UNIVERSIDAD INTERNACIONAL DEL ECUADOR

Nombre: Jair Naranjo

MATERIA: PROGRAMACION ORIENTADA A OBJETOS

FECHA: 1/03/2026

TEMA: Aprendizaje Autónomo 2

Índice

Etapa: 1 - Planeación del Software

1. INTRODUCCIÓN Y OBJETIVO DEL SISTEMA

El presente documento detalla la planificación del sistema "SecureStream", una plataforma de gestión de contenido multimedia diseñada bajo principios de programación funcional, el objetivo primordial es implementar un sistema que no solo gestione contenido, sino que aplique políticas de seguridad estrictas para la protección de la integridad de la plataforma.

2. Alcance del Proyecto

El sistema abarcará las siguientes funcionalidades críticas, garantizando un rendimiento óptimo y una superficie de ataque reducida:

- Gestión de identidades: Registro y validación de usuarios mediante estructuras de datos optimizadas (Maps) para búsquedas de tiempo constante $O(1)$.
- Control de acceso basado en atributos (RBAC): Restricción de contenido basada en el nivel de suscripción y la edad del usuario, utilizando lógica booleana y operadores de comparación.
- Auditoría de seguridad: Generación de logs automáticos ante intentos de acceso no autorizados mediante funciones flexibles.
- Administración de catálogo: Listado dinámico de contenido multimedia utilizando Slices para una gestión de memoria eficiente.

3. Estructura de los Módulos

Basado en los temas de la Unidad 1, el sistema se divide en los siguientes módulos funcionales:

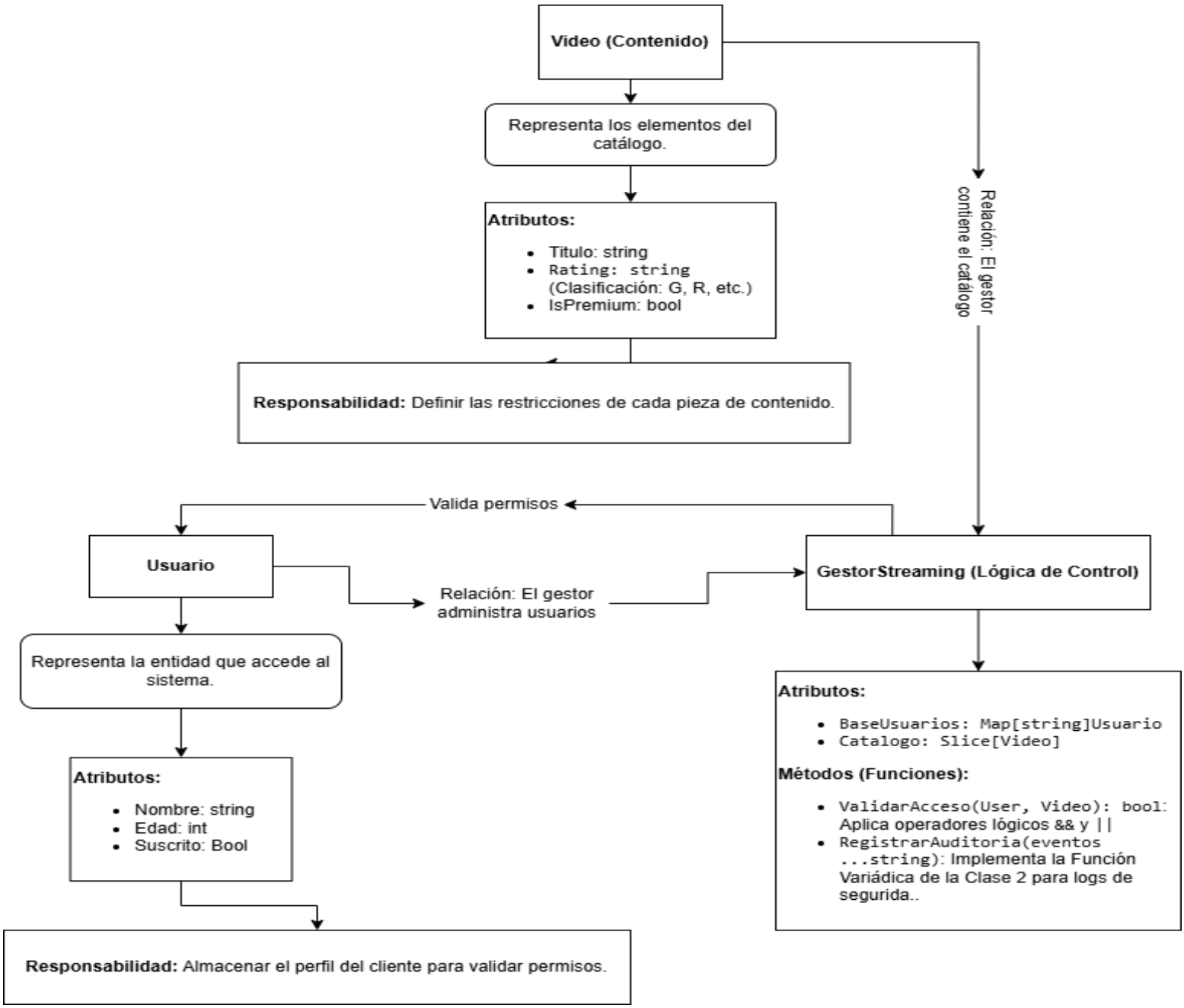
Módulo	Funcionalidad Principal	Concepto Técnico Aplicado
Módulo de Seguridad	Validación de credenciales y permisos de acceso.	Operadores Lógicos (&&, `
Módulo de Usuarios	Almacenamiento y recuperación de perfiles.	Uso de Maps y Structs para persistencia en memoria.
Módulo de Catálogo	Gestión de películas, series y ratings.	Manejo de Slices y recorrido dinámico con range.
Módulo de Auditoría	Registro de eventos y alertas del sistema.	Funciones Variádicas (...string) y manipulación de cadenas.

4. Justificación Técnica

El software se desarrollará bajo el paradigma de programación funcional solicitado, se ha seleccionado el lenguaje Golang debido a su alta eficiencia en sistemas concurrentes y su tipado fuerte, lo cual es un estándar en el desarrollo de herramientas de Ciberseguridad.

Se utilizará el **identificador en blanco (_)** para la optimización de ciclos, evitando el desperdicio de recursos de memoria al procesar grandes volúmenes de datos en los Slices de catálogo.

5. DIAGRAMA DE CLASES



Arquitectura modular del sistema SecureStream: Se observa el desacoplamiento entre las entidades de datos (Usuario/Video) y la lógica de control, garantizando una auditoría centralizada y una gestión de memoria eficiente mediante estructuras dinámicas de Go.

6. RECURSOS Y HERRAMIENTAS

- IDE: Visual Studio Code.
- Control de Versiones: Repositorio en GitHub con configuración de ramas para desarrollo seguro.
- Documentación: Estructura en PDF con diagramas de flujo y alcance detallado.

7. Link del repositorio de github

https://github.com/muhaccho/SecureStream-POO-G1

Etapa 2 - Desarrollo y Estructuras Avanzadas

8. IDENTIFICACIÓN DE NUEVAS CLASES Y ESTRUCTURAS

En esta segunda etapa, el sistema SecureStream ha evolucionado de un modelo funcional hacia una arquitectura basada en Interfaces y Encapsulamiento, alineándose con los temas de la Unidad 3:

- Interfaz ContenidoMultimedia: Se definió esta interfaz para establecer un contrato de comportamiento obligatorio para cualquier tipo de video. Incluye los métodos Reproducir(edadUsuario int) y ObtenerInfo(), lo que permite que el sistema sea polimórfico.
- Clase (Struct) pelicula: Es la implementación concreta de la interfaz, se diseñó utilizando Encapsulamiento, definiendo sus atributos (titulo, minimaEdad, esPremium) en minúsculas para proteger la integridad de los datos de ciberseguridad ante accesos externos no autorizados.
- Acoplamiento y Lógica de Control: La clase pelicula se acopla a la interfaz mediante métodos receptores de punteros (*pelicula), esto permite que el sistema valide dinámicamente si un usuario cumple con las restricciones de edad antes de permitir la ejecución del método de reproducción, integrando así el manejo de errores directamente en la lógica de negocio.

9. DOCUMENTACIÓN DE FUNCIONALIDADES AVANZADAS

En el desarrollo del prototipo funcional, se han integrado pilares fundamentales de la Programación Orientada a Objetos para fortalecer la robustez del sistema:

- Uso de Encapsulamiento: Se ha implementado la restricción de visibilidad mediante el uso de identificadores en minúsculas para los atributos del struct pelicula (ej. minimaEdad, esPremium), esta decisión técnica asegura que el estado interno del objeto no pueda ser alterado de forma arbitraria desde otros paquetes, garantizando que las políticas de acceso de SecureStream solo se modifiquen a través de métodos controlados Setters y validados.
- Manejo de Errores e Integridad: A diferencia de los modelos básicos, este avance incorpora la interfaz nativa error de Go en el método Reproducir. Esto permite que el sistema detecte proactivamente violaciones de seguridad, tales como un usuario menor de edad intentando acceder a contenido restringido, y retorne un objeto de error descriptivo, esta técnica es vital en el desarrollo de software seguro. Porque evita fallos catastróficos y permite generar Alertas de Auditoría precisas en tiempo real.
- Interacción mediante Menú Dinámico: Se desarrolló una interfaz de consola utilizando la estructura de control switch, lo que permite una navegación fluida y segmentada de las funcionalidades, facilitando las pruebas de penetración y validación de permisos en cada módulo.

10. JUSTIFICACIÓN TÉCNICA - UNIDAD 3

La evolución del sistema SecureStream en esta etapa se fundamenta en la implementación de estructuras avanzadas que permiten una transición de la programación funcional hacia un entorno orientado a objetos profesional:

- Implementación de Interfaces: Siguiendo las directrices de la Unidad 3, se ha integrado la interfaz ContenidoMultimedia. Esta decisión no es meramente organizativa; permite establecer un "contrato" de métodos que cualquier tipo de contenido como, Películas, Series o Documentales, debe

cumplir, garantizando que la lógica de seguridad sea uniforme y no presente fugas de validación.

- Polimorfismo mediante Interfaces: El uso de interfaces permite que el sistema trate a diversos objetos de manera genérica, esto facilita la escalabilidad del software, ya que se pueden agregar nuevas funcionalidades o tipos de contenido sin necesidad de reescribir el código base, cumpliendo con el principio de abierto/cerrado de la POO.
- Punteros y Eficiencia de Memoria: Se ha priorizado el uso de receptores de punteros (*) en los métodos de las clases. Tal como se analizó en clase, esto evita la duplicación innecesaria de objetos en la memoria Slices, optimizando el rendimiento del sistema al manejar catálogos extensos y permitiendo que los métodos modifiquen el estado real de los objetos cuando sea necesario.

11. CONCLUSIONES Y LOGROS RELACIONADOS

- Optimización de la Arquitectura: La implementación de Interfaces ha permitido desacoplar la lógica de reproducción de los datos de contenido, logrando un sistema polimórfico que facilita la escalabilidad hacia nuevos formatos multimedia sin alterar el núcleo del software.
- Fortalecimiento de la Integridad: Mediante el uso de Encapsulamiento, se garantiza que atributos críticos como el rating de edad y el estado premium de los contenidos sean inaccesibles para modificaciones externas directas, cumpliendo con los estándares de seguridad exigidos en la disciplina.
- Gestión Eficiente de Excepciones: El diseño de un sistema de Manejo de Errores personalizado ha mejorado la auditoría de ciberseguridad, permitiendo que el sistema responda de manera controlada y descriptiva ante intentos de acceso no autorizados, en lugar de generar fallos inesperados.

- Integración de Servicios: La simulación exitosa de una conexión a base de datos mediante un servidor HTTP en el puerto 8000 demuestra la viabilidad de integrar SecureStream con entornos de persistencia profesional en futuras etapas.

Evaluación en Contacto con el Docente

12. ETAPA 3: APLICATIVO WEB E INTEGRACIÓN PROFESIONAL

En esta fase final el sistema SecureStream ha dejado de ser un programa de consola para convertirse en una solución de software integral, uniendo el Backend (Go), Persistencia (MySQL) y Frontend (Bulma CSS).

A. Implementación de Servicios Web (API JSON)

Siguiendo los requerimientos de la asignatura, se han desarrollado 8 servicios web que permiten la interoperabilidad del sistema mediante serialización JSON:

- GET /api/peliculas: Servicio principal que consulta la base de datos y devuelve el catálogo completo.
- GET /api/estadisticas: Proporciona métricas en tiempo real para el Dashboard.
- POST /api/login: Módulo de seguridad que valida credenciales y gestiona el acceso al sistema.
- Otros servicios: /api/logs, /api/usuarios, /api/verificar, /api/config y la ruta raíz / para la interfaz visual.

B. Persistencia en Base de Datos MySQL

Se integró MySQL mediante el driver oficial de Go, permitiendo que la gestión de datos sea persistente y no solo en memoria:

- Conectividad: Uso del paquete database/sql para realizar consultas dinámicas (SELECT).
- Integridad: Los datos se almacenan en la base securestream_db, garantizando la seguridad y disponibilidad de la información.

C. Interfaz de Usuario y Visualización del Futuro

El aplicativo utiliza Bulma CSS para ofrecer una experiencia de usuario (UX) moderna y funcional.

- Visualización: Se ha proyectado el uso de estas tecnologías para crear sistemas de streaming altamente escalables, donde el rendimiento de Go y la flexibilidad de JSON permiten una respuesta inmediata ante millones de peticiones concurrentes.

13. CONCLUSIÓN FINAL DEL PROYECTO INTEGRADOR

- Aprendizaje: Se integraron con éxito las 4 unidades, logrando un software robusto que cumple con estándares de ciberseguridad y arquitectura web.
- Dificultades: El manejo de módulos en Go y la configuración de rutas para servir archivos estáticos fueron los retos técnicos más significativos.
- Impacto Profesional: Este proyecto demuestra que la POO es la base para construir aplicaciones escalables, seguras y profesionales en la industria actual

14. Link de github

https://github.com/muhaccho/SecureStream-POO-G1

15. Link de Canvas

https://www.canva.com/design/DAHCu5Kyr7o/1TOhEoKSv9_Fq_68B5SWeA/edit?ui=eyJBIjp7fX0