

**INSTITUTO TECNOLÓGICO AUTÓNOMO DE MÉXICO**



**DISEÑO E IMPLEMENTACIÓN DE UNA  
PLATAFORMA ONLINE DE RENTA DE EQUIPO DE  
CINE**

**T E S I S**

**QUE PARA OBTENER EL TÍTULO DE  
INGENIERO EN COMPUTACIÓN**

**P R E S E N T A**

**JOSÉ DE JESÚS SÁNCHEZ AGUILAR**

**ASESOR:**

**DR. JOSÉ OCTAVIO GUTIÉRREZ GARCÍA**

**CIUDAD DE MÉXICO**

**2025**

«Con fundamento en los artículos 21 y 27 de la Ley Federal del Derecho de Autor y como titular de los derechos moral y patrimonial de la obra titulada “**DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA ONLINE DE RENTA DE EQUIPO DE CINE**”, otorgo de manera gratuita y permanente al Instituto Tecnológico Autónomo de México y a la Biblioteca Raúl Baillères Jr., la autorización para que fijen la obra en cualquier medio, incluido el electrónico, y la divulguen entre sus usuarios, profesores, estudiantes o terceras personas, sin que pueda percibir por tal divulgación una contraprestación.»

---

Fecha

---

José de Jesús Sánchez Aguilar

# Índice

<b>Introducción</b>	<b>6</b>
1.1 Contexto	6
1.2 Identificación del Problema	7
1.3 Objetivos	9
1.4 Organización del documento	11
<b>Análisis</b>	<b>13</b>
2.1 Análisis de Requerimientos	13
2.1.1 Requerimientos Funcionales	13
2.1.2 Requerimientos No Funcionales	14
2.2 Análisis de Restricciones	15
<b>Diseño</b>	<b>17</b>
3.1 Soluciones Alternativas	17
3.2 Descripción completa del diseño	18
1. Frontend (Interfaz de Usuario)	18
2. Backend Lógico (Google Cloud Functions)	20
3. Autenticación (Firebase Auth)	23
4. Almacenamiento de Objetos (Firebase Storage)	25
5. Integración Continua y Despliegue Continuo (CI/CD)	27
6. Integraciones y Servicios Externos	29
6.1 Procesamiento de Pagos (Stripe)	30
6.2 Orquestación de CI/CD (GitHub Actions)	31
3.3. Estándares Utilizados	32
<b>Implementación</b>	<b>35</b>
4.1 Especificaciones de Software y Hardware	35
4.2 Descripción completa de la implementación	35
<b>Pruebas y resultados</b>	<b>35</b>
5.1 Descripción de las pruebas	36

5.2 Resultados	36
5. 3 Análisis de resultados	36
<b>Conclusiones</b>	<b>36</b>
<b>Referencias</b>	<b>36</b>

# Capítulo 1

## Introducción

### 1.1 Contexto

La industria audiovisual se encuentra en un momento de expansión sin precedentes. El acceso a nuevas tecnologías, la proliferación de plataformas digitales y el creciente interés del público por contenidos audiovisuales de diversa índole han impulsado la producción de cine en todo el mundo, especialmente en el ámbito de los cortometrajes. Este formato, por su menor duración y presupuesto, se ha convertido en una puerta de entrada para nuevos talentos y un espacio ideal para la experimentación y la innovación.

Sin embargo, a pesar del auge de la producción audiovisual, el acceso a equipo de cine profesional sigue siendo un obstáculo significativo para muchos cineastas independientes y estudiantes. El alto costo de cámaras, lentes, iluminación y otros equipos esenciales limita la capacidad de las pequeñas producciones para materializar sus proyectos con la calidad que desean. Si bien la compra de equipo puede ser una opción para grandes productoras, resulta inviable para la mayoría de los realizadores de cortometrajes, quienes a menudo se ven obligados a sacrificar la calidad técnica o a renunciar a sus proyectos por falta de recursos.

En este contexto, la renta de equipo se presenta como una alternativa viable y accesible para democratizar el acceso a la tecnología y fomentar la producción audiovisual. Alquilar el equipo necesario en lugar de comprarlo permite a los cineastas reducir costos, acceder a tecnología de punta y concentrar sus

recursos en otros aspectos de la producción, como el desarrollo del guión, la dirección de actores o la postproducción.

La creciente digitalización de la industria ha facilitado la conexión entre proveedores y clientes a través de plataformas online. Estas plataformas permiten a los usuarios buscar y comparar equipos, gestionar reservas y realizar pagos de forma rápida y segura. Sin embargo, aún existe una necesidad por plataformas especializadas que se enfoquen en las necesidades específicas de las pequeñas producciones de cine, ofreciendo equipos adecuados a sus presupuestos y proyectos, y brindando un servicio personalizado y accesible.

El presente trabajo se centra en el diseño de una plataforma de renta de equipo de cine que busca responder a esta necesidad, facilitando el acceso a tecnología de calidad para pequeñas producciones y cortometrajes.

## **1.2 Identificación del Problema**

El cine, como forma de expresión artística y medio de comunicación, ha experimentado un crecimiento exponencial en las últimas décadas. La democratización de la tecnología y la proliferación de plataformas digitales han impulsado la creación audiovisual, especialmente en el ámbito de las pequeñas producciones y cortometrajes. Sin embargo, este florecimiento creativo se ve obstaculizado por una serie de limitaciones que dificultan el acceso a los recursos necesarios para la producción cinematográfica.

Una de las principales barreras que enfrentan los cineastas independientes y estudiantes es la limitación financiera. La adquisición de equipo profesional de cine, como cámaras, lentes,

iluminación y sonido, implica una inversión considerable que muchas veces resulta inaccesible para producciones de bajo presupuesto. Esta restricción económica puede obligar a los realizadores a sacrificar la calidad técnica de sus proyectos o incluso a abandonar sus ideas por falta de recursos.

La falta de capital no solo limita la compra de equipo, sino que también restringe el acceso a la tecnología. Las pequeñas producciones se ven obligadas a trabajar con equipo obsoleto o de gama baja, lo que puede afectar la calidad de la imagen, el sonido y la postproducción. Esto limita las posibilidades creativas de los cineastas y reduce la competitividad de sus proyectos en un mercado cada vez más exigente.

Si bien la renta de equipo se presenta como una alternativa viable para solventar estas limitaciones, las pequeñas producciones también enfrentan dificultades para encontrar proveedores confiables. La oferta de equipo en renta puede ser escasa, especialmente en ciudades con menor desarrollo de la industria cinematográfica. Además, es común encontrar precios elevados, equipos en mal estado o falta de disponibilidad en las fechas requeridas.

En este contexto, se identifica la falta de una plataforma especializada que facilite la conexión entre proveedores de equipo y pequeñas producciones. Si bien existen plataformas de renta de equipo en general, no hay una que se enfoque específicamente en las necesidades de la industria cinematográfica, ofreciendo una selección de equipo curada, precios accesibles y un proceso de renta transparente y eficiente.

Esta problemática limita el desarrollo de las pequeñas producciones y cortometrajes, que representan un espacio fundamental para la experimentación, la innovación y el surgimiento de nuevos talentos en la industria cinematográfica.

### 1.3 Objetivos

El presente proyecto tiene como objetivo principal el diseño y desarrollo de una plataforma online de renta de equipo de cine robusta, escalable y segura, que permita la gestión eficiente de inventario y la interacción fluida entre proveedores y clientes.

Para lograr este objetivo general, se han definido los siguientes objetivos específicos:

- **Reducir los costos de mantenimiento a través del uso estratégico de recursos en la nube:** Se diseñará la arquitectura de la aplicación para minimizar los costos de mantenimiento, aprovechando al máximo las capas gratuitas y los recursos de bajo costo ofrecidos por los proveedores de la nube. Se implementarán estrategias de escalabilidad que permitan ajustar los recursos en función de la demanda, optimizando el uso de la infraestructura y controlando los costos operativos.
- **Desarrollar una arquitectura de software eficiente:** Se definirá la arquitectura de la plataforma, incluyendo la selección de tecnologías de desarrollo, bases de datos, servidores y protocolos de comunicación, para garantizar un rendimiento óptimo, escalabilidad y seguridad.
- **Diseñar una base de datos robusta:** Se diseñará una base de datos que permita almacenar y gestionar de forma eficiente la información del equipo de cine (especificaciones



técnicas, disponibilidad, historial de rentas, etc.), los usuarios de la plataforma (proveedores y clientes) y las transacciones.

- **Implementar un sistema de gestión de inventario:** Se desarrollará un sistema que permita a los proveedores administrar su inventario de equipo de cine, incluyendo la actualización de la disponibilidad, el registro de mantenimiento y la generación de reportes.
- **Crear una interfaz de usuario intuitiva y accesible:** Se diseñará una interfaz de usuario (UI) que sea fácil de usar y navegar, con un diseño responsivo que se adapte a diferentes dispositivos (computadoras, tablets, smartphones), y que cumpla con los estándares de accesibilidad web.
- **Implementar un sistema de búsqueda y filtrado avanzado:** Se desarrollará un sistema que permita a los clientes buscar equipo de cine de forma rápida y precisa, utilizando diferentes criterios de búsqueda (tipo de equipo, marca, modelo, precio, disponibilidad) y filtros avanzados.
- **Desarrollar un sistema de reservas y pagos seguro:** Se implementará un sistema que permita a los clientes realizar reservas de equipo de forma segura, gestionar sus reservas y realizar pagos online a través de pasarelas de pago confiables.
- **Implementar medidas de seguridad:** Se implementarán medidas de seguridad para proteger la información de los usuarios y las transacciones, incluyendo la encriptación de datos, la autenticación de usuarios y la protección contra ataques cibernéticos.
- **Realizar pruebas de usabilidad y rendimiento:** Se realizarán pruebas para evaluar la usabilidad de la plataforma, la eficiencia del sistema de búsqueda, la

seguridad de las transacciones y el rendimiento general del sistema bajo diferentes condiciones de carga.

Alcanzar estos objetivos permitirá construir una plataforma online de renta de equipo de cine confiable, eficiente y segura, que satisfaga las necesidades tanto de proveedores como de clientes.

El cumplimiento de estos objetivos permitirá crear una plataforma online de renta de equipo de cine que sea funcional, accesible y que responda a las necesidades específicas de las pequeñas producciones audiovisuales.

## **1.4 Organización del documento**

Esta tesis se organiza en cinco capítulos principales que guían al lector a través del proceso de diseño, desarrollo e implementación de la plataforma de renta de equipo de cine. A continuación, se presenta una breve descripción de cada capítulo:

**Capítulo 1: Introducción.** Este capítulo proporciona el contexto general de la tesis, planteando la problemática que se busca resolver y los objetivos que se pretenden alcanzar. Se define el alcance del proyecto y se presenta la organización del documento.

**Capítulo 2: Análisis.** En este capítulo se realiza un análisis exhaustivo de los requerimientos de la plataforma. Se estudian las necesidades de los usuarios (proveedores y clientes), se investigan las tecnologías de desarrollo más adecuadas y se define la arquitectura de la plataforma, incluyendo la selección de bases de datos, servidores y protocolos de comunicación.

**Capítulo 3: Diseño.** Este capítulo se centra en el diseño de la plataforma. Se describe el diseño de la base de datos, la interfaz de

usuario (UI) y las funcionalidades principales del sistema, como la gestión de inventario, el sistema de búsqueda y filtrado, el proceso de reservas y el sistema de pagos. Se incluyen diagramas, mockups y prototipos para ilustrar el diseño. Esta información sirve como referencia y complemento a la documentación del sistema (Apéndice A).

**Capítulo 4: Implementación.** En este capítulo se detalla el proceso de implementación de la plataforma. Se describe el entorno de desarrollo, las herramientas utilizadas, el código fuente desarrollado y la configuración de la infraestructura en la nube. Se documentan las decisiones tomadas durante la implementación y se explican las soluciones a los desafíos encontrados.

**Capítulo 5: Pruebas y resultados.** Este capítulo se dedica a la evaluación de la plataforma. Se describen las pruebas realizadas para verificar la funcionalidad, la usabilidad, el rendimiento y la seguridad del sistema. Se presentan los resultados obtenidos en las pruebas y se realiza un análisis de los mismos.

**Capítulo 6: Conclusiones.** En este capítulo final se presentan las conclusiones del trabajo, resumiendo los logros alcanzados, las limitaciones encontradas y las posibles líneas de trabajo futuro. Se reflexiona sobre el impacto del proyecto y se discuten las contribuciones a la ingeniería de software.

Esta estructura permite una presentación clara y ordenada del proceso de desarrollo de la plataforma, facilitando la comprensión del proyecto y sus resultados.

# Capítulo 2

## Análisis

Este capítulo se centra en el análisis de los requerimientos y restricciones que influyen en el diseño e implementación de la plataforma de renta de equipo de cine. Este análisis es fundamental para asegurar que la plataforma satisfaga las necesidades de los usuarios y se ajuste a las limitaciones del proyecto.

### 2.1 Análisis de Requerimientos

En esta sección se identifican y detallan las necesidades y expectativas de los usuarios de la plataforma, tanto de los proveedores de equipo como de los clientes que lo rentan.

#### 2.1.1 Requerimientos Funcionales

Describen las funciones específicas que debe realizar la plataforma para cumplir con los objetivos del proyecto.

##### Para proveedores:

- **Gestión de perfil:** Permitir a los proveedores crear, editar y gestionar su perfil, incluyendo información de contacto, ubicación y datos de la empresa.
- **Publicación de anuncios:** Facilitar la publicación de anuncios de equipo de cine, incluyendo la descripción del equipo, fotos, precio de renta y disponibilidad.
- **Gestión de inventario:** Permitir a los proveedores gestionar su inventario de equipo, incluyendo la actualización de la disponibilidad, el registro de mantenimiento y la generación de reportes.

- **Procesamiento de solicitudes:** Gestionar las solicitudes de renta, incluyendo la comunicación con los clientes, la confirmación de reservas y la gestión de pagos.

#### **Para clientes:**

- **Búsqueda y filtrado:** Permitir a los clientes buscar equipo de cine utilizando diferentes criterios de búsqueda (tipo de equipo, marca, modelo, precio, disponibilidad) y filtros avanzados.
- **Visualización de información:** Mostrar información detallada del equipo, incluyendo especificaciones técnicas, fotos y condiciones de renta.
- **Gestión de reservas:** Permitir a los clientes realizar reservas de equipo, gestionar sus reservas (modificar fechas, cancelar reservas) y realizar el seguimiento de sus pedidos.
- **Sistema de pagos:** Facilitar la realización de pagos seguros online a través de pasarelas de pago confiables.
- **Comunicación con proveedores:** Permitir la comunicación directa entre clientes y proveedores a través de la plataforma.

#### **2.1.2 Requerimientos No Funcionales**

Describen las características de calidad que debe tener la plataforma para asegurar una buena experiencia de usuario y un funcionamiento óptimo.

- **Usabilidad:** La interfaz de usuario debe ser intuitiva, fácil de usar y accesible para usuarios con diferentes niveles de experiencia.
- **Rendimiento:** La plataforma debe ser rápida y eficiente en el procesamiento de información y la ejecución de las funciones.

- **Seguridad:** Se deben implementar medidas de seguridad para proteger la información de los usuarios, las transacciones y la integridad del sistema.
- **Escalabilidad:** La plataforma debe ser capaz de manejar un número creciente de usuarios y transacciones sin afectar el rendimiento.
- **Disponibilidad:** La plataforma debe estar disponible en todo momento, con un alto nivel de uptime.
- **Fiabilidad:** La plataforma debe ser confiable y libre de errores, minimizando las fallas y garantizando la integridad de los datos.

## 2.2 Análisis de Restricciones

En esta sección se identifican las limitaciones que pueden afectar el desarrollo e implementación de la plataforma.

### Restricciones técnicas:

- Limitaciones en la capacidad de procesamiento y almacenamiento de los servidores.
- Compatibilidad con diferentes dispositivos, navegadores y sistemas operativos.
- Complejidad en la integración de pasarelas de pago.

### Restricciones económicas:

- Presupuesto limitado para el desarrollo y mantenimiento de la plataforma.
- Costos de las licencias de software y servicios en la nube.

### Restricciones legales:

- Cumplimiento de las leyes y regulaciones relacionadas con el comercio electrónico, la protección de datos personales y la renta de bienes.
- Obtención de permisos y licencias para operar la plataforma.

Este análisis de requerimientos y restricciones proporciona una base sólida para el diseño e implementación de la plataforma, permitiendo tomar decisiones informadas y desarrollar una solución que satisfaga las necesidades de los usuarios y se ajuste a las limitaciones del proyecto.

# Capítulo 3

## Diseño

### 3.1 Soluciones Alternativas

En la búsqueda de soluciones para facilitar el acceso a equipos de cine a bajo costo, se exploraron diversos enfoques existentes. A nivel internacional, se identificó la plataforma ShareGrid, un mercado en línea que conecta a dueños de equipos con arrendatarios en los Estados Unidos. Sin embargo, hasta el momento, no existe una solución equivalente con la misma estructura y alcance en México.

Actualmente, la renta de equipo de cine en México se realiza principalmente a través de dos vías informales. La primera involucra redes y comunidades de pequeños cineastas y estudiantes, donde el intercambio y la renta de equipos se gestionan de manera descentralizada y, a menudo, con un alcance limitado a los círculos cercanos. Si bien estas redes pueden facilitar el acceso a equipos a costos más accesibles, suelen carecer de la formalidad, la seguridad y la variedad de inventario que una plataforma dedicada podría ofrecer.

Otra alternativa identificada es la renta de equipos a través de casas productoras. Algunas productoras publicitan sus equipos disponibles para renta, pero esta opción presenta limitaciones significativas. El catálogo de equipos se restringe al inventario propio de la casa productora, lo que reduce considerablemente las opciones disponibles para los cineastas. Además, al tratarse de empresas con una estructura de costos mayor, las tarifas de renta



pueden resultar elevadas, especialmente para producciones independientes con presupuestos ajustados.

Estas limitaciones en las soluciones existentes en México resaltan la necesidad de una plataforma como la que se propone en este proyecto, que pueda centralizar la oferta y demanda de equipos de cine, facilitar transacciones seguras y ofrecer precios competitivos para la comunidad de cineastas con recursos limitados.

## **3.2 Descripción completa del diseño**

La arquitectura de la plataforma de renta de equipo de cine está diseñada para ser escalable, robusta y costo-efectiva, aprovechando al máximo los servicios de la nube de Google Cloud Platform (GCP) y Firebase. Se optó por una arquitectura predominantemente serverless (sin servidor), lo que minimiza la sobrecarga de gestión de infraestructura y permite que la lógica del backend se ejecute bajo demanda.

### **1. Frontend (Interfaz de Usuario)**

El frontend de la plataforma, la capa con la que los usuarios interactúan directamente, se construyó con **Vue.js** y **Tailwind CSS**.

**Propósito Principal:** Proporcionar una experiencia de usuario intuitiva, reactiva y visualmente atractiva. Es responsable de presentar el catálogo de equipos, guiar a los usuarios a través del proceso de renta, permitir la gestión de perfiles y publicaciones para los arrendadores, y reflejar el estado actual de la plataforma.

**Interacción con Otros Componentes:** Se comunica principalmente con el Backend Lógico (Google Cloud Functions) a

través de solicitudes HTTP (API RESTful). Envía datos de usuario y solicitudes de operación (ej. búsqueda, solicitud de renta, actualización de perfil) a las funciones y recibe respuestas para actualizar la interfaz. También interactúa directamente con Firebase Auth para el flujo de autenticación y con Firebase Storage para la carga y visualización de imágenes.

**Alojamiento:** El frontend se aloja de forma estática en Firebase Hosting, lo que garantiza una alta disponibilidad y baja latencia. Adicionalmente, se configuró un dominio personalizado y su DNS se conectó a través de Firebase Hosting, proporcionando una dirección web profesional y fácil de recordar para la plataforma.

### **Funcionalidades Específicas:**

- **Navegación y Búsqueda de Equipos:** Permite a los usuarios explorar el inventario completo, aplicar filtros por tipo de equipo, marca, precio, ubicación y fechas de disponibilidad.
- **Detalles del Producto:** Muestra información exhaustiva de cada equipo, incluyendo descripciones, especificaciones técnicas, precios por día/semana, galería de imágenes y datos del arrendador.
- **Proceso de Reserva y Renta:** Guía a los usuarios a través de los pasos necesarios para seleccionar las fechas de renta, revisar los costos y el resumen de la orden antes de iniciar el proceso de pago.
- **Gestión de Perfil de Usuario:** Permite a los usuarios actualizar su información personal, ver su historial de rentas (como arrendatario) y gestionar sus publicaciones (como arrendador).

- **Publicación y Gestión de Equipos (para Arrendadores):** Ofrece una interfaz para que los dueños de equipos puedan añadir nuevos ítems al catálogo, subir imágenes, definir precios, marcar disponibilidad y gestionar las solicitudes de renta de sus equipos.
- **Responsividad:** Gracias a Tailwind CSS, la interfaz se adapta fluidamente a diferentes dispositivos (escritorio, tabletas, móviles), optimizando la experiencia de usuario sin importar el tamaño de pantalla.

## 2. Backend Lógico (Google Cloud Functions)

El corazón de la lógica de negocio y la interactividad de la plataforma reside en las **Google Cloud Functions**. Este servicio serverless de Google Cloud Platform permite ejecutar código de backend en un entorno escalable y gestionado, eliminando la necesidad de provisionar y mantener servidores dedicados. Las funciones se activan en respuesta a diversos eventos, lo que facilita una arquitectura basada en eventos y altamente desacoplada.

**Propósito Principal:** Las Google Cloud Functions son responsables de ejecutar toda la lógica de negocio crítica y las operaciones del lado del servidor que no pueden o no deben ser realizadas directamente por el frontend. Esto incluye el procesamiento seguro de transacciones, la automatización de tareas y la orquestación entre los diferentes servicios de Firebase y Google Cloud Platform.

### Interacción con Otros Componentes:

- **Frontend (Interfaz de Usuario):** Reciben solicitudes HTTP del frontend (actuando como una API RESTful) para procesar acciones iniciadas por el usuario, como la creación

de una solicitud de renta, la actualización de un perfil o la publicación de un nuevo equipo.

- **Base de Datos (Firestore):** Interactúan intensivamente con Firestore para leer, escribir, actualizar y eliminar datos de usuarios, equipos, transacciones y cualquier otra información estructurada de la aplicación.
- **Almacenamiento de Objetos (Firebase Storage):** Acceden a Firebase Storage para gestionar archivos, por ejemplo, para procesar imágenes subidas (ej. redimensionarlas o crear miniaturas) o para manejar documentos de usuarios.
- **Servicios Externos (Stripe):** Se integran con pasarelas de pago externas, específicamente **Stripe**, para manejar todo el ciclo de vida de las transacciones monetarias.
- **Tareas Programadas (Cloud Scheduler):** Son invocadas por Cloud Scheduler para ejecutar lógica de negocio de forma recurrente y automatizada.
- **Servicios de Correo Electrónico:** Se utilizan para enviar notificaciones transaccionales y de marketing a los usuarios.

### Funcionalidades Específicas:

- **Procesamiento de Pagos Seguro:** Una de las funcionalidades más críticas es la gestión de pagos. Las Cloud Functions utilizan la **extensión oficial de Stripe para Firebase** para procesar los pagos de las rentas de forma segura y compatible con los estándares PCI. Esto incluye la creación de intenciones de pago, la confirmación de cargos, la gestión de reembolsos y la sincronización del estado de los pagos con la base de datos de Firestore. Al manejar los pagos en el backend, se garantiza que la información sensible de la tarjeta de crédito nunca toque el frontend de la aplicación.

- **Envío de Notificaciones Automatizadas:** Implementan la lógica para enviar correos electrónicos transaccionales a los usuarios en puntos clave del flujo de renta. Esto incluye confirmaciones de reserva, notificaciones de aceptación o rechazo de solicitudes, recordatorios de fechas de entrega o devolución de equipos, y recibos de pago.
- **Gestión Automatizada del Catálogo:** Contienen lógica para realizar actualizaciones automatizadas en el inventario de equipos. Por ejemplo, pueden cambiar el estado de una publicación de "disponible" a "rentado" una vez que una orden es confirmada, o dar de baja automáticamente publicaciones que han expirado o que no cumplen con ciertos criterios después de un período de inactividad.
- **Actualización del Estado de Órdenes:** Las funciones monitorean y actualizan el estado de las órdenes de renta basándose en fechas programadas o eventos de usuario, asegurando que la información sea siempre precisa.
- **Validación y Saneamiento de Datos:** Antes de que los datos enviados desde el frontend se escriban en Firestore o Storage, las Cloud Functions realizan validaciones y saneamiento para asegurar la integridad, formato correcto y seguridad de la información.
- **Sincronización de Datos:** Permiten la sincronización de datos complejos entre diferentes colecciones de Firestore o la agregación de información para reportes.

En resumen, las Google Cloud Functions actúan como el cerebro operativo de la plataforma, permitiendo la ejecución de lógica de negocio compleja, transacciones seguras y automatizaciones, todo ello dentro de un entorno sin servidor altamente escalable y eficiente en costos.

### 3. Autenticación (Firebase Auth)

Para la gestión de identidades y el control de acceso de los usuarios, la plataforma se apalanca en **Firebase Authentication (Firebase Auth)**. Este servicio robusto y fácil de implementar de Firebase proporciona una solución completa para registrar y autenticar usuarios, eliminando la necesidad de construir y mantener un sistema de autenticación desde cero.

**Propósito Principal:** Firebase Auth tiene como objetivo principal asegurar que solo los usuarios legítimos puedan acceder a la plataforma y a sus funcionalidades. Maneja de forma segura el proceso de registro, inicio de sesión y la gestión de sesiones de usuario, proporcionando una identidad verificable para cada usuario.

#### **Interacción con Otros Componentes:**

- **Frontend (Interfaz de Usuario):** El frontend utiliza el SDK de Firebase Auth para interactuar directamente con el servicio. Esto incluye la presentación de interfaces de usuario para el registro y el inicio de sesión, la captura de credenciales del usuario y la gestión del estado de autenticación (ej., redirigiendo al usuario una vez que ha iniciado sesión).
- **Base de Datos (Firestore) y Almacenamiento de Objetos (Firebase Storage):** Una vez que un usuario es autenticado, Firebase Auth proporciona un token de autenticación. Este token es automáticamente utilizado por los SDKs de Firebase en el frontend y en las Cloud Functions para interactuar de forma segura con Firestore y Firebase Storage. Las **Reglas de Seguridad de Firebase** para estas bases de

datos y almacenamiento utilizan la información de autenticación proporcionada por este token (como el `uid` del usuario) para permitir o denegar el acceso a datos específicos.

- **Backend Lógico (Google Cloud Functions):** Las Cloud Functions pueden verificar la autenticación del usuario que realiza una solicitud HTTP para asegurar que solo los usuarios autenticados y autorizados puedan ejecutar ciertas operaciones críticas (ej., procesar un pago, modificar una publicación). El `uid` del usuario autenticado se puede acceder dentro de la función para realizar validaciones basadas en la identidad.

### Funcionalidades Específicas:

- **Registro e Inicio de Sesión por Correo Electrónico y Contraseña:** Permite a los usuarios crear una cuenta y acceder utilizando su dirección de correo electrónico y una contraseña segura, con funcionalidades incorporadas para la recuperación de contraseña.
- **Inicio de Sesión con Google:** Facilita el proceso de registro e inicio de sesión al permitir a los usuarios utilizar sus cuentas existentes de Google, lo que mejora la conveniencia y reduce la fricción en el proceso de onboarding.
- **Verificación por Número Telefónico (SMS):** Implementa un sistema de autenticación adicional que permite a los usuarios verificar su identidad mediante el envío de un código SMS a su número de teléfono. Esto puede ser útil para agregar una capa extra de seguridad o para usuarios que prefieren esta opción.
- **Gestión de Sesiones:** Firebase Auth maneja automáticamente las sesiones de usuario, persistiendo el

estado de autenticación entre diferentes visitas o recargas de página, mejorando la usabilidad.

- **Seguridad Integrada:** Proporciona medidas de seguridad robustas contra ataques comunes como fuerza bruta y reutilización de credenciales, y gestiona el almacenamiento seguro de las credenciales de usuario.

En síntesis, Firebase Auth es fundamental para establecer un entorno seguro y confiable dentro de la plataforma, garantizando que cada interacción del usuario esté ligada a una identidad verificada y que los accesos a los recursos sean debidamente controlados.

#### **4. Almacenamiento de Objetos (Firebase Storage)**

Para la gestión eficiente de archivos no estructurados, como imágenes y documentos, la plataforma utiliza **Firebase Storage**. Este servicio robusto, escalable y seguro de Google Cloud permite almacenar y servir contenido generado por el usuario con gran fiabilidad.

**Propósito Principal:** Proporcionar una solución de almacenamiento segura y escalable para todos los archivos binarios que la aplicación necesita. Esto incluye, de manera primordial, las fotografías de los equipos de cine listados en la plataforma, así como cualquier otro documento relacionado con los usuarios, como podrían ser identificaciones para verificación (si fuera el caso en el futuro).

#### **Interacción con Otros Componentes:**

- **Frontend (Interfaz de Usuario):** Los usuarios interactúan con Firebase Storage a través del frontend para subir



imágenes de sus equipos cuando crean o editan una publicación. El SDK de Firebase Storage en el frontend facilita esta interacción, manejando la carga de archivos directamente desde el navegador del usuario al bucket de Storage. El frontend también recupera y muestra las URLs de las imágenes almacenadas para exhibirlas en el catálogo.

- **Backend Lógico (Google Cloud Functions):** Aunque la carga inicial suele ser desde el frontend, las Cloud Functions pueden interactuar con Firebase Storage para realizar tareas de procesamiento de archivos (ej. redimensionamiento de imágenes, creación de miniaturas) después de una carga, o para gestionar documentos de forma automatizada (ej. eliminar archivos relacionados con publicaciones inactivas).
- **Reglas de Seguridad de Firebase:** La seguridad de Firebase Storage se gestiona a través de reglas personalizadas que determinan quién puede leer y escribir archivos. Estas reglas utilizan la información de autenticación de **Firebase Auth** para permitir o denegar el acceso a diferentes rutas de almacenamiento (ej. un usuario solo puede escribir en su propia carpeta de imágenes).

### Funcionalidades Específicas:

- **Almacenamiento de Imágenes de Productos:** La función principal de Firebase Storage es alojar las fotografías de alta calidad de los equipos de cine que los arrendadores suben. Estas imágenes son cruciales para que los potenciales arrendatarios puedan visualizar y evaluar el equipo.
- **Almacenamiento de Documentos de Usuario:** Puede ser utilizado para almacenar documentos de verificación de identidad de los usuarios, contratos de renta digitalizados o

cualquier otro archivo relevante que necesite ser asociado a un usuario o transacción específica.

- **Escalabilidad Automática:** Firebase Storage se escala automáticamente para manejar cualquier volumen de carga y descarga de archivos, asegurando un rendimiento óptimo sin importar el número de usuarios o el tamaño del catálogo.
- **Seguridad de Archivos:** A través de las reglas de seguridad de Firebase, se controla el acceso a los archivos. Por ejemplo, las imágenes públicas del catálogo pueden ser accesibles para todos, mientras que los documentos privados de un usuario solo pueden ser vistos por el propietario del documento y por la administración de la plataforma.
- **Acceso Global y Baja Latencia:** Al ser parte de Google Cloud, los datos almacenados en Firebase Storage se replican globalmente, lo que garantiza una baja latencia y una alta disponibilidad para los usuarios en diferentes ubicaciones geográficas.

En resumen, Firebase Storage proporciona una solución de almacenamiento de objetos robusta y segura, esencial para el manejo de contenido multimedia y documental de la plataforma, integrándose de manera fluida con el resto del ecosistema de Firebase.

## **5. Integración Continua y Despliegue Continuo (CI/CD)**

Para garantizar un ciclo de desarrollo ágil y una entrega de software rápida y confiable, la plataforma implementa un proceso de Integración Continua (CI) y Despliegue Continuo (CD) utilizando **GitHub Actions** en conjunto con **Firebase Hosting**.

**Propósito Principal:** El objetivo de este pipeline de CI/CD es automatizar los pasos de construcción, prueba (implícito en la fase de "build" si hubiera tests unitarios) y despliegue de la aplicación frontend. Esto minimiza los errores humanos, reduce el tiempo entre el desarrollo de una característica y su disponibilidad en producción, y mantiene el código base en un estado desplegable en todo momento.

### **Interacción con Otros Componentes:**

- **Repositorio de GitHub:** El proceso se inicia con eventos específicos en el repositorio de código fuente en GitHub (commits, pull requests).
- **Frontend (Vue.js):** GitHub Actions es responsable de tomar el código fuente del frontend desarrollado en Vue.js, construirlo y prepararlo para su despliegue.
- **Firebase Hosting:** Una vez que la construcción es exitosa, GitHub Actions interactúa directamente con Firebase Hosting para cargar y hacer disponible la versión más reciente de la aplicación web estática.

### **Funcionalidades Específicas del Pipeline:**

- **Automatización del Build:** Cada vez que se realiza un **push** a una rama específica (comúnmente **main** o **master**) o se abre/actualiza un **pull request**, GitHub Actions se activa automáticamente. Este flujo de trabajo instala las dependencias del proyecto (Node.js, npm/yarn) y ejecuta el comando de construcción (**npm run build** o **yarn build**) para compilar la aplicación Vue.js en archivos estáticos optimizados.

- **Despliegue Automatizado a Firebase Hosting:** Tras una construcción exitosa, el flujo de trabajo procede a desplegar los archivos estáticos generados al entorno de **Firebase Hosting**. Esto se logra utilizando la acción oficial de Firebase CLI o una acción personalizada que interactúa con la cuenta de Firebase del proyecto. El despliegue a Firebase Hosting también incluye la configuración de la CDN global y la gestión de certificados SSL para el dominio personalizado previamente configurado.
- **Flujo de Trabajo Robusto:** La configuración de GitHub Actions permite definir pasos condicionales, como ejecutar el despliegue solo si la construcción es exitosa, o desplegar a un entorno de **staging** para **pull requests** y a **producción** para **pushes** a la rama principal. Esto asegura que solo el código estable llegue a los usuarios finales.
- **Retroalimentación Rápida:** Los desarrolladores reciben retroalimentación inmediata sobre el éxito o el fracaso de las compilaciones y despliegues directamente en la interfaz de GitHub, lo que facilita la identificación y corrección temprana de problemas.

Este enfoque de CI/CD, al integrar GitHub Actions con Firebase Hosting, establece un flujo de trabajo de desarrollo eficiente que soporta la entrega continua de nuevas características y mejoras a la plataforma, manteniendo la fiabilidad y consistencia del despliegue.

## 6. Integraciones y Servicios Externos

Además de la robusta infraestructura proporcionada por Google Cloud Platform y Firebase, la plataforma integra servicios externos

clave para ampliar sus funcionalidades y garantizar operaciones críticas.

## 6.1 Procesamiento de Pagos (Stripe)

Para manejar todas las transacciones monetarias relacionadas con la renta de equipos, la plataforma se integra con **Stripe**, una de las principales pasarelas de pago a nivel mundial.

**Propósito Principal:** Proporcionar una solución segura, escalable y conforme a las regulaciones para procesar pagos en línea, permitiendo a los arrendatarios pagar por las rentas y a los arrendadores recibir sus ganancias.

**Interacción con la Plataforma:** La integración con Stripe se realiza principalmente a través de las **Google Cloud Functions**. Cuando un usuario inicia una transacción de renta en el frontend, la solicitud se envía a una Cloud Function, que a su vez se comunica con la API de Stripe (utilizando la extensión oficial de Stripe para Firebase). Esta arquitectura asegura que la información sensible de la tarjeta de crédito nunca toque los servidores de la aplicación, minimizando los riesgos de seguridad y la carga de cumplimiento de PCI.

### Funcionalidades Específicas:

- **Creación de Cargos:** Permite la creación y procesamiento de cargos a las tarjetas de crédito o débito de los arrendatarios por el monto total de la renta.
- **Gestión de Reembolsos:** Facilita la emisión de reembolsos parciales o totales en caso de cancelaciones o disputas.
- **Manejo de Webhooks:** Stripe envía webhooks a las Cloud Functions para notificar sobre el estado de los pagos (ej.,

pago exitoso, fallo, disputa), lo que permite a la plataforma actualizar el estado de las órdenes en Firestore en tiempo real.

- **Seguridad y Cumplimiento:** Stripe se encarga de la seguridad de los datos de pago y del cumplimiento de normativas como PCI DSS, aliviando esta responsabilidad de la plataforma.

## 6.2 Orquestación de CI/CD (GitHub Actions)

Aunque ya se mencionó en el contexto del despliegue del frontend, **GitHub Actions** merece ser destacado como un servicio externo fundamental para la automatización del flujo de trabajo de desarrollo y despliegue de la plataforma.

**Propósito Principal:** Proporcionar una plataforma de automatización de flujos de trabajo directamente integrada en el repositorio de código, permitiendo la implementación de prácticas de Integración Continua y Despliegue Continuo (CI/CD).

**Interacción con la Plataforma:** GitHub Actions se configura en el repositorio de código de la aplicación. Cuando se detectan eventos específicos (como un **push** a una rama, la apertura de un **pull request** o la creación de un *release*), GitHub Actions ejecuta un *workflow* (flujo de trabajo) predefinido. Este *workflow* utiliza herramientas y credenciales para interactuar con **Firestore** y desplegar el frontend.

### Funcionalidades Específicas:

- **Disparo Basado en Eventos:** Los *workflows* se activan automáticamente por eventos en el repositorio, eliminando la necesidad de activación manual.

- **Ejecución de Scripts de Build:** Ejecuta comandos como `npm install` y `npm run build` para compilar el código fuente de Vue.js y optimizarlo para producción.
- **Despliegue Programático:** Utiliza las credenciales de Firebase configuradas de forma segura (como secretos de GitHub) para autenticarse con Firebase y realizar el despliegue de la aplicación estática en **Firebase Hosting**, asegurando que la última versión del código esté siempre disponible en línea de manera eficiente.
- **Reporte de Estado:** Proporciona visibilidad del progreso de la construcción y el despliegue directamente en la interfaz de GitHub, lo que facilita la colaboración y la resolución de problemas.

### 3.3. Estándares Utilizados

Durante el diseño y la concepción de la plataforma, se adhirió a un conjunto de estándares y mejores prácticas con el fin de asegurar la robustez, seguridad, escalabilidad y mantenibilidad del sistema.

**Estándares Web Abiertos (HTML5, CSS3, ECMAScript 2015+):** El desarrollo del frontend se basó en las últimas versiones de los estándares web, garantizando la compatibilidad con navegadores modernos y el acceso a funcionalidades avanzadas del lenguaje JavaScript.

**Principios de Diseño de API RESTful:** La comunicación entre el frontend (Vue.js) y el backend (Google Cloud Functions) se diseñó siguiendo los principios de las API RESTful. Esto implica el uso de verbos HTTP estándar (GET, POST, PUT, DELETE) para realizar operaciones sobre recursos, URLs descriptivas y respuestas en

formato JSON, facilitando la comprensión, el consumo y la evolución de la API.

**Diseño Responsivo:** Mediante el uso de **Tailwind CSS** y siguiendo principios de diseño mobile-first, la interfaz de usuario se adaptó para proporcionar una experiencia óptima y consistente en una amplia gama de dispositivos, desde teléfonos móviles hasta monitores de escritorio.

**Seguridad en la Capa de Transporte (HTTPS):** Todas las comunicaciones entre el cliente y el servidor se realizan a través de HTTPS. Firebase Hosting implementa automáticamente certificados SSL, garantizando que los datos transmitidos estén encriptados y protegidos contra interceptaciones.

**Autenticación Estándar (OAuth 2.0 / OpenID Connect):** Firebase Auth, al integrarse con proveedores como Google, utiliza internamente estándares de la industria como OAuth 2.0 y OpenID Connect para una autenticación segura y delegada.

**Reglas de Seguridad Declarativas:** Se implementaron reglas de seguridad declarativas en **Firestore** y **Firebase Storage**. Estas reglas siguen un lenguaje específico de Firebase para definir de manera precisa y auditable las políticas de acceso a los datos y archivos, garantizando que solo los usuarios autorizados puedan leer y escribir información específica.

**Principios de CI/CD:** El flujo de trabajo de desarrollo sigue principios de Integración Continua y Despliegue Continuo (CI/CD) a través de **GitHub Actions**. Esto asegura que cada cambio de código se integra y prueba (implícitamente en el **build**) de manera



automática y se despliega de forma consistente, lo que mejora la calidad del software y la velocidad de entrega.

**Estándares de Procesamiento de Pagos (PCI DSS):** Aunque la plataforma no maneja directamente la información sensible de tarjetas, la integración con **Stripe** significa que las operaciones de pago cumplen con los estrictos estándares de seguridad de datos de la industria de tarjetas de pago (PCI DSS), ya que Stripe es un proveedor de servicios certificado.

La adopción de estos estándares y principios en cada etapa del diseño y desarrollo contribuye a la creación de una plataforma robusta, segura, escalable y mantenible a largo plazo.

# Capítulo 4

## Implementación

### 4.1 Especificaciones de Software y Hardware

En esta subsección, listaremos las herramientas y tecnologías que utilizaste. Es importante ser preciso y describir el rol de cada una.

#### Especificaciones de Software:

- **Frontend:**
  - **Vue.js:** Framework progresivo de JavaScript para construir la interfaz de usuario.
  - **Tailwind CSS:** Framework CSS utilitario para un diseño rápido y personalizable.
- **Backend y Servicios en la Nube:**
  - **Google Cloud Platform (GCP):** Plataforma de servicios en la nube.
  - **Firebase:** Plataforma de desarrollo de aplicaciones móviles y web de Google, utilizada para:
    - **Firebase Hosting:** Alojamiento del sitio web.
    - **Firestore:** Base de datos NoSQL para gestionar usuarios, inventario y transacciones.
    - **Firebase Authentication:** Sistema de autenticación de usuarios por correo, contraseña, Google y teléfono (SMS).
    - **Firebase Storage:** Almacenamiento de fotos de productos y documentos de usuarios.

- **Google Cloud Functions:** Entorno de ejecución de código sin servidor para:
  - Procesar pagos a través de la extensión oficial de **Stripe**.
  - Enviar correos electrónicos a los usuarios.
  - Realizar cambios automatizados en el catálogo.
- **Google Cloud Scheduler:** Programador de tareas en la nube, utilizado para ejecutar funciones de Cloud Functions de manera periódica (ej. una vez al día).
- **Herramientas de Desarrollo y Flujo de Trabajo:**
  - **GitHub:** Control de versiones del código fuente.
  - **GitHub Actions:** Servicio de integración continua y despliegue continuo (CI/CD) para automatizar el **build** y **deploy** a Firebase Hosting.
- **Servicios de Terceros:**
  - **Stripe:** Procesador de pagos para transacciones financieras.

### Especificaciones de Hardware:

- Dado que el proyecto se apalancó en servicios de computación sin servidor (serverless) de Google Cloud, no se requirió la adquisición o configuración de hardware físico. La infraestructura fue gestionada en su totalidad por la plataforma en la nube, optimizando costos y escalabilidad.

## 4.2 Descripción completa de la implementación

Aquí, detallaremos cómo se conectan todas las piezas que mencionaste. La idea es explicar el flujo de la información y la

interacción entre los diferentes componentes, desde que un usuario entra a la plataforma hasta que completa una transacción.

**Introducción:** El proyecto se implementó como una aplicación web de página única (SPA) con una arquitectura basada en microservicios y servicios en la nube. Se buscó una solución que fuera escalable, de bajo costo y fácil de mantener, aprovechando la capa gratuita y las herramientas de Google Cloud y Firebase.

### **Flujo de la Plataforma:**

1. **Frontend y Hosting:** El sitio web, desarrollado con Vue.js y Tailwind CSS, se aloja en **Firebase Hosting**. Este entorno proporciona una entrega rápida y segura del contenido. El proceso de despliegue se automatizó a través de **GitHub Actions**, lo que asegura que cada cambio en el repositorio principal se refleje de manera automática en la plataforma, implementando un flujo de **integración y despliegue continuo (CI/CD)**.
2. **Gestión de Usuarios y Seguridad:** Para la autenticación, se implementó **Firebase Authentication**. Los usuarios pueden registrarse y acceder usando su correo y contraseña, su cuenta de Google o su número telefónico validado por SMS. Las credenciales de los usuarios y sus perfiles se almacenan en **Firestore**, donde las **Reglas de Seguridad** garantizan que la información privada, como datos de perfil, solo sea accesible por el usuario propietario.
3. **Catálogo y Transacciones:** El inventario de equipos y los detalles de las publicaciones se gestionan en **Firestore**. Las imágenes de los productos se almacenan en **Firebase Storage** y se accede a ellas mediante reglas que distinguen entre archivos públicos (fotos de productos) y privados

(documentos de usuarios). Cuando un usuario decide rentar un equipo, la transacción se inicia y se procesa a través de una **Cloud Function**. Esta función se encarga de la comunicación segura con la extensión oficial de **Stripe**, manejando el flujo de pago sin exponer información sensible en el frontend.

4. **Automatización y Comunicación:** Además de procesar pagos, las **Cloud Functions** se utilizan para la comunicación con los usuarios. Por ejemplo, al completar una transacción, se dispara una función que envía un correo de confirmación. Para tareas repetitivas, como el monitoreo del catálogo, se utiliza **Cloud Scheduler** para ejecutar funciones de manera periódica. Por ejemplo, una función se ejecuta diariamente para dar de baja publicaciones inactivas o actualizar el estado de las órdenes, asegurando que la información sea siempre precisa y esté al día.
5. **Reglas de Seguridad:** Un aspecto crítico de la implementación es la seguridad. Tanto **Firestore** como **Firebase Storage** utilizan reglas de seguridad que controlan el acceso a los datos. Estas reglas permiten definir quién puede leer, escribir o eliminar información, protegiendo la integridad y privacidad del sistema. Por ejemplo, un usuario solo puede modificar sus propios datos de perfil o las publicaciones que creó, pero no las de otros.

# **Capítulo 5**

## **Pruebas y resultados**

### **5.1 Descripción de las pruebas**

### **5.2 Resultados**

### **5.3 Análisis de resultados**

---

# **Capítulo 6**

## **Conclusiones**

---

## Referencias