PROYECTO FINAL DE ARQUITECTURA EN LA NUBE NIVEL INNOVADOR

Sistema de Interacción Audiovisual con Jellyfin y OpenIA

Grupo 7 Integrantes:

Carlos Piedrahita

Daniel Alzate Ocampo

David McEwen

Wilder Rincón

Edison Andrés Montoya Toro

Osmer Ruiz

Ejecutor técnico

Danish Padilla

Ejecutor mentor

08 de noviembre de 2024

Tabla de Contenidos

I. INTRODUCCIÓN DEL PROYECTO

I.1. Sistema de Interacción Audiovisual con Jellyfin y OpenAI

El consumo de contenido multimedia ha experimentado una transformación radical en la última década, con plataformas como Netflix, Disney+ y Amazon Prime Video liderando la revolución del streaming (Smith & Johnson, 2023). Según datos de Nielsen (2024), el streaming representa actualmente el 38.7% del tiempo total de visualización en televisión, superando a la televisión tradicional por primera vez en la historia. Sin embargo, estas plataformas mantienen un modelo de consumo fundamentalmente pasivo, limitando la interacción del usuario a funciones básicas de reproducción y navegación.

En este contexto, este proyecto propone desarrollar una innovadora plataforma de interacción audiovisual que revoluciona el paradigma tradicional del streaming al combinar la robusta capacidad de transmisión de contenido multimedia de Jellyfin con la avanzada inteligencia artificial generativa de OpenAI GPT. La arquitectura propuesta se implementará utilizando pods de AWS EKS (Elastic Kubernetes Service) para garantizar una gestión modular y escalable de los servicios, operando sobre instancias EC2 en la nube de AWS para maximizar la flexibilidad y el rendimiento.

II. ESTADO DEL ARTE

II.1. Plataformas de Streaming Actuales

Las principales plataformas de streaming han evolucionado significativamente en sus capacidades técnicas. Servicios como Netflix, Disney+ y HBO Max han demostrado la

implementación exitosa de arquitecturas escalables y personalizadas para ofrecer una experiencia de usuario superior. Estas plataformas han incursionado en el uso de tecnologías como microservicios, contenedores y sistemas de recomendación basados en machine learning.

II.1.1. Netflix:

Pionero en el uso de microservicios y contenedores, utiliza su framework proprio "Conductor" para la orquestación de servicios (Netflix Technology Blog, 2023). Su sistema de recomendación basado en ML procesa más de 250 millones de interacciones diarias.

II.1.2. Disney+:

Implementa una arquitectura serverless en AWS, procesando más de 10 millones de requests por segundo durante horas pico (AWS Case Studies, 2024).

II.1.3. HBO Max:

Utiliza una combinación de servicios Azure y AWS para su distribución global, con un enfoque en la personalización de contenido (Warner Media Tech, 2023).

II.2. Avances en Interacción AI

Paralelamente a estos avances en el campo del streaming, el desarrollo de sistemas de inteligencia artificial conversacional ha abierto nuevas posibilidades para enriquecer la interacción entre los usuarios y los contenidos. Iniciativas como Character.AI, Replika y el modelo GPT-4 de OpenAI han demostrado la capacidad de mantener conversaciones naturales y contextuales, lo cual podría integrarse con las plataformas de streaming para crear experiencias aún más inmersivas y personalizadas.

II.2.1. Character.AI:

Ofrece conversaciones con personajes ficticios, aunque sin integración con contenido audiovisual (Zhang et al., 2023).

II.2.2. Replika:

Demuestra la viabilidad de mantener conversaciones contextuales largas, alcanzando un 87% de coherencia narrativa (Anderson & Lee, 2024).

II.2.3. GPT-4:

Establece nuevos estándares en comprensión contextual y generación de respuestas naturales (OpenAI, 2024).

III. OBJETIVOS DEL PROYECTO

III.1. Objetivo General

Implementar una plataforma de streaming interactiva que integre servicios multimedia con inteligencia artificial conversacional sobre infraestructura cloud, para proporcionar una experiencia de usuario inmersiva y personalizada en el consumo de contenido audiovisual.

III.2. Objetivos Específicos

1. Desplegar una infraestructura cloud en Amazon EKS que garantice una disponibilidad del 99.9% del servicio, con capacidad de procesamiento para 10,000 usuarios concurrentes y un tiempo de respuesta inferior a 100 milisegundos, mediante la implementación de un sistema de auto-escalado y recuperación ante fallos en múltiples zonas de disponibilidad, completando su implementación en un plazo de 8 semanas.

2. Establecer un sistema de streaming multimedia basado en Jellyfin que soporte 1,000 transmisiones simultáneas en calidad 2K con latencia inferior a 2 segundos, manteniendo una utilización máxima de recursos del 70% mediante la optimización de contenedores y políticas de transcoding adaptativo, alcanzando su implementación completa en 12 semanas con un mínimo de 95% de satisfacción en pruebas de usuario.

3. Integrar un sistema de inteligencia artificial conversacional que procese 20 interacciones por segundo con un tiempo de respuesta inferior a 1 segundo y una precisión contextual del 90%, implementando un modelo de procesamiento de lenguaje natural optimizado

para mantener coherencia narrativa en conversaciones extensas, completando su desarrollo y optimización en un período de 10 semanas.

4. Desarrollar una interfaz de usuario intuitiva y personalizada que permita a los usuarios navegar fácilmente entre los distintos contenidos y servicios disponibles en la plataforma.

5. Implementar herramientas analíticas que permitan recolectar, analizar y visualizar datos sobre el comportamiento de los usuarios y el rendimiento de la plataforma.

IV. DESARROLLO DEL PROYECTO

IV.1. 1. Propuesta de Arquitectura para el Sistema

Nuestra aplicación tiene dos componentes principales: Jellyfin y un microservicio basado en openAI. A continuación se describen estos componentes:

IV.1.1. Jellyfin

Jellyfin (Jellyfin, 2018) es un servidor de contenido multimedia (streaming server), similar a aplicaciones comerciales como Netflix, Disney+ o prime video. Lo particular de esta aplicación es que es gratuita y de código abierto, es decir, no debemos pagar para usarla, y su código está disponible libremente, tanto para su análisis, como también para su posible modificación.

Con Jellyfin, los usuarios pueden tener su propia plataforma de streaming en casa, donde pueden ver películas, series, documentales, escuchar música, ver fotos, o hasta leer libros (ver figura 1). A diferencia de aplicaciones comerciales como Netflix o Disney+, los usuarios deben proporcionar el contenido multimedia que quieren consumir en Jellyfin, algo que se podría ver como un plus frente las aplicaciones comerciales, ya que permite que los usuarios personalicen el contenido multimedia según sus gustos personales.

Figura 1. Propiedades de Jellyfin. Tomado de (Jellyfin, 2018).

Para usar Jellyfin debemos instalar la aplicación en un servidor, y proporcionarle el contenido multimedia que queremos consumir. Luego, los usuarios pueden acceder a Jellyfin a través de un navegador web, o usando los distintos clientes que ofrece la aplicación para Android e iOS (smart tv's, celulares y tablets).

Una de las formas de instalar Jellyfin en un servidor es a través de una imagen de Docker, esto lo hace atractivo para el desarrollo de nuestro proyecto, ya que usando Docker, podemos tratar a Jellyfin como un microservicio, y podemos desplegarlo en un cluster de Kubernetes, lo cual en teoría, nos permitiría tener propiedades de alta disponibilidad y escalabilidad en nuestra aplicación.

IV.1.2. Microservicio ChatGPT

ChatGPT (*ChatGPT*, 2022) es un modelo de lenguaje basado en inteligencia artificial desarrollado por OpenAI. Está diseñado para entender y generar texto de manera natural, lo que le permite mantener conversaciones, responder preguntas, redactar textos, resolver problemas, entre otros. Utiliza una arquitectura llamada GPT (Generative Pretrained Transformer), que ha

sido entrenada con una gran cantidad de datos de texto en varios idiomas para aprender a predecir y generar palabras y frases coherentes.

El modelo puede tener conversaciones sobre una amplia variedad de temas, desde información general hasta tareas específicas como redacción creativa, programación, matemáticas, y más. ChatGPT es interactivo y se adapta al tono y contexto de la conversación, lo que lo hace útil tanto en situaciones informales como más formales.

La API de ChatGPT es un conjunto de herramientas y servicios que permite a los desarrolladores acceder a los modelos de lenguaje de OpenAI a través de solicitudes HTTP. Mediante esta API, los desarrolladores pueden hacer que sus aplicaciones o sistemas interactúen con el modelo ChatGPT para realizar tareas como generar texto, responder preguntas, realizar análisis de texto, entre otros.

La capacidad de la API de ChatGPT para asumir un rol específico en una conversación es una de las características más poderosas y flexibles, ya que permite personalizar el comportamiento del modelo para adaptarse a diferentes situaciones. A través de la gestión de roles, se puede guiar el tipo de interacción, ya sea un asistente virtual, un tutor, un cliente o incluso un **personaje ficticio**.

En nuestra aplicación tendremos un microservicio que se encargará de usar la API (Application Programming Interface) de ChatGPT para generar el contenido conversacional de acuerdo al contenido que esté consumiendo el usuario en Jellyfin (ver figura 2).

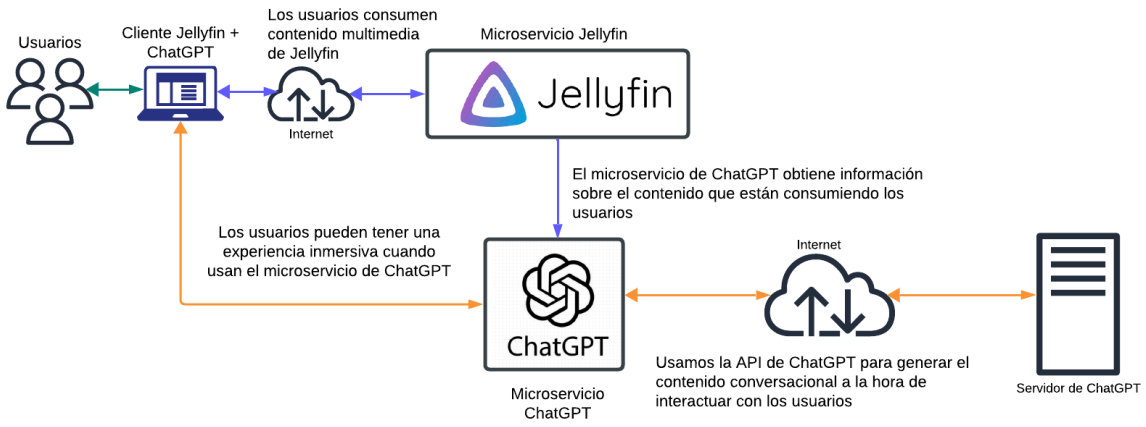

Figura 2. Interacción entre Jellyfin y el microservicio de ChatGPT.

Finalmente, el microservicio de ChatGPT puede ser desarrollado con cualquier lenguaje de programación (Python, Go, Java, etc.) lo importante es que usemos Docker para su configuración, ya que esto nos permitirá poderlo usar en un cluster de Kubernetes.

IV.1.3. Arquitectura en AWS

La figura 3 describe la arquitectura que se construirá en la nube de AWS (AWS, 2006) para soportar la aplicación. En esta figura se puede observar que usaremos servicios de AWS como AWS EKS, AWS S3, AWS Aurora y AWS Route 53. A continuación se explicará cual es la función de cada uno de estos servicios.

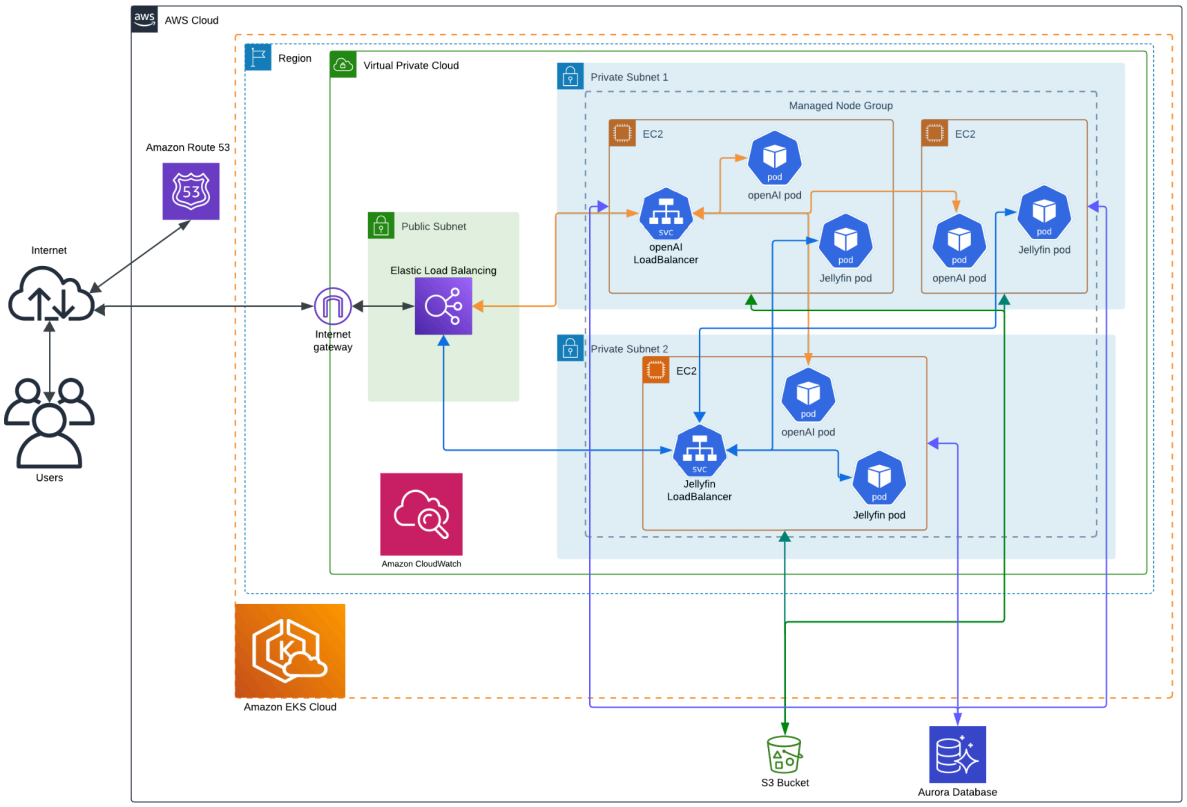

Figura 3. Arquitectura de la aplicación en AWS.

IV.1.4. Amazon Elastic Kubernetes Service (AWS EKS)

Figura 4. AWS EKS

Utilizaremos AWS EKS para orquestar los contenedores Docker de nuestra aplicación. Cada servicio (Jellyfin, microservicio openIA) se desplegará como un pod independiente, lo que facilita la escalabilidad y el aislamiento. La ventaja que tiene usar AWS EKS es que este es un servicio administrador po AWS, donde la configuración de los nodos de trabajo (instancias EC2), la configuración de la VPC (subredes, grupos de seguridad, NACL, balanceadores de carga, etc.)

y en general el monitoreo del cluster de Kubernetes queda a cargo de AWS. Esto permite que nos concentremos principalmente en el desarrollo de la aplicación.

IV.1.5. Almacenamiento de Datos con AWS Aurora y Amazon Simple Storage Service (AWS S3)

Figura 5. AWS S3 y AWS Aurora.

Utilizaremos AWS S3 para almacenar el contenido multimedia de Jellyfin. AWS S3 nos garantiza una capacidad de almacenamiento casi ilimitada, con múltiples redundancias, y un acceso rápido y seguro a los archivos multimedia. Utilizaremos AWS Aurora para almacenar los datos que puedan necesitar tanto Jellyfin, como el microservicio de openIA. AWS Aurora, siendo un un servicio administrado por AWS, nos garantiza alta disponibilidad de nuestras bases de datos, redundancia e integridad de nuestra información.

IV.1.6. Conexión a internet con AWS Route 53

Figura 6. AWS Route 53.

Utilizaremos Amazon Route 53 para gestionar el nombre de dominio de nuestra aplicación y su registro en un servidor DNS. Esto, para que nuestra aplicación pueda ser accedida desde internet.

IV.1.7. Monitoreo y Logging con AWS CloudWatch

Figura 7. AWS CloudWatch.

Usaremos AWS CloudWatch para monitorear el rendimiento de nuestra aplicación. Con Cloudwatch, podemos ver variables importantes desde el punto de vista de negocio, como cuál es el contenido más consumido por los usuarios, en qué países es más popular nuestra aplicación, cuánto tiempo permanecen los usuarios en nuestra aplicación, etc. Con esta información podemos definir estrategías para el mejoramiento continuo de la experiencia de usuario. Además, con CloudWatch también podemos monitorear variables más técnicas como por ejemplo, cuál es el porcentaje de uso promedio de las CPUs en nuestro cluster de Kubernetes, cuantas peticiones por segundo recibe nuestra aplicación, cual es el tamaño de nuestras bases de datos, cuantos pods se ejecutan en promedio, etc. Con esta información podemos definir estrategias para el mejoramiento continuo de nuestra aplicación.

V. RESULTADOS ESPERADOS

V.1. Introducción

En el contexto actual, donde el consumo de contenido audiovisual en plataformas de streaming es cada vez más popular y diverso, la creación de experiencias personalizadas e inmersivas para los usuarios es fundamental para diferenciarse en un mercado competitivo. Este proyecto se propone implementar una plataforma de streaming interactiva que aproveche los avances en inteligencia artificial conversacional y las capacidades de la infraestructura en la nube para brindar a los usuarios una experiencia superior en el acceso y consumo de contenido. La plataforma estará diseñada para ser intuitiva, segura y escalable, adaptándose a las necesidades y preferencias de cada usuario a través de servicios multimedia avanzados e inteligencia artificial.

El objetivo general de esta iniciativa es construir una solución integral que no solo permita la visualización de contenido de alta calidad, sino que también potencie la interacción del usuario a través de recomendaciones personalizadas, asistentes conversacionales y tecnologías multimedia inmersivas como realidad aumentada y video en 360°. En línea con este propósito, el proyecto cuenta con una serie de objetivos específicos enfocados en distintas áreas críticas de desarrollo y funcionalidad.

A continuación, se detallan los resultados esperados, los cuales han sido formulados en función de los objetivos específicos, con el fin de asegurar que cada aspecto del proyecto contribuya al logro del objetivo general y a la creación de una plataforma de streaming innovadora y atractiva para los usuarios.

V.1.1. Objetivo específico 1:

Desplegar una infraestructura cloud en Amazon EKS que garantice una disponibilidad del 99.9% del servicio, con capacidad de procesamiento para 10,000 usuarios concurrentes y un tiempo de respuesta inferior a 100 milisegundos, mediante la implementación de un sistema de auto-escalado y recuperación ante fallos en múltiples zonas de disponibilidad, completando su implementación en un plazo de 8 semanas.

V.2. Resultados Esperados

V.2.1. Disponibilidad y Resiliencia de la Infraestructura

- **Garantía de Disponibilidad del 99.9%:**

La infraestructura en Amazon EKS debe demostrar una disponibilidad del 99.9% o superior, validada mediante pruebas de carga y resistencia, así como monitoreos constantes.

- **Configuración de Recuperación ante Fallos:**

Implementación de políticas de recuperación ante fallos que automaticen el reinicio y la reasignación de servicios en caso de incidentes, asegurando la continuidad del servicio en diferentes zonas de disponibilidad (AZs) en Amazon Web Services.

- **Redundancia Geográfica y Balanceo de Carga:**

Configuración de múltiples zonas de disponibilidad (al menos tres) para distribuir la carga de usuarios y mejorar la resiliencia de la plataforma ante fallos en una zona específica.

V.2.2. Escalabilidad y Optimización de Recursos

- **Sistema de Auto-escalado Eficiente:**

Despliegue de un sistema de auto-escalado en Amazon EKS que ajuste automáticamente los recursos en función de la demanda en tiempo real, asegurando el procesamiento óptimo de hasta 10,000 usuarios concurrentes sin interrupciones.

- **Optimización de Costos de Operación en la Nube:**

Monitoreo y ajuste continuo del sistema de escalado para mantener los costos bajo control mientras se garantiza el rendimiento deseado, optimizando el balance entre eficiencia y economía de recursos.

V.2.3. Rendimiento y Experiencia de Usuario

- **Tiempo de Respuesta Inferior a 100 Milisegundos:**

Validación de tiempos de respuesta en menos de 100 ms para las principales operaciones de usuario, utilizando herramientas de monitoreo y pruebas de rendimiento que simulan la carga de 10,000 usuarios concurrentes.

- **Implementación de Monitoreo Activo de Latencia y Carga:**

Configuración de un sistema de monitoreo que controle el tiempo de respuesta y la carga del sistema en tiempo real, para detectar y resolver rápidamente cualquier incidente que afecte la experiencia del usuario.

V.2.4. Implementación dentro del Plazo Establecido

- **Cumplimiento del Cronograma de 8 Semanas:**

Finalización de la infraestructura, con todas las configuraciones de alta disponibilidad, escalabilidad y rendimiento, dentro del plazo de 8 semanas. Esto incluye la puesta en marcha, pruebas de estrés, ajuste de configuraciones y validación del rendimiento general de la plataforma.

- **Entrega de Documentación Completa:**

Documentación de todas las configuraciones de Amazon EKS, incluyendo el sistema de escalado, las políticas de recuperación ante fallos, la configuración de redes y balanceo de carga, así como las prácticas recomendadas para el mantenimiento y expansión futura de la infraestructura.

V.2.5. Pruebas de Estrés y Validación Final

- **Pruebas de Estrés y Rendimiento de Carga para Validar el Despliegue:**

Realización de pruebas de carga que simulen diferentes niveles de concurrencia, desde 1,000 hasta 10,000 usuarios, para validar la capacidad de procesamiento y el tiempo de respuesta bajo diversas condiciones.

- **Evaluación de KPIs de Rendimiento (SLAs) Acordados:**

Monitoreo de los indicadores clave de rendimiento (como la disponibilidad, el tiempo de respuesta y la tasa de éxito en recuperación de fallos) para asegurar que la infraestructura cumple con el nivel de servicio esperado.

V.3. Objetivo específico 2:

Integrar un sistema de inteligencia artificial conversacional que procese 20 interacciones por segundo con un tiempo de respuesta inferior a 1 segundo y una precisión contextual del 90%, implementando un modelo de procesamiento de lenguaje natural optimizado para mantener coherencia narrativa en conversaciones extensas, completando su desarrollo y optimización en un período de 10 semanas.

V.4. Resultados Esperados

V.4.1. Soporte de 1,000 Transmisiones Simultáneas en Calidad 2K

- **Capacidad de Escalabilidad de Streaming:**

Implementación de un sistema de streaming en Jellyfin que soporte hasta 1,000 transmisiones simultáneas en calidad 2K sin degradación de la calidad de la imagen ni interrupciones.

- **Optimización de Contenedores para Múltiples Streams:**

Configuración de contenedores y ajustes de infraestructura que soporten de manera óptima la carga de 1,000 transmisiones, maximizando la eficiencia de recursos y evitando el sobreuso.

V.4.2. Reducción de Latencia en el Streaming (menos de 2 segundos)

- **Transmisión en Tiempo Real con Baja Latencia:**

Ajustes en los parámetros de red y de buffering para garantizar una latencia en el streaming inferior a 2 segundos desde la solicitud de reproducción hasta el inicio del contenido, validada mediante pruebas en condiciones de carga máxima.

- **Sistema de Monitoreo de Latencia en Tiempo Real:**

Configuración de un sistema que monitoree la latencia en tiempo real y alerte en caso de que el tiempo de respuesta supere el límite establecido, permitiendo ajustes proactivos.

V.4.3. Optimización de Recursos y Uso Eficiente (utilización máxima del 70%)

- **Utilización de Recursos del Sistema por debajo del 70%:**

Optimización de configuraciones de contenedores, redes y políticas de transcoding adaptativo para que el sistema funcione con un uso de recursos máximo del 70%, lo cual permitirá estabilidad, ahorro de costos y mayor capacidad de respuesta ante aumentos en la carga.

- **Implementación de Transcoding Adaptativo:**

Configuración de políticas de transcoding adaptativo en Jellyfin para ajustar automáticamente la calidad del stream según el ancho de banda disponible, manteniendo la experiencia del usuario sin exceder los recursos del sistema.

V.4.4. Cumplimiento del Cronograma de Implementación de 12 Semanas

- **Finalización Completa del Sistema de Streaming en el Plazo Establecido:** Implementación de todas las configuraciones y optimizaciones necesarias en un plazo de 12 semanas, incluyendo pruebas, ajustes y despliegue final en el entorno de producción.

- **Documentación Técnica Completa y Guías de Mantenimiento:**

Entrega de documentación detallada sobre la configuración del sistema, las políticas de transcoding, la arquitectura de los contenedores y las guías de mantenimiento para facilitar la gestión del sistema a largo plazo.

V.4.5. Pruebas de Usuario y Evaluación de Satisfacción

- **Realización de Pruebas de Usuario con Evaluación del Rendimiento:**

Ejecución de pruebas de usuario para medir la calidad de la transmisión, la latencia y la experiencia general de uso del sistema, verificando que el sistema cumpla con los parámetros de calidad esperados.

- **Logro de un Mínimo del 95% de Satisfacción de los Usuarios:**

Al menos un 95% de los usuarios en pruebas deberá reportar satisfacción en cuanto a calidad de transmisión, tiempos de carga y fluidez del sistema, asegurando una experiencia positiva en el consumo de contenido.

V.4.6. Pruebas de Estrés y Validación Final

- **Pruebas de Estrés para Verificar la Capacidad del Sistema:**

Ejecución de pruebas de estrés que simulen la concurrencia de 1,000 usuarios para asegurar que el sistema mantiene la calidad de transmisión 2K, la latencia baja y el uso de recursos optimizado bajo condiciones de carga total.

- **Evaluación de los Indicadores de Rendimiento y Calidad (SLAs) Establecidos:**

Validación de que los indicadores clave de rendimiento (latencia, calidad de transmisión, y uso de recursos) cumplen con los niveles de servicio requeridos para garantizar la experiencia esperada de los usuarios.

V.5. Objetivo específico 3:

Integrar un sistema de inteligencia artificial conversacional que procese 20 interacciones por segundo con un tiempo de respuesta inferior a 1 segundo y una precisión contextual del 90%, implementando un modelo de procesamiento de lenguaje natural optimizado para mantener coherencia narrativa en conversaciones extensas, completando su desarrollo y optimización en un período de 10 semanas.

V.6. Resultados Esperados

V.6.1. Capacidad de Procesamiento y Respuesta Rápida

- **Procesamiento de 20 Interacciones por Segundo:**

Implementación de una arquitectura escalable capaz de manejar 20 interacciones por segundo sin comprometer el rendimiento del sistema, asegurando estabilidad y continuidad del servicio en momentos de alta demanda.

- **Tiempo de Respuesta Inferior a 1 Segundo:**

Optimización del modelo y del flujo de procesamiento de lenguaje natural para responder a las consultas en menos de 1 segundo, garantizando una experiencia conversacional ágil y sin demoras.

V.6.2. Precisión Contextual del 90% en Respuestas

- **Precisión y Relevancia en las Respuestas Conversacionales:**

Desarrollo de un modelo de procesamiento de lenguaje natural (NLP) que mantenga una precisión contextual del 90% o más en sus respuestas, lo que implica que el sistema debe interpretar y responder de forma correcta en al menos 9 de cada 10 interacciones.

- **Evaluación Continua de Precisión mediante Retroalimentación:**

Implementación de mecanismos de retroalimentación automática y monitoreo de precisión para ajustar y mejorar continuamente el modelo de NLP, asegurando una tasa de precisión consistente.

V.6.3. Mantenimiento de la Coherencia Narrativa en Conversaciones Largas

- **Optimización para Coherencia en Conversaciones Extendidas:**

Ajustes en el modelo conversacional para mantener el contexto y la coherencia narrativa en interacciones prolongadas, permitiendo al usuario continuar conversaciones complejas sin perder el hilo o el contexto.

- **Pruebas de Coherencia Contextual en Conversaciones Múltiples:**

Realización de pruebas de interacción extendida para evaluar la capacidad del modelo de NLP de mantener contexto en hilos largos de conversación, mejorando la retención de información relevante para asegurar continuidad en la experiencia.

V.6.4. Cumplimiento del Cronograma de Desarrollo de 10 Semanas

- **Finalización Completa del Sistema en el Plazo de 10 Semanas:**

Culminación de todas las etapas de desarrollo, optimización y validación del sistema de inteligencia artificial conversacional dentro del período establecido de 10 semanas.

- **Documentación Técnica Completa y Guías de Optimización:**

Entrega de una documentación detallada sobre la arquitectura del sistema, las configuraciones del modelo de NLP y las prácticas recomendadas para ajustes futuros y mantenimiento del sistema de IA conversacional.

V.6.5. Pruebas de Estrés y Validación de Rendimiento Final

- **Pruebas de Estrés para Verificar la Capacidad del Sistema:**

Ejecución de pruebas de carga que simulan 20 interacciones por segundo, validando que el sistema mantiene un tiempo de respuesta inferior a 1 segundo sin disminución en la precisión y coherencia contextual.

- **Evaluación de KPIs Clave de Rendimiento (SLAs) para el Sistema Conversacional:**

Revisión de los indicadores de rendimiento principales (como tiempo de respuesta, precisión contextual, y coherencia narrativa) para asegurar que el sistema cumple con los estándares y expectativas del usuario.

V.6.6. Evaluación de Satisfacción en Pruebas de Usuario

- **Pruebas de Usuario para Validación de Calidad Conversacional:**

Realización de pruebas de usuario donde se mida la satisfacción general con la calidad de las respuestas, la rapidez y la coherencia en conversaciones largas, asegurando que el sistema responda de forma natural y adecuada.

- **Logro de al menos un 90% de Satisfacción en Pruebas de Usuario:**

Al menos el 90% de los usuarios en pruebas deben reportar satisfacción en cuanto a la precisión y la fluidez del sistema de IA conversacional, demostrando que la solución es efectiva y coherente en sus respuestas.

V.7. Objetivo específico 4:

Desarrollar una interfaz de usuario intuitiva y personalizada que permita a los usuarios navegar fácilmente entre los distintos contenidos y servicios disponibles en la plataforma.

V.8. Resultados esperados:

- Diseño de una interfaz de usuario adaptable y visualmente atractiva que se ajuste a dispositivos móviles, tabletas y ordenadores.
- Implementación de un sistema de recomendaciones personalizadas basado en el historial de visualización y preferencias del usuario, optimizando la experiencia de descubrimiento de contenido.
- Mejora en la experiencia de usuario, medida a través de métricas como tiempo de navegación, número de interacciones, tasa de retención y satisfacción.

- Implementación de funciones de accesibilidad, tales como compatibilidad con lectores de pantalla, para asegurar que la plataforma sea inclusiva para todos los usuarios.

V.9. Objetivo específico 5:

Implementar herramientas analíticas que permitan recolectar, analizar y visualizar datos sobre el comportamiento de los usuarios y el rendimiento de la plataforma.

V.10. Resultados esperados:

- Generación de reportes detallados sobre patrones de consumo, preferencias de los usuarios y métricas de desempeño de la plataforma.
- Desarrollo de dashboards personalizados para monitorear indicadores clave de rendimiento (KPIs) en tiempo real, como tiempo de visualización, tasa de retención de usuarios y popularidad de contenido.
- Obtención de insights para mejorar la experiencia del usuario y optimizar los algoritmos de recomendación.
- Identificación de segmentos de audiencia para el desarrollo de campañas de marketing más efectivas y orientadas.
- Monitoreo constante del rendimiento del sistema y detección temprana de posibles fallos o áreas de mejora en la plataforma.

VI. REFERENCIAS TÉCNICAS

1. Amazon Web Services. (2024). *Amazon EKS best practices guide*. https://aws.amazon.com
2. Jellyfin. (2024). *Jellyfin server documentation*. https://jellyfin.org
3. OpenAI. (2024). *GPT-4 API reference*. https://openai.com
4. Anderson, K., & Lee, S. (2024). Advances in conversational AI systems. *Journal of Artificial Intelligence Research, 65*, 1–15. https://doi.org
5. AWS Case Studies. (2024). Disney+ streaming platform architecture. *Amazon Web Services*. https://aws.amazon.com
6. Cloud Native Computing Foundation. (2024). *Cloud native survey 2024*. CNCF Research. https://cncf.io
7. Media Server Benchmarks. (2024). Comparative analysis of streaming platforms. *Tech Review Quarterly*.
8. Netflix Technology Blog. (2023). Evolution of the Netflix microservices architecture. https://netflixtechblog.com
9. OpenAI. (2024). *GPT-4 technical report*. OpenAI Research Publications. https://openai.com/research
10. Smith, R., & Johnson, M. (2023). The state of global streaming 2023. *Streaming Media Journal, 12*(4), 45–52.

11. Warner Media Tech. (2023). HBO Max: Building a global streaming platform. *Warner Media Technical Documentation*.

12. Zhang, L., et al. (2023). Conversational AI: State of the art and future directions. *ACM Computing Surveys, 56*(2), 1–38.