabilidades indicadas para el desarrollador en el taller.

Nota: La base de datos a utilizar puede ser libremente escogida por los miembros del equipo. Sin embargo, tal como se indicó en la descripción de los módulos, **no** se permite el uso de PostgreSQL.

A continuación se incluye una imagen de un diagrama de arquitectura para microservicios con cola de mensajería, esto **es un ejemplo, no representa la arquitectura del taller**.



Notar que los servicios no se comunican entre sí, se comunican con la cola de mensajería.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

RESPONSABILIDADES DE LOS MICROSERVICIOS

MICROSERVICIO DE USUARIOS

Maneja la creación y gestión de cuentas de usuario, debe incluir las siguientes operaciones usando **gRPC**:

1. Crear usuario.
2. Obtener usuario por ID.
3. Actualizar usuario.
4. Eliminar usuario.
5. Listar todos los usuarios.

Crear usuario

Registrar un usuario nuevo en el sistema, se debe ingresar: Nombre, apellido, correo electrónico, contraseña, confirmación de contraseña y rol. La respuesta a la petición debe incluir los datos del usuario.

Requerimientos adicionales:

- La contraseña debe almacenarse utilizando hashing.
- La contraseña y confirmación de contraseña deben ser iguales.
- La respuesta del módulo debe incluir todos los datos del usuario, excepto el hash de la contraseña.
- El correo electrónico debe ser único en el sistema.
- Los roles solo pueden ser “Administrador” y “Cliente”.
- Para registrar a un usuario con rol Administrador, el usuario que realiza la acción debe haber iniciado sesión y tener el rol de “Administrador”.
- **Importante:** El endpoint POST /usuarios debe ser público. Solo en el caso de que se intente registrar un usuario con rol Administrador, se debe validar que el solicitante esté autenticado y autorizado. En cualquier otro caso, el registro debe estar disponible sin autenticación.

Obtener usuario por ID

Permite obtener los datos de un usuario por su ID. La respuesta debe contener: ID del usuario, nombre, apellido, correo electrónico, rol y fecha de registro del usuario. La respuesta a la petición debe incluir los datos del usuario.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- La respuesta debe excluir la contraseña.

- Solo los administradores pueden solicitar los datos de otros usuarios.
- El usuario que realiza la petición puede solicitar sus datos, si el usuario no es administrador, solamente podrá acceder a sus datos.

Actualizar usuario

Permite actualizar los datos de un usuario existente. Se pueden actualizar los siguientes campos del usuario: Nombre, apellido y correo electrónico. La respuesta a la petición debe incluir los datos del usuario.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- La respuesta debe excluir la contraseña.
- Si se intenta modificar la contraseña, la aplicación debe entregar un mensaje indicando que no se puede modificar la contraseña aquí.
- Un cliente únicamente puede editarse a sí mismo.
- Un administrador puede editar a cualquier usuario, incluyendo a otro administrador.

Eliminar usuario

Permite a un usuario con rol de administrador, eliminar un usuario del sistema.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Se debe realizar un **soft delete** del usuario.
- Solo los administradores pueden eliminar usuarios.
- La respuesta debe de estar vacía.

Listar todos los usuarios

Obtiene la lista de todos los usuarios registrados en el sistema. La respuesta debe contener un listado de los usuarios y por cada uno incluir: ID del usuario, nombre, apellido, correo electrónico, rol y fecha de registro del usuario.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- No se deben incluir contraseñas ni hashes en la respuesta.
- Se debe poder buscar por correo electrónico, nombre y apellido utilizando query params, uno para correo electrónico y otro para nombre y apellido. La búsqueda debe

ser flexible: Si se ingresa un valor parcial, el sistema debe devolver todos los resultados que lo contengan. Si se ingresan dos valores, como nombre y apellido, la búsqueda debe responder con los usuarios que cumplan ambas condiciones, tener el nombre y apellido ingresados.

- Solo los administradores pueden acceder al listado de todos los usuarios.
- No se deben mostrar los usuarios eliminados.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

MICROSERVICIO DE AUTENTICACIÓN

Maneja la autenticación y cambio de contraseña de los usuarios, debe incluir endpoints para las siguientes operaciones:

1. Iniciar sesión.
2. Actualizar contraseña.
3. Cerrar sesión.

Iniciar sesión (POST /auth/login)

Permite a un usuario autenticarse en el sistema mediante su correo electrónico y contraseña.

Requerimientos adicionales:

- Si las credenciales son correctas, el sistema debe generar y devolver un token JWT para la autenticación del usuario.
- La respuesta del endpoint debe incluir todos los datos del usuario, excepto la contraseña.
- Si el usuario ha sido eliminado, debe rechazarse el inicio de sesión.

Actualizar contraseña (PATCH /auth/usuarios/{id})

Permite actualizar la contraseña de un usuario. Para actualizar la contraseña se debe solicitar: Contraseña actual, nueva contraseña y confirmación de nueva contraseña.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- La respuesta no debe incluir la contraseña.
- Un cliente únicamente puede actualizar su contraseña, no puede actualizar la de otro cliente o administrador.
- Un administrador puede editar la contraseña de cualquier usuario.

Cerrar sesión (POST /auth/logout)

Permite a un usuario cerrar sesión, ingresando su token actual a la blacklist.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- La respuesta debe incluir un mensaje con el éxito de cierre de sesión.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

MICROSERVICIO DE FACTURACIÓN

Se encarga de la gestión de pagos y suscripciones, debe incluir las siguientes operaciones usando **gRPC**:

1. Crear factura
2. Obtener factura por ID
3. Actualizar estado de factura
4. Eliminar factura
5. Listar facturas por usuario

Crear factura

Registra una nueva factura para un usuario en el sistema, se debe ingresar: ID del usuario, estado de la factura y Monto a pagar.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden crear facturas.
- El monto debe ser un número entero positivo.
- Los estados de las facturas pueden ser “Pendiente”, “Pagado” y “Vencido”.
- La respuesta debe incluir todos los datos de la factura registrada y la fecha de emisión de la factura, que es la fecha en la que se registró la factura en la aplicación.

Obtener factura por ID

Obtiene los datos de una factura específica. La respuesta debe incluir: ID de la factura, estado de la factura, ID de usuario asociado a la factura, monto a pagar, fecha de emisión de la factura y fecha de pago de la factura.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores y el cliente dueño de la factura pueden consultarla.

Actualizar estado de factura

Permite actualizar el estado de una factura. Los campos que se pueden actualizar: Estado de la factura.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden actualizar facturas.
- Si se actualiza el estado de la factura a “Pagado”, se debe registrar la fecha de pago como la fecha en la que se realiza la actualización.
- Se debe responder a la petición con los datos de la factura actualizada.

Eliminar factura

Permite eliminar una factura del sistema.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solamente los administradores pueden eliminar facturas.
- Se debe realizar un **soft delete** de la factura.
- No se puede eliminar una factura en estado pagado.

Listar facturas por usuario

Obtiene todas las facturas. La respuesta debe contener un listado de las facturas y por cada una incluir: ID de la factura, estado de la factura, ID de usuario asociado a la factura, monto a pagar, fecha de emisión de la factura y fecha de pago de la factura.

Requerimientos adicionales:

- Se debe permitir filtrar por estado de factura utilizando query params.
- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Un administrador puede ver las facturas de cualquier usuario, un cliente recibe únicamente sus facturas.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

MICROSERVICIO DE VIDEOS

Administra el contenido audiovisual disponible en la plataforma, debe incluir las siguientes operaciones usando **gRPC**:

1. Subir
2. Obtener video por ID
3. Actualizar video
4. Eliminar video
5. Listar todos los videos

Subir vídeo

Registra un nuevo vídeo en el sistema, se deben ingresar: título, descripción y género.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden subir videos.
- La respuesta debe incluir todos los datos del video registrado.

Obtener video por ID

Obtiene los detalles de un video específico. La respuesta debe incluir: ID del video, título, descripción**, cantidad de likes** y género del video.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

Actualizar video

Permite actualizar la información de un video existente. Se pueden actualizar los siguientes campos del video: Título, descripción y género.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden realizar modificaciones
- La respuesta debe incluir todos los datos actualizados del video.
- No se puede modificar la cantidad de likes.

Eliminar video

Permite eliminar un video del sistema.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Se debe realizar un **soft delete** del video.
- Solo los administradores pueden eliminar videos.
- La respuesta debe estar vacía.

Listar todos los videos

Obtiene la lista de todos los videos disponibles en la plataforma.

Requerimientos adicionales:

- Se debe permitir la búsqueda por título y género utilizando query params. Si se ingresa un valor parcial, el sistema debe devolver todos los resultados que lo contengan.
- No deben incluirse videos eliminados.
- En cada video debe mostrarse la cantidad de likes.

MICROSERVICIO DE MONITOREO

Registra todas las acciones y errores realizadas en el sistema de manera automática y permite a los administradores ver estas acciones, debe incluir las siguientes operaciones usando **gRPC**:

1. Listar todas las acciones
2. Listar todos los errores

Listar todas las acciones

Entrega un listado de todas las acciones realizadas en el sistema mostrando por cada una: ID, ID del usuario que realizó la acción, la URL con el método, correo electrónico del usuario que realizó la acción, fecha de la acción y la acción.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden ver el listado de las acciones.

- Si el usuario no inició sesión cuando sucedió el error, el ID y correo electrónico del usuario que realizó la acción deben estar vacíos.

Nota: Algunas acciones podrían ser CREAR USUARIO, INICIAR SESIÓN, OBTENER FACTURA POR ID, ACTUALIZAR VIDEO, ELIMINAR LISTA DE REPRODUCCIÓN, etc.

Listar todos los errores

Entrega un listado de todos los errores que sucedieron en el sistema, mostrando por cada uno: ID, ID del usuario que realizó la acción, correo electrónico del usuario que realizó la acción, fecha del error y el error.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solo los administradores pueden ver el listado de las acciones.
- El error debe representar lo que sucedió y puede ser el mensaje entregado al usuario cuando sucede un error
- Si el usuario no inició sesión cuando sucedió el error, el ID y correo electrónico del usuario que realizó la acción deben estar vacíos.

Nota: Algunos errores se pueden representar como “No tiene permisos para esta acción”, “Factura no encontrada”, etc.

MICROSERVICIO DE LISTAS DE REPRODUCCIÓN

Permite a los usuarios crear listas de reproducción de videos, debe incluir las siguientes operaciones usando **gRPC**:

1. Crear lista de reproducción
2. Añadir video a lista de reproducción
3. Eliminar video de lista de reproducción
4. Ver listas de reproducción
5. Ver videos de lista de reproducción
6. Eliminar lista de reproducción

Crear lista de reproducción

Permite al usuario crear una lista de reproducción, el usuario debe ingresar el nombre de la lista.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

Añadir video a lista de reproducción

Permite al usuario añadir un vídeo a la lista de reproducción, el usuario debe ingresar el ID del video que desea añadir.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Solamente el usuario que creó la lista puede añadir videos a esta.

Ver listas de reproducción

Permite al usuario obtener todas las listas de reproducción, por cada una se debe mostrar el nombre e ID de la lista

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

Ver videos de lista de reproducción

Permite al usuario obtener los videos de una lista de reproducción, por cada uno se debe mostrar ID y nombre del video.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

Eliminar video de lista de reproducción

Permite al usuario eliminar un video a la lista de reproducción, el usuario debe ingresar el ID del video que desea eliminar.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- **Solamente el usuario que creó la lista puede eliminar videos de la lista.**

Eliminar lista de reproducción

Permite al usuario eliminar una lista de reproducción, el usuario debe ingresar el ID de la lista que desea eliminar.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- **Solamente el usuario que creó la lista puede eliminar la lista.**

MICROSERVICIO DE INTERACCIONES SOCIALES

Permite al usuario dejar likes y comentarios en videos, debe incluir las siguientes operaciones usando **gRPC**:

1. Dar like
2. Dejar comentario
3. Obtener likes y comentarios de un video

Dar like

Permite al usuario darle like a un video, el usuario debe ingresar el ID del video al que le dará like.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.
- Un usuario puede darle múltiples likes al mismo video.

Dejar comentario

Permite al usuario dejar un comentario a un video, el usuario debe ingresar el ID del video al que le dejará el comentario y el comentario.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

Obtener likes y comentarios de un video

Permite obtener los likes y comentarios de un video, el usuario debe ingresar el ID del video.

Requerimientos adicionales:

- El usuario debe haber iniciado sesión, de lo contrario debe indicarlo con un mensaje de error.

MICROSERVICIO DE ENVÍO DE CORREOS

Permite el envío de correos dentro de la aplicación, debe incluir las siguientes operaciones usando **gRPC**:

1. Enviar correo de actualización de factura

Enviar correo de actualización de factura

Permite al sistema enviar un correo cuando se actualiza la factura, el correo debe ser enviado al usuario asociado a la factura, en este se debe indicar ID, monto y estado de la factura.

Requerimientos adicionales:

- Esta funcionalidad no está expuesta a los usuarios, se debe realizar automáticamente al actualizar el estado de la factura.

API GATEWAY

Actúa como punto único de entrada a los microservicios del sistema encargándose de asegurar la autenticación y seguridad en el sistema. Este expone los siguientes endpoints usando **HTTP**:

- Crear usuario (POST /usuarios)
- Obtener usuario por ID (GET /usuarios/{id})
- Actualizar usuario (PATCH /usuarios/{id})
- Eliminar usuario (DELETE /usuarios/{id})
- Listar todos los usuarios (GET /usuarios)
- Iniciar sesión (POST /auth/login)
- Actualizar contraseña (PATCH /auth/usuarios/{id})
- Cerrar sesión (POST /auth/logout)
- Crear factura (POST /facturas)
- Obtener factura por ID (GET /facturas/{id})
- Actualizar estado de factura (PATCH /facturas/{id})
- Eliminar factura (DELETE /facturas/{id})
- Listar facturas por usuario (GET /facturas)
- Subir vídeo (POST /videos)
- Obtener video por ID (GET /videos/{id})
- Actualizar video (PATCH /videos/{id})
- Eliminar video (DELETE /videos/{id})
- Listar todos los videos (GET /videos)
- Listar todas las acciones (GET /monitoreo/acciones)
- Listar todos los errores (GET /monitoreo/errores)
- Crear lista de reproducción (POST /listas-reproduccion)
- Añadir video a lista de reproducción (POST /listas-reproduccion/{id}/videos)
- Eliminar video de lista de reproducción (DELETE /listas-reproduccion/{id}/videos)
- Ver listas de reproducción (GET /listas-reproduccion)
- Ver videos de lista de reproducción (GET /listas-reproduccion/{id}/videos)
- Eliminar lista de reproducción (DELETE /listas-reproduccion/{id})
- Dar like (POST /interacciones/{id}/likes)
- Dejar comentario (POST /interacciones/{id}/comentarios)
- Obtener likes y comentarios de un video (GET /interacciones/{id})

Todos los endpoints deben de responder con el código HTTP adecuado, según lo enseñado en clases.

Nota: Las operaciones de gRPC serán evaluadas a través de los endpoints de la API Gateway, así que si el endpoint no funciona, la operación tendrá puntaje 0.

Las responsabilidades para el desarrollador “A” y “B” serán divididas de la siguiente manera:

- Desarrollador “A” tomará el microservicio de autenticación.
- Desarrollador “A” tomará el microservicio de usuarios.
- Desarrollador “A” tomará el microservicio de listas de reproducción
- Desarrollador “A” tomará el microservicio de envío de correos.
- Desarrollador “B” tomará el microservicio de videos.
- Desarrollador “B” tomará el microservicio de facturas.
- Desarrollador “B” tomará el microservicio de monitoreo.
- Desarrollador “B” tomará el microservicio de interacciones sociales.

Ambos desarrolladores, “A” y “B” trabajarán en la API GATEWAY.

Notas:

- El integrante que desarrolle el microservicio de autenticación debe realizar todo lo relacionado con esta, no solamente los endpoints.
- En caso de cualquier error el sistema debe devolver un mensaje indicando que causó el error, si sucede un error la respuesta no puede estar vacía.
- Se deben mantener las relaciones entre las bases de datos, por ejemplo, no se puede crear una factura para un usuario que no se encuentre en el microservicio de usuarios.
- Los microservicios deben ser completamente independientes, no se pueden llamar los unos a otros (Pista: Esto se puede conseguir con redundancia en las bases de datos y colas de mensajería).
- Soft delete es una técnica que marca un registro como eliminado sin borrarlo físicamente de la base de datos. En lugar de eliminarlo con DELETE, se actualiza un campo como deleted_at o is_deleted.

SEEDER

La aplicación debe incluir un script seeder o endpoint que automatice el llenado de la base de datos con datos de prueba. En concreto, se deben generar:

- Entre 100 y 200 usuarios.
- Entre 300 y 400 facturas.
- Entre 400 y 600 videos.
- Entre 50 y 100 likes.
- Entre 20 y 50 comentarios.

Notas:

- En el README.md del repositorio debe indicarse como ejecutar el seeder de la aplicación. Si requiere que todos los microservicios o cola de mensajeria estén funcionando, debe ser indicado.
- Se debe indicar en el README.md las credenciales para iniciar sesión con un usuario de rol “Administrador”.
- No es necesario incluir los datos de monitoreo en el seeder.

NGINX

Se deberá entregar una configuración de nginx y archivos relacionados que permitan realizar las siguientes acciones:

- Escribir en un log el cuerpo de una petición.
- Actuar como balanceador de carga para 3 API Gateways diferentes
    - Nota: Se debe levantar el servicio de la API Gateway 3 veces, apuntando a puertos diferentes.
- Que al realizar una petición GET a la URL “/comedia” se responda con una frase cómica y código 200.
    - Nota: Se debe de hacer usando únicamente las herramientas que nginx entrega.
- Usar certificados SSL propios para comunicación HTTPS
- Redirección automática hacia HTTPS

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

INFORME

Se debe crear un informe que contenga una portada que incluya, el logo de la universidad, fecha de entrega del informe, asignatura e integrantes del equipo, un índice y lo indicado a continuación **según el contexto indicado en el taller**, todos los puntos a continuación (excepto los Modelos ER) **deben ser fundamentados**:

1. Modelos ER (Entidad Relación) de cada base de datos realizado en DBDiagram.
2. Diagrama C4 de la arquitectura.
3. ¿Por qué crees que se eligió la arquitectura de microservicios?
    a. ¿Existen otras opciones de arquitectura, enseñadas en el curso, que podrían aplicarse en el contexto del taller?
    b. ¿Consideran que la arquitectura de microservicios fue la elección correcta para el taller?
4. ¿Cuáles son los beneficios de la arquitectura utilizada y por qué?
5. Respecto a la migración del monolito modular del taller 1 a los microservicios del taller 2
    a. Si utilizara el patrón de migración Strangler Fig, ¿En qué orden migraría los módulos, y por qué en ese orden?
    b. ¿Qué patrón migración recomienda para esta migración y porque?
    c. ¿Utilizaría un patrón de migración de base de datos?, de ser así, ¿Cuál y porque?, de no ser así, ¿Por qué no usaría un patrón de migración de base de datos?
6. ¿Cuáles son los beneficios de utilizar una API Gateway en este contexto?
7. ¿Cuáles son los beneficios de usar gRPC para la mayoría de servicios?

Contenido extra del informe, no forma parte de la rúbrica, pero entregará 0,3 décimas en la nota final del taller si se fundamenta correctamente:

8. Posibles mejoras para la arquitectura indicada.

El informe debe cumplir con el siguiente formato:

- Texto justificado.
- Times New Roman 12.

Notas:

- El informe debe ser desarrollado por ambos desarrolladores, “A” y “B” en conjunto.
- Si alguno de los puntos del informe no incluye una fundamentación (a excepción del modelo ER de las bases de datos), ese punto será evaluado con puntaje 0.

POSTMAN

Para cada módulo deberán realizar colecciones de postman para probar flujos que un usuario realizaría normalmente. Cada uno de estos deberá además incorporar pre y post scripts para poder obtener y establecer variables de entorno para poder ser utilizadas durante el flujo.

Los flujos que se espera que realicen los usuarios son los siguientes:

1. Obtener el listado de todos los videos, registrar un nuevo usuario de rol cliente, iniciar sesión con el usuario creado, obtener un único vídeo por su ID y darle like al video obtenido anteriormente.
2. Iniciar sesión con usuario administrador, obtener todas las facturas, marcar una factura como pagada (se debe enviar el correo) y revisar el listado de todas las acciones.
3. Iniciar sesión como usuario administrador, obtener todos los usuarios, eliminar un usuario y crear un nuevo video.
4. Iniciar sesión como usuario cliente, cambiar la contraseña, crear una lista de reproducción, obtener listado de todos los videos y añadir un vídeo a la lista de reproducción creada previamente.

El/la Desarrollador “A” tiene asignado los flujos 1 y 2, mientras que el/la Desarrollador “B” los flujos 3 y 4.

Además, dentro del repositorio, junto a las colecciones de los flujos, la colección de postman debe incluir todos los endpoints de la aplicación.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

COMPOSICIÓN DE PAREJAS

El curso Arquitectura de Sistemas se encuentra compuesto por 12 estudiantes, por lo que sé requiere la creación de 6 parejas.

TAREAS POR INTEGRANTE

Cada integrante de la pareja será responsable de las tareas asignadas que tenga, según el rol que elija, desarrollador “A” o “B”. Estará encargado desde la concepción y desarrollo de este, hasta su despliegue y correcto funcionamiento en la nube. Se le evaluará el desarrollo de su módulo en base a los commits del repositorio.

INSCRIPCIÓN PAREJAS

Las parejas designarán a un representante, el cual debe enviar un correo a más tardar el 04/05/2025 - 23:59 al profesor, con copia al ayudante, de los integrantes: El nombre, apellidos, RUT y el desarrollador que tomará cada integrante.

Una vez se reciba una respuesta al correo de inscripción, estará autorizada y conformada la pareja para entregar el taller.

BÚSQUEDA DE PAREJA

Aquellos/as estudiantes que no logren conformar una pareja, deberán comunicarse con el profesor, con copia al ayudante, a más tardar el 04/05/2025 - 23:59, explicando el motivo o complicación.

Aquellos/as estudiantes que no posean equipo pasado el plazo, serán sancionados con 10 décimas de su nota de taller 1 y posteriormente asignados a algún equipo.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

ENTREGA

Se debe entregar mediante Campus Virtual UCN un archivo de .zip o .rar que contenga:

1. Archivo texto o PDF con el enlace al repositorio y el listado de con los nombres, apellidos y RUT de cada integrante de la pareja.
    a. El repositorio debe ser configurado con visibilidad pública. Se sugiere trabajar con visibilidad privada y posterior a la entrega actualizar. Considere que se tiene un plazo de hasta 1 hora pasado el límite de entrega para realizar este cambio.
2. Archivo PDF con las respuestas a las preguntas mencionadas en el apartado de INFORME.

El nombre del .zip o .rar debe estar compuesto por los apellidos de los integrantes de la pareja, seguido de "-TALLER2", por ejemplo "ALARCON-FONTECILLA-TALLER2.rar"

Fecha límite de entrega: 08/06/2025 - 23:59 hrs.

DESCUENTOS Y CONSIDERACIONES

Los descuentos serán aplicados conforme se cumplan con las faltas que se mencionan en este apartado.

Se describe la falta y la cantidad de décimas que descuenta de la nota:

- Los descuentos por atrasos se encuentran descritos como:
    - Entrega entre las 00:00 y 00:59 - 10 décimas.
    - Entrega entre las 01:00 y 01:59 - 20 décimas.
    - Entrega entre las 02:00 y 02:59 - 30 décimas.
    - Entrega entre las 03:00 y 03:59 - 40 décimas.
    - Entrega entre las 04:00 y 04:59 - 50 décimas.
    - Entrega pasado las 05:00 am será calificada con nota mínima.
- No entregar un archivo de extensión .txt o .pdf (texto o PDF) con el repositorio y el enlace a la aplicación - 10 décimas.
- No se entrega el informe nota mínima.
- Usar PostgreSQL como base de datos para el microservicio de usuarios nota máxima 4.0.
- No incluir nombres, apellidos y RUT de todos los integrantes en la entrega - 10 décimas.
- No configurar visibilidad pública de los repositorios a más tardar 1 hora después de la entrega - 10 décimas.
- No se levanta la aplicación siguiendo los pasos del README.md del repositorio nota mínima.
- No se entrega colección de postman con todos los endpoints -30 décimas.
- No se cuenta con un seeder o no se indica cómo se debe ejecutar en el readme.md -30 décimas.

Nota: Atraso considera tanto la entrega de los enlaces al repositorio en la nube, cómo commits o cambios en el repositorio. Es decir: Si se sube en el periodo la entrega a campus, pero se realizan commits fuera del plazo, se aplicarán igualmente las sanciones.

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

RÚBRICA

La rúbrica será publicada a más tardar el 06/05/2025.

No fundamenta porque se implementaría de esa manera