



UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD DE TECNOLOGÍAS INTERACTIVAS

SISTEMA PARA LA CONFIGURACIÓN DINÁMICA DE MATRICES EN TV INTELIGENTE.

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Fernando Parra Masjuan

Tutor: Ing. Julio Alberto Leyva Durán

La Habana, 2024

Si haces lo que siempre has hecho, llegarás donde siempre has llegado.
Tony Robbins

Dedicatoria

Dedico esta tesis con todo mi cariño a mis padres, quienes han sido la luz y el pilar de mi vida. Su amor incondicional y su apoyo constante me han permitido alcanzar este importante logro. También dedico este trabajo a mis compañeros de universidad, con quienes compartí momentos inolvidables y aprendizajes valiosos. Gracias a todos ustedes por ser una parte esencial de mi viaje académico.

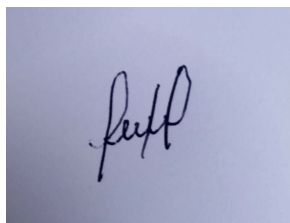
Agradecimientos

Quiero expresar mi más sincero agradecimiento a mi madre y a mi padre, quienes han sido mi mayor apoyo incondicional a lo largo de este camino, brindándome amor, motivación y la confianza necesaria para seguir mis sueños. A mi familia, por su comprensión y aliento constante, que me han inspirado a dar lo mejor de mí en cada paso. A mis compañeros de la universidad, con quienes compartí risas, desvelos y aprendizajes, su camaradería ha hecho de esta experiencia algo inolvidable. Y, por supuesto, a todos mis profesores, quienes con su dedicación y sabiduría han guiado mi formación académica y personal. A cada uno de ustedes, gracias por ser parte fundamental de este logro.

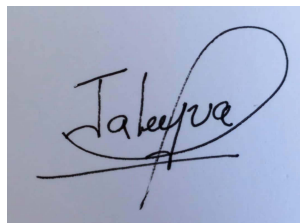
Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los 09 días del mes de Diciembre del año 2024.



Fernando Parra Masjuan
Autor



Ing. Julio Alberto Leyva Durán
Tutor

La presente investigación se centra en el desarrollo de un sistema automatizado para el control de matrices de televisores o video walls, mediante plataformas de hardware libre como Raspberry Pi. La motivación detrás de este estudio radica en la creciente necesidad de soluciones eficientes para la gestión de múltiples pantallas en entornos comerciales y de entretenimiento. El objetivo principal es diseñar un sistema que permita gestionar la configuración de control en las matrices de Televisores Inteligentes, mejorando así la experiencia del usuario y optimizando recursos. Para ello, se empleó la metodología Extreme Programming (XP) que fomenta la entrega continua de software mediante ciclos de desarrollo cortos y la integración constante. Este sistema incluye un software accesible a través de una interfaz web segura, permitiendo realizar ajustes remotos y eliminando la necesidad de operar manualmente cada televisor. Además, se integró un sistema de almacenamiento para respaldar configuraciones. Para garantizar la seguridad, se implementó un sistema de autenticación que restringe el acceso al personal. El sistema permite programar configuraciones automáticas para que el videowall esté siempre listo para su uso sin intervención manual.

Palabras clave: Video walls, hardware libre , Raspberry Pi..

Introducción	1
1 FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE LOS PROCESOS DE CONFIGURACIÓN DE TV INTELIGENTES.	5
1.1 Matriz de televisores	5
1.1.1 Televisores inteligentes	6
1.1.2 Raspberry pi	7
1.2 Análisis de sistemas homólogos	8
1.3 Tecnologías y herramientas	11
1.3.1 Accesorios para el control de la matriz	14
1.4 Metodología de desarrollo de software	17
2 DISEÑO DE LA SOLUCIÓN PARA LA CONFIGURACIÓN DINÁMICA DE MATRICES EN TV INTELIGENTE	20
2.1 Descripción de la propuesta de solución	20
2.2 Requisitos, análisis y modelado del sistema	21
2.3 Historias de usuario	23
2.4 Plan de Iteraciones	25
2.5 Plan de Entrega	26
2.6 Diseño arquitectónico	27
2.7 Tarjetas CRC	32
3 VALIDACIÓN DE LA SOLUCIÓN PROPUESTA	34
3.1 Tareas ingenieriles	34
3.2 Plan de pruebas	35
3.3 Pruebas Unitarias	36
3.4 Pruebas de Aceptación	38
Conclusiones	41

Recomendaciones	42
Referencias bibliográficas	43
Apéndices	47
A Historias de Usuario	48
B Tareas Ingenieriles	55
C Pruebas unitarias	60
D Pruebas de aceptación	62

Índice de figuras

1.1	Matriz de televisores. Fuente:(Digital, 2024).	6
1.2	Raspberry Pi. Fuente:(cgarcia, 2024).	8
1.3	Accesorios del controlador de la matriz. Fuente: (<i>ManualUsuarioMatrizTV-LG</i> n.d.).	14
1.4	Vista frontal del Dispositivo Controlador, con los correspondientes conectores. Fuente:(<i>ManualUsuarioMatrizTV-LG</i> n.d.).	15
1.5	Conexión serial de los televisores. Fuente:(<i>ManualUsuarioMatrizTV-LG</i> n.d.).	17
2.1	Digrama de paquete del Modelo-Vista-Plantilla. Fuente:(Elaboración propia)	28
2.2	Patrón Controlador. Fuente:(Elaboración propia)	29
2.3	Patrón Creador. Fuente:(Elaboración propia)	29
2.4	Patrón Alta cohesión. Fuente:(Elaboración propia)	30
2.5	Patrón Bajo acoplamiento. Fuente:(Elaboración propia)	30
2.6	Patrón Mediador. Fuente:(Elaboración propia)	31
2.7	Patrón Mediador. Fuente:(Elaboración propia)	31
3.1	Resultado de las pruebas unitarias Primera iteración. Fuente:(Elaboración propia)	36
3.2	Resultado de las pruebas unitarias Segunda iteración. Fuente:(Elaboración propia)	36
3.3	Resultado de las pruebas unitarias Tercera iteración. Fuente:(Elaboración propia)	36
3.4	Resultado de las pruebas unitarias Cuarta iteración. Fuente:(Elaboración propia)	37
3.5	Resultado de las pruebas unitarias Quinta iteración. Fuente:(Elaboración propia)	37
3.6	Resultado de las pruebas unitarias Quinta iteración. Fuente:(Elaboración propia)	37
3.7	Resultado de las pruebas unitarias final. Fuente:(Elaboración propia)	38
C.1	Pruebas unitarias Fuente: Elaboración propia	60
C.2	Pruebas unitarias Fuente: Elaboración propia	60
C.3	Pruebas unitarias Fuente: Elaboración propia	61
C.4	Pruebas unitarias Fuente: Elaboración propia	61

Índice de tablas

1.1	Características de los sistemas homólogos. Fuente: Elaboración propia.	10
2.1	Requerimientos funcionales. Fuente: Elaboración propia	21
2.2	Historia de usuario # 1	24
2.3	Historia de usuario # 2	24
2.4	Historia de usuario # 3	25
2.5	Plan de iteraciones. Fuente: Elaboración propia.	26
2.6	Plan de entregas. Fuente: Elaboración propia.	26
2.7	Tarjeta CRC # 1	32
2.8	Tarjeta CRC # 2	32
3.1	Tarea de ingeniería # 1	34
3.2	Tarea de ingeniería # 2	35
3.3	Tarea de ingeniería # 3	35
3.4	Pruebas y Validación. Fuente: Elaboración propia	35
3.5	Prueba de aceptación # 1	38
3.6	Prueba de aceptación # 2	39
3.7	Prueba de aceptación # 3	39
A.1	Historia de usuario # 4	48
A.2	Historia de usuario # 5	48
A.3	Historia de usuario # 6	49
A.4	Historia de usuario # 7	49
A.5	Historia de usuario # 8	50
A.6	Historia de usuario # 9	50
A.7	Historia de usuario # 10	50
A.8	Historia de usuario # 11	51
A.9	Historia de usuario # 12	51
A.10	Historia de usuario # 13	51

A.11 Historia de usuario # 14	52
A.12 Historia de usuario # 15	52
A.13 Historia de usuario # 16	52
A.14 Historia de usuario # 17	53
A.15 Historia de usuario # 18	53
A.16 Historia de usuario # 19	54
A.17 Historia de usuario # 20	54
B.1 Tarea de ingeniería # 4	55
B.2 Tarea de ingeniería # 5	55
B.3 Tarea de ingeniería # 6	55
B.4 Tarea de ingeniería # 7	56
B.5 Tarea de ingeniería # 8	56
B.6 Tarea de ingeniería # 9	56
B.7 Tarea de ingeniería # 10	56
B.8 Tarea de ingeniería # 11	57
B.9 Tarea de ingeniería # 12	57
B.10 Tarea de ingeniería # 13	57
B.11 Tarea de ingeniería # 14	58
B.12 Tarea de ingeniería # 15	58
B.13 Tarea de ingeniería # 16	58
B.14 Tarea de ingeniería # 17	58
B.15 Tarea de ingeniería # 18	59
B.16 Tarea de ingeniería # 19	59
B.17 Tarea de ingeniería # 20	59
D.1 Prueba de aceptación # 4	62
D.2 Prueba de aceptación # 5	62
D.3 Prueba de aceptación # 6	63
D.4 Prueba de aceptación # 7	63
D.5 Prueba de aceptación # 8	63
D.6 Prueba de aceptación # 9	64
D.7 Prueba de aceptación # 10	64
D.8 Prueba de aceptación # 11	64
D.9 Prueba de aceptación # 12	64
D.10 Prueba de aceptación # 13	65
D.11 Prueba de aceptación # 14	65
D.12 Prueba de aceptación # 15	65

D.13 Prueba de aceptación # 16	66
D.14 Prueba de aceptación # 17	66
D.15 Prueba de aceptación # 18	66
D.16 Prueba de aceptación # 19	67
D.17 Prueba de aceptación # 20	67

En la actualidad, los videos walls o matriz de televisores se han consolidado como una herramienta esencial en la comunicación visual, utilizada en una variedad de sectores a nivel internacional. Estas configuraciones, compuestas por múltiples pantallas que se combinan para formar una superficie de visualización continua, permiten la presentación de contenido dinámico y atractivo que capta la atención del público. Desde su origen en entornos de control y seguridad hasta su implementación en retail, educación y entretenimiento, los videos walls han demostrado ser versátiles y efectivos en la mejora de la experiencia del espectador.

Los videos walls encuentran aplicaciones en una variedad de industrias. En el sector retail, se utilizan para crear experiencias visuales impactantes que atraen a los clientes y promueven productos de manera efectiva. Un estudio indica que el uso de video walls en tiendas puede aumentar el tráfico peatonal hasta en un 30 (Digital, 2024). En el ámbito educativo, estas pantallas mejoran la enseñanza al facilitar presentaciones interactivas que mantienen el interés de los estudiantes. Además, en entornos corporativos y de control, los video walls permiten a los equipos monitorear múltiples fuentes de información simultáneamente, lo que es crucial para la toma de decisiones informadas (Byers, 2024).

A medida que la tecnología avanza, también lo hacen las capacidades de los videos walls. La integración con sistemas basados en IP (AVoIP) ha revolucionado la forma en que se gestionan estos sistemas, permitiendo una mayor flexibilidad y escalabilidad en la distribución del contenido (Eli Ofek, 2024). Esta capacidad para adaptarse rápidamente a diferentes configuraciones y requisitos es fundamental en un mundo donde la rapidez y la efectividad son esenciales para el éxito organizacional.

Los videos wall en Cuba han comenzado a tener un uso creciente, especialmente en entornos urbanos. Impulsada por su versatilidad y capacidad para mejorar la comunicación visual en diversos sectores. A medida que la tecnología avanza y se vuelve más accesible, es probable que su uso se expanda aún más, beneficiando tanto a empresas como a instituciones públicas y educativas (CORREOS DE CUBA, 2024).

Algunas de estas instituciones están comenzando a implementar video wall para presentaciones interactivas, exposiciones y actividades académicas, aprovechando su capacidad para mostrar información. A pesar de

su creciente popularidad, la instalación y mantenimiento de video wall en Cuba enfrenta desafíos debido a la infraestructura tecnológica limitada y a la disponibilidad de recursos financieros poniendo obstáculos al desarrollo de esta tecnología y sistemas controlen estas matrices de televisores.

La Universidad de las Ciencias Informáticas (UCI) es una institución educativa dedicada a la formación de profesionales en el campo de la informática y las tecnologías de la información, jugando un papel crucial en el desarrollo de la educación tecnológica en el país, promoviendo la investigación y la innovación en diversas áreas relacionadas con la computación. La UCI ha estado involucrada en el desarrollo de software que puede ser utilizado para gestionar y controlar videowalls. Esto incluye aplicaciones que permiten la integración y sincronización de múltiples fuentes de contenido, facilitando su uso en entornos comerciales y de monitoreo (*Universidad de las Ciencias Informáticas 2024*).

La UCI, como institución educativa, necesita un sistema eficiente para gestionar y controlar las configuraciones de una matriz de televisores inteligentes de forma automática.

Actualmente, las configuraciones de los videos walls se realizan manualmente a través de los mandos de cada televisor. Esto se vuelve especialmente complicado en una matriz de $n \times n$ pantallas, donde el tiempo y el esfuerzo requeridos aumentan considerablemente, lo que puede llevar a errores humanos ya una gestión ineficiente del tiempo. Un riesgo crítico es la posible pérdida de configuración en caso de un corte eléctrico prolongado. Esto obligaría al personal a reiniciar la configuración desde cero, lo que no solo consumirá tiempo, sino que también puede causar interrupciones en las operaciones. También existe el peligro de que personas ajenas al personal calificado puedan manipular el sistema, apretando botones o cambiando configuraciones sin autorización. Esto podría resultar en una pérdida accidental de la configuración o en un mal funcionamiento del sistema.

A partir de la situación problemática descrita anteriormente se plantea como **problema de investigación**: ¿Cómo garantizar la configuración adecuada en una matriz en TV Inteligente? Por lo tanto el **objeto de estudio** está enfocado en los procesos de configuración de TV Inteligente. Se asume como **campo de acción**: la configuración automática de matrices de TV inteligente mediante plataforma de hardware libre. Con el fin de solucionar el problema planteado se define como **objetivo general**: Desarrollar un Sistema para la configuración dinámica de matrices en TV Inteligente.

Para lograr el objetivo general se definen las siguientes **tareas de la investigación**:

- Estudio del Estado del Arte del Tema en Cuestión:
Investigar y recopilar literatura relevante sobre sistemas de video wall, incluyendo tecnologías actuales, métodos de configuración y gestión, y estudios previos sobre y automatización control centralizado.
Examinar ejemplos de implementación exitosa de videowalls en diferentes entornos (hospitales, centros de control, eventos) para identificar mejores prácticas y lecciones aprendidas.

- **Generación de los Artefactos Relacionados con el Análisis y Diseño de la Solución Propuesta:**
Definir los requisitos del sistema, incluyendo funcionalidad, rendimiento, escalabilidad y seguridad.
Crear diagramas arquitectónicos y flujos de trabajo que representen la interacción entre los componentes del sistema (pantallas, controladores, interfaz de usuario).
Desarrollar prototipos iniciales para validar conceptos clave y funcionalidades antes de la implementación completa.
- **Diseño e Implementación de la Solución Propuesta:**
Programar la aplicación que permitirá la gestión centralizada del video wall, incluyendo la interfaz gráfica y la lógica para la sincronización y configuración dinámica.
Seleccionar e integrar los componentes físicos necesarios (pantallas, controladores, redes) para crear el sistema completo.
Realizar pruebas unitarias y de integración para asegurar que cada componente funcione correctamente dentro del sistema.
- **Validación de la propuesta de solución:**
Evaluar el sistema en condiciones reales para verificar que cumple con los requisitos establecidos en las especificaciones.
Medir el rendimiento del sistema en términos de sincronización, respuesta a cambios en tiempo real y estabilidad durante el uso prolongado.
Realizar sesiones con usuarios finales para obtener retroalimentación sobre la usabilidad y efectividad del sistema, realizando ajustes según sea necesario.
Elaborar un informe detallado que incluya los resultados obtenidos, las lecciones aprendidas durante el proceso y recomendaciones para futuras mejoras.

Para obtener los conocimientos necesarios que hagan posible el cumplimiento del objetivo trazado, se lleva a cabo una investigación en las que se utilizan algunos de los métodos científicos existentes, tanto teóricos como empíricos.

Métodos teóricos:

Modelación: Es empleada en la representación mediante diagramas de las características, procesos y componentes de la solución propuesta, así como la relación existente entre ellos.

Analítico-sintético: Se emplearon en el proceso de análisis documental y revisión bibliográfica, con el objetivo de extraer las ideas esenciales que permitirán fundamentar desde el punto de vista teórico la investigación, así como la propuesta que se realiza.

Hipotético-deductivo: Para guiar la investigación desde el planteamiento del problema hasta la verificación de la solución a partir de las validaciones, orientando la secuencia lógica de las tareas que se realizaron.

Métodos empíricos:

Entrevista: Se emplea en encuentros con el cliente para conocer la necesidad del desarrollo de la propuesta de solución, definir sus funcionalidades e identificar a la vez particularidades de cada usuario y las restricciones que se imponen.(Anexo 1)

Observación: Se empleó como método referencial al observar distintos sistemas de control de video wall que sirvieron como objeto de análisis y comparación para establecer las características y elementos fundamentales de la solución propuesta.

El uso de este método es esencial porque permite realizar un análisis comparativo que identifica las mejores prácticas y características clave de diversas tecnologías, lo que ayuda a entender qué elementos son cruciales para el éxito del proyecto. Este enfoque facilita la adaptación de la solución a las necesidades específicas del cliente y fomenta la innovación al inspirarse en tendencias emergentes. En conjunto, estas ventajas garantizan que la solución propuesta sea efectiva, eficiente y alineada con las expectativas del usuario final.

El presente trabajo está compuesto por tres capítulos, estructurados de la siguiente forma:

- **Capítulo 1 FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE LOS PROCESOS DE CONFIGURACIÓN DE TV INTELIGENTES:**

En él se realiza un estudio de los referentes teóricos fundamentales relacionados con el objeto de estudio, se realiza además un estudio de soluciones homólogas necesario para el establecimiento de una base sólida para el diseño y posterior implementación de la propuesta de solución.

- **Capítulo 2 DISEÑO DE LA SOLUCIÓN PARA LA CONFIGURACIÓN DINÁMICA DE MATRICES EN TV INTELIGENTE:**

En este capítulo se definen los requisitos de la propuesta de solución, además de una breve descripción de la misma. Se abordan aspectos como la arquitectura y funcionalidades descritas a través del uso de historias de usuario y tarjetas CRC (Clase Responsabilidad y Control).

- **Capítulo 3 VALIDACIÓN DE LA SOLUCIÓN PROPUESTA:**

Describe como se llevó a cabo las pruebas realizadas para asegurar el perfecto funcionamiento del código y para verificar si lo realizado cumple con las especificaciones definidas por el cliente.

FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE LOS PROCESOS DE CONFIGURACIÓN DE TV INTELIGENTES.

Este capítulo se centra en los aspectos teóricos y conceptos fundamentales relacionados con el objeto de estudio y el campo de acción de la investigación. Se analizan las soluciones existentes tanto en el ámbito internacional como en el nacional y se describe el entorno de desarrollo a partir de la fundamentación del uso de la metodología, tecnologías y herramientas propuestas para el desarrollo de la propuesta de solución.

1.1. Matriz de televisores

Una matriz de televisores o videowall es una configuración compuesta por múltiples pantallas que se combinan para crear una única superficie visual de gran tamaño. Esta tecnología permite la visualización simultánea de diferentes fuentes de contenido, lo que resulta esencial en entornos como centros de control, auditorios y espacios públicos. La capacidad de presentar información en tiempo real y en formatos variados lo convierte en una herramienta poderosa para la toma de decisiones y la comunicación efectiva (Digital, 2024).



Figura 1.1. Matriz de televisores. Fuente:(Digital, 2024).

El análisis del concepto de videowall es crucial en este proyecto de sistema de control de videowall debido a su papel fundamental en la visualización y gestión de información en entornos críticos. Comprender cómo funcionan estas matrices de pantallas permite identificar sus ventajas, como la capacidad de mostrar múltiples fuentes de contenido simultáneamente, lo que es esencial para la toma de decisiones informadas en tiempo real. Posteriormente, es necesario evaluar el papel de los **televisores inteligentes** en la creación de videowalls, ya que su evolución tecnológica ha permitido optimizar la experiencia visual y funcionalidad en entornos profesionales.

1.1.1. Televisores inteligentes

Los televisores inteligentes (Smart TVs) han evolucionado para convertirse en componentes clave en la creación de videowalls, ofreciendo características que optimizan la experiencia visual y la funcionalidad en entornos profesionales. Existen diferentes tipos de televisores inteligentes, los cuales pueden variar en función de sus características y funcionalidades (*webmasters*, 2024). Algunos de los tipos más comunes incluyen:

- Televisores con sistemas operativos integrados, como Android TV, Tizen (Samsung), webOS (LG), Roku TV, entre otros.
- Televisores con asistentes de voz integrados, como Amazon Alexa, Google Assistant o Siri.
- Televisores con paneles de control remoto o aplicaciones móviles que permiten la gestión de contenido y funciones extras.
- Televisores con capacidades de pantalla dividida o multitarea para visualizar múltiples fuentes de contenido a la vez.

Entre las características comunes de un televisor inteligente se incluyen:

- **Conexión a Internet:** Un televisor inteligente puede conectarse a internet a través de Wi-Fi o un cable de red, lo que permite acceder a contenido en línea, servicios de streaming, redes sociales, navegadores

web y mucho más.

- **Aplicaciones y Servicios:** Un Smart TV generalmente incluye un sistema operativo que permite instalar y ejecutar aplicaciones. Esto puede incluir aplicaciones de streaming como Netflix, Hulu, Amazon Prime Video, Disney+, YouTube, entre otros. También puede ofrecer acceso a servicios de música, juegos, noticias, clima y otras aplicaciones personalizadas.
- **Navegación web:** Algunos Smart TV permiten navegar por internet usando un navegador web incorporado. Esto facilita el acceso a sitios web y servicios en línea directamente desde el televisor.
- **Conectividad con otros dispositivos:** Los televisores inteligentes suelen ser compatibles con otros dispositivos inteligentes, como teléfonos, tabletas y altavoces inteligentes, lo que permite la transmisión de contenido y control remoto mediante aplicaciones o comandos de voz.
- **Integración con asistentes virtuales:** Muchos televisores inteligentes son compatibles con asistentes virtuales como Amazon Alexa o Google Assistant, lo que permite controlar el televisor mediante comandos de voz y realizar otras tareas inteligentes.

Los televisores inteligentes en una matriz suelen estar conectados a través de una red o mediante cables para formar una pantalla única y coordinada (TCL, 2022).

Al comprender las capacidades específicas que ofrecen estos televisores, como la conectividad, la calidad de imagen y las opciones de control, se puede garantizar que el videowall no solo cumpla con los requisitos técnicos, sino que también optimice la experiencia visual y la interacción del usuario. Además es paso crucial para este proyecto analizar las funcionalidades de una matriz de televisores inteligentes debido a que estas características son clave para determinar la efectividad y el rendimiento del sistema de videowall.

Las funcionalidades de una matriz de televisores inteligentes pueden incluir:

- Reproducción simultánea de contenido en múltiples televisores.
- Gestión centralizada de configuraciones y actualizaciones de software.
- Control remoto unificado para manejar varios televisores desde un solo dispositivo.
- Uso compartido de contenido y aplicaciones entre los televisores de la matriz.
- Posibilidad de mostrar contenido personalizado o informativo en todos los televisores al mismo tiempo.

1.1.2. Raspberry pi

Una Raspberry Pi es un pequeño ordenador que tiene un tamaño similar a una placa de arduino y que también se utiliza a menudo en el mundo maker para construir proyectos de electrónica. Sin embargo, son dos herramientas que nos pueden parecer similares visualmente pero que no son lo mismo: la Raspberry Pi es una herramienta mucho más potente que Arduino que sirve para hacer proyectos más complejos y de diferentes tipos, ya que es un microordenador y, como todos los ordenadores, requiere de un sistema

operativo para funcionar (cgarcia, 2024).

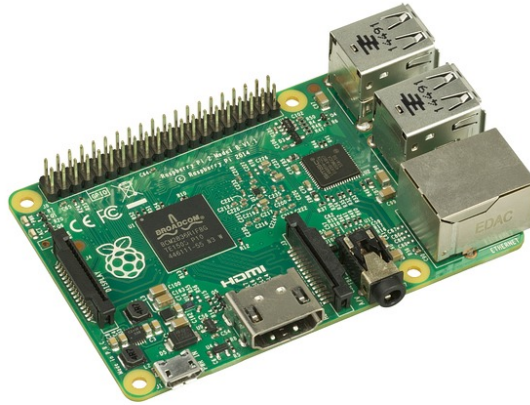


Figura 1.2. Raspberry Pi. Fuente:(cgarcia, 2024).

Se va a utilizar la Raspberry Pi en este proyecto porque servirá como un servidor web que facilitará la interacción del usuario con las opciones del sistema, permitiendo un control intuitivo y accesible de los televisores conectados. Además, su capacidad para gestionar la comunicación a través de TCP/IP garantiza un control eficiente y en tiempo real de los dispositivos, lo que es crucial para la funcionalidad del videowall. La Raspberry Pi ofrece una solución compacta y económica, ideal para implementar un sistema que responda a las necesidades específicas de visualización y gestión de contenido en entornos profesionales.

1.2. Análisis de sistemas homólogos

El uso de video walls se ha expandido en diversas industrias, siendo adoptado por empresas y organizaciones que requieren soluciones visuales de gran impacto (Bescan, 2024). A continuación se presentan algunas de las principales empresas y sectores que utilizan video walls:

- Corporación Sony
- Electrónica Samsung
- Electrónica LG
- Corporación Toshiba
- Corporación Panasonic

Estas empresas no solo fabrican tecnología de video walls, sino que también los implementan en sus propias instalaciones para exhibiciones y presentaciones.

Sectores Comunes de Uso:

- Seguridad y Monitoreo
- Salas de control para vigilancia y gestión de emergencias.
- Monitoreo de múltiples cámaras en tiempo real.
- Comercial y Marketing
- Exposiciones en ferias comerciales.
- Publicidad en espacios públicos y tiendas.
- Transporte
- Aeropuertos y estaciones de tren para información al pasajero.
- Control de tráfico en centros de transporte masivo.
- Entretenimiento y medios
- Producciones cinematográficas y eventos en vivo.
- Instalaciones artísticas interactivas.

En Estados Unidos, algunos proveedores destacados incluyen:

- Daktronics : Líder en pantallas LED.
- PixelFLEX : Especialista en soluciones personalizadas.
- Neoti : Ofrece pantallas LED versátiles para diversos eventos

El mercado de video walls sigue creciendo, impulsado por la demanda de soluciones visuales efectivas en múltiples aplicaciones comerciales y técnicas. Existen diversas aplicaciones web y software especializados que se utilizan para el control y gestión de video walls. A continuación, se presentan algunas de las más destacadas:

- Matrox MuraControl
Matrox MuraControl es un software de gestión de video walls que permite controlar pantallas Matrox desde una interfaz intuitiva. Este sistema es accesible tanto localmente como de forma remota y es compatible con Windows (*MATROX*, 2024).
- Useful Infinity
Useful ofrece una plataforma de software que permite gestionar y transmitir contenido a video walls en tiempo real a través de una interfaz web. Es una solución AV sobre IP que se adapta a cualquier pantalla (*UsefulTM*, 2024).
- Soluciones C2H
C2H Solutions proporciona plataformas integradas para el control de video walls, especialmente en centros de comando, permitiendo la visualización simultánea de múltiples fuentes de datos (*C2H Solutions*, 2024).
- GESAB Video Wall Solutions

GESAB ofrece soluciones completas para video walls en centros de control, incluyendo hardware y software diseñados para optimizar la visualización y gestión del contenido (GESAB, 2024).

- **VuWall TRx**

Una plataforma centralizada que facilita la gestión de dispositivos AV sobre IP, permitiendo configurar y controlar todos los dispositivos desde un solo lugar. Ofrece una interfaz intuitiva para arrastrar y soltar, simplificando el manejo de escenarios operativos (VuWall, 2024).

Como resultado del análisis de los sistemas antes mencionados se muestra indicadores que se consideran relevantes para la investigación:

- **Interfaz Única:** La mayoría de los sistemas permiten controlar múltiples pantallas desde una única interfaz, lo que facilita la administración y el monitoreo del contenido en tiempo real.
- **Acceso Remoto:** Muchos sistemas ofrecen la posibilidad de gestionar el contenido de forma remota, lo que permite a los operadores realizar ajustes desde diferentes ubicaciones.
- **Expansión Sencilla:** Los sistemas están diseñados para ser escalables, permitiendo a los usuarios agregar más pantallas o fuentes sin necesidad de reconfigurar todo el sistema.
- **Flexibilidad en configuración:** Se pueden ajustar fácilmente para adaptarse a diferentes configuraciones y tamaños de video wall.

Tabla 1.1. Características de los sistemas homólogos. Fuente: Elaboración propia.

Características	Matrox MuraControl	Userful Infinity	Soluciones C2H	Video Wall Solutions	GESAB Video Wall Solutions
Interfaz de Usuario	Interfaz intuitiva	Interfaz basada en navegador (HTML5)	Interfaz centralizada	Interfaz intuitiva	Interfaz personalizable
Tipo de Software	Costo	Costo	Costo	Costo	Costo
Gestión de Contenido	Sí	Sí	Sí	Sí	Sí
Flexibilidad y Escalabilidad	Sí	Sí	Sí	Sí	Sí

El análisis de sistemas homólogos ha demostrado ser una herramienta invaluable en el proceso de desarrollo del sistema. Al examinar las características, funcionalidades y enfoques de operadores similares, se ha logrado identificar las prácticas y los errores comunes ayudando a la comprensión de las necesidades y expectativas de los usuarios. En consecuencia, se puede afirmar que la implementación de este enfoque no solo optimiza los recursos y el tiempo de desarrollo, sino que también contribuye a la creación de un sistema robusto, innovador y con alto potencial de aceptación en el contexto actual.

1.3. Tecnologías y herramientas

Para desarrollar un sistema, es fundamental contar con herramientas y tecnologías adecuadas que faciliten el proceso de creación, gestión y despliegue. A continuación, se presentan algunas de las herramientas más relevantes para este propósito:

Entornos de Desarrollo Integrados (IDE)

- **Visual Studio Code:** un editor de código fuente ligero y altamente personalizable que admite múltiples lenguajes de programación. Ofrece extensiones para facilitar el desarrollo y la depuración. (versión 1.94.2)
- El **Arduino IDE** (Entorno de Desarrollo Integrado) es una herramienta fundamental para programar placas de Arduino. Sus características principales, funcionalidades y aspectos técnicos basados en la información disponible es multiplataforma, Bibliotecas Integradas, Compilación y Carga, Interfaz de Usuario Intuitiva y Soporte para Terceros. (versión 2.3.3)

Lenguaje de modelado Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) es un estándar ampliamente utilizado para la visualización, especificación, construcción y documentación de sistemas de software (UML, 2024). A continuación se presentan sus características, ventajas y tipos de diagramas.

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para ingenieros de software, analistas de sistemas y arquitectos de sistemas que están interesados en la construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos (Free UML Tool, 2024).

Lenguajes de programación

HTML5

HTML5 es la última versión del lenguaje de marcado de hipertexto, diseñado para estructurar y presentar contenido en la web (HTML en español, 2024).

Introduce nuevas etiquetas semánticas como <header>, <footer>, <article>, <section>, y <nav>, que mejoran la estructura del documento y ayudan a los motores de búsqueda a entender el contenido de la página. Ofrece nuevos tipos de entrada para formularios, como selectores de fecha, campos numéricos y validación nativa, lo que facilita la creación de formularios más interactivos y fáciles de usar.

CSS

CSS (Cascading Style Sheets) es un lenguaje utilizado para describir la presentación de un documento HTML (¿Qué es CSS? - CSS en español 2024). Permite separar el contenido (HTML) de su presentación

(estilos), lo que facilita el mantenimiento del código y mejora la organización del proyecto. Con CSS se pueden crear diseños responsivos utilizando media queries, lo que permite que el diseño se ajuste automáticamente a diferentes tamaños de pantalla y dispositivos.

HTML5 y CSS son tecnologías fundamentales en el desarrollo web moderno. HTML5 proporciona una estructura semántica rica y soporte multimedia nativo, mientras que CSS permite diseñar interfaces atractivas y responsivas. Juntas, estas tecnologías facilitan la creación de aplicaciones web interactivas y accesibles que mejoran la experiencia del usuario.

JavaScript

JavaScript es un lenguaje de programación fundamental en el desarrollo web moderno, conocido por su capacidad para crear contenido dinámico e interactivo en las páginas web (*Javascript en español*, 2024).

Lenguaje de programación del lado del servidor:

C++ es un lenguaje de programación de propósito general, ampliamente utilizado en diversas aplicaciones debido a su eficiencia y versatilidad (*El lenguaje C++ Fundamentos de Programación en C++* 2024). A continuación, se presenta un resumen detallado sobre C++, incluyendo sus características, usos, y ventajas.

Características de C++:

- **Orientado a Objetos:**

C++ es un lenguaje orientado a objetos (OOP), lo que significa que permite la creación de clases y objetos. Esto facilita la organización del código y la reutilización de componentes, promoviendo prácticas de programación más limpias y estructuradas.

- **Compilado:**

C++ es generalmente un lenguaje compilado, lo que significa que el código fuente se traduce a código máquina antes de ser ejecutado. Esto contribuye a un rendimiento más rápido en comparación con los lenguajes interpretados.

- **Control de Recursos:**

Ofrece un alto nivel de control sobre los recursos del sistema y la memoria, lo que permite a los programadores gestionar directamente la memoria dinámica mediante punteros y referencias.

- **Eficiencia:**

C++ es conocido por su eficiencia en términos de rendimiento, lo que lo hace ideal para aplicaciones donde la velocidad y el uso óptimo de recursos son críticos, como en sistemas embebidos, videojuegos y aplicaciones en tiempo real.

- **Bibliotecas Estándar:**

Incluye una rica biblioteca estándar (STL) que proporciona estructuras de datos como vectores, listas y mapas, así como algoritmos para realizar operaciones comunes como búsqueda y ordenamiento.

- **Portabilidad:**

Los programas escritos en C++ pueden ser ejecutados en diferentes plataformas con mínimas modificaciones, lo que lo convierte en un lenguaje altamente portable.

Usos Comunes de C++:

- **Desarrollo de Software:**

Ampliamente utilizado en la creación de aplicaciones de software complejas, incluyendo sistemas operativos (como Windows y Mac OS), navegadores web (como Mozilla Firefox), y herramientas de desarrollo.

- **Sistemas Embebidos:**

Ideal para programar dispositivos embebidos que requieren un control directo sobre el hardware, como electrodomésticos inteligentes y dispositivos médicos.

Python (versión 3.12.5) es un lenguaje de programación versátil y popular que se utiliza en diversos campos, desde el desarrollo web hasta la ciencia de datos y la inteligencia artificial (Python documentation, 2023). A continuación, se detallan aspectos clave sobre Python, incluyendo sus características, frameworks y aplicaciones.

Características de Python:

1. **Sintaxis Simple y Clara:** Python tiene una sintaxis fácil de entender, lo que lo convierte en un excelente lenguaje para principiantes. Su diseño permite escribir código limpio y legible.
2. **Tipado Dinámico:** No es necesario declarar el tipo de variable al crearla, lo que facilita la escritura de código más rápido y flexible.
3. **Gran Comunidad y Soporte:** Python cuenta con una amplia comunidad de desarrolladores que contribuyen a su crecimiento mediante bibliotecas y frameworks, así como foros de discusión y documentación extensa.
4. **Multiplataforma:** Python se puede ejecutar en diferentes sistemas operativos, incluyendo Windows, macOS y Linux, lo que permite a los desarrolladores trabajar en diversas plataformas.
5. **Bibliotecas Ricas:** Dispone de una vasta colección de bibliotecas estándar y de terceros que facilitan tareas comunes como manipulación de datos, creación de interfaces gráficas y desarrollo web.

Django

Descripción: Un framework web de alto nivel que promueve el desarrollo rápido y limpio.

Características:

- Incluye un ORM para manejar bases de datos.
- Proporciona herramientas para autenticación, administración y manejo de URL.

- Ideal para aplicaciones complejas como Instagram y Pinterest.

Se ha seleccionado el framework Django (versión 5.1) para el desarrollo de este proyecto de control de configuración de un videowall debido a su robustez y flexibilidad. Django es un framework web de alto nivel que promueve el desarrollo rápido y limpio, lo que resulta ideal para gestionar la complejidad de un sistema de videowall. Además incluye características integradas como autenticación, administración y manejo de bases de datos, lo que agiliza el proceso de desarrollo y reduce el tiempo necesario para implementar funcionalidades críticas. Su capacidad para manejar múltiples solicitudes simultáneamente y su escalabilidad son esenciales para un sistema que debe gestionar contenido visual en tiempo real en diversas pantallas.

Python ofrece una combinación efectiva de simplicidad, potencia y capacidad de integración que lo convierte en una excelente elección para el desarrollo de una aplicación web destinada al control de un video wall. Esto permite crear una solución eficiente y escalable que puede adaptarse a las necesidades cambiantes de los usuarios y del sistema.

Además Python tiene excelentes capacidades de integración con diferentes dispositivos y hardware, lo que es fundamental en el contexto de un video wall. Esto incluye controladores para gestionar las pantallas, así como la comunicación con otras herramientas y sistemas. Gracias a su arquitectura modular y a la calidad de los frameworks disponibles, las aplicaciones desarrolladas en Python pueden escalar de manera eficiente, lo que es importante si se planea aumentar la complejidad o las funcionalidades del sistema en el futuro.

1.3.1. Accesorios para el control de la matriz

En la figura a continuación se muestran los accesorios para el control de la matriz de televisores inteligentes.



Figura 1.3. Accesorios del controlador de la matriz. Fuente: (*ManualUsuarioMatrizTV-LG* n.d.).

Leyenda:

1. Dispositivo Controlador de la Matriz: Dispositivo responsable de la transmisión de contenidos multimedia mediante una interface HDMI sobre una conexión TCP-IP.
2. Fuente de alimentación 5V, 6A. : Fuente de corriente continua para proveerle energía al dispositivo controlador.
3. Cable HDMI: Este cable se emplea para conectar el Controlador hacia la fuente de entrada de video de la matriz de televisores LG, con el objetivo de transportar la señal de multimedia sobre TCP-IP.
4. Cable RS-232: Este cable se conecta hacia el televisor maestro de la matriz de televisores LG para realizar el control automático de las configuraciones.
5. Cable Ethernet: Este cable se conecta a una red Ethernet determinada para establecer la conexión TCP-IP con el Controlador de la Matriz.

Conexión de los accesorios al controlador de Matriz

A continuación se muestra una vista frontal del Dispositivo Controlador de la Matriz, señalándose las interfaces físicas con los correspondientes accesorios:

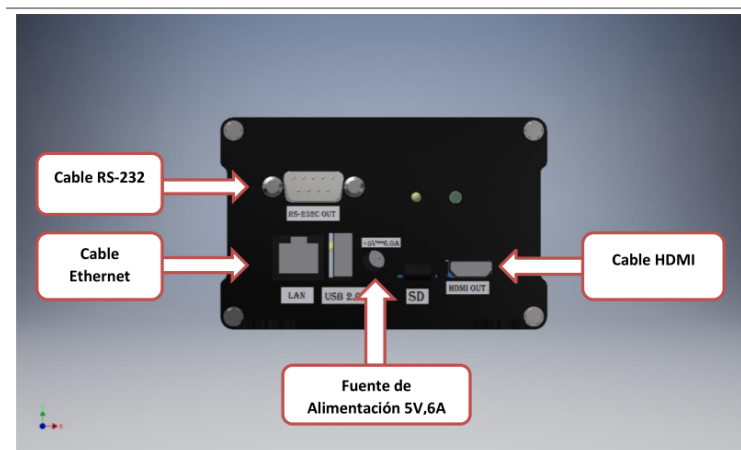


Figura 1.4. Vista frontal del Dispositivo Controlador, con los correspondientes conectores. Fuente:(*ManualUsuarioMatrizTV-LG* n.d.).

Dentro del dispositivo utilizado, se encuentra una Raspberry Pi conectada por RS232, la cual actúa como el controlador central de la matriz de televisores. La Raspberry Pi, al estar conectada a través de RS232, asegura una comunicación robusta y confiable con los componentes del sistema, permitiendo ajustes y cambios sin necesidad de intervención manual. El dispositivo cuenta con otros puertos como Puerto USB que permite conectar periféricos como teclados, ratones, cámaras y dispositivos de almacenamiento externo, el Ethernet que facilita la conexión a Internet o redes internas, la ranura para tarjeta microSD que almacena el sistema operativo y los datos del usuario.

Monitores

El monitor es un dispositivo electrónico que permite la salida de video el utilizado en para la matriz de televisores fueron, los televisores inteligentes LG al destacar por su avanzada tecnología y diseño innovador, ofreciendo una experiencia de visualización excepcional. A continuación, se presentan las características más relevantes de estos dispositivos:

Tipos de Pantalla

Tecnología: Los televisores OLED de LG utilizan píxeles que producen su propia luz, lo que permite un control preciso del brillo y un contraste infinito. Esto resulta en negros más profundos y colores más vibrantes.

Modelos destacados: Incluyen el LG OLED GX Diseño Galería, que tiene un diseño ultra delgado y se monta al ras de la pared, convirtiéndose en una pieza decorativa.

Funcionalidades de los Televisores LG

Sistema Operativo: La mayoría de los televisores LG funcionan con webOS, que permite acceder a diversas aplicaciones y servicios de streaming.

Conectividad: Cuentan con múltiples puertos HDMI y USB, así como conectividad Wi-Fi y Bluetooth, facilitando la conexión con otros dispositivos.

Calidad de Imagen: Los televisores LG son conocidos por su capacidad para ofrecer imágenes realistas gracias a tecnologías como HDR10 Pro y Dolby Vision.

Lista de entradas

HDMI, DVI, DISPLAYPORT (DP)

Monitores en cadena

La conexión en cadena de televisores, también conocida como daisy chaining, permite interconectar múltiples dispositivos de visualización para que funcionen como una sola unidad. La conexión en cadena implica conectar un televisor a otro mediante un solo cable, lo que reduce el desorden de cables y simplifica la configuración. Esto se logra utilizando puertos compatibles, como DisplayPort, que permite transmitir señales de video y audio a través de una única conexión.

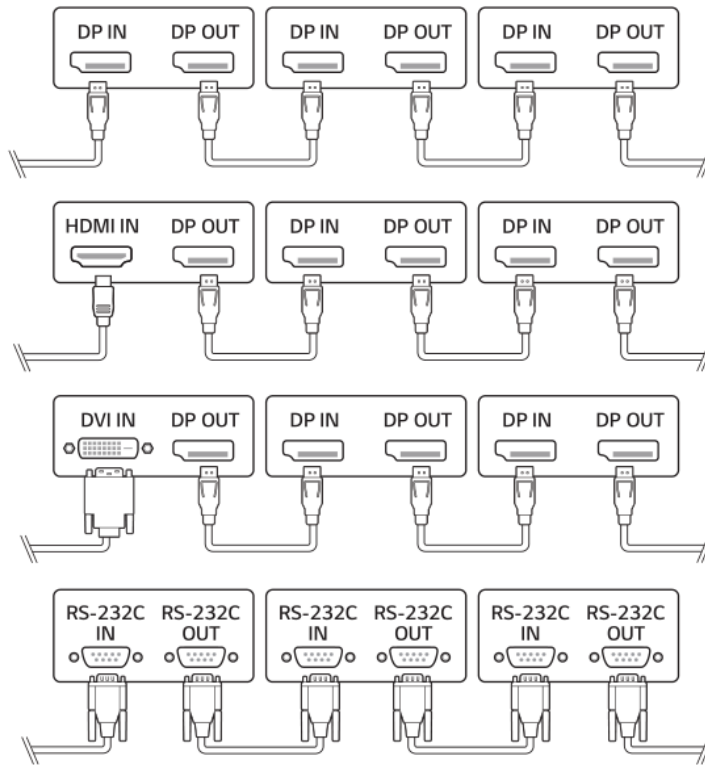


Figura 1.5. Conexión serial de los televisores. Fuente: (Manual Usuario Matriz TV-LG n.d.).

Los televisores inteligentes LG combinan tecnología de punta con un diseño elegante, proporcionando a los usuarios una experiencia audiovisual envolvente y personalizada. Su enfoque en la calidad de imagen y sonido, junto con funcionalidades inteligentes, los convierte en una opción destacada en el sistema.

1.4. Metodología de desarrollo de software

La metodología XP fue creada por Kent Beck en el año 1996. Tiene sus antecedentes desde mediados de los años 80 cuando Kent Beck y Ward Cunningham probaron formas de desarrollar software diferente a las que se habían utilizado hasta el momento, trabajando en un grupo de investigación de Tektronix. Posteriormente, en los años 90, Beck comenzó un proyecto en DaimlerChrysler, conocido como C3 (Chrysler Comprehensive Compensation) que dio paso al surgimiento de esta metodología (K. Beck, 2000). Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, la comunicación fluida entre todos los participantes, la simplicidad en las soluciones implementadas y el

coraje para enfrentar los cambios. Es adecuada para proyectos con equipos de desarrollo pequeños y con requisitos imprecisos y muy cambiantes (K. Beck, 2000), (K. Beck, 1999).

Principios de XP:

Comunicación: Fomenta la interacción constante entre los desarrolladores, los usuarios y otros interesados. Para un sistema de control de un video wall, esto implica reuniones regulares con los operadores de la instalación para entender sus necesidades y expectativas.

Simplicidad: Se apoya en construir solo lo necesario, evitando la sobrecarga funcional. En el contexto de un video wall, esto podría significar desarrollar solo las características que los usuarios requieren inicialmente, y luego ir añadiendo funcionalidades basadas en los comentarios recibidos.

Retroalimentación: Se busca obtener feedback constante a través de pruebas frecuentes y revisiones. Esto es esencial en un sistema visual como un video wall, donde la calidad de la visualización y la interacción del usuario son clave.

Valor: La entrega de funcionalidades que añadan valor al cliente es primordial. Cada iteración en el desarrollo debería facilitar el control y la personalización del video wall para maximizar su utilidad.

Prácticas de XP:

Desarrollo Iterativo e Incremental: Dividir el desarrollo en ciclos cortos (iteraciones) permite lanzar versiones tempranas del sistema y ajustar las funcionalidades según las necesidades del usuario.

Programación en Parejas: Esto promueve la colaboración entre dos programadores que trabajan juntos en el código, lo que puede ayudar a identificar problemas rápidamente y mejorar la calidad del código.

Pruebas Unitarias: Implementar pruebas automatizadas para cada componente del sistema. Esto asegura que cada parte del código funciona correctamente, lo que es crucial en un sistema que controla hardware como un video wall.

Integración Continua: Realizar integraciones frecuentes del código para detectar errores rápidamente y asegurar que el sistema siempre esté en un estado funcional.

Propiedad Colectiva del Código: Permite que todo el equipo sea responsable del código, fomentando la colaboración y el intercambio de conocimientos. En un sistema de control, esto puede llevar a una mejor comprensión de cómo funciona todo el sistema, facilitar el mantenimiento y mejorar la calidad.

La metodología Extreme Programming (XP) es crucial para el desarrollo de este sistema de control del video wall debido a su enfoque en la flexibilidad y adaptación a cambios rápidos. Las iteraciones cortas permiten responder eficazmente a las necesidades del cliente, mientras que las pruebas automatizadas garantizan la alta calidad y fiabilidad del software. Además, la colaboración constante entre el equipo de desarrollo y el

cliente fomenta una comunicación efectiva y una retroalimentación continua, lo que resulta en un producto final que satisface las expectativas y requisitos.

Conclusiones del capítulo

En este capítulo se establece un marco teórico y metodológico sobre la configuración de televisores inteligentes, resaltando su evolución y relevancia en la creación de videowalls. Se enfatiza que estos dispositivos son clave para ofrecer experiencias visuales mejoradas en entornos profesionales y comerciales. El análisis del estado del arte revela que empresas líderes reconocen la importancia de los videowalls en sus operaciones. Además, se identifican herramientas y tecnologías esenciales para desarrollar sistemas de control de videowalls. La elección de la metodología eXtreme Programming (XP) se considera crucial para asegurar la calidad y adaptabilidad del software. Por lo cual se sientan las bases para una comprensión profunda de los procesos de configuración de televisores inteligentes.

DISEÑO DE LA SOLUCIÓN PARA LA CONFIGURACIÓN DINÁMICA DE MATRICES EN TV INTELIGENTE

En este capítulo se llevará a cabo la caracterización de la propuesta de solución. Se especifican los requisitos funcionales y no funcionales a partir de la metodología seleccionada, y se realiza el diseño ingenieril donde se describen los patrones de diseño definidos como buenas prácticas durante el ciclo de desarrollo del software, la realización del modelado de diagramas, los elementos fundamentales del diseño y de la arquitectura que se deben tener en cuenta para llegar a la propuesta de solución.

2.1. Descripción de la propuesta de solución

La propuesta de solución se basa en implementar un sistema automatizado de gestión que combine hardware y software para controlar las configuraciones de forma centralizada. Este sistema incluiría un software de gestión de videowalls accesible a través de una interfaz web segura, que permita realizar cambios y ajustes de manera remota, eliminando la necesidad de operar manualmente cada televisor. Además, se integraría un sistema de almacenamiento para respaldar las configuraciones, de modo que en caso de fallos eléctricos, las configuraciones puedan restaurarse automáticamente, al restablecer la energía, minimizando el tiempo de inactividad y los esfuerzos requeridos para reiniciar las configuraciones. Para prevenir la manipulación no autorizada, se implementaría un sistema de autenticación que limite el acceso exclusivo al personal calificado y registre todas las acciones realizadas en el sistema, asegurando transparencia y trazabilidad.

El sistema permitirá programar configuraciones automáticas, asegurando que el videowall esté siempre listo para su uso sin necesidad de intervención manual. Se garantizará que el sistema sea compatible con los televisores inteligentes actuales mediante estándares como HDMI y protocolos IP, facilitando su integración.

sin necesidad de reemplazar equipos existentes.

La implementación del sistema representa una solución integral que aborda de manera efectiva los desafíos actuales asociados con la gestión manual de televisores inteligentes. A continuación, se destacan las características más relevantes:

1. **Eficiencia Operativa:** La centralización del control a través de un software accesible por interfaz web elimina la necesidad de operar manualmente cada televisor, lo que reduce significativamente el tiempo y esfuerzo requeridos para gestionar configuraciones en matrices grandes. Esto permite al personal enfocarse en tareas más críticas y productivas.

2. **Minimización de Riesgos:** La integración de un sistema de almacenamiento para respaldar configuraciones garantiza que, en caso de cortes eléctricos, las configuraciones puedan restaurarse automáticamente. Esta funcionalidad no solo minimiza el tiempo de inactividad, sino que también previene la pérdida accidental de configuraciones, asegurando una continuidad operativa.

3. **Seguridad y Control de Acceso:** La implementación de un sistema de autenticación que limita el acceso al personal calificado proporciona una capa adicional de seguridad. Al registrar todas las acciones realizadas en el sistema, se asegura la transparencia y trazabilidad, lo que es fundamental para mantener la integridad del sistema.

2.2. Requisitos, análisis y modelado del sistema

Los requisitos funcionales, como su nombre indica, describen las funciones del sistema que se va a diseñar. Es una descripción de lo que será el sistema y cómo funcionará para satisfacer las necesidades del usuario. Proporcionan una descripción clara de cómo se supone que el sistema debe responder a un comando en particular, las características y lo que esperan los usuarios (Pressman, 2010).

Tabla 2.1. Requerimientos funcionales. Fuente: Elaboración propia

No.	Nombre	Descripción	Prioridad	Complejidad
RF1	Iniciar Sesión	La interfaz debe permitir a los usuarios iniciar sesión mediante un sistema de autenticación (mediante usuario y contraseña).	Alta	Baja
RF2	Visualizar configuración	El sistema debe permitir visualizar una configuración.	Alta	Media
RF3	Aplicar configuración	El sistema debe permitir aplicar una configuración.	Alta	Media
RF4	Listar configuraciones	El sistema debe permitir listar las configuraciones.	Alta	Baja
RF5	Eliminar configuración	El sistema debe permitir eliminar una configuración.	Alta	Baja

CAPÍTULO 2. DISEÑO DE LA SOLUCIÓN PARA LA CONFIGURACIÓN DINÁMICA DE MATRICES EN TV INTELIGENTE

No.	Nombre	Descripción	Prioridad	Complejidad
RF6	Crear configuración	El sistema debe permitir crear una configuración.	Alta	Alta
RF7	Añadir conexión	El sistema debe permitir añadir conexión.	Alta	Alta
RF8	Seleccionar entrada de video	El sistema debe permitir seleccionar la entrada de video deseada (HDMI1, HDMI2, DVI o DP).	Alta	Media
RF9	Visualizar en pantalla completa	El sistema debe permitir visualizar en pantalla completa.	Media	Media
RF10	Controlar el encendido	El sistema debe permitir encender todos los televisores.	Media	Media
RF11	Controlar el apagado	El sistema debe permitir apagar todos los televisores.	Media	Media
RF12	Controlar el restablecimiento	El sistema debe permitir restablecer todos los televisores.	Media	Media
RF13	Añadir televisor	El sistema debe permitir añadir nuevo TV a la matriz.	Baja	Media
RF14	Listar televisores	El sistema debe permitir listar los televisores.	Baja	Baja
RF15	Eliminar televisores	El sistema debe permitir eliminar televisores.	Baja	Baja
RF16	Añadir usuario	El sistema debe permitir añadir usuario.	Baja	Baja
RF17	Listar usuarios	El sistema debe permitir listar los usuarios.	Baja	Baja
RF18	Editar usuario	El sistema debe permitir editar usuario.	Baja	Baja
RF19	Eliminar usuario	El sistema debe permitir eliminar usuario.	Baja	Baja
RF20	Cerrar sesión	El sistema debe permitir cerrar sesión.	Baja	Baja

Requisitos no funcionales

Los requisitos no funcionales son tan importantes como los requisitos funcionales. Si los requisitos funcionales especifican lo que debe hacer un sistema, los requisitos no funcionales describen cómo lo hará (Pressman, 2010).

Seguridad: El sistema debe implementar un robusto mecanismo de autenticación para restringir el acceso solo al personal autorizado. Además, debe registrar todas las acciones realizadas en el sistema para asegurar la trazabilidad y transparencia.

- **Disponibilidad:** El sistema debe estar disponible para su uso en todo momento, garantizando una alta disponibilidad (por ejemplo, 99,9 por ciento) para minimizar el tiempo de inactividad. Esto es crucial para mantener la operatividad del videowall en situaciones críticas.
- **Rendimiento:** El software de gestión debe ser capaz de manejar múltiples configuraciones y ajustes simultáneamente sin degradar su rendimiento. Los tiempos de respuesta ante comandos del usuario deben ser rápidos (por ejemplo, menos de 2 segundos).
- **Escalabilidad:** El sistema debe ser escalable, permitiendo la integración de más dispositivos o funcionalidades sin necesidad de reestructurar completamente la infraestructura existente.

- **Usabilidad:** La interfaz web del sistema debe ser intuitiva y fácil de usar, permitiendo que el personal calificado realice ajustes sin necesidad de formación extensa. Esto incluye aspectos como la accesibilidad y la experiencia del usuario.
- **Mantenibilidad:** El sistema debe ser fácil de mantener y actualizar, con una arquitectura que permita realizar cambios o agregar nuevas funcionalidades sin complicaciones significativas.
- **Compatibilidad:** Debe ser compatible con los estándares actuales (como HDMI y protocolos IP) para asegurar la integración sin necesidad de reemplazar equipos existentes.
- **Fiabilidad:** El sistema debe garantizar la recuperación automática de configuraciones tras cortes eléctricos o fallos del sistema, minimizando así la pérdida de datos y asegurando una continuidad operativa.

Requisitos de Software

- Sistema operativo Windows o Linux
- Utilizar navegador Google Chrome, Mozilla Firefox, Opera.

Requisitos de Hardware

Para el cliente

Una pc que tenga un navegador.

Para el servidor

Capacidad de almacenamiento de 500GB, con un mínimo 4GB de memoria RAM.

2.3. Historias de usuario

El primer paso de cualquier proyecto que siga la metodología XP es definir las historias de usuario con el cliente. Las historias de usuario tienen la misma finalidad que los casos de uso pero con algunas diferencias: Constan de 3 ó 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles, no se debe hablar ni de posibles algoritmos para su implementación ni de diseños de base de datos adecuados, etc. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el programa cumple con lo que especifica la historia de usuario. Cuando llega la hora de implementar una historia de usuario, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer dicha historia (*XP - Extreme Programming Ingeniería de Software* n.d.).

Tabla 2.2. Historia de usuario # 1

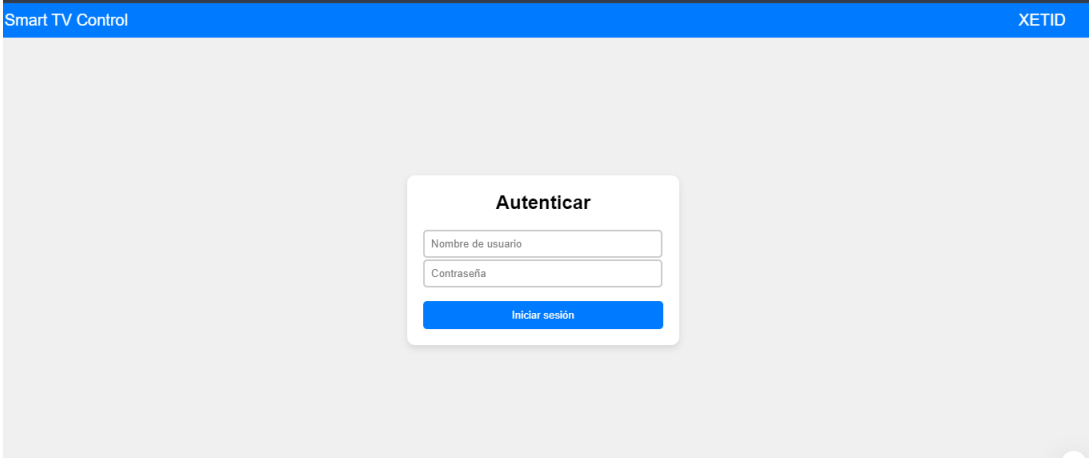
Historia de usuario	
Número: 1	Nombre: Iniciar sesión.
Usuario: Administrador	
Prioridad en negocio: Alta	Riesgo en desarrollo: Medio
Puntos estimados: 0.2	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir autenticar al administrador, insertando su usuario y contraseña.	
Observaciones: No aplica.	
Interfaz: 	

Tabla 2.3. Historia de usuario # 2

Historia de usuario	
Número: 2	Nombre: Visualizar Configuración
Usuario: Administrador	
Prioridad en negocio: Alta	Riesgo en desarrollo: Medio
Puntos estimados: 0.4	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir visualizar las configuraciones.	
Observaciones: Debe estar ya autenticado.	

Continúa en la próxima página

Tabla 2.3. Continuación de la página anterior

Interfaz:

Tabla 2.4. Historia de usuario # 3

Historia de usuario	
Número: 3	Nombre: Aplicar configuración
Usuario: Administrador	
Prioridad en negocio: Alta	Riesgo en desarrollo: Medio
Puntos estimados: 0.6	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir aplicar las configuraciones.	
Observaciones: El administrador debe estar ya autenticado.	
Interfaz:	

2.4. Plan de Iteraciones

El plan de iteración define el alcance, las tareas, las estimaciones y las pruebas de aceptación para cada historia de usuario. El plan de iteración también es flexible y se puede ajustar durante la iteración, a medida que el equipo descubre nueva información o enfrenta nuevos desafíos (*Introduction to Django* n.d.).

Iteración	HU a implementar	Duración Total (Semanas)
-----------	------------------	--------------------------

Tabla 2.5. Plan de iteraciones. Fuente: Elaboración propia.

Iteración	HU a implementar	Duración Total (Semanas)
1	1-8	4
2	9-15	3
3	16-20	2

2.5. Plan de Entrega

Después de tener ya definidas las historias de usuario es necesario crear un plan de entrega, en inglés "Release plan", donde se indiquen las historias de usuario que se crearán para cada versión del programa y las fechas en las que se publicarán estas versiones. Un Release plan^{es} una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa (K. Beck, 2000).

Tabla 2.6. Plan de entregas. Fuente: Elaboración propia.

Entregable	Duración (semanas)	Fecha de entrega
Autenticar administrador, visualizar configuración y funciones generales, Crear configuración, eliminar configuración, visualizar configuración, ,aplicar configuración,añadir conexión , Seleccionar entradas de video,visualizar pantalla completa.	4	Septiembre 2024
Controlar encendido,apagado,restablecer Listar conexiones, eliminar conexión, editar conexión, añadir televisores,.listar televisores,eliminar televisores.	3	Octubre 2024
Añadir ,listar,editar,eliminar usuario, cerrar sesión .	2	Noviembre 2024

2.6. Diseño arquitectónico

Django es un framework de desarrollo web de alto nivel que facilita la creación rápida y eficiente de sitios web seguros y mantenibles, permitiendo a los desarrolladores construir aplicaciones web con un enfoque en la reutilización de código y una estructura modular, lo que se traduce en un menor tiempo de desarrollo y mayor eficiencia en la creación de aplicaciones complejas (*Django Tutorial: Python Web Development* n.d.). Entre sus características más destacadas se encuentra su enfoque proactivo hacia la seguridad, ya que incluye protecciones integradas contra amenazas comunes como inyecciones SQL, falsificación de solicitudes entre sitios (CSRF) y clickhacking, ayudando a mantener la integridad y confidencialidad de los datos (W. Wizard, n.d.).

La arquitectura Modelo-Vista-Template (MVT) es fundamental en Django por varias razones que contribuyen a la eficiencia y calidad del desarrollo web. En primer lugar, MVT permite una clara separación de responsabilidades, donde la lógica del negocio se gestiona en el Modelo, la interfaz del usuario se presenta en la Vista y la presentación visual se maneja a través del Template, esta separación facilita el mantenimiento del código y mejora su legibilidad (CodeScouts", "n.d.").

Además, los componentes MVT son intercambiables, lo que significa que los desarrolladores pueden modificar o reemplazar partes del sistema sin afectar el resto, fomentando así la creación de aplicaciones modulares y escalables. Por último, esta arquitectura proporciona una facilidad de pruebas significativa, al tener una clara separación entre las diferentes capas, se simplifica el proceso de pruebas unitarias y funcionales, permitiendo a los desarrolladores asegurarse de que cada parte del sistema funciona correctamente antes de integrarlo todo.

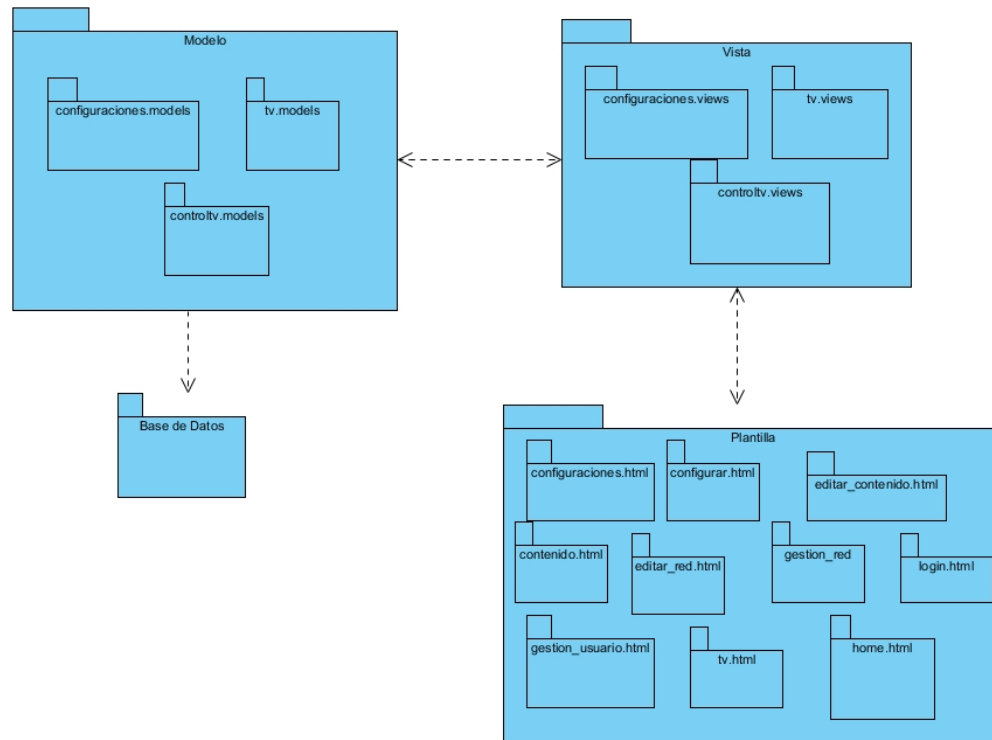


Figura 2.1. Digrama de paquete del Modelo-Vista-Plantilla. Fuente:(Elaboración propia)

Patrones de Diseño

Al diseñar el sistema utilizando la arquitectura Modelo-Vista-Template (MTV) de Django, se pueden aplicar varios patrones de diseño tanto de GRASP (General Responsibility Assignment Software Patterns) como de GoF (Gang of Four). Estos patrones ayudan a estructurar el código, mejorar la mantenibilidad y facilitar la escalabilidad del sistema. A continuación, se presentan algunos de estos patrones, su utilidad y ejemplos en el código.

Patrones GRASP

Craig Larman introdujo unos patrones muy interesantes, estos son los patrones de GRASP, el acrónimo de General Responsibility Assignment Software Patterns. Los patrones de GRASP, no compiten con los patrones de diseño de GOF, los patrones de GRASP, nos guían para ayudarnos a encontrar los patrones de diseño (Larman, 2004).

Controlador

El patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado (ibíd.).

Las vistas en Django (definidas en 'views.py') actúan como controladores, gestionando las solicitudes HTTP y delegando la lógica de negocio a los modelos.

```
from django.shortcuts import render, redirect, get_object_or_404
from .models import ConfiguracionTV
from .forms import ConfiguracionTVForm
from tv.models import Televisor, Contenido

def configuraciones(request):
    if request.method == 'POST':
        form = ConfiguracionTVForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('configuraciones') # Redirige a la misma página después de guardar
    else:
        form = ConfiguracionTVForm()

    configuraciones = ConfiguracionTV.objects.all()
    return render(request, 'configuraciones.html', {'form': form, 'configuraciones': configuraciones})

def eliminar_configuracion(request, id):
    config = get_object_or_404(ConfiguracionTV, id=id)
    config.delete()
    return redirect('configuraciones')

def aplicar_configuracion(request, id):
    config = get_object_or_404(ConfiguracionTV, id=id)
    # Establecer esta configuración como la activa
    request.session['config_activa_id'] = config.id
    return redirect('home')
```

Figura 2.2. Patrón Controlador. Fuente:(Elaboración propia)

Creator

El patrón Creator establece que una clase debe ser responsable de crear instancias de otras clases si tiene una relación con ellas. Este patrón es útil para mantener la cohesión y la claridad en el código(Larman, 2004), (Gamma; Helm; Johnson y Vlissides, 1994).

Puedes definir métodos en los modelos que sean responsables de crear instancias de otros modelos.

```
from django.db import models

class Televisor(models.Model):
    id = models.AutoField(primary_key=True) # Campo ID explícito como clave primaria
    marca = models.CharField(max_length=100)

    entrada = models.CharField(max_length=20, null=True) # Campo para la entrada
    (HDMI, DVI, etc.)
    def __str__(self):
        return self.marca

class Contenido(models.Model):
    televisor = models.ForeignKey(Televisor, on_delete=models.CASCADE)
    archivo = models.FileField(upload_to='contenidos/')

    def __str__(self):
        return f"{self.televisor} - {self.archivo.name}"
```

Figura 2.3. Patrón Creador. Fuente:(Elaboración propia)

High Cohesion

La alta cohesión implica que las clases deben tener responsabilidades bien definidas y relacionadas entre sí. Esto simplifica el mantenimiento y mejora la comprensión del sistema (*Patrones de diseño / Design patterns* n.d.). Mantener los modelos, vistas y plantillas centradas en responsabilidades específicas y relacionadas.

```
class ConfiguracionTV(models.Model):
    id = models.AutoField(primary_key=True)
    filas = models.PositiveIntegerField()
    columnas = models.PositiveIntegerField()
    nombre = models.CharField(max_length=100)

    def __str__(self):
        return f"{self.nombre}: {self.filas}x{self.columnas}"

class Televisor(models.Model):
    id = models.AutoField(primary_key=True) # Campo ID explícito como clave primaria
    marca = models.CharField(max_length=100)

    entrada = models.CharField(max_length=20, null=True) # Campo para la entrada
    (HDMI, DVI, etc.)
    def __str__(self):
        return self.marca
```

Figura 2.4. Patrón Alta cohesión. Fuente:(Elaboración propia)

Low Coupling

El bajo acoplamiento se refiere a la independencia entre clases. En Django, esto se puede lograr utilizando señales para desacoplar la lógica entre modelos y vistas (Larman, 2004), (Gamma; Helm; Johnson y Vlissides, 1994). Mantén las interacciones entre modelos y vistas bien definidas y evita dependencias innecesarias.

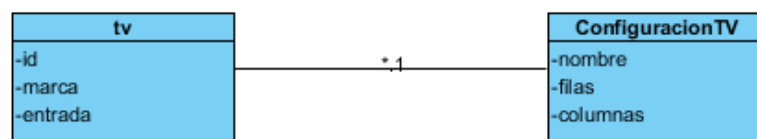


Figura 2.5. Patrón Bajo acoplamiento. Fuente:(Elaboración propia)

Patrones GoF

Los patrones de diseño (design patterns) son soluciones habituales a problemas comunes en el diseño de software. Cada patrón es como un plano que se puede personalizar para resolver un problema de diseño

particular de tu código (*Command* n.d.).

Patrón Command

Command es un patrón de diseño de comportamiento que convierte una solicitud en un objeto independiente que contiene toda la información sobre la solicitud. Esta transformación le permite pasar solicitudes como argumentos de un método, retrasar o poner en cola la ejecución de una solicitud y admitir operaciones que se pueden deshacer (ibíd.). El uso de dicho patrón en este proyecto resultó muy útil para implementar comandos que controlen los televisores, como encender, apagar, cambiar de canal, etc.

Patrón mediator

El Patrón Mediator es un patrón de diseño de comportamiento que promueve la comunicación entre objetos al introducir un objeto mediador que gestiona las interacciones entre ellos. Este patrón ayuda a reducir el acoplamiento entre los componentes al centralizar la comunicación, lo que simplifica el mantenimiento y la evolución del sistema. Utiliza un mediador para gestionar la comunicación entre componentes, como modelos y vistas (ibíd.).

```
<div class="modal-body">
  <form method="post" id="configForm">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit" class="btn btn-primary">Crear Configuración</button>
  </form>
</div>
```

Figura 2.6. Patrón Mediator. Fuente:(Elaboración propia)

```
<div class="card">
  <div class="card-header">
    <h4>Editar Configuración para {{ config.pantalla }}</h4>
  </div>
  <div class="card-body">
    <form method="post">
      {% csrf_token %}
      <div class="form-group">
        <label for="{{ form.pantalla.id_for_label }}">Pantalla:</label>
        {{ form.pantalla }}
      </div>
      <div class="form-group">
        <label for="{{ form.direccion_ip.id_for_label }}">Dirección IP:</label>
        {{ form.direccion_ip }}
      </div>
      <div class="form-group">
        <label for="{{ form.mascara_bits.id_for_label }}">Nº Bits a 1 máscara red:</label>
        {{ form.mascara_bits }}
      </div>
      <button type="submit" class="btn btn-primary">Guardar Cambios</button>
      <a href="{% url 'gestion_red' %}" class="btn btn-secondary">Cancelar</a>
    </form>
  </div>
</div>
```

Figura 2.7. Patrón Mediator. Fuente:(Elaboración propia)

En los HTML proporcionados se evidencia el patrón mediador debido a que el formulario actúa como mediador entre el usuario y el backend.

Patrón Strategy

La estrategia es un patrón de diseño de comportamiento que le permite definir una familia de algoritmos, colocar cada uno de ellos en una clase separada y hacer que sus objetos sean intercambiables (*crc.pdf* n.d.). El patrón Strategy se puede emplear para implementar diferentes estrategias de control de video, como métodos de ajuste de brillo, contraste, etc.

2.7. Tarjetas CRC

Las tarjetas CRC (Clase, Responsabilidad y Colaboración) son una herramienta de diseño, las cuales se dividen en tres secciones que contienen la información del nombre de las clases, sus responsabilidades y sus colaboraciones. Estas normalmente son usadas cuando se determina primero las clases que se necesitan, su iteración y después la implementación de la solución. La estructura de una tarjeta CRC está compuesta por el nombre de la clase, a la izquierda todas las responsabilidades, y a la derecha las clases implicadas en cada responsabilidad. En fin, las tarjetas CRC contribuyen al diseño a todo el equipo del proyecto. A continuación se presentan algunas de las tarjetas CRC de la herramienta (*Admin_{Donetonic}*, n.d.).

Tabla 2.7. Tarjeta CRC # 1

Tarjeta CRC	
Clase: ConfiguracionTV	
Responsabilidad	Colaboración
• Visualizar configuraciones existentes. • Crear nuevas configuraciones. • Eliminar configuraciones no deseadas. • Listar todas las configuraciones disponibles.	Televisores

Tabla 2.8. Tarjeta CRC # 2

Tarjeta CRC	
Clase: Televisores	
Responsabilidad	Colaboración
• Agregar televisor • Listar televisores registrados • Eliminar televisor.	ConfiguracionTV

Conclusiones del capítulo

La metodología XP aportó los artefactos necesarios para comprender el desarrollo del sistema, las funcio-

nalidades y características que permitieron definir las opciones que tendrá el sistema teniendo en cuenta las necesidades del cliente. Mientras las HU permitieron delimitar las tareas a realizar para llevar a cabo la implementación de cada funcionalidad, cómo establecer el tiempo aproximado que se dedicará para el cumplimiento de las mismas, así como para desarrollar un plan de iteración y un plan de entrega. Los patrones de diseño utilizados mejoraron significativamente el desarrollo de la aplicación, las tarjetas CRC confeccionadas ayudaron a establecer la relación entre las funcionalidades a implementar y determinar la responsabilidad y colaboración entre ellas.

VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

En el siguiente capítulo se analizan los procesos referentes al desarrollo de la solución y las pruebas realizadas para validar el correcto funcionamiento del sistema.

3.1. Tareas ingenieriles

Las tareas de ingeniería son un conjunto de actividades específicas que cumplen con cada Historia de Usuario (HU). Su descripción debe ser detallada y técnica para facilitar su construcción. Estas tareas pueden escribirse utilizando un lenguaje técnico y no necesariamente deben de ser entendibles para el cliente (Admin_{Donetonic}, n.d.). En las siguientes tablas se muestran las correspondientes Tareas de Ingeniería para las Historias de Usuario.

Tabla 3.1. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	Número de Historia de usuario: 1
Nombre de la tarea: Iniciar sesión	
Tipo de tarea: Desarrollo	Puntos estimados: 0.2
Fecha de inicio: 1 de agosto de 2024	Fecha de fin: 2 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Iniciar sesión.	

Tabla 3.2. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	Número de Historia de usuario: 2
Nombre de la tarea: Visualizar Configuración.	
Tipo de tarea: Desarrollo	Puntos estimados: 0.4
Fecha de inicio: 3 de agosto de 2024	Fecha de fin: 5 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Visualizar Configuración.	

Tabla 3.3. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	Número de Historia de usuario: 3
Nombre de la tarea: Aplicar configuración.	
Tipo de tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 6 de agosto de 2024	Fecha de fin: 10 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Aplicar configuración.	

3.2. Plan de pruebas

Un plan de pruebas para un proyecto es una guía en el proceso de verificación y validación del software, asegurando que cumpla con los requisitos establecidos y funcione correctamente antes de su lanzamiento. Se contemplan diversos tipos de pruebas, incluyendo funcionales, de usabilidad, rendimiento y seguridad. La importancia de un plan de pruebas radica en garantizar la calidad del producto final, reducir costos al identificar errores antes del lanzamiento y aumentar la satisfacción del usuario mediante un funcionamiento adecuado (Marcillo y Cruz, 2019).

Tabla 3.4. Pruebas y Validación. Fuente: Elaboración propia

Pruebas	Validación	Breve explicación
Unitarias	Unittest en Django	Verifica que cada unidad de código funcione correctamente de manera aislada.
Aceptación	Aceptación del cliente	Verifica que el sistema cumple con los requisitos y expectativas del usuario final.

3.3. Pruebas Unitarias

Para comprobar el correcto funcionamiento de la implementación del componente desarrollado, se realizaron pruebas unitarias. Se utilizó la biblioteca de Python (unit testing) y la definición de las pruebas se realizaron dentro del archivo tests.py, creado por el marco de trabajo Django. Para ejecutar los test o iniciar el servidor de prueba se hace uso del comando python manage.py test. Se realizaron pruebas unitarias en models en el views en los métodos.

Primera iteración:

```
-----  
Ran 7 tests in 1.278s  
  
FAILED (failures=1)  
Destroying test database for alias 'default'...
```

Figura 3.1. Resultado de las pruebas unitarias Primera iteración. Fuente:(Elaboración propia)

Segunda iteración:

```
-----  
Ran 10 tests in 5.337s  
Ran 10 tests in 5.337s  
  
FAILED (failures=1)
```

Figura 3.2. Resultado de las pruebas unitarias Segunda iteración. Fuente:(Elaboración propia)

Tercera iteración:

```
-----  
Ran 11 tests in 5.562s  
  
FAILED (failures=2)
```

Figura 3.3. Resultado de las pruebas unitarias Tercera iteración. Fuente:(Elaboración propia)

Cuarta iteración:

```

-----
Ran 18 tests in 5.620s

FAILED (failures=2)

```

Figura 3.4. Resultado de las pruebas unitarias Cuarta iteración. Fuente:(Elaboración propia)

Quinta iteración:

```

-----
Ran 15 tests in 9.486s

OK

```

Figura 3.5. Resultado de las pruebas unitarias Quinta iteración. Fuente:(Elaboración propia)

```

test_form_invalid (tv.tests.NetworkConfigFormTestCase.test_form_invalid) ... ok
test_form_valid (tv.tests.NetworkConfigFormTestCase.test_form_valid) ... ok
test_str_method (tv.tests.NetworkConfigModelTest.test_str_method) ... ok
test_gestion_red_view_get (tv.tests.NetworkConfigViewsTest.test_gestion_red_view_get) ... ok
test_form_invalid (tv.tests.TelevisorFormTestCase.test_form_invalid) ... ok
test_form_valid (tv.tests.TelevisorFormTestCase.test_form_valid) ... ok
test_str_method (tv.tests.TelevisorModelTest.test_str_method) ... ok
test_eliminar_contenido (tv.tests.TelevisorViewsTest.test_eliminar_contenido) ... ok
test_tv_view_get (tv.tests.TelevisorViewsTest.test_tv_view_get) ... ok
test_tv_view_post_valid (tv.tests.TelevisorViewsTest.test_tv_view_post_valid) ... ok
test_login_invalid (tv.tests.UserAuthenticationTestCase.test_login_invalid) ... ok
test_login_valid (tv.tests.UserAuthenticationTestCase.test_login_valid) ... ok
test_protected_view (tv.tests.UserAuthenticationTestCase.test_protected_view) ... ok
test_str_method (tv.tests.UserProfileModelTest.test_str_method) ... ok
test_home_view (tv.tests.UserViewsTestCase.test_home_view) ... ok

```

Figura 3.6. Resultado de las pruebas unitarias Quinta iteración. Fuente:(Elaboración propia)

Luego de varias interacciones se logró ejecutar las pruebas unitarias con éxito logrando encontrar los fallos y errores en las pruebas los cuales se evidencian a continuación:

```
test_editar_red_view_post (tv.tests.NetworkConfigViewsTest.test_editar_red_view_post) ... ERROR

test_login_message (tv.tests.UserAuthenticationTestCase.test_login_message) ... FAIL

test_form_valid (tv.tests.ContenidoFormTestCase.test_form_valid) ... FAIL
test_str_method (tv.tests.ContenidoModelTest.test_str_method) ... FAIL
```

Figura 3.7. Resultado de las pruebas unitarias final. Fuente:(Elaboración propia)

La corrección de errores gracias a las pruebas unitarias no solo mejora la estabilidad y la calidad, sino que también ahorra tiempo y recursos, ya que reduce la necesidad de depuración y trabajo en el futuro.

3.4. Pruebas de Aceptación

Las pruebas de aceptación son un componente esencial en el desarrollo de una página web de control de video wall, ya que determinan si el software cumple con los requisitos y expectativas del usuario final. Estas pruebas se realizan al final del ciclo de desarrollo, después de que se han completado las pruebas unitarias. Las pruebas de aceptación son un tipo de evaluación que verifica si el sistema desarrollado cumple con los criterios establecidos en las historias de usuario y si es adecuado para su uso en un entorno real. Se centran en validar que el software satisface las necesidades del cliente y los usuarios finales, asegurando que la funcionalidad y la usabilidad sean las esperadas:

Tabla 3.5. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: P1_HU1	Historia de usuario: 1
Nombre: Iniciar sesión.	
Descripción: Se debe probar la autenticación exitosa.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • El usuario abre la interfaz web. • El usuario rellena los campos para el acceso. • El usuario accede al Sistema.	
Resultados esperados: Autenticación exitosa.	

Tabla 3.6. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: P2_HU2	Historia de usuario: 2
Nombre: Visualizar Configuración.	
Descripción: Se debe probar que el usuario pueda visualizar la configuración de la matriz.	
Condiciones de ejecución: El usuario debe estar logueado en el sistema.	
Pasos de ejecución: • No se requieren pasos específicos para este caso.	
Resultados esperados: El usuario visualiza la configuración de la matriz exitosamente.	

Tabla 3.7. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: P3_HU3	Historia de usuario: 3
Nombre: Aplicar configuración.	
Descripción: Se debe probar que el usuario pueda aplicar la configuración de la matriz.	
Condiciones de ejecución: El usuario debe estar logueado en el sistema.	
Pasos de ejecución: • Ir a configuraciones y dar click en el botón aplicar. • Dar click en el botón volver.	
Resultados esperados: Se aplicó exitosamente la configuración de la matriz.	

Resultado de las pruebas

La realización exitosa de pruebas unitarias y de aceptación proporciona una validación integral del sistema, garantizando que cada componente funcione correctamente en aislamiento y que, en conjunto, satisfaga los requisitos y expectativas del usuario final. Estas pruebas no solo confirman que las funcionalidades específicas, como encender/apagar televisores, operan según lo esperado, sino que también aseguran que la interfaz de usuario es intuitiva y eficiente, y que los flujos críticos, como la autenticación y gestión de red, son robustos y fiables. En resumen, las pruebas unitarias y de aceptación validan tanto la integridad del código individual como la usabilidad y funcionalidad completa del sistema, proporcionando confianza en la calidad y estabilidad del software desarrollado.

Conclusiones del capítulo

Luego del desarrollo del presente capítulo sobre la validación mediante pruebas de la propuesta de solución, se arribó a la siguiente conclusión:

La integración de estas prácticas ha proporcionado una validación integral del sistema, asegurando tanto la integridad del código individual como la funcionalidad y satisfacción del usuario final. Esta combinación de

tareas de ingeniería, plan de pruebas, pruebas unitarias y de aceptación ha permitido desarrollar un sistema de control de video wall robusto, confiable y alineado con los objetivos.

Conclusiones

La investigación desarrollada ha permitido destacar los resultados obtenidos en diversas áreas. Entre ellos se encuentran:

- Se utilizó la metodología Extreme Programming (XP) debido a su enfoque en la flexibilidad y adaptación a cambios rápidos. Esta metodología permitió realizar iteraciones cortas, facilitando la respuesta a las necesidades del cliente y garantizando la calidad del software a través de pruebas automatizadas.
- El análisis de sistemas similares reveló características clave que son esenciales para el desarrollo del sistema propuesto. Se identificaron prácticas efectivas y errores comunes, lo que permitió optimizar recursos y tiempo en el desarrollo.
- La solución propuesta consistió en un sistema integral que combina hardware y software para el control centralizado de las configuraciones de video walls. Se diseñó una interfaz web segura que permite realizar ajustes remotos, respaldar configuraciones y programar cambios automáticos. Además, se implementó un sistema de autenticación para asegurar el acceso solo al personal calificado, garantizando así la integridad del sistema.
- Las pruebas unitarias y de aceptación confirmaron que todos los componentes del sistema funcionaban correctamente tanto en aislamiento como en conjunto. Los resultados fueron satisfactorios, validando no solo las funcionalidades específicas como encender y apagar televisores, sino también la usabilidad de la interfaz. Estas pruebas aseguraron que los flujos críticos del sistema eran robustos y fiables, proporcionando confianza en la calidad y estabilidad del software desarrollado.

Este sistema establece una solución integral que aborda eficazmente los desafíos en la gestión de videowalls y optimiza los recursos disponibles en entornos como presentaciones, publicidad digital y eventos en vivo.

Recomendaciones

Para proyectos futuros que involucren el desarrollo de proyectos similares al control de video wall, se recomienda considerar los siguientes aspectos:

- Implementar una aplicación móvil para el control de la matriz de televisores facilitando la comodidad del usuario.
- Implementar funcionalidades como la gestión de contenidos para tener mayor control en lo que se quiere mostrar en cada televisor.
- Implementar un mecanismo de notificaciones ante anomalías de configuraciones indebidas para que el usuario tenga conocimiento de cuando hacer un restablecimiento en la configuración.

Referencias bibliográficas

- ADMIN_DONETONIC, n.d. User Story Mapping, qué es y cómo crearlo DoneTonic. Url: <https://donetonic.com/es/user-story-mapping/>. Accedido: 29 diciembre del año no disponible" (vid. págs. 32, 34).
- Applications of LED Video Walls: Everything You Need to Know*, 2024. Url: <https://hartfordrents.com/blog/led-video-wall-applications/>. Accedido: 28 de noviembre de 2024 (vid. pág. 6).
- BYERS, K., 2024. *The Best Video Wall Uses For Your Industry*. Url: <https://www.lamasatech.com/blog/video-walls-uses/>. Accedido: 28 de noviembre de 2024 (vid. pág. 1).
- CGARCIA, 2024. *¿Qué es Raspberry Pi y para qué sirve? | Escuela de programación, robótica y pensamiento computacional | Codelearn.es*. Url: <https://codelearn.es/blog/que-es-raspberry-pi-y-para-que-sirve/>. Accedido: 29 de noviembre de 2024 (vid. pág. 8).
- CODESCOUTS
, "n.d." *"Patrones GRASP, el viejo SOLID"*. Url: <https://www.codescouts.academy/blog/grasp/>. Accedido: 29 diciembre del año no disponible" (vid. pág. 27).
- Command*, n.d. Url: <https://refactoring.guru/es/design-patterns/command>. Accedido: 28 diciembre del año no disponible" (vid. pág. 31).
- crc.pdf*, n.d. Url: <https://lsi2.ugr.es/~mvega/docis/crc.pdf>. Accedido: 29 diciembre del año no disponible" (vid. pág. 32).
- DIGITAL, I. N. S., 2024. *Video Wall: Todo lo que debes saber*. Url: <https://www.ins-digital.com/video-wall-que-es/>. Accedido: 28 de noviembre de 2024 (vid. págs. 1, 5, 6).
- Django Tutorial: Python Web Development*, n.d. Url: %7B<https://www.datacamp.com/tutorial/web-development-django>%7D. Accedido: 28 de noviembre del año no disponible (vid. pág. 27).
- El lenguaje C++ Fundamentos de Programación en C++*, 2024. Url: https://www2.eii.uva.es/fund_inf/cpp/temas/1_introduccion/introduccion.html. Accedido: 29 de noviembre de 2024 (vid. pág. 12).
- Free UML Tool*, 2024. Url: <https://www.visual-paradigm.com/solution/freeumltool/>. Accedido: 29 de noviembre de 2024 (vid. pág. 11).

- GAMMA, E.; HELM, R.; JOHNSON, R. y VLISSIDES, J., 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Pearson Education (vid. págs. 29, 30).
- HTML y la semántica web - HTML en español, 2024. Url: <https://lenguajehtml.com/html/introduccion/que-es-html/>. Accedido:29 de noviembre de 2024 (vid. pág. 11).
- INAUGURAN NUEVOS SISTEMAS DE VIDEO EN LA OSDE Y LA ESCUELA DE CORREOS DE CUBA, 2024. Url: <https://www.correos.cu/2021/02/08/inauguran-nuevos-sistemas-de-video-conferencia-en-la-osde-y-la-escuela-de-correos-de-cuba/>. Accedido: 28 de noviembre de 2024 (vid. pág. 1).
- Introduction to Django, n.d. Url: https://www.w3schools.com/django/django_intro.php. Accedido:28 de noviembre del año no disponible (vid. pág. 25).
- K. BECK, 2000. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional (vid. págs. 17, 18, 26).
- K. BECK, 1999. Embracing change with extreme programming. *Computer*. Vol. 32, n.º 10, págs. 70-77. Disponible desde DOI: {10.1109/2.796139} (vid. pág. 18).
- LARMAN, C., 2004. *Agile and Iterative Development A Managers Guide*. Boston: Addison-Wesley Profesional (vid. págs. 28-30).
- Lenguaje Javascript - Javascript en español, 2024. Url: <https://lenguajejs.com/javascript/>. Accedido:29 de noviembre de 2024 (vid. pág. 12).
- ManualUsuarioMatrizTV-LG, n.d. Url: <https://www.lg.com/colombia/usuarios/manuales>. Accedido:29 de noviembre del año no disponible (vid. págs. 14, 15, 17).
- MARCILLO, S. R. Marquínez y CRUZ, G. R. Poveda, 2019. *Implementación de cartelera digital multimedia Video Wall mediante el uso de Raspberries*. Quito. Url: <http://bibdigital.epn.edu.ec/handle/15000/20388>. bachelorThesis. .Accedido:28 diciembre del año no disponible" (vid. pág. 35).
- MATROX MURACONTROL SOFTWARE CONTROL VIDEOWALL, 2024. Url: <https://charmex.net/matrox-muracontrol--mura-ctrlwf>. Accedido: 28 de noviembre de 2024 (vid. pág. 9).
- Noticias - Los 50 principales proveedores de videowall LED en EE. UU. 2024. Url: <https://www.bescan-led.com/es/news/top-50-led-video-wall-suppliers-in-usa/>. Accedido: 29 de noviembre de 2024 (vid. pág. 8).
- Patrones de diseño / Design patterns, n.d. Url: <https://refactoring.guru/es/design-patterns>. .Accedido:29 diciembre del año no disponible" (vid. pág. 30).
- Plataforma líder de controladores de videowall | UsefulTM, 2024. Url: <https://es.userful.com/video-wall-controller>. Accedido:28 de noviembre de 2024 (vid. pág. 9).

- PRESSMAN, Roger S., 2010. *Ingeniería del software: Un enfoque práctico*. 7ma. McGraw-Hill. Url: <https://github.com/jmartingimenez/UNLaM/blob/master/Introducci%C3%B3n%20a%20la%20Administra-ci%C3%B3n%20de%20Proyectos/Ingenieria%20del%20software.%20Un%20enfoque%20practico,%207ma%20Edicion%20-%20Roger%20S.%20Pressman.pdf>. Accedido: 28 de noviembre de 2024 (vid. págs. 21, 22).
- PYTHON DOCUMENTATION, 2023. *El tutorial de Python*. Url: <https://docs.python.org/3/tutorial/index.html>. Accedido: 29 de noviembre de 2023 (vid. pág. 13).
- ¿Qué es CSS? - CSS en español, 2024. Url: <https://lenguajecss.com/css/introduccion/que-es-css/>. Accedido: 29 de noviembre de 2024 (vid. pág. 11).
- ¿Qué es y para qué sirve UML? Versiones de UML (Lenguaje Unificado de Modelado). Tipos de diagramas UML. 2024. Url: https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=688:ique-es-y-para-que-sirve-uml-versiones-de-uml-lenguaje-unificado-de-modelado-tipos-de-diagramas-uml&catid=46&Itemid=163. Accedido: 29 de noviembre de 2024 (vid. pág. 11).
- Tecnología y mobiliario técnico para centros de control, 2024. Url: <https://gesab.com/>. Accedido: 28 de noviembre de 2024 (vid. pág. 10).
- Una guía básica de Smart TV que necesita saber, 2024. Url: <https://www.tcl.com/latin/es/blog/guide-to-smart-tv>. Accedido: 29 de noviembre de 2024 (vid. pág. 7).
- Universidad de las Ciencias Informáticas, 2024. Url: <https://www.uci.cu/>. Accedido: 28 de noviembre de 2024 (vid. pág. 2).
- Video Wall para Centros de Comando: Toma de Decisiones en Tiempo Real - Seguridad con Innovación y Confiabilidad - C2H Solutions, 2024. Url: <https://c2hsolutions.com.br/es/videowall-para-centros-de-comando-toma-de-decisiones-en-tiempo-real/>. Accedido: 28 de noviembre de 2024 (vid. pág. 9).
- VuWall Video Wall Control Systems and KVM stations | VuWall, 2024. Url: https://vuwall.com/?gad_source=1&gclid=CjwKCAiAxqC6BhBcEiwALXp455BaSLsRaGJnQAG2zCxnQGr_NN1-aZymv2s6NZh0es2MiFp3DBdFwBwE. Accedido: 28 de noviembre de 2024 (vid. pág. 10).
- W. WIZARD, n.d. *Understanding Django's MVT Architecture: A Beginners Guide to Models, Views, and Templates* | Web Wizard. Url: <https://www.webwizard.ie/blog/understanding-djangos-mvt-architecture-a-beginners-guide-to-models-views-and-templates>. Accedido: 28 de noviembre del año no disponible (vid. pág. 27).
- What is a Video Wall? Its techs (AVSM vs. AVoIP) & use cases, 2024. Url: <https://www.kramerav.com/content-hub/blog/what-is-a-video-wall/>. Accedido: 28 de noviembre de 2024 (vid. pág. 1).

XP - Extreme Programing Ingenieria de Software, n.d. Url: %7Bhttp://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html%7D. Accedido:28 de noviembre del año no disponible (vid. pág. 23).

Apéndices

Historias de Usuario

Tabla A.1. Historia de usuario # 4

Historia de usuario	
Número: 4	Nombre: Listar Configuraciones
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.3	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir listar las configuraciones.	
Observaciones: No aplica.	
Interfaz: Configuraciones Existentes mat1: 2x2 mat2: 2x3 mat3: 3x3	

Tabla A.2. Historia de usuario # 5

Historia de usuario	
Número: 5	Nombre: Eliminar Configuración
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.4	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	

Continúa en la próxima página

Tabla A.2. Continuación de la página anterior

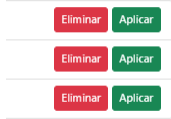
Descripción: El sistema debe permitir eliminar las configuraciones.
Observaciones: No aplica.
Interfaz: 

Tabla A.3. Historia de usuario # 6

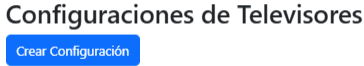
Historia de usuario	
Número: 6	Nombre: Crear Configuración
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.6	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir crear las configuraciones.	
Observaciones: No aplica.	
Interfaz: 	

Tabla A.4. Historia de usuario # 7

Historia de usuario	
Número: 7	Nombre: Añadir conexión
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.8	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir añadir conexión IP.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.5. Historia de usuario # 8

Historia de usuario	
Número: 8	Nombre: Seleccionar entrada de video
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.7	Iteración asignada: 1
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir seleccionar la entrada de video.	
Observaciones: No aplica.	
Interfaz: 	

Tabla A.6. Historia de usuario # 9

Historia de usuario	
Número: 9	Nombre: Visualizar en pantalla completa
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.4	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir ver en pantalla completa.	
Observaciones: No aplica.	
Interfaz: 	

Tabla A.7. Historia de usuario # 10

Historia de usuario	
Número: 10	Nombre: Controlar el encendido
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.5	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir encender los televisores.	
Observaciones: No aplica.	

Continúa en la próxima página

Tabla A.7. Continuación de la página anterior

Interfaz:

Tabla A.8. Historia de usuario # 11

Historia de usuario	
Número: 11	Nombre: Controlar el apagado
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.5	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir apagar los televisores.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.9. Historia de usuario # 12

Historia de usuario	
Número: 12	Nombre: Controlar el restablecimiento
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.4	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir restablecer los televisores.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.10. Historia de usuario # 13

Historia de usuario	
Número: 13	Nombre: Añadir televisor
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.3	Iteración asignada: 2

Continúa en la próxima página

Tabla A.10. Continuación de la página anterior

Programador responsable: Fernando Parra Masjuan
Descripción: El sistema debe permitir añadir televisor.
Observaciones: No aplica.
Interfaz:

Tabla A.11. Historia de usuario # 14

Historia de usuario	
Número: 14	Nombre: Listar televisores
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.3	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir listar los televisores.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.12. Historia de usuario # 15

Historia de usuario	
Número: 15	Nombre: Eliminar televisores
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.6	Iteración asignada: 2
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir editar las conexiones.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.13. Historia de usuario # 16

Historia de usuario	
Número: 16	Nombre: Añadir Usuario

Continúa en la próxima página

Tabla A.13. Continuación de la página anterior

Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.3	Iteración asignada: 3
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir añadir usuario.	
Observaciones: No aplica.	
Interfaz: 	

Tabla A.14. Historia de usuario # 17

Historia de usuario	
Número: 17	Nombre: Listar usuarios
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.2	Iteración asignada: 3
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir listar usuario.	
Observaciones: No aplica.	
Interfaz: 	

Tabla A.15. Historia de usuario # 18

Historia de usuario	
Número: 18	Nombre: Editar usuario

Continúa en la próxima página

Tabla A.15. Continuación de la página anterior

Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.4	Iteración asignada: 3
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir editar usuario.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.16. Historia de usuario # 19

Historia de usuario	
Número: 19	Nombre: Eliminar usuario
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.2	Iteración asignada: 3
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir eliminar usuario.	
Observaciones: No aplica.	
Interfaz:	

Tabla A.17. Historia de usuario # 20

Historia de usuario	
Número: 20	Nombre: Cerrar sesión
Usuario: Administrador	
Prioridad en negocio: Media	Riesgo en desarrollo: No aplica
Puntos estimados: 0.2	Iteración asignada: 3
Programador responsable: Fernando Parra Masjuan	
Descripción: El sistema debe permitir cerrar sesión.	
Observaciones: No aplica.	
Interfaz:	

Tareas Ingenieriles

Tabla B.1. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	Número de Historia de usuario: 4
Nombre de la tarea: Listar configuraciones	
Tipo de tarea: Desarrollo	Puntos estimados: 0.3
Fecha de inicio: 11 de agosto de 2024	Fecha de fin: 13 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Listar configuraciones.	

Tabla B.2. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	Número de Historia de usuario: 5
Nombre de la tarea: Eliminar configuración	
Tipo de tarea: Desarrollo	Puntos estimados: 0.4
Fecha de inicio: 13 de agosto de 2024	Fecha de fin: 16 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Eliminar configuración.	

Tabla B.3. Tarea de ingeniería # 6

Tarea	
Número de tarea: 6	Número de Historia de usuario: 6
Nombre de la tarea: Crear configuración	

Continúa en la próxima página

Tabla B.3. Continuación de la página anterior

Tipo de tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 17 de agosto de 2024	Fecha de fin: 21 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Crear configuración.	

Tabla B.4. Tarea de ingeniería # 7

Tarea	
Número de tarea: 7	Número de Historia de usuario: 7
Nombre de la tarea: Añadir conexión	
Tipo de tarea: Desarrollo	Puntos estimados: 0.8
Fecha de inicio: 22 de agosto de 2024	Fecha de fin: 27 de agosto de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Añadir conexión.	

Tabla B.5. Tarea de ingeniería # 8

Tarea	
Número de tarea: 8	Número de Historia de usuario: 8
Nombre de la tarea: Seleccionar entrada de video	
Tipo de tarea: Desarrollo	Puntos estimados: 0.7
Fecha de inicio: 28 de agosto de 2024	Fecha de fin: 1 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Seleccionar entrada de video.	

Tabla B.6. Tarea de ingeniería # 9

Tarea	
Número de tarea: 9	Número de Historia de usuario: 9
Nombre de la tarea: Visualizar en pantalla completa	
Tipo de tarea: Desarrollo	Puntos estimados: 0.4
Fecha de inicio: 2 de septiembre de 2024	Fecha de fin: 4 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Visualizar en pantalla completa.	

Tabla B.7. Tarea de ingeniería # 10

Tarea
Continúa en la próxima página

Tabla B.7. Continuación de la página anterior

Número de tarea: 10	Número de Historia de usuario: 10
Nombre de la tarea: Controlar el encendido	
Tipo de tarea: Desarrollo	Puntos estimados: 0.5
Fecha de inicio: 5 de septiembre de 2024	Fecha de fin: 7 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Controlar el encendido.	

Tabla B.8. Tarea de ingeniería # 11

Tarea	
Número de tarea: 11	Número de Historia de usuario: 11
Nombre de la tarea: Controlar el apagado	
Tipo de tarea: Desarrollo	Puntos estimados: 0.5
Fecha de inicio: 8 de septiembre de 2024	Fecha de fin: 10 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Controlar el apagado.	

Tabla B.9. Tarea de ingeniería # 12

Tarea	
Número de tarea: 12	Número de Historia de usuario: 12
Nombre de la tarea: Controlar el restablecimiento	
Tipo de tarea: Desarrollo	Puntos estimados: 0.4
Fecha de inicio: 11 de septiembre de 2024	Fecha de fin: 13 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Controlar el restablecimiento.	

Tabla B.10. Tarea de ingeniería # 13

Tarea	
Número de tarea: 13	Número de Historia de usuario: 13
Nombre de la tarea: Añadir televisor	
Tipo de tarea: Desarrollo	Puntos estimados: 0.3
Fecha de inicio: 13 de septiembre de 2024	Fecha de fin: 15 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de añadir televisor.	

Tabla B.11. Tarea de ingeniería # 14

Tarea	
Número de tarea: 14	Número de Historia de usuario: 14
Nombre de la tarea: Listar televisores	
Tipo de tarea: Desarrollo	Puntos estimados: 0.3
Fecha de inicio: 16 de septiembre de 2024	Fecha de fin: 18 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de listar televisores.	

Tabla B.12. Tarea de ingeniería # 15

Tarea	
Número de tarea: 15	Número de Historia de usuario: 15
Nombre de la tarea: Eliminar televisores	
Tipo de tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 18 de septiembre de 2024	Fecha de fin: 21 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Eliminar televisores.	

Tabla B.13. Tarea de ingeniería # 16

Tarea	
Número de tarea: 16	Número de Historia de usuario: 16
Nombre de la tarea: Añadir usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.3
Fecha de inicio: 25 de septiembre de 2024	Fecha de fin: 27 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Añadir usuario	

Tabla B.14. Tarea de ingeniería # 17

Tarea	
Número de tarea: 17	Número de Historia de usuario: 17
Nombre de la tarea: Listar usuarios	
Tipo de tarea: Desarrollo	Puntos estimados: 0.2
Fecha de inicio: 27 de septiembre de 2024	Fecha de fin: 28 de septiembre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Listar usuarios	

Tabla B.15. Tarea de ingeniería # 18

Tarea	
Número de tarea: 18	Número de Historia de usuario: 18
Nombre de la tarea: Editar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.4
Fecha de inicio: 29 de septiembre de 2024	Fecha de fin: 1 de octubre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Editar usuario	

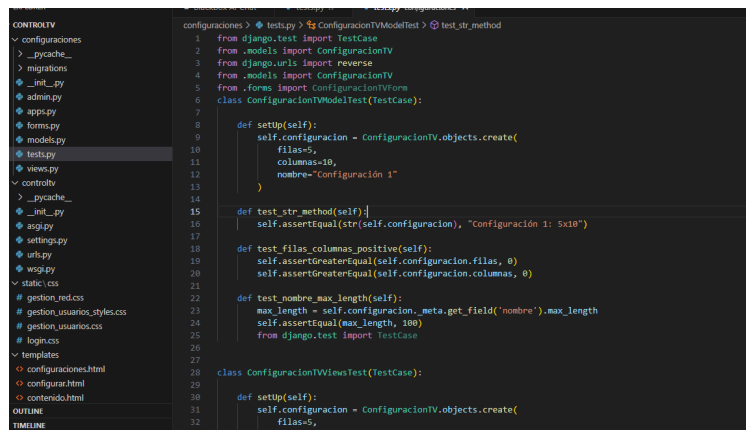
Tabla B.16. Tarea de ingeniería # 19

Tarea	
Número de tarea: 19	Número de Historia de usuario: 19
Nombre de la tarea: Eliminar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 0.3
Fecha de inicio: 2 de octubre de 2024	Fecha de fin: 4 de octubre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Eliminar usuario	

Tabla B.17. Tarea de ingeniería # 20

Tarea	
Número de tarea: 20	Número de Historia de usuario: 20
Nombre de la tarea: Cerrar sesión	
Tipo de tarea: Desarrollo	Puntos estimados: 0.2
Fecha de inicio: 4 de octubre de 2024	Fecha de fin: 5 de octubre de 2024
Programador responsable: Fernando Parra Masjuan	
Descripción: Implementar la funcionalidad de Cerrar sesión	

Pruebas unitarias



```

1 from django.test import TestCase
2 from .models import ConfiguracionTV
3 from django.urls import reverse
4 from .models import ConfiguracionTV
5 from .forms import ConfiguracionTVForm
6 class ConfiguracionTVModelTest(TestCase):
7
8     def setUp(self):
9         self.configuracion = ConfiguracionTV.objects.create(
10             filas=5,
11             columnas=10,
12             nombre="Configuración 1"
13         )
14
15     def test_str_method(self):
16         self.assertEqual(str(self.configuracion), "Configuración 1: 5x10")
17
18     def test_filas_columnas_positivo(self):
19         self.assertGreaterEqual(self.configuracion.filas, 0)
20         self.assertGreaterEqual(self.configuracion.columnas, 0)
21
22     def test_nombre_max_length(self):
23         max_length = self.configuracion._meta.get_field('nombre').max_length
24         self.assertEqual(max_length, 100)
25
26 from django.test import TestCase
27
28 class ConfiguracionTVViewsTest(TestCase):
29
30     def setUp(self):
31         self.configuracion = ConfiguracionTV.objects.create(
32             filas=5,

```

Figura C.1. Pruebas unitarias Fuente: Elaboración propia



```

class TelevisorViewsTest(TestCase):
    def setUp(self):
        self.user = User.objects.create_user(username='admin', password='adminpass', is_superuser=True)
        self.client.login(username='admin', password='adminpass')
        self.televisor = Televisor.objects.create(marca='Samsung', entrada='HDMI')

    def test_tv_view_get(self):
        response = self.client.get(reverse('tv'))
        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'tv.html')
        self.assertContains(response, 'Samsung')

    def test_tv_view_post_valid(self):
        response = self.client.post(reverse('tv'), {
            'marca': 'LG',
            'entrada': 'DVI'
        })
        self.assertEqual(response.status_code, 302) # Redirección
        self.assertTrue(Televisor.objects.filter(marca='LG').exists())

    def test_eliminar_contenido(self):
        contenido = Contenido.objects.create(televisor=self.televisor, archivo='path/to/file.mp4')
        response = self.client.post(reverse('eliminar_contenido', args=[contenido.id]))
        self.assertEqual(response.status_code, 302) # Redirección
        self.assertFalse(Contenido.objects.filter(id=contenido.id).exists())

```

Figura C.2. Pruebas unitarias Fuente: Elaboración propia

```

configuraciones > tests.py > ConfiguraciónTVModelTest > test_str_method
28 class ConfiguraciónTVViewsTest(TestCase):
29     def setUp(self):
30         filas=5,
31         columnas=10,
32         nombre='Configuración 1'
33     )
34
35     def test_configuraciones_view_get(self):
36         response = self.client.get(reverse('configuraciones'))
37         self.assertEqual(response.status_code, 200)
38         self.assertTemplateUsed(response, 'configuraciones.html')
39         self.assertContains(response, "Configuración 1")
40
41     def test_configuraciones_view_post_valid(self):
42         response = self.client.post(reverse('configuraciones'), {
43             'filas': 3,
44             'columnas': 6,
45             'nombre': 'Configuración 2'
46         })
47         self.assertEqual(response.status_code, 302) # Redirección
48         self.assertTrue(ConfiguraciónTV.objects.filter(nombre='Configuración 2').exists())
49
50     def test_eliminar_configuracion(self):
51         response = self.client.post(reverse('eliminar_configuracion', args=[self.configuracion.id]))
52         self.assertEqual(response.status_code, 302) # Redirección
53         self.assertFalse(ConfiguraciónTV.objects.filter(id=self.configuracion.id).exists())
54
55     def test_aplicar_configuracion(self):
56         response = self.client.get(reverse('aplicar_configuracion', args=[self.configuracion.id]))
57         self.assertEqual(response.status_code, 302) # Redirección
58         self.assertEqual(self.client.session['config_activa_id'], self.configuracion.id)
59
60

```

Figura C.3. Pruebas unitarias Fuente: Elaboración propia

```

class UserProfileModelTest(TestCase):
    def setUp(self):
        self.user = User.objects.create_user(username='testuser', password='testpass')
        self.profile = UserProfile.objects.create(user=self.user)

    def test_str_method(self):
        self.assertEqual(str(self.profile), "testuser - Usuario Nivel 1")

class TelevisorModelTest(TestCase):
class ContenidoFormTest(TestCase):
    def setUp(self):
        self.televisor = Televisor.objects.create(marca='Samsung', entrada='HDMI 1')

class NetworkConfigModelTest(TestCase):
    def setUp(self):
        self.televisor = Televisor.objects.create(marca='Sony', entrada='HDMI')
        self.network_config = NetworkConfig.objects.create(pantalla=self.televisor, direccion_ip='192.168.1.1', mascara_bits=24)

    def test_str_method(self):
        self.assertEqual(str(self.network_config), f"{self.televisor} - 192.168.1.1/24")

class TelevisorViewsTest(TestCase):
    def setUp(self):
        self.user = User.objects.create_user(username='admin', password='adminpass', is_superuser=True)
        self.client.login(username='admin', password='adminpass')
        self.televisor = Televisor.objects.create(marca='Samsung', entrada='HDMI')

    def test_tv_view_get(self):
        response = self.client.get(reverse('tv'))

```

Figura C.4. Pruebas unitarias Fuente: Elaboración propia

Pruebas de aceptación

Tabla D.1. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: P4_HU4	Historia de usuario: 4
Nombre: Listar Configuraciones	
Descripción: Se debe probar el listar configuraciones.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir Configuraciones.	
Resultados esperados: Listar configuraciones exitosamente.	

Tabla D.2. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: P5_HU5	Historia de usuario: 5
Nombre: Eliminar Configuración	
Descripción: Se debe probar eliminar configuración.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir Configuraciones. • Dar click en eliminar.	
Resultados esperados: Eliminar configuración exitosa.	

Tabla D.3. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: P6_HU6	Historia de usuario: 6
Nombre: Crear Configuración	
Descripción: Se debe probar crear configuración exitosamente.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir Configuraciones. • Dar click en crear configuración. • Rellenar los campos correspondientes.	
Resultados esperados: Crear configuración exitosamente.	

Tabla D.4. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: P7_HU7	Historia de usuario: 7
Nombre: Añadir Conexión	
Descripción: Se debe probar añadir conexión.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir Gestión de Red. • Llenar los campos necesarios.	
Resultados esperados: Añadir conexión exitosamente.	

Tabla D.5. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: P8_HU8	Historia de usuario: 8
Nombre: Seleccionar entrada de video	
Descripción: Se debe probar seleccionar entrada de video.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Seleccionar en entrada.	
Resultados esperados: Entrada cambiada exitosamente.	

Tabla D.6. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: P9_HU9	Historia de usuario: 9
Nombre: Visualizar en pantalla completa	
Descripción: Se debe probar el visualizar TV en pantalla completa.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: ● Dar click en pantalla completa.	
Resultados esperados: Mostrar TV en pantalla completa exitoso.	

Tabla D.7. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: P10_HU10	Historia de usuario: 10
Nombre: Controlar el encendido	
Descripción: Se debe probar el encender televisores.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: ● Dar click en Encender Todos.	
Resultados esperados: Encender televisores exitoso.	

Tabla D.8. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: P11_HU11	Historia de usuario: 11
Nombre: Controlar el apagado	
Descripción: Se debe probar el apagado de TV.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: ● Dar click en Apagar Todos.	
Resultados esperados: Apagar todos los TV exitosamente.	

Tabla D.9. Prueba de aceptación # 12

Caso de prueba de aceptación	
Código: P12_HU12	Historia de usuario: 12
Nombre: Controlar el restablecimiento	
Descripción: Se debe probar controlar el restablecimiento.	

Continúa en la próxima página

Tabla D.9. Continuación de la página anterior

Condiciones de ejecución: El usuario debe estar registrado en el sistema.
Pasos de ejecución: • Dar click en Restablecer.
Resultados esperados: Restablecimiento de las configuraciones exitoso.

Tabla D.10. Prueba de aceptación # 13

Caso de prueba de aceptación	
Código: P13_HU13	Historia de usuario: 13
Nombre: Añadir televisor	
Descripción: Se debe probar añadir televisor.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir TV. • Añadir televisor.	
Resultados esperados: Añadir televisor exitoso.	

Tabla D.11. Prueba de aceptación # 14

Caso de prueba de aceptación	
Código: P14_HU14	Historia de usuario: 14
Nombre: Listar televisores	
Descripción: Se debe probar listar televisores.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir TV.	
Resultados esperados: Listar televisores exitosa.	

Tabla D.12. Prueba de aceptación # 15

Caso de prueba de aceptación	
Código: P15_HU15	Historia de usuario: 15
Nombre: Eliminar televisores	
Descripción: Se debe probar eliminar televisores.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	

Continúa en la próxima página

Tabla D.12. Continuación de la página anterior

Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir TV. • Dar click en eliminar.
Resultados esperados: Eliminar televisores exitosamente.

Tabla D.13. Prueba de aceptación # 16

Caso de prueba de aceptación	
Código: P16_HU16	Historia de usuario: 16
Nombre: Añadir Usuario	
Descripción: Se debe probar añadir usuario.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir gestionar usuario. • Rellenar los campos y dar click en añadir usuario.	
Resultados esperados: Añadir usuario exitosamente.	

Tabla D.14. Prueba de aceptación # 17

Caso de prueba de aceptación	
Código: P17_HU17	Historia de usuario: 17
Nombre: Listar usuarios	
Descripción: Se debe probar el listar usuarios.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: • Abrir el menú dando click en la página principal. • Abrir gestión de usuarios.	
Resultados esperados: Se muestra la lista de usuarios correctamente.	

Tabla D.15. Prueba de aceptación # 18

Caso de prueba de aceptación	
Código: P18_HU18	Historia de usuario: 18
Nombre: Editar usuario	
Descripción: Se debe editar usuario.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	

Continúa en la próxima página

Tabla D.15. Continuación de la página anterior

Pasos de ejecución: ● Abrir el menú dando click en la página principal. ● Abrir gestión de usuarios. ● Dar click en editar usuario en la lista de usuarios.
Resultados esperados: Editar con éxito el usuario.

Tabla D.16. Prueba de aceptación # 19

Caso de prueba de aceptación	
Código: P19_HU19	Historia de usuario: 19
Nombre: Eliminar usuario	
Descripción: Se debe probar el eliminar usuario.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: ● Abrir el menú dando click en la página principal. ● Abrir gestión de usuarios. ● Dar click en eliminar en la lista de usuarios.	
Resultados esperados: Eliminación exitosa.	

Tabla D.17. Prueba de aceptación # 20

Caso de prueba de aceptación	
Código: P20_HU20	Historia de usuario: 20
Nombre: Cerrar sesión	
Descripción: Se debe probar el cierre de sesión.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Pasos de ejecución: ● Abrir el menú dando click en la página principal. ● Dar click en logout.	
Resultados esperados: Cerrar sesión exitoso.	