RESUMEN TRABAJO DE GRADO

AUTOR(ES):

NOMBRE(S): JUAN ANDRES APELLIDOS: RODRIGUEZ ARENAS

NOMBRE(S): APELLIDOS:

FACULTAD: INGENIERIA

PLAN DE ESTUDIOS: INGENIERÍA ELECTRÓNICA

DIRECTOR:

NOMBRE(S): KARLA CECILIA APELLIDOS: PUERTO LÓPEZ

CO-DIRECTOR:

NOMBRE(S): JUAN SEBASTIÁN APELLIDOS: GALINDO LIZCANO

TÍTULO DEL TRABAJO (TESIS): SISTEMA DE MONITOREO EN MÁQUINAS VIRTUALES HYPER-V PARA LA EMPRESA PATIÑO Y CONTRERAS CIA SAS

RESUMEN

Este proyecto se basó en desarrollo del sistema de monitoreo Hyper-V para la empresa Patiño y Contreras CIA SAS. Para ello, se implementó una investigación tipo descriptiva, donde la información se obtuvo mediante la aplicación de sistemas de monitoreo y tecnologías móviles de la Web. La población y muestra correspondió al procesador, discos de almacenamiento y procesadores de la empresa Patiño y Contreras CIA SAS. Se logró desarrollar un agente recopilatorio y una API para procesar las variables captadas del servidor Hyper-V. Posteriormente, se elaboró una aplicación Web administrativa para obtener el historial de las máquinas y recibir notificaciones de alerta. Finalmente, se evaluó el desempeño del sistema de monitoreo.

PALABRAS CLAVE: sistema de monitoreo, Hyper-V, Web administrativa, aplicación móvil Android.

CARACTERÍSTICAS:

PÁGINAS: 101 PLANOS: ____ ILUSTRACIONES: ____ CD ROOM: 1

Copia No Controlada

SISTEMA DE MONITOREO EN MÁQUINAS VIRTUALES HYPER-V PARA LA EMPRESA PATIÑO Y CONTRERAS CIA SAS

JUAN ANDRES RODRIGUEZ ARENAS

UNIVERSIDAD FRANCISCO DE PAULA SANTANDER

FACULTAD DE INGENIERÍA

PLAN DE ESTUDIOS DE INGENIERÍA ELECTRÓNICA

SAN JOSÉ DE CÚCUTA

2022

SISTEMA DE MONITOREO EN MÁQUINAS VIRTUALES HYPER-V PARA LA EMPRESA PATIÑO Y CONTRERAS CIA SAS

JUAN ANDRES RODRIGUEZ ARENAS

Trabajo de grado presentado como requisito para optar al título de:

Ingeniero Electrónico

Director:

KARLA CECILIA PUERTO LÓPEZ

Ingeniera Electrónica

Codirector:

JUAN SEBASTIÁN GALINDO LIZCANO

Ingeniero de sistemas

UNIVERSIDAD FRANCISCO DE PAULA SANTANDER

FACULTAD DE INGENIERÍA

PLAN DE ESTUDIOS DE INGENIERÍA ELECTRÓNICA

SAN JOSÉ DE CÚCUTA

2022

Universidad Francisco de Paula Santander
Vigilada Mineducación
NIT. 890500622 - 6

ACTA DE SUSTENTACIÓN DE UN TRABAJO DE GRADO

Fecha:	CÚCUTA, 16 DE NOVIEMBRE DE 2022
Hora:	16:00
Lugar:	SALON SD 302
Plan de Estudios:	INGENIERÍA ELECTRÓNICA

Título de la Tesis: "SISTEMA DE MONITOREO EN MÁQUINAS VIRTUALES HYPER-V PARA LA EMPRESA PATIÑO Y CONTRERAS CIA SAS"

Jurados:	IE MSc. EDWIN JOSÉ VERA ROZO IE MSc. DARWIN CARDOZO SARMIENTO
Director:	IE. MSc. KARLA CECILIA PUERTO LOPEZ
Codirector:	IE. JUAN SEBASTIÁN GALINDO LIZCANO

Nombre del Estudiante:	**Código:**	**Calificación:**	
		Número	Letra
JUAN ANDRES RODRIGUEZ ARENAS	1160961	4,0	Cuatro, cero

APROBADA

EDWIN JOSÉ VERA ROZO

DARWIN CARDOZO SARMIENTO

SERGIO SEPÚLVEDA MORA
Coordinador Comité Curricular
Ingeniería Electrónica

Contenido

Lista de Figuras

Lista de Tablas

pág.

Resumen

Este proyecto se basó en el sistema de monitoreo en máquinas virtuales Hyper-V para la empresa Patiño y Contreras CIA SAS. Para ello, se implementó una investigación tipo descriptiva, ya que se basó en el análisis de los sistemas de monitoreo. La información se obtuvo mediante la aplicación de sistemas de monitoreo y tecnologías móviles de la Web. La población y muestra correspondió al procesador, discos de almacenamiento y procesadores de la empresa Patiño y Contreras CIA SAS. Se logró diseñar un sistema de monitoreo de máquinas virtuales Hyper-V para la empresa Patiño y Contreras CIA SAS. Seguidamente, se desarrolló un agente recopilatorio y una API para procesar las variables captadas del servidor Hyper-V. Posteriormente, se elaboró una aplicación Web administrativa para obtener el historial de las máquinas y recibir notificaciones de alerta. Finalmente, se evaluó el desempeño del sistema de monitoreo.

Introducción

El programa Microsoft Hyper-V hace parte del software hipervisores, el cual se utiliza para la creación e incorporación del equipamiento virtual en los canales de red. El Hyper-V, básicamente está conformado por un host Hyper-V instalado en un dispositivo tipo Windows. Los equipos virtuales Hyper es uno de los tantos recursos de red que requieren de un constante monitoreo con el fin de asegurar una duración de inactividad mínima y el adecuado rendimiento. Cabe resaltar que existen distintas herramientas gratuitas de monitoreo de Hyper-V donde en algunas ocasiones los proveedores de mayor importancia ofrecen un soporte para adherir el sistema monitoreo de Hyper-V, como herramienta de supervisión (Manage Engine OpManager, 2022).

Esta pasantía se basa en la implementación de un sistema de monitoreo en máquinas virtuales Hyper-V para la empresa Patiño y Contreras CIA SAS, esto con el fin de garantizar el procesamiento de datos y la calidad de sus servicios a los clientes. Cabe resaltar que el desarrollo de esta pasantia lleva a la práctica nuevos conocimientos, que fortalecen procesos desarrollando este sistema de monitoreo.

En base a lo anterior se presenta el desarrollo de esta investigación la cual se encuentra estructurada de la siguiente manera: en el primer capítulo se describe el planteamiento, la justificación, el alcance, las limitaciones y delimitaciones y los objetivos del problema. En el segundo apartado se analizan los antecedentes, el marco teórico y el marco legal. En el tercer itenm se establece el diseño metodológico en el cual se plantea el tipo de investigación, la metodología, las técnicas de recolección de información. En el capítulo cuatro se evidencian los resultados de los objetivos ejecutados durante el proceso. Finalmente, en el capítulo 5 se hayan las conclusiones del presente trabajo.

1. Problema

1.1 Título

SISTEMA DE MONITOREO EN MÁQUINAS VIRTUALES HYPER-V PARA LA EMPRESA PATIÑO Y CONTRERAS CIA SAS.

1.2 Planteamiento del Problema

En el presente se vive en un mundo virtual donde en su mayoría los procesos pasaron a ser registros de base de datos y programas de administración, cualquier negocio que dependa de las tecnologías para administrarse debe tener alta disponibilidad en los servicios, ya que estar un tiempo prolongado sin servicios conlleva a pérdidas económicas e incluso legales por pérdida de información.

Soluciones como virtualizar las bases de datos o aplicaciones mediante sistemas como Hyper-V forman parte de las medidas para evitar estos riesgos (Hernández, 2017), puesto que permite tener una copia de la configuración e información del sistema administrativo que corre el negocio, pero estas copias por si solas no garantizan una alta disponibilidad (Costas Santos, 2014), ya que, si ocurre una falla alguien debe poner en marcha un plan de recuperación.

Los sistemas de virtualización permiten tener replicación de las copias de las máquinas en servidores con las mismas o más características tales como la memoria RAM, el procesador, discos de almacenamiento y procesadores; en el caso de una falla estos sistemas automáticamente lanzan un encendido al servidor de respaldo el cual contiene las copias de la replicación, levantando las máquinas virtuales reduciendo el tiempo de desconexión de las bases de datos o aplicaciones administrativas mejorando la alta disponibilidad.

A partir de la replicación de máquinas virtuales Patiño y Contreras SAS se plantea la siguiente pregunta en los servicios ofrecidos a sus clientes: ¿Cómo desarrollar un sistema de supervisión de monitoreo de la replicación en máquinas virtuales para garantizar que el proceso se esté llevando a cabo correctamente, garantizando la alta disponibilidad en el caso de alguna falla o desastre informático? (Anchuelo, 2021).

1.3 Justificación

A razón de algunos reportes a clientes por parte del equipo de soporte técnico en las revisiones periódicas, la empresa Patiño y Contreras CIA SAS determinó que, en varios servicios instalados, la replicación tenía problemas o anomalías con un tiempo prolongado lo cual no garantiza la alta disponibilidad a sus clientes (Jayaseelan & Charles, 2014).

La empresa Patiño y Contreras CIA SAS con el fin de garantizar la calidad de sus servicios y evitar problemas de tipo legal por pérdida de información e incumplimiento de contrato, decide llevar a cabo la realización de un mejoramiento en la seguridad de los servidores de sus clientes mediante la solución del sistema de monitoreo de máquinas virtuales, con lo cual se deseaban obtener los siguientes beneficios:

1.3.1 Impacto esperado. El diseño de un sistema de monitoreo de la virtualización en Hyper-V de la empresa Patiño y Contreras CIA SAS, es una herramienta que ofrece seguridad y calidad en sus servicios prestados, con el propósito de atraer clientes y mejorar la confianza de los que actualmente usan los servicios de la empresa.

Por esta razón, el sistema ofrece una solución en seguridad y además de rápida acción ante problemas que afecten la integridad de los negocios reduciendo el daño ocasionado por

problemas informáticos de manera eficiente.

1.3.2 Beneficios tecnológicos. Por medio de este proyecto se tiene una herramienta que permite tener seguridad en el estado de los servicios de replicación configurados por la empresa. Esto es posible por medio de un sistema que cuenta con aplicación Web y móvil para el monitoreo de las máquinas virtuales que interactúan con un servicio en la nube, el cual almacena los registros de los monitores de los servidores y alertará de anomalías mediante SMS, llamada o correo electrónico.

1.3.3 Beneficios institucionales y empresariales. Este proyecto es un trabajo en conjunto entre un pasante adscrito a la Universidad Francisco de Paula Santander y la empresa Patiño y Contreras CIA SAS, con el cual se obtuvieron evidencias del desarrollo productivo de la región con miras a la política de proceso de la acreditación del programa de Ingeniería Electrónica.

Patiño y Contreras CIA SAS tienen el beneficio de la adquisición del sistema de monitoreo desarrollada para ser utilizada en sus clientes, además de brindar la oportunidad a estudiantes pasantes de la Universidad Francisco de Paula Santander encaminado a fortalecer lazos de cooperación institucionales.

1.3.4 Beneficios científicos. Se fomentó en el estudiante el pensamiento investigativo en las múltiples áreas del saber que se ejecutaron en este proyecto, en especial el desarrollo de soluciones de sistemas de información de monitoreos y alertas.

Este proyecto permitió continuar la mejora de los procesos en seguridad y alta disponibilidad de negocios que ofrecen soluciones de sistemas informáticos para el manejo y control de procesos.

1.4 Alcance

Este sistema es desplegado en los servidores de máquinas virtuales Hyper-V usado por los clientes en modo de servicio agente de monitoreo, además sus respectivas aplicaciones tales como el cliente Web y el API, instaladas en un VPS con sistema operativo distribución Linux Ubuntu Server 20.04.3 LTS configurado para su correcto funcionamiento teniendo en cuenta que solo será compatible con navegadores Chrome como mínimo en su versión 96.0.4664.45, Microsoft Edge 96.0.1054.43 o Mozilla Firefox 94.0.2 y Android como requisito mínimo 10.0.

1.4.1 Tipo de proyecto. El proyecto se clasifica como pasantía, ya que este se dirige en la mejora y desarrollo de nuevos conocimientos, que fortalecen procesos ya existentes por otros autores en el desarrollo de sistemas de monitoreo con tecnologías móviles, Web y servicios de recopilación de información.

Además de integrar muchos de los conocimientos adquiridos en la formación académica como la programación de software, con el fin de dar solución al problema de presentado por la empresa Patiño y Contreras CIA SAS.

1.4.2 Resultados esperados. Se obtuvo el cumplimiento de los objetivos, los resultados directos e indirectos después de culminar el desarrollo son las siguientes:

1.4.2.1 Resultados directos. Patiño y Contreras CIA SAS al tener el sistema de monitoreo de la virtualización y replicación de máquinas en Hyper-V, será beneficiado con la seguridad de que sus clientes cuentan con el respaldo de un soporte que va a estar atento de alguna falla o imprevisto en la copia de sus máquinas que tienen la lógica e información de la empresa, con esto pueden garantizar la alta disponibilidad con la cual ellos ofrecen sus servicios ya que contaran

con un sistema que respaldara sus operaciones mediante notificaciones.

1.4.2.2 Resultados indirectos. Con el desarrollo del sistema de monitoreo de la virtualización y replicación de máquinas en Hyper-V, la empresa Patiño y Contreras CIA SAS fue capacitada en el uso de la interfaz gráfica del monitor y tendrá experiencia en el desarrollo de sistemas de información.

1.5 Limitaciones y Delimitaciones

El sistema de monitoreo y alertas presenta ciertas limitaciones y delimitaciones propias de este tipo de proyecto y son enunciadas a continuación.

1.5.1 Limitaciones. Compatibilidad con múltiples navegadores de internet, el aplicativo Web debe funcionar con la mayoría de navegadores modernos como Chrome, Firefox, Edge y Safari, los cuales cuentan con diferentes motores en ejecución JavaScript e interpretación de hojas de estilo CSS (MDN Contributors, 11).

Compatibilidad con los diferentes dispositivos Android, ya varían los tamaños de pantalla y versiones del sistema operativo (Google Developers, 2020).

Adquisición de licencias correspondientes del software o librerías a emplear, es por este motivo que se hace necesario la búsqueda de opciones de código abierto.

Adquisición y configuración de un VPS donde se ejecutará la aplicación API REST en internet, para hacer pruebas de los servicios como el sistema de notificaciones.

Contratar servicios de mensajería de texto y llamadas para el sistema de alertas.

1.5.2 Delimitaciones. El diseño del sistema de monitoreo y alertas está limitado a la recepción, registro y notificación de los estados de las máquinas virtuales enviados por el agente instalado en el sistema operativo Hyper-V.

El sistema de alertas solamente notifica en el caso de no hacerse la replicación de las máquinas virtuales.

Las delimitaciones globales del proyecto son las siguientes.

1.5.2.1 Delimitación espacial. Este proyecto se ejecutó de manera semipresencial por medio de herramientas TIC para el desarrollo del software usando simulaciones de las máquinas virtuales y en las instalaciones de la empresa Patiño y Contreras ubicadas en la ciudad de Cúcuta para pruebas y orientaciones del proyecto.

1.5.2.2 Delimitación temporal. El desarrollo del presente proyecto comenzó con la aprobación por parte de la empresa y por parte del comité curricular de Ingeniería Electrónica, y tuvo una duración según lo establecido por la UFPS y el contrato contraído en Patiño y Contreras CIA SAS.

1.6 Objetivos

1.6.1 Objetivo general. Diseñar un sistema de monitoreo de máquinas virtuales Hyper-V para la empresa Patiño y Contreras CIA SAS.

1.6.2 Objetivos específicos. A continuación, se presentan los objetivos específicos:

Desarrollar un agente recopilatorio y una API para procesar las variables captadas del servidor Hyper-V.

Elaborar una aplicación Web administrativa.

Crear una aplicación móvil Android para obtener el historial de las máquinas y recibir notificaciones de alerta.

Evaluar el desempeño del sistema de monitoreo.

2. Marco Referencial

En este proyecto se encuentran áreas y principios de estudio el cual son de importancia en los procesos administrativos: sistemas de comunicaciones, arquitectura e infraestructura de equipos de cómputo y control.

2.1 Antecedentes

Título: Performance monitoring of operating systems in Hyper-V environment.

Autor: Katarzyna Paula Piórkowska (FEIT/ICS).

País: Polonia.

Institución: The Institute of Computer Science (FEIT/ICS).

Resumen: En este proyecto cuyo objetivo es el estudio del rendimiento del entorno Hyper-V implementando un agente de monitoreo y un software de arquitectura cliente-servidor el cual almacena los datos en un histórico, para proporcionar información sobre el rendimiento del sistema. La lectura es calculada si esta no detecta alguna anomalía en los datos que recopila, de lo contrario si detecta alguna falla en el sistema se realizara una lectura detallada en las muestras que se envían a la base de datos. Este trabajo aporta experiencia e información sobre los tipos de datos que se pueden obtener mediante el agente y además de tener como referente el manejo de los datos que el sistema va a almacenar en el histórico (The Institute of Computer Science, 2020).

Título: Implementación de una herramienta de monitoreo sobre la emulación de bienes de infraestructura tecnológica en una red LAN.

Autor: Garzón Vásquez, Christian Camilo.

País: Colombia.

Institución: Universidad de Antioquia.

Resumen: El monitoreo es un trabajo de alta necesidad sin importar el enfoque en que se aplique, pero su eficacia depende de la herramienta que se usa y de la capacitación de personal que la maneja para determinar el rendimiento. Este proyecto emula una tienda o sede del Grupo Éxito, el cual, por alta disponibilidad en el servicio y fallas eléctricas, se diseña una herramienta de monitoreo para aprovechar las capacidades analíticas y de entrega. La elección de las herramientas se basó en las necesidades de la empresa y en las experiencias en las aplicaciones, software con su respectivo licenciamiento, etc. Al integrar el sistema de monitoreo con los datos recolectados se pudieron organizar y visualizar para un mejor análisis de la red emulada, lo cual aporta al proyecto la base para la visualización de los datos recolectados (Garzón, 2021).

Título: Advanced Hyper-V Replica Monitoring via #PowerShell #HyperV.

Autor: Charbel Nemnom.

País: Estados Unidos.

Institución: Charbel Nemnom's Cloud & CyberSecurity Blog.

Resumen: Se has usado Hyper-V replica conoces el visualizador de estados de la réplica en la consola Hyper-V Manager, en esta interfaz el usuario puede visualizar el estado de la réplica con las alertas: Normal, Advertencia o Crítico.

Uno de sus problemas es que el estado de la replicación este en un estado crítico durante un largo periodo de tiempo ya que el disco de diferenciación crecerá y esto puede ocasionar un gran problema, debido a que es necesario saberlo lo más pronto posible es necesario una notificación del fallo (Nemnom, 2022).

Por esta razón se ha creado un ingenioso script Powershell que genera un correo electrónico con un elegante HTML con detalles del problema, Este script se usa en conjunto a el programador de tareas para programar su frecuencia de ejecución dependiendo los fines de supervisión necesaria 9.

2.2 Marco Teórico

A continuación, se presentan las definiciones y conceptos de los elementos que conforman el proyecto.

2.2.1 Máquina virtual. Una máquina virtual es una recreación de todo el sistema PC en un sistema operativo instalado en un hardware físico o PC como en la Figura 1, al cual se le pueden asignar recursos del PC anfitrión como la RAM, Procesador, o algunos periféricos como tarjetas de red, mouses, teclados, entre otros. Además, en algunos casos se recrean los componentes como el disco duro con ficheros en los discos reales del anfitrión 10.

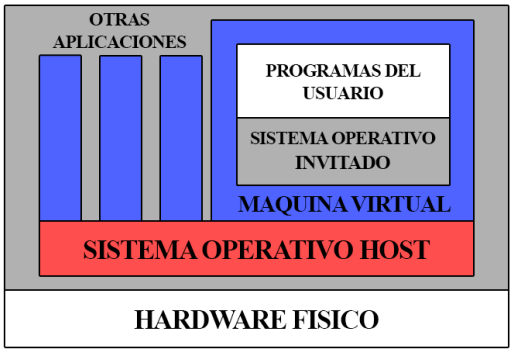


Figura 1. Máquina virtual en un sistema operativo anfitrión

Fuente: Microsoft Corporation (2018).

2.2.2 Hyper-V. Es un producto de Microsoft basado en hipervisor (sistema para crear y ejecutar máquinas virtuales) en una capa de software, en la cual se permite la ejecución de múltiples sistemas operativos en un solo hardware al mismo tiempo, con el fin de aislar la lógica donde se ejecutan los sistemas operativos para múltiples propósitos como ahorrar recursos o tener seguridad por medio de copias de las máquinas virtuales o la replicación de ellas a otro anfitrión con características similares figura 2 (Microsoft Corporation, 2018).

Figura 2. Logo Hyper-V

Fuente: Microsoft Corporation (2018).

2.2.3 Replicación máquinas virtuales. Un sistema de información instanciado de una máquina virtual debe estar en ejecución la mayoría del tiempo, teniendo una inactividad nula y poca intervención de alguna persona por lo cual requiere unos métodos de restauración y de recuperación. La replicación de una máquina virtual es el almacenamiento de los datos que la

conforman en uno o más sitios de manera simultánea, para otorgar a los sistemas condiciones de alta disponibilidad y seguir trabajando sin interrupciones (Medina, 2013).

2.2.4 Powershell. Es un Shell de comandos multiplataforma Windows, Linux y macOS para la automatización de tareas, el cual maneja un sistema de comandos de lenguaje de scripts y un asistente para configuración del equipo. Dentro del ecosistema de Microsoft que permite manipular muchas de sus herramientas como Hyper-V, figura 3 (Microsoft Corporation, 2021).

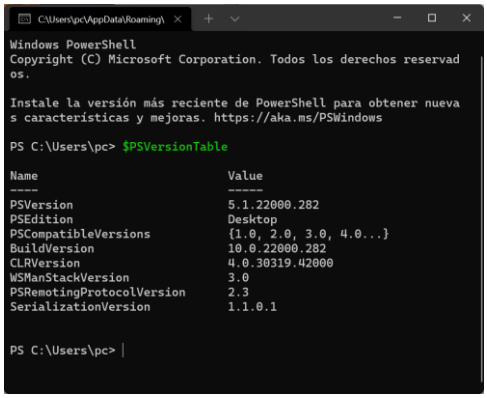


Figura 3. Interfaz PowerShell 5.1.22000.282

Fuente: Microsoft Corporation (2021).

2.2.5 REST API. Es una interfaz de programación de aplicaciones que usa la arquitectura REST, para la relación entre servicios atreves de la red HTTP, por medio de un conjunto de operaciones para sus llamados como:

- GET: Para pedir registros.
- POST: Para enviar múltiples datos en un paquete y pedir registros.
- PUT: Para editar registros.

- DELETE: Para eliminar registros.

- Una API es un servicio que permite obtener datos y ejecutar funciones entre un equipo de cómputo o algún sistema (Red Hat, 2020).

2.2.6 Aplicación Web. Es un tipo de página Web accedida mediante un navegador por medio de internet o intranet, con contenidos dinámicos provenientes de alguna base de datos o de alguna función calculada desde una aplicación remota o local, mediante servicios en el DOM de los navegadores o por la ejecución de JavaScript (Luján, 2002).

2.2.7 Android. Es una plataforma de desarrollo que cuenta con sistemas operativos para dispositivos móviles como teléfonos inteligentes o tables, televisores o tv box Android TV, relojes inteligentes Wear OS y automóviles Android Auto. Usa como base el Kernel Linux y fue desarrollado por Google bajo la licencia de código abierto dotándolo de gran popularidad y aceptación llegando a ocupar el 74% del mercado de los dispositivos móviles, lo cual facilita su uso en el desarrollo de aplicaciones, figura 4 (Google, 2021).

Figura 4. Logo Android

Fuente: Google (2021).

2.2.8 CMS. Un sistema de gestión de contenidos es una librería o programa encargado de crear espacios de trabajos en elaboración o administración de contenidos, por lo general cuentan con una interfaz de desarrollo o administración para manejar el contenido o diseño de su

contenido (Wikipedia, 2021).

2.2.9 Flutter. Es un marco de desarrollo código abierto creado por Google para la creación de aplicaciones en Dart, figura 5, compiladas de manera nativa con un solo código base multiplataforma Android e IOS (Google, 2021).

Figura 5. Logo Flutter

Fuente: Google (2021).

2.2.10 Dart. Dart es un lenguaje de programación desarrollado por Google, figura 6 pensado como una alternativa moderna para aplicaciones móviles rápidas y multiplataforma basado en JavaScript, C#, Java y CoffeScript (Google, 2021).

Figura 6. Logo Dart

Fuente: Google (2021).

2.2.11 VPS (Virtual Private Server). Es una solución conformada por una máquina virtual con un sistema operativo y una dirección IP dedicada, teniendo todas las prestaciones de un servidor dedicado configurado según los requerimientos de memoria RAM, espacio de almacenamiento y procesamiento (Hostingred, 2021).

2.2.12 NodeJS. Un entorno de ejecución JavaScript multiplataforma basado en el motor V8 de Google, figura 7, producido para la creación de aplicaciones de red escalables, el cual cuenta con gran popularidad entre las comunidades de desarrollo de software y un gran número librerías (OpenJS Foundation, 2021).

Figura 7. Logo NodeJS

Fuente: OpenJS Foundation (2021).

2.2.13 RBAC. El control de acceso basado en roles (role based access control) es un método de seguridad que se basa en la delegación de las funciones y autoridad en un sistema informático como se puede observar en la figura 8. En este modelo las funciones o autorizaciones se basan en roles y permisos en usuarios o grupos los cuales se pueden configurar mediante un módulo administrativo (Ramirez, 2021).

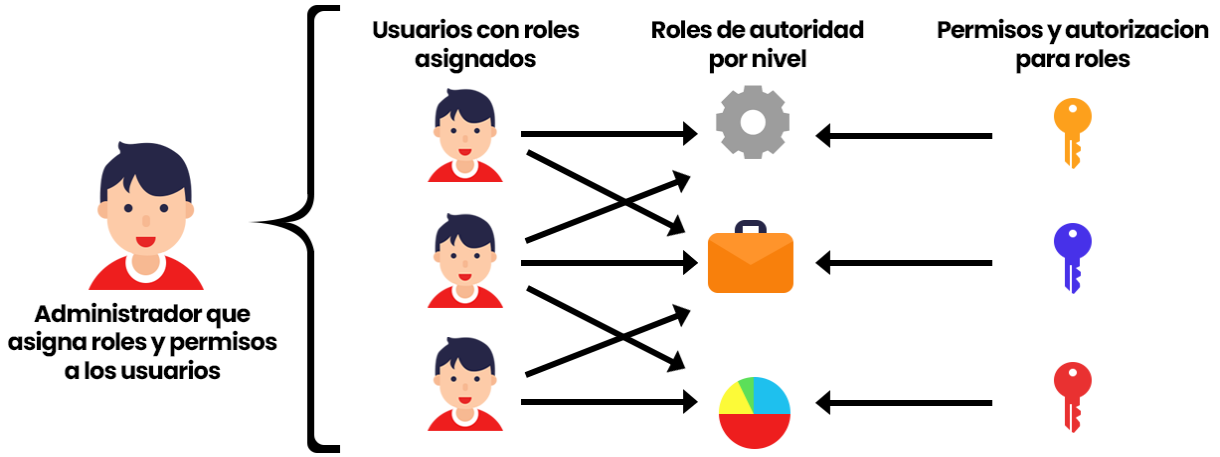


Figura 8. Diagrama RBAC

2.2.14 Strapi. Es un CMS (sistema de gestión de contenidos) código abierto de NodeJS para el desarrollo de aplicaciones API de manera y sencilla, figura 9, con sistema de autenticación RBAC (control de acceso basado en roles) y gestión de contenidos multimedia (Strapi, 2021).

Figura 9. Logo Strapi

Fuente: Strapi (2021).

2.3 Marco Legal

A continuación, se presenta como estructura legal de la investigación y desarrollo del proyecto bajo la normatividad de la constitución política de Colombia dentro de los fundamentos, leyes y decretos para el buen desempeño del ministerio de la ciencia, tecnología e innovación.

2.3.1 Normatividad vigente para comerciantes y el uso del comercio de la tecnología. En la ley 16 del año 1968, el cual expide en las facultades extraordinarias que confiere en el numeral 15, decreta (Sistema Unico de Informacion Normativa, 1968):

- Artículo 1: En objeto a la ley, la aplicabilidad del comercio y asuntos mercantiles para el buen uso del ejercicio bajo las disposiciones sujetas a la norma, regulados y decididos por la norma vigente.
- Artículo 3: Manifiesta en la autoridad y costumbre mercantil, la cual tendrá la misma autoridad en la ley, la cual concede, siempre que cumpla y no contrario la norma para el mecanismo comercial la prestación de servicios públicos, uniformes y reiterados en el lugar donde hayan de cumplirse las debidas prestaciones y requisitos establecidos.
- Artículo 5: Aplicación del comercio y costumbre mercantil, para la prestación de servicios, con el fin de determinar técnicas de comercio y para interpretar los actos y convenios mercantiles.

2.3.2 Decretos vigentes aplicables a entidades del sector de las TIC. Según la ley 1266 de 2008, para el tratamiento de datos y el comercio electrónico, por la cual se actualiza el sector de las tecnologías e información y las comunicaciones – TIC, se determinan competencias según la regulación establecida en la norma, decreta (Congreso de Colombia, 2008):

- Artículo 1: Se tiene como objeto alinear los estímulos de los agentes y autoridades encargadas de las tecnologías de la información y las comunicaciones – TIC, con el fin aumentar, simplificar y modernizar el marco constitucional, así, velar y mejorar la prestación de servicios del sector privado en el desarrollo de proyectos innovadores asociados.

- Artículo 2: Cumplimiento de la normatividad, el estado y todos los agentes del sector y las tecnologías y las comunicaciones deberán colaborar dentro del marco legal vigente, para priorizar el acceso y el uso a las tecnologías de la información y las comunicaciones en pro de los bienes y servicios, en condiciones no discriminatorias en la conectividad, educación y tratamiento de datos.

2.3.3 Pilares normativos e institucionales según el ministerio de Ciencia, Tecnología e innovación. En la ley 1951 de 2019, en consideración de las agencias de información comercial, para el tratamiento de datos de usuarios, según sea el caso, en objeto a la ley y con el fin de crear el ministerio de Ciencia, Tecnología e innovación, se aplicarán lo siguiente (Congreso de Colombia, 2019):

- Artículo 2: Objetivos y consideraciones, por medio de la cual se reconocen y actualizan los derechos y deberes del estado en materia del desarrollo científico, tecnológico e innovador, la cual busca formular la política pública en función de la tecnología e información, establecer estrategias en pro de las agencias del sector de la ciencia, tecnología e innovación, impulsar y fomentar el desarrollo, garantizar y velar por las condiciones necesarias para el desarrollo de proyectos innovadores que favorezcan la productividad y la competitividad.
- Artículo 3: La creación del ministerio de ciencia, tecnología e innovación, como organismo para la gestión de la adecuada administración pública encargada de formular, orientar, dirigir, ejecutar, implementar y controlar la política de estado en materia, en la participación de la comunidad científica y de la tecnología, teniendo en cuenta la planeación de programas de desarrollo, de acuerdo a la ley y su respectiva normativa.

3. Diseño Metodológico

En la elaboración de este proyecto se presentaron los siguientes procesos para el cumplimiento de los objetivos específicos planteados.

3.1 Objetivo 1. Desarrollar un Agente Recopilatorio y una API para Procesar las Variables Captadas del Servidor Hyper-V

- Identificar comandos PowerShell para la recopilación de datos.

Metodología: Identificar las variables del sistema para establecer parámetros y características principales de los comandos necesarios que se pueden obtener mediante la documentación de Hyper-V, además de establecer las condiciones necesarias el testeo de estos.

Se analizará el intervalo de tiempo de recolección de datos para establecer el periodo de toma de muestras.

- Diseño agente recopilación en .Net.

Metodología: Desarrollar una aplicación de servicio mediante Visual Studio 2021 y el marco de desarrollo .NET en C# que consuma las funciones de recopilación PowerShell en el intervalo previamente definido.

Desarrollar de una interfaz para la administración del servicio del agente monitor y probar el funcionamiento del agente.

- Diseño de la base de datos.

Metodología: Modelar la base de datos con las variables adquiridas por el agente monitor.

- Desarrollar un API REST con Strapi.

Metodología: Por medio del marco de desarrollo Strapi en NodeJS crear la aplicación API REST del sistema de monitoreo configurándola con la base de datos MariaDB.

Configurar los permisos de Strapi para acceder a los métodos HTTP los cuales permiten crear, buscar, actualizar, eliminar y listar los registros almacenados.

- Integración servicios SMS y llamadas.

Metodología: Compra e integracion los servicios de Plivo para envío de mensajes SMS (Plivo, 2021) y llamadas (Plivo, 2021) de acuerdo con las necesidades de la aplicación.

- Despliegue del API.

Metodología: Comprar y configurar el servidor VPS con las aplicaciones o librerías necesarias para la ejecución del API REST en relación con el costo beneficio teniendo en cuenta la memoria RAM, procesador, espacio en disco y transferencia de datos.

Configurar el servidor de dominio suministrados por Patiño y Contreras CIA SAS para exponer a internet el VPS.

Migrar la base de datos y crear un servidor de dominios NGINX para instalar los certificados SSL de la aplicación HTTPS y poder tener seguridad en la aplicación.

Ejecutar en segundo plano la aplicación Strapi con la librería de NodeJS PM2 para tener funcionando la aplicación API REST.

Configurar el servidor de dominios con la API REST y verificar el funcionamiento con el cliente HTTP Postman (Postman, Inc., 2021).

3.2 Objetivo 2. Elaborar una Aplicación Web Administrativa

- Programación aplicación Web NUXT.

Metodología: Crear una aplicación para el cliente Web con el marco de desarrollo NUXT e instalar librerías y marcos de desarrollo HTML necesarios para crear la aplicación como Bootstrap 5.0.2 y Moment.JS 2.29.

- Diseño del modelo VUEX.

Metodología: Crear el Store de VUEX (base de datos del lado cliente que almacena en el DOM del navegador) para guardar y obtener los registros a listar, configuraciones y demás datos necesarios para el funcionamiento del aplicativo Web, el Store además contiene las funciones para llamar al API conectándola al sistema de monitoreo.

- Diseño de las vistas.

Metodología: Diseñar y crear las páginas de la aplicación las cuales tienen el sistema de autenticación, listado de máquinas virtuales y detalles de las mismas mediante llamados al Store.

- Configuración VPS.

Metodología: Configurar en el servidor NGINX el dominio de la aplicación cliente y desplegar la aplicación NUXT.

3.3 Objetivo 3. Crear una Aplicación móvil Android para Obtener el Historial de las Máquinas y Recibir Notificaciones de Alerta

- Diseño de las vistas aplicación móvil.

Metodología: Crear la aplicación móvil Android con el marco de desarrollo Flutter 2.8 y Diseñar las vistas de la aplicación.

- Programación consumo API.

Metodología: Crear funciones para llamar al API conectándola al sistema de monitoreo por medio de peticiones HTTP.

Integrar las funciones con las vistas de la aplicación móvil y con ayuda de un simulador Android probar la aplicación.

Crear el paquete APK para verificar desde un dispositivo en físico mediante depuración ADB.

3.4 Objetivo 4. Evaluar el Desempeño del Sistema de Monitoreo

- Elaborar pruebas del sistema.

Metodología: Se debe crear un escenario de replicación con dos servidores Hyper-V en los cuales uno actuara como el servidor principal y el otro como el de replicación.

- Simular pruebas.

Metodología: Instalación del sistema de monitoreo y crear errores simulados para observar el funcionamiento del sistema.

- Socialización resultados del sistema con la empresa Patiño y contreras SAS.

Metodología: Con los datos obtenidos mediante la simulación se presentará la configuración y ejecución del sistema a la empresa Patiño y Contreras SAS.

4. Resultados

Al culminar el proyecto se cumplen de manera satisfactoria todos los objetivos específicos, donde se obtuvieron los siguientes resultados.

4.1 Desarrollar un Agente Recopilatorio y una API para Procesar las Variables Captadas del Servidor Hyper-V

Por medio de diversas fuentes a fines con el proyecto, se logró comprender qué tipo de tecnologías eran las necesarias para llevar a cabo dicha tarea. Donde la documentación de la empresa Microsoft sobre los comandos PoweShell para el uso del hipervisor Hyper-V, sirvió como rumbo de partida para conocer el alcance de los datos que podían recolectar de las replicaciones de las máquinas virtuales, además de dicha documentación a través de diversos sitios de la internet el proyecto se nutrió de experiencia y conocimientos, brindando la capacidad y confianza para su ejecución. Por otro lado, fue necesario probar e incluso crear aplicaciones mínimas que corroboraron que estos datos obtenidos funcionaran a cabalidad, además de haberse realizado una investigación de otras tecnologías a fines de nuestros objetivos pero estas necesitaban recursos de terceros como Hyper-V WMI Provider de Java o librerías de NodeJS como hyperv, donde se debían de instalar en los servidores las plataforma de desarrollo para que estas aplicaciones se pudieran ejecutar, se concluyó optar por la opción más viable de recolección de datos con PowerShell ya que esta derivaba del proveedor oficial de Hyper-V con .Net framework la cual está instalada en los servidores de manera predeterminada evitando usar recursos de terceros.

La empresa Patiño y Contreras S.A.S evaluó la propuesta con ayuda de su equipo técnico, donde su líder tecnológico y codirector del proyecto Juan Sebastián Galindo Lizcano aporto

dudas referentes a los recursos tecnológicos necesarios para realizar los objetivos planteados y su tiempo de ejecución. Para responder a estas dudas se hace entrega a la empresa de un anteproyecto con los objetivos específicos, cronogramas y metodología a aplicar.

4.1.1 Identificar comandos PowerShell para la recopilación de datos. Por medio de la consola PowerShell se ejecuta el comando Measure-VMReplication el cual trae datos útiles para nuestro sistema de monitoreo como:

- Health: Especifica el estado de replicación de las máquinas virtuales cuyas estadísticas de replicación desea obtener. Los valores válidos son "Critical", "Warning", "Normal" y "NotApplicable".
- State: Especifica el estado de replicación de las máquinas virtuales para las que desea obtener estadísticas de replicación. Los valores válidos son “Error”, “FailOverWaitingCompletion”, “FailedOver”, “NotApplicable”, “ReadyForInitialReplication”, “Replicating”, “Resynchronizing”, “ResynchronizeSuspended”, “Suspended”, “SyncedReplicationComplete”, “WaitingForInitialReplication” y “WaitingForStartResynchronize”.
- LReplTime: Fecha de la última replicación.
- AvgReplSize: Tamaño promedio de la replicación.

4.1.2 Diseñar agente recopilación en .Net. Para llevar a cabo el desarrollo del agente recopilatorio fue necesario que la empresa Patiño y Contreras suministrara una máquina virtual con Windows Server como se muestra en la figura 10, en la cual se tiene configurado Hyper-V en modo replicación para poder ejecutar los comandos PowerShell y tener los datos de prueba.

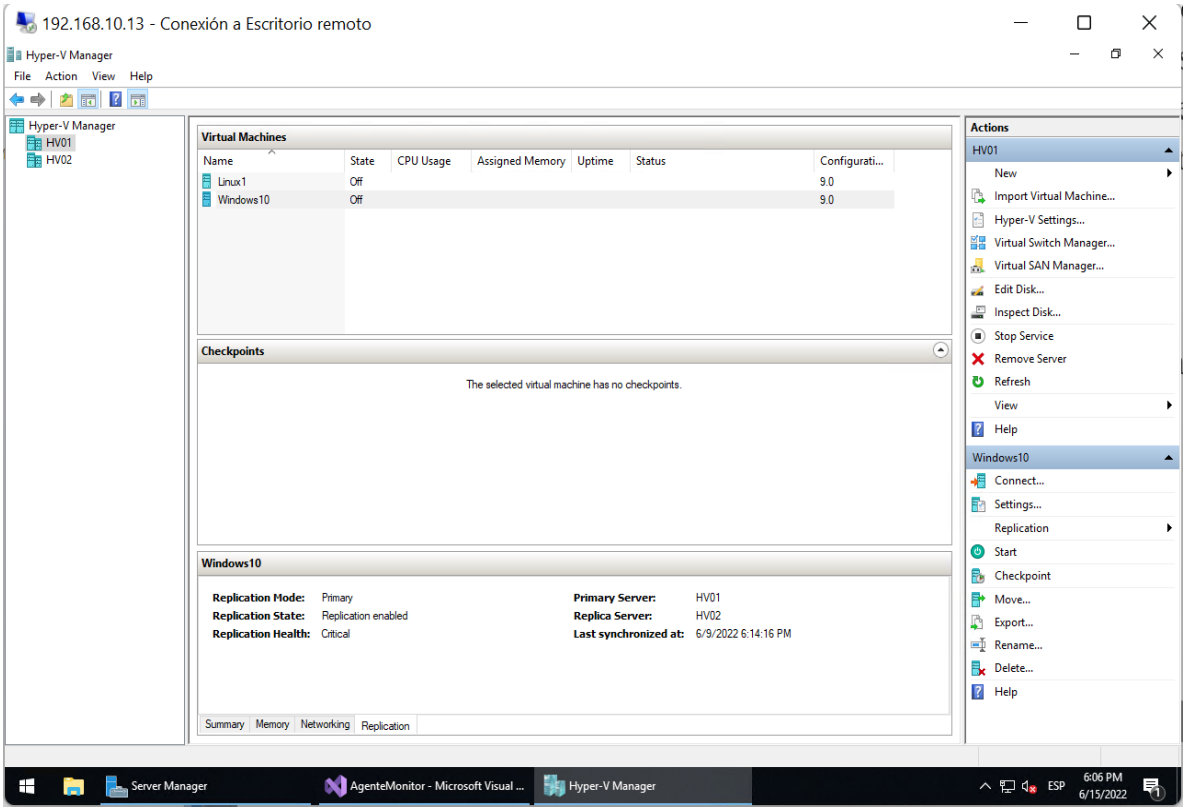


Figura 10. Máquina virtual con Windows server y Hyper-V replicando

Se crea una solución como se muestra en la figura 11 donde se agrega un proyecto WinForms para la interfaz gráfica o panel del agente monitor, una aplicación de servicios en C# para el servicio de monitoreo del agente y una aplicación de instalación también conocida como Setup.

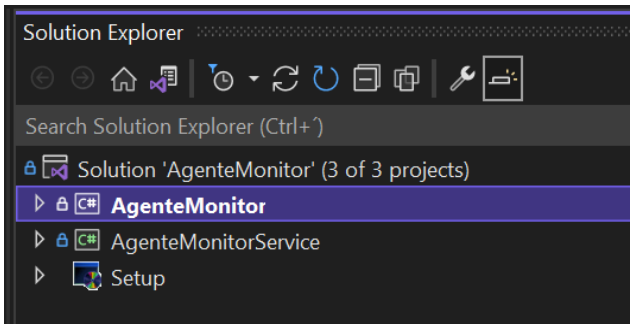


Figura 11. Solución para el agente monitor

Se instalaron las librerías o paquetes necesarios para la aplicación agente monitor con el gestor de paquetes de Visual Studio el NuGet package mostrada en la figura 12, entre los paquetes instalados se tiene Cake en conjunto con System.Management.Automatation para ejecutar comandos PowerShell en nuestra aplicación de servicios e interfaz gráfica WinForms, Flurl para enviar peticiones HTTP a él API, y Newtonsoft.Json para manejar los datos obtenidos del API.

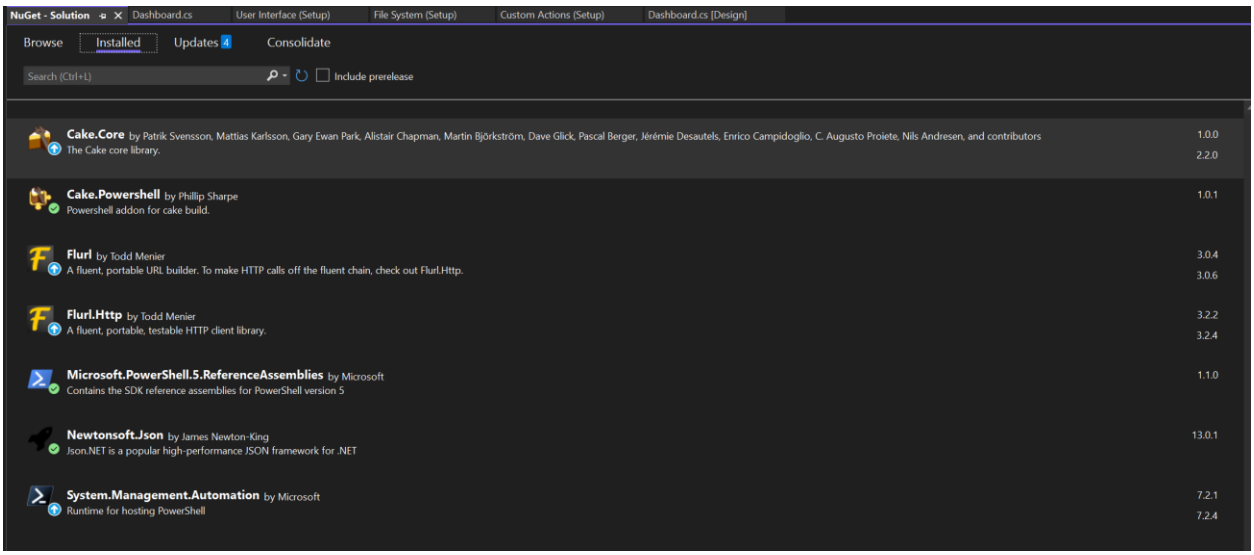


Figura 12. Administrador de librerías Nuget Package

Se diseñó la interfaz de inicio de sesión mostrada en la figura 13 y se añade un evento click al botón “Login”, en el evento asíncrono buttonLogin_Click se obtienen los datos de las cajas de texto textBoxEmail y textBoxPassword para crear un JSON el cual será enviado a la url “/auth/local” en una petición POST del API.

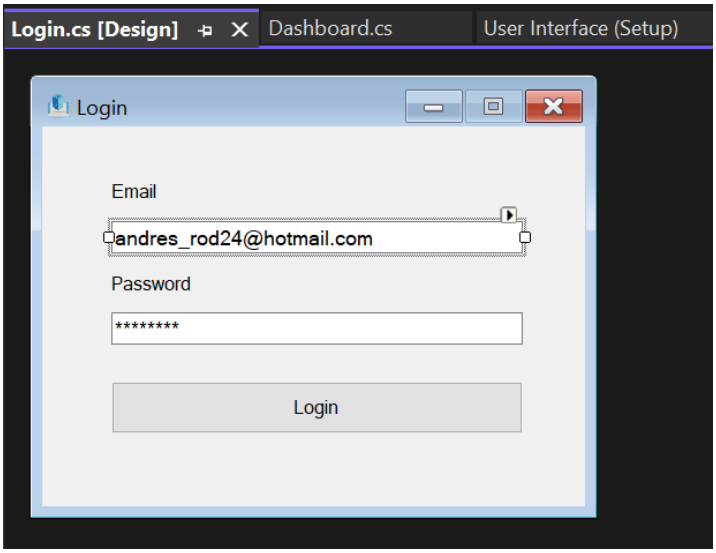


Figura 13. Formulario de inicio de sesión

Si los datos de acceso son correctos se guarda el JWT (token de validación de usuario) en las propiedades de configuración de la aplicación para luego cambiar de vista a la seleccionadora de clientes de la figura 14 o mostrara un cuadro de dialogo con los datos del error como en la figura 15.

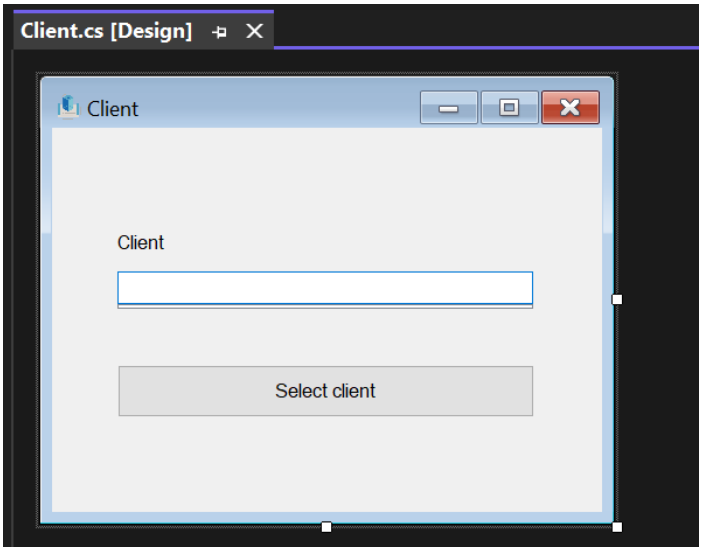


Figura 14. Seleccionador clientes

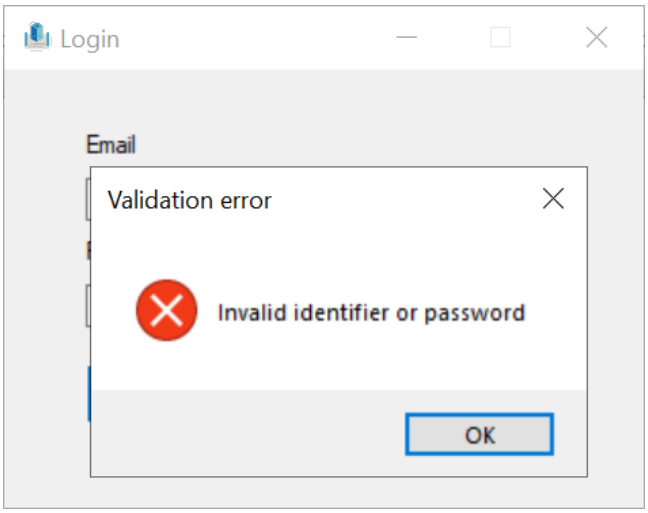


Figura 15. Ventana de datos inválidos

En el formulario de cliente se le pide al API traer los clientes asignados al usuario y en el caso que este solo tenga 1 la vista no se mostrara ya que por defecto seleccionara este único cliente, pasando a preguntar si el nombre de este servidor ya está registrado en el cliente de lo contrario preguntar si desea registrarlo, una vez ya este validado el servidor se cambia a el formulario dashboard mostrada en la figura 16.

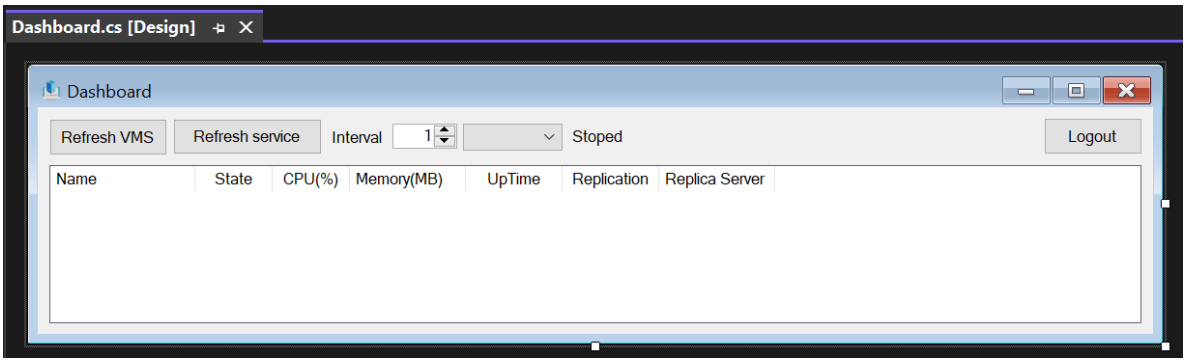


Figura 16. Formulario dashboard

El formulario dashboard cuenta con un script PowerShell para listar las máquinas virtuales una vez abierto o al presionar el botón de refresco de máquinas virtuales ubicado en la parte superior izquierda como se puede observar en la figura 17, en este listado de máquinas virtuales

se tiene la opción de activar o desactivar el monitoreo a través de un checkbox el cual se hará valido presionando en el botón de refrescar el servicio que se encuentra en la parte superior izquierda de este formulario, además se puede cambiar el tiempo de intervalos en el cual se hace el monitoreo en minutos o horas cambiando este valor en el seleccionador de tiempos como se observa en la figura 18.

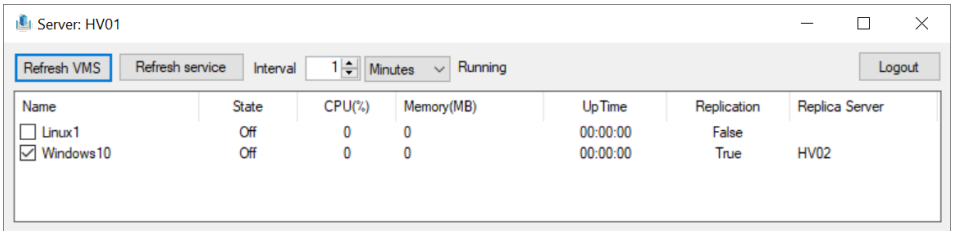


Figura 17. Listado de máquinas en la dashboard

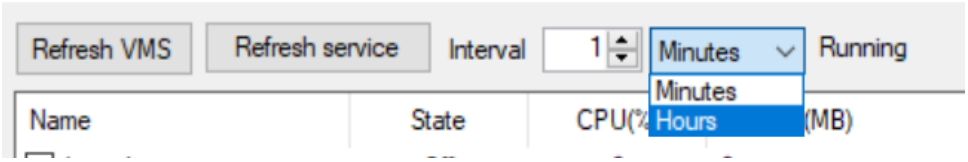


Figura 18. Acciones de la dashboard

Para probar las funciones la aplicación de servicios es necesario compilarla e instalarla mediante comandos InstallUtil del framework .Net el cual en este caso se encuentra instalado en la ruta “C:\Windows\Microsoft.NET\Framework\v4.0.30319”, en el desarrollo de esta aplicación se usó “InstallUtil "C:\Users\Administrator\Desktop\Projects\monitor-replica\Agente\AgenteMonitorService\bin\Debug\AgenteMonitorService.exe" -i” para instalar y “-u” para desinstalarla que pueden observarse en el listado de servicios de Windows observada en la figura 19.

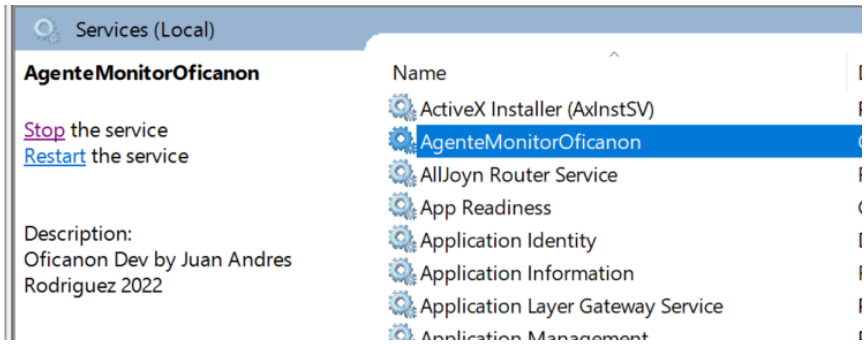


Figura 19. Listado de servicios Windows

La aplicación de servicios recibe un parámetro String que traduce a un JSON el cual contendrá la información a guardar en las propiedades de configuración del monitor tal como el intervalo, maquinas a monitorear, la identificación del cliente, el JWT del usuario y el id del servidor, si este inicia sin parámetros los leerá de las propiedades de configuración y si estas están vacías no ninguna acción entonces.

La aplicación de servicio ejecuta una tarea en la cual mediante el comando PowerShell "Measure-VMReplication" que tomara los datos de cada máquina señalada en el JSON con las instrucciones mediante un argumento "VMName", una vez tiene las propiedades necesarias las envia en un paquete JSON al API por medio una petición POST a la url "/replications".

Se creó un paquete de instalación Setup como se observa en la figura 20 con las dependencias necesarias para el funcionamiento de la dashboard y el servicio "AgenteMonitorOficanon".

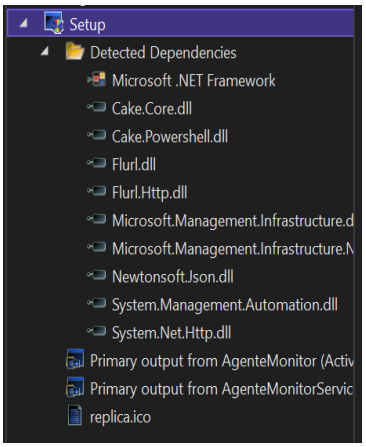


Figura 20. Paquete de instalación

En el sistema de archivos del instalador vista en la figura 21 en la carpeta de la aplicación agregan todas las dependencias y en el escritorio se crea un acceso directo como en la figura 22.

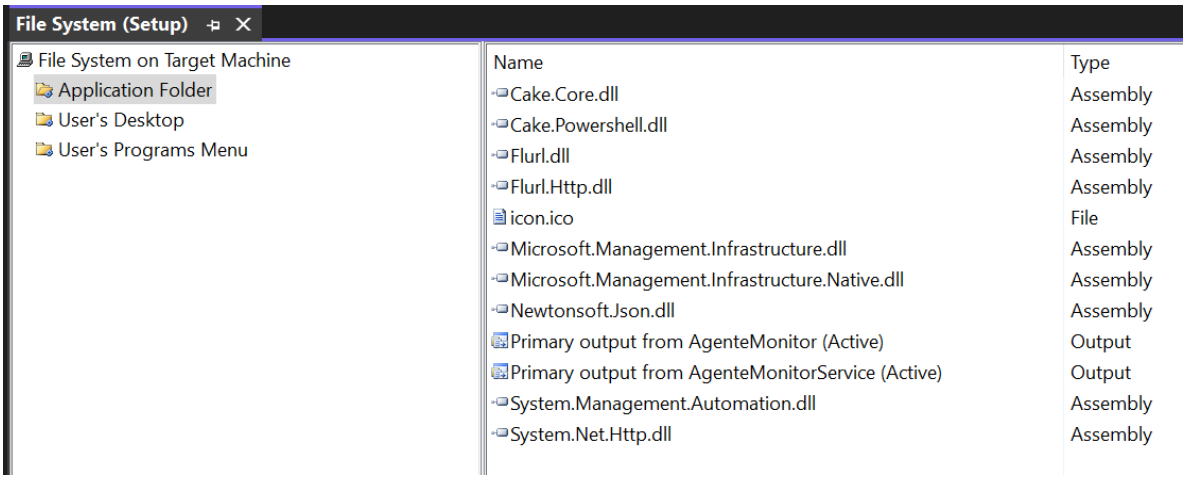


Figura 21. Sistema de archivos de la instalación

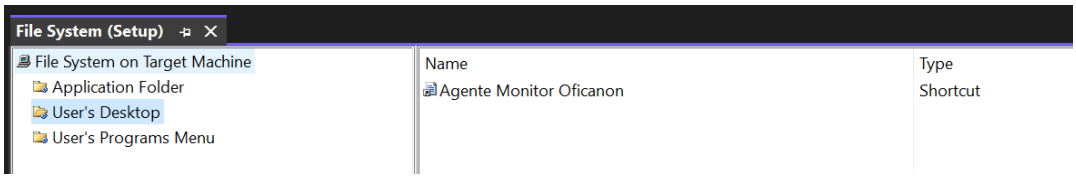


Figura 22. Acceso directo en el escritorio al instalar la aplicación

Una vez creado el paquete de instalación observada en la figura 23 se instala en el servidor para comprobar su funcionamiento.

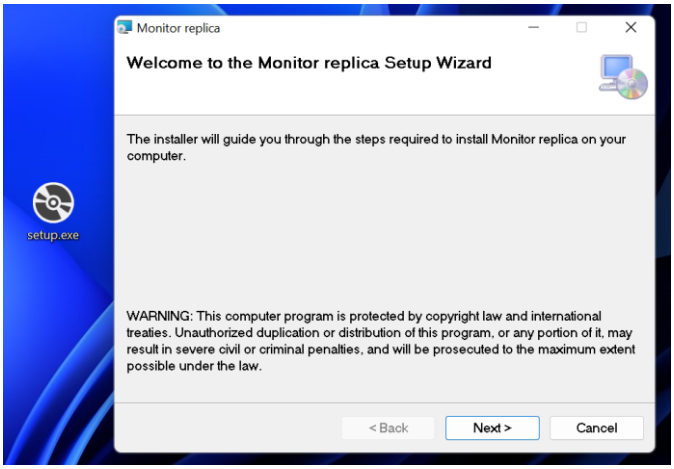


Figura 23. Instalador del agente monitor

4.1.3 Diseño de la base de datos. Se diseña la base de datos como se puede ver en la figura 24 la cual satisface la necesidad del sistema monitoreo con las siguientes tablas:

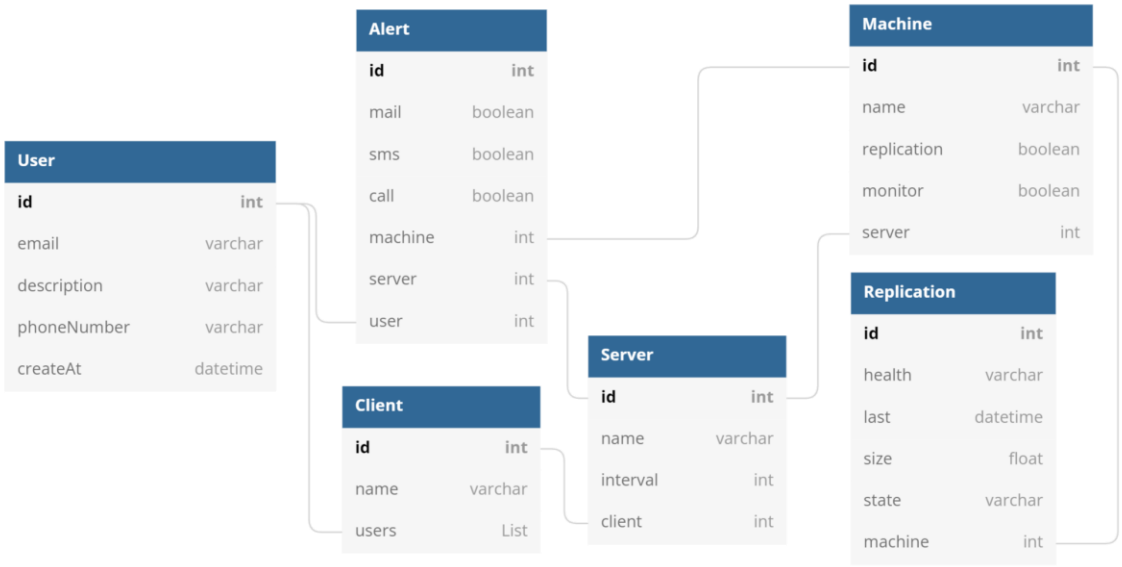


Figura 24. Diagrama base de datos

- User: Esta tabla alberga los datos de los usuarios de la aplicación además del inicio de sesión.
- Client: Un cliente es una empresa que requiera el servicio, donde un usuario puede pertenecer a múltiples clientes.
- Server: Es la maquina Hyper-V que tiene las máquinas virtuales, un cliente puede tener múltiples servers, pero estos no pueden tener el mismo nombre.
- Machine: La máquina virtual solo puede pertenecer a un servidor.
- Replication: Son los datos obtenidos en la replicación de cada máquina virtual.
- Alert: Es la tabla que tiene las configuraciones de alertas y cuando ocurra un evento es donde el sistema va a decidir a qué usuario notificar y si enviar un mail, sms o una llamada.

4.1.4 Desarrollar un API con Strapi. Mediante la terminal de Windows se ejecuta en la carpeta del repositorio del proyecto el comando “npx create-strapi-app@latest api-monitor-replica” para crear la aplicación Api de Strapi, una vez esta esté creada arrojara el mensaje de que está disponible en la Url localhost:1337/admin en la salida de la figura 25.

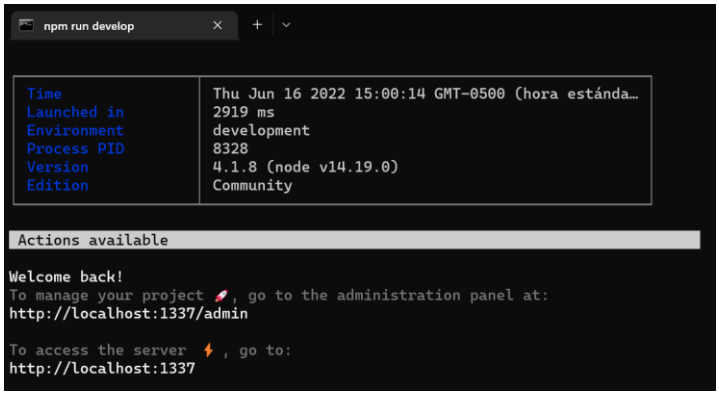


Figura 25. Strapi corriendo en Windows

Strapi solicitara unos datos para crear las credenciales del administrador del Api, ya suministrada esta información se ingresa a ella cambiando a una ventana de inicio de sesión como se puede ver en la figura 26 en la cual se introducen los datos previamente registrados.

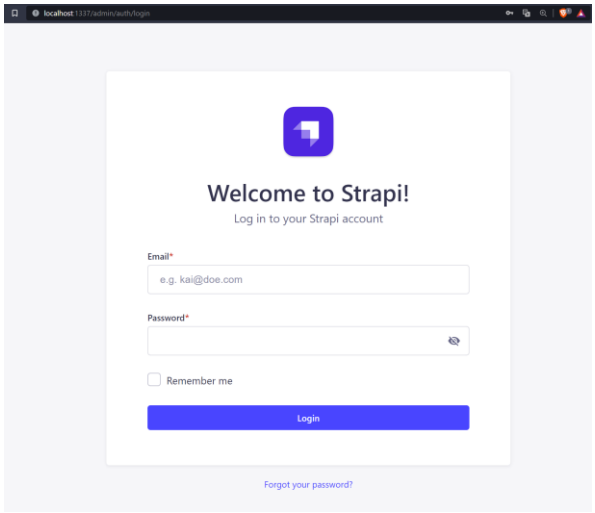


Figura 26. Strapi ventana inicio de sesión

Dentro de la aplicación Strapi se procede a crear el Content-Type Builder con el modelo de la base de datos de la figura 27.

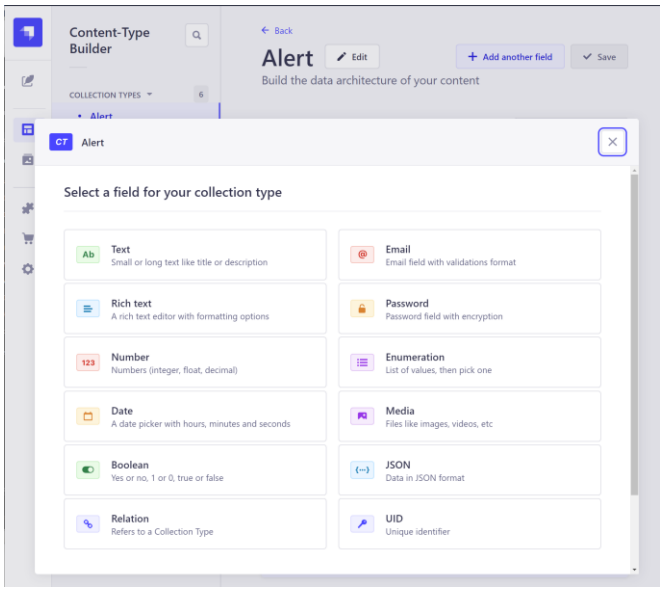


Figura 27. Creación del modelo en Strapi

Ya creado el modelo del api se pasa a dar permisos de acceso a los roles de la figura 28.

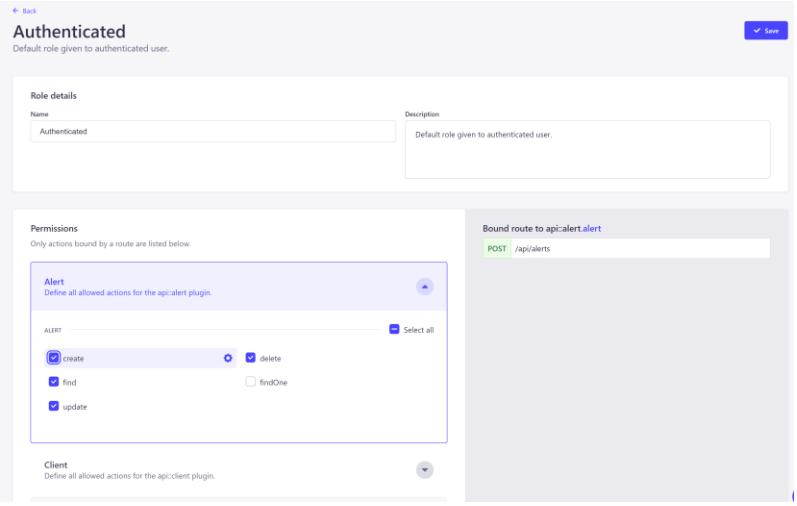


Figura 28. Configuración de los permisos en los roles

Con la ayuda del cliente Postman programa visto en la figura 29 se crean las peticiones HTTP necesarias para usar el API, con esto se modifican los controladores para validar si el usuario pertenece al cliente, si el servidor pertenece a cliente, si la maquina pertenece a el servidor, etc… entre las peticiones creadas se tienen las de autenticación, consultas a los modelos, edición de los modelos, creación de alertas y eliminación de alertas.

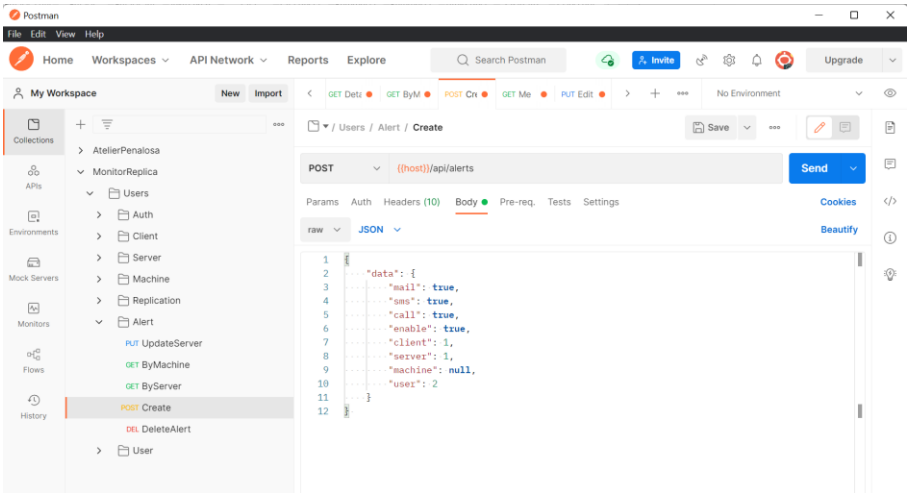


Figura 29. Cliente HTTP Postman

4.1.5 Servicios SMS y llamadas. Son adquiridos los servicios de Plivo que tiene un valor expresado en la Tabla 1 la cual fue tomada con un dólar a 4,232.34 pesos colombianos:

Tabla 1. Precios recursos de Plivo

Servicio	Precio Dólares	Precio Pesos Colombianos
SMS	$0.0410/sms	$173.53
Llamada	$0.0330/min	$139.67

Plivo es un sistema prepagado que a medida que se envíen mensajes o llamadas va descontando del saldo y la empresa para el desarrollo recargo 20 dólares de la figura 30.

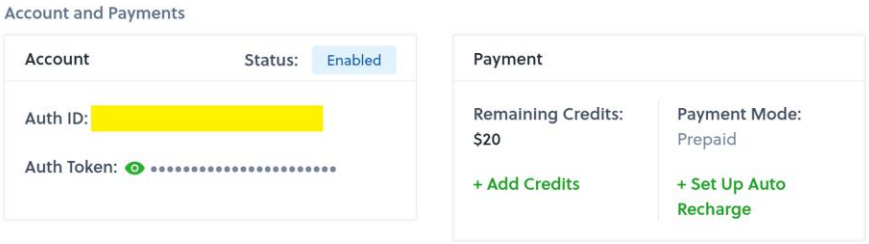


Figura 30. Recursos de Plivo para el desarrollo de la aplicación

Para usar el servicio de llamadas de Plivo fue necesario crear una pequeña aplicación que a partir de un texto que retorne un archivo XML con el formato requerido en la documentación (https://www.plivo.com/docs/voice/xml/overview/), esta aplicación para pruebas corre en la dirección http://speak.monitor.juanrodriguez.tech/speak/es-US/MAN/texto ejemplo visto en la figura 31.

```
<Response>
  <Speak voice="MAN" language="es-US">texto ejemplo</Speak>
</Response>
```

Figura 31. Respuesta XML para la llamada

En la funcion de alertas donde almacenan los datos de las replicaciones, es decir en el créate del controlador replication del Api Strapi, se debe llamar o enviar mensajes de texto según el tipo de alertas, para poder usar estos servicios se instancia la librería de Plivo con los parámetros de inicio de sesión Auth ID y Auth Token.

Para las alertas por llamadas se usan la funcion calls.create, esta funcion pide un número from donde se usa el 897781 correspondiente a un número informativo, además del número con indicativo al cual se llamara y un mensaje formato XML, el mensaje dispuesto para la llamada es "Alerta Monitor Cliente {Nombre cliente} Servidor {Servidor nombre} Máquina virtual {Maquina nombre} Estado {Estado} {Fecha-Hora}"

En el caso de los mensajes de texto se usa la funcion messages.create, que recibe el parámetro src con el número 897781, el dst que es el numero con indicativo del usuario a notificar y el text el cual tiene la siguiente estructura "Alerta Monitor HV: {Cliente nombre} (Cliente) - {Servidor nombre} (Servidor) - ${Maquina nombre} (MV). Estado: {Estado} - {Fecha-Hora}".

Una vez se configuran las llamadas y mensajes de texto se prueba con una alerta como se muestra en la figura 32 para la llamada y figura 33 mensajes de texto.

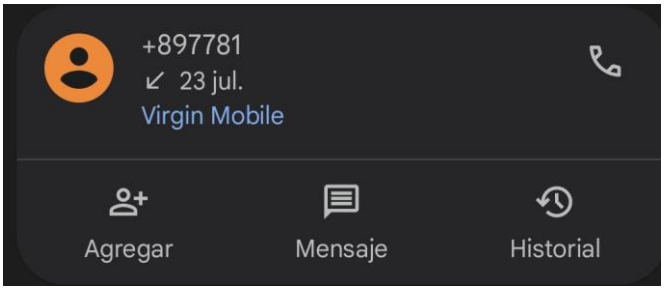


Figura 32. Prueba llamada Plivo

Figura 33. Prueba mensaje de texto Plivo

4.1.6 Despliegue del API. Se usa como servidor de pruebas una computadora suministrada por Patiño y Contreras en la cual se trabajó este proyecto, esta computadora cuenta con 20GB de memoria RAM y un procesador Core i5 de 11va generación, en ella creo sobre su sistema operativo Windows 11 una instalación sencilla de NodeJS y un servidor de base de datos MariaDB con un programa libre llamado XAMMP el cual provee de PHPMyAdmin para administrar la base de datos de la figura 34.

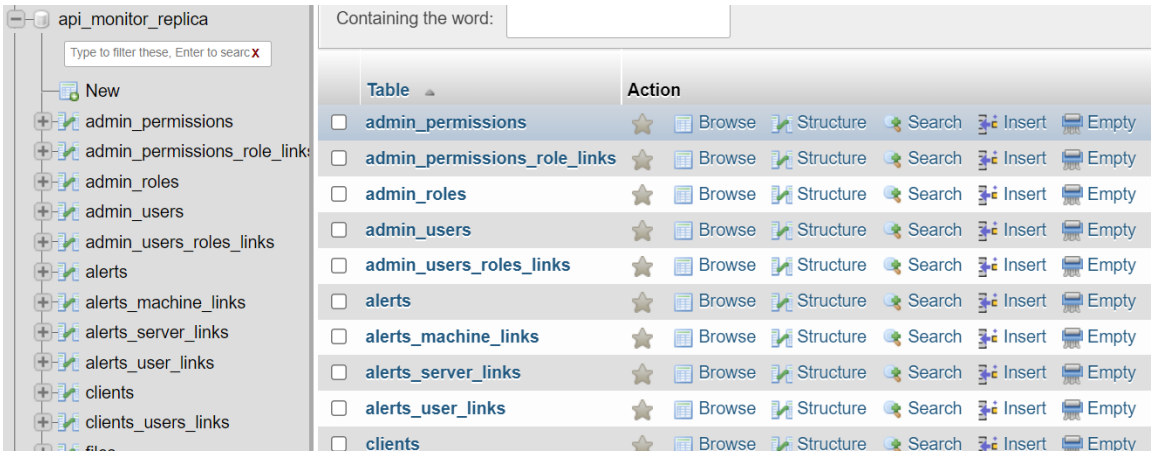


Figura 34. Base de datos en PHPMyAdmin

Se instala de manera global en NodeJS un administrador de ejecución en segundo plano conocido como PM2, se crea la tarea y se ejecuta verificando su funcionamiento con la salida

mostrada en la terminal de la figura 35.

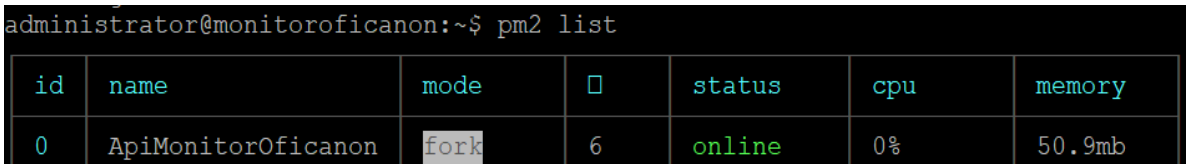


Figura 35. Administrador de ejecución en segundo plano PM2

Se configura el servidor NGINX instalada en el sistema operativo donde se estará ejecutando la máquina de pruebas, en donde se le indica que todas las peticiones que lleguen por medio del dominio de pruebas sean atendidas por este servidor en el puerto del API de la figura 36.

Figura 36. Configuración del sitio en NGINX para el API

Ahora se procedió a apuntar el dominio de prueba a la dirección IP en la cual funciona nuestro servidor, esta dirección está en una zona abierta del modem con la autorización del proveedor de internet observado en la figura 37.

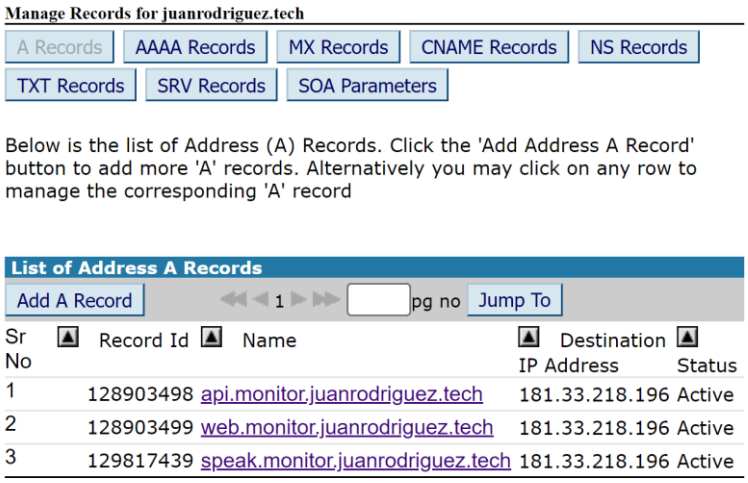


Figura 37. Registro de dominios para la aplicación

4.2 Elaborar una Aplicación Web Administrativa

Una web administrativa es necesaria para tener acceso a los datos sin tener que instalar aplicaciones en el dispositivo usado, es recomendable usar marcos de desarrollo ya que estos están optimizados para el manejo de interfaces de usuarios y de datos, en este proyecto se usa como marco de desarrollo NUXT un paquete de VUE y VUEX preconfigurado ya que se buscaba un desarrollo ágil.

En el diseño de una aplicación Web se debe tomar nota de cada vista que sea necesaria para el manejo de sus usuarios, debe ser intuitiva es decir que el usuario no tenga que aprender muchas cosas para poder usarla y además reactiva ya que es recomendable por rendimiento no tener que estar refrescando el navegador de internet ya que esto genera tráfico.

4.2.1 Programación aplicación Web NUXT. Por medio del comando “npx create-nuxt-app project-name” se crea la aplicación Nuxt con los parámetros de aplicación universal, una vez creada abrimos el proyecto con la ayuda del IDE de desarrollo Visual Studio Code de la figura 38.

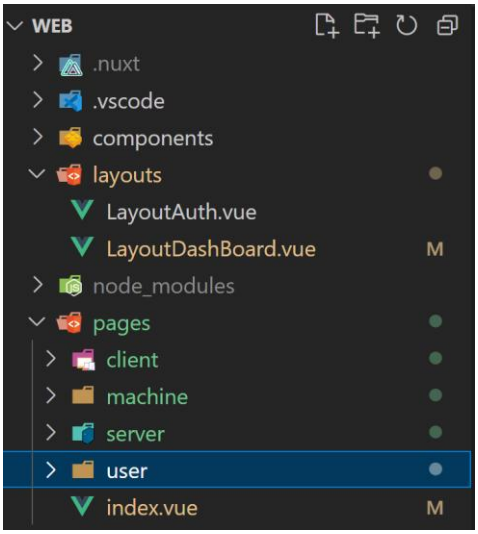


Figura 38. Proyecto abierto en el visual studio

Por medio del comando “npm run dev” se ejecuta la aplicación en la consola visto en la figura 39, esta mostrara el “Environment” o el modo de desarrollo que puede ser desarrollo o producción, el “Rendering” es donde se está generando las vistas el cual puede ser lado cliente o lado server, el “Target” que es donde se va a ejecuta que es este caso es server y la ruta del sitio con sus respectivos procesos ejecutados:

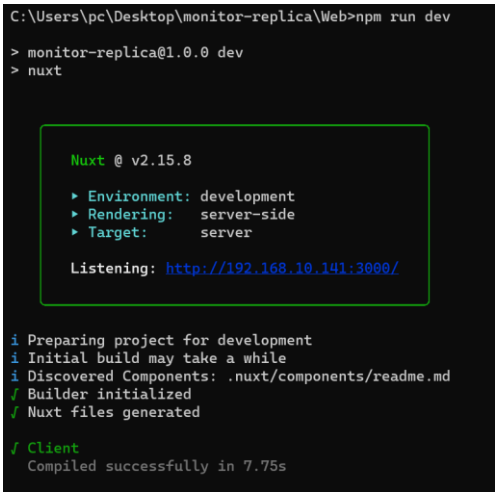


Figura 39. NUXT corriendo en modo depuración

Una vez abierto el proyecto se crea la estructura layouts (paginas maestras que contendrán las vistas) y las carpetas de las paginas para configurar el enrutamiento del sitio basado en el sistema de archivos árbol:

---| client/

------| _id.vue

---| machine/

------| _id.vue

---| server/

------| _id.vue

---| user/

------| edit.vue

------| login.vue

------| register.vue

Que se traduce automáticamente en un Json siendo: id un parámetro recibido en la ruta de la aplicación:

```
routes: [
   {
     name: 'client_details,
     path: '/client/:id',
     component: 'pages/client/_id.vue'
   },
  {
     name: “machine_details,
     path: '/machine/:id',
     component: 'pages/machine/_id.vue'
```

```
  },
  {
    name: “server_details,
    path: '/server/:id',
    component: 'pages/server/_id.vue'
  },
  {
    name: “user_edit,
    path: '/user/edit',
    component: 'pages/user/edit.vue'
  },
  {
    name: “user_login,
    path: '/user/login,
    component: 'pages/user/ login.vue'
  },
  {
    name: “user_register,
    path: '/user/register,
    component: 'pages/user/ register.vue'
  }
]
```

4.2.2 Diseño del módulo VUEX. Nuxt viene con VUEX precargado en su paquete de instalación, solamente basta con crear el archivo index.js y los demás necesarios para su funcionamiento con el modelado de la documentación como se muestra en la figura 40.

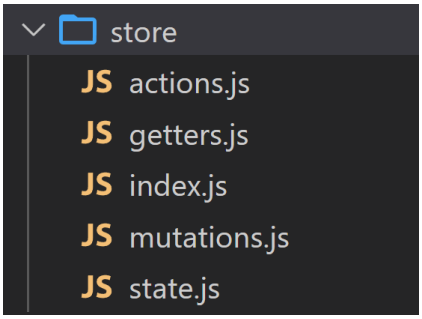


Figura 40. Proyecto abierto en el visual studio

El modelado del store de VUEX se compone de:

state.js: Es el modelo de datos que usa VUEX se le denomina único árbol de estados, esto quiere decir que se trata de un objeto JavaScript con todos los estados o variables de la aplicación, Un solo árbol de estados permite acceder de manera sencilla a una variable desde cualquier componente de la aplicación sin tener que estar enviando Props entre componentes ya que esto se tornaría engorroso por estar pasando parámetros entre muchos componentes.

Para la aplicación Web se creó el siguiente state:

```
export default {
  clients: [],
  client: {users: [], servers: []},
  servers: [],
  server: {
    client: {},
    machines: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
    alerts: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
  },
```

```
    resources: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
    machines: [],
    machine: {
      server: {},
      alerts: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
      replications: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
      resources: {results: [], pagination: {page: 0, pageSize: 0, pageCount: 0, total: 0}},
    },
    selected: {client: 0, server: 0, machine: 0},
    load: {clients: false}
  }
```

Getters.js: En ocasiones es necesario calcular o procesar previamente un estado de los datos para ser representado en varios componentes, VUEX permite tener estos estados calculados como una variable de acceso rápido por medio de un getter.

Los getters que se usaron en la aplicación Web del monitor son los siguientes:

```
export default {
  isAuthenticated (state) {return state.auth.loggedIn},
  loggedInUser (state) {return state.auth.user},
  clients (state) {return state.clients},
  client (state) {return state.client},
  servers (state) {return state.servers},
  server (state) {return state.server},
  machines (state) {return state.machines},
```

```
    machine (state) {return state.machine},
    selected (state) {return state.selected},
    load (state) {return state.load}
}
```

Mutations.js: Para cambiar los valores de un state en Vuex se recomienda usar mutations, la cuales son funciones similares a eventos que aparte de cambiar el estado de los state, se encarga de notificar a los componentes que usen estos valores que deben refrescarlos.

Las mutations que se usan en el cliente son:

```
export default {
    setClients (state, payload) {...},
    setClient (state, payload) {...},
    setClientUsers (state, payload) {...},
    setSeletedClient (state, payload) {...},
    setServers (state, payload) {...},
    setServer (state, payload) {...},
    setServerClient (state, payload) {...},
    setServerAlerts (state, payload) {...},
    setServerResources (state, payload) {...},
    setServerMachines (state, payload) {...},
    setSeletedServer (state, payload) {...},
    setMachines (state, payload) {...},
    setSeletedMachine (state, payload) {...},
    setReplications (state, payload) {...},
```

```
    setMachine (state, payload) {...},
    setMachineResources (state, payload) {...},
    setMachineAlerts (state, payload) {...},
    setMachineReplications (state, payload) {...},
    updateUser (state, payload) {...},
    deleteAlert (state, payload) {...},
    clearStore (state) {…}
}
```

Actions.js: Las actions son funciones parecidas a la mutations pero en vez de mutar o cambiar states estas llaman a las mutations que son las expertas en hacer estos cambios, estas actions pueden tener funciones de tipo asincrónicas como peticiones HTTP.

En los actions con axios para las peticiones HTTP del API y con qs para el formato de las consultas se tiene la siguiente estructura:

```
import axios from 'axios'
import qs from 'qs'
const apiBaseUrl = `${process.env.apiBaseUrl}/api`
export default {
  async fetchClients ({commit, dispatch, state}) {...},
  async fetchClient ({ commit, state }) { ... },
  async fetchClientUsers ({ commit, state }) { ... },
  async fetchServers ({ commit, state, dispatch }) { ... },
  async fetchServer ({ commit, state }) { ... },
  async fetchServerAlerts ({ commit, state }, { pagination }) { ... },
```

```
    async fetchServerResource ({ commit, state }) { ... },
    async fetchServerMachines ({ commit, state }, { pagination }) { ... },
    async fetchServerClient ({ commit, state }) { ... },
    async updateAlert ({ commit, state }, { alert, type }) { ... },
    async fetchMachines ({ commit, state }) { ... },
    async fetchMachine ({ commit, state }) { ... },
    async fetchMachineResource ({ commit, state }) { ... },
    async fetchMachineReplications ({ commit, state }, pagination) { ... },
    async fetchMachineAlerts ({ commit, state }, pagination) { ... },
    async updateUser ({ commit, state }, dataParameter) { ... },
    async deleteAlert ({ commit, state }, { id, machine }) { ... },
    addAlert ({ state }, dataParameter) { ... },
    async addUserClientByMail ({ state, dispatch }, email) { ... },
    async deleteClientUser ({ dispatch, state }, { userId }) { ... }
  }
```

4.2.3 Diseño de las vistas. Patiño y Contreras suministro una plantilla de un panel administrativo cuya licencia cubre proyectos que se hagan en la empresa, la plantilla “StarAdmin 2 Pro” vista en la figura 41 contiene múltiples vistas o componentes como el inicio de sesión, tablas, tarjetas, iconos, etc...

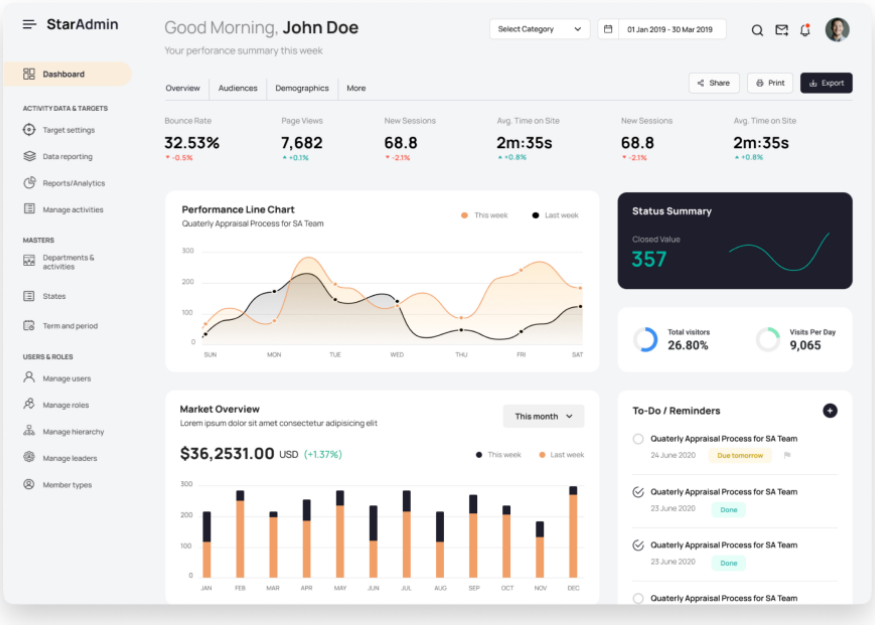


Figura 41. Vista del panel administrativo StarAdmin 2 Pro

Un layout es un contenedor o marco en el cual se van a mostrar las vistas en NUXT carpeta de la figura 42, el layout de autenticación “LayoutAuth.vue” contiene la página de login ya que esta no contiene elementos como un menú de navegación, el layout de los paneles administrativos “LayoutDashBoard.vue” contiene el marco de navegación superior con las opciones del usuario y el lateral izquierdo de navegación de servidores.

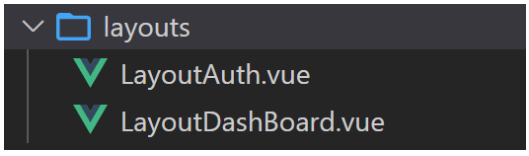


Figura 42. Carpeta contenedora de los layouts en el proyecto NUXT

Con la plantilla de “Login” del panel administrativo, se crea la vista de inicio de sesión vista de la figura 43 donde se agregan los métodos y las variables del modelo para hacer la petición al API consumiendo la acción del login de NUXT.

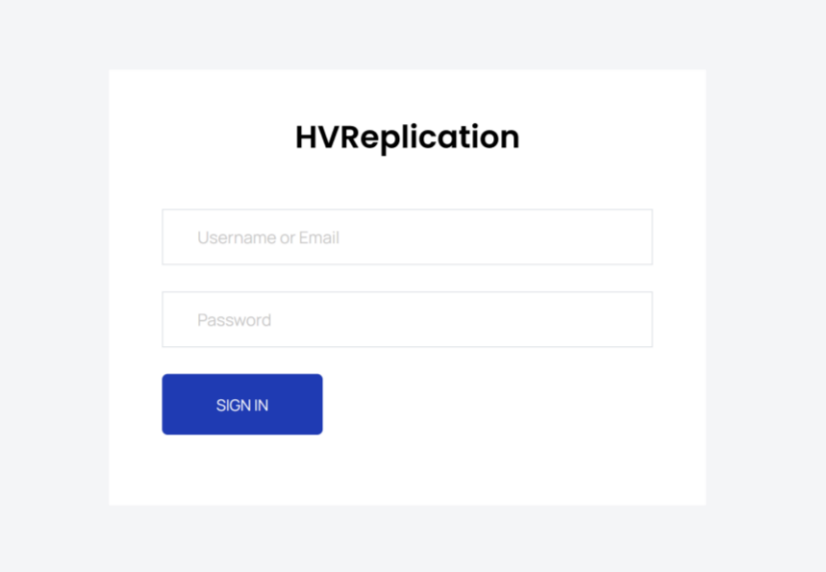


Figura 43. Vista del inicio de sesión en el cliente Web

Una vez se pasa al inicio la sesión el API retorna una llave para hacer las futuras peticiones y que él sepa que usuario las realiza, nuestra aplicación valida en el layout de los paneles administrativos si él tiene acceso a los datos de los usuarios para saber si este está con una sesión correcta mostrando la vista del panel administrativo como se observa en la figura 44.

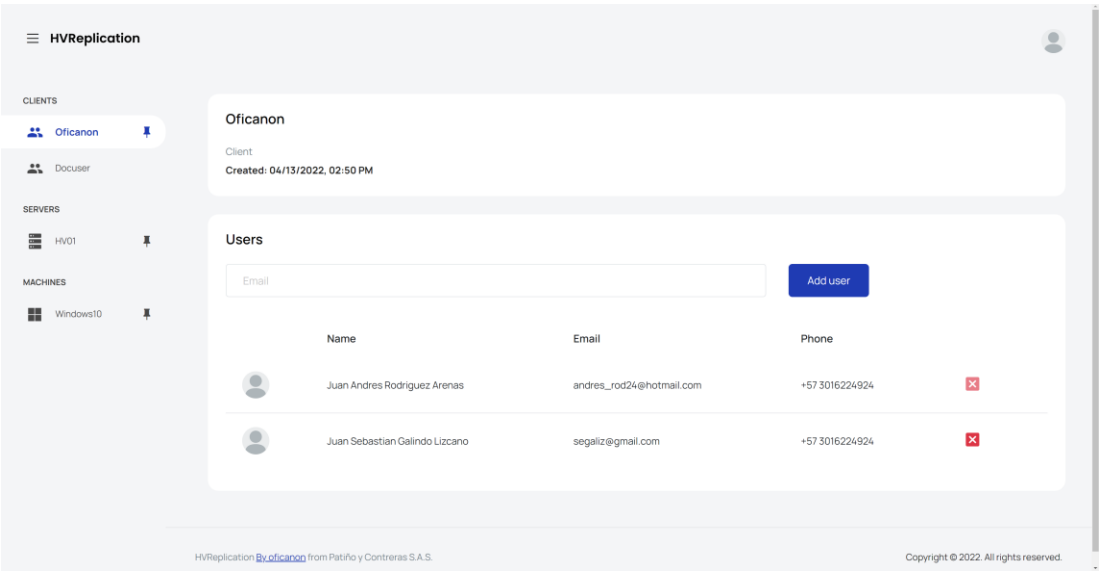


Figura 44. Vista del panel administrativo

Para el manejo de la navegación entre clientes, servidores y maquinas, se diseñó un menú lateral observado en la figura 45 dinámico donde dependiendo de las opciones seleccionadas estas cambian su contenido, por ejemplo, si se cambia el cliente este refrescara el listado de servidores que a su vez actualizara la lista de máquinas virtuales.

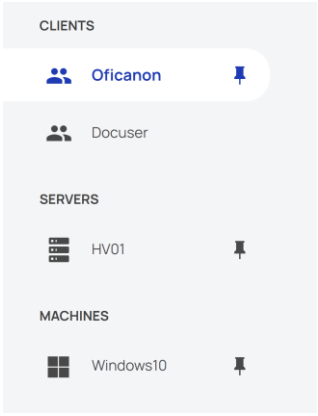


Figura 45. Menú lateral izquierdo de navegación de servidores

En la parte superior derecha de layout de los paneles administrativos se topará con la imagen del usuario y si se presiona click en ella las opciones de cerrar sesión y editar el usuario visto en la figura 46.

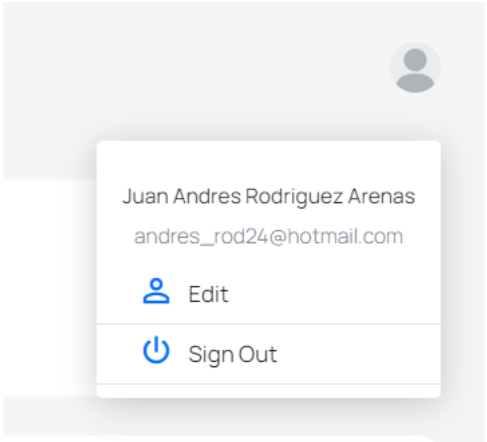


Figura 46. Opciones del usuario menú superior

La edición de usuarios se realiza en una vista observada en la figura 47 que cuenta con un formulario con los datos del usuario con la sesión abierta, los únicos datos que no se pueden cambiar son el usuario y correo electrónico, al dar click en la imagen de este permite seleccionar una nueva para ser subida al servidor.

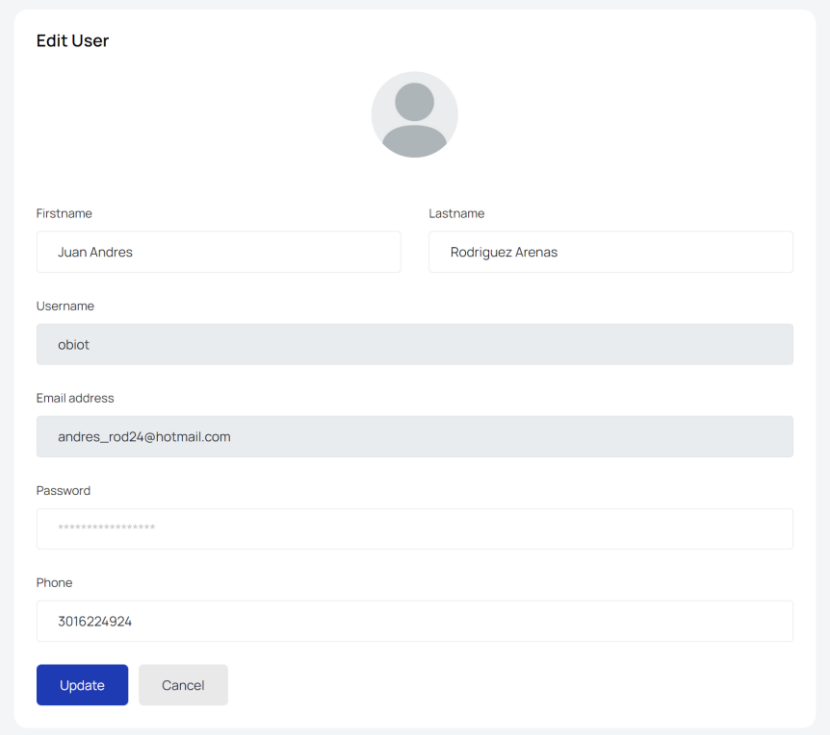


Figura 47. Vista de edición de usuarios

En la opción del cliente se permite agregar usuarios registrados en el API del sistema como se puede observar en la figura 48, una vez se han añadido se podrán agregar a las alertas de los servidores o máquinas virtuales, para agregar el usuario solo bastara con introducir el correo electrónico y dar click en agregar como en la figura 49.

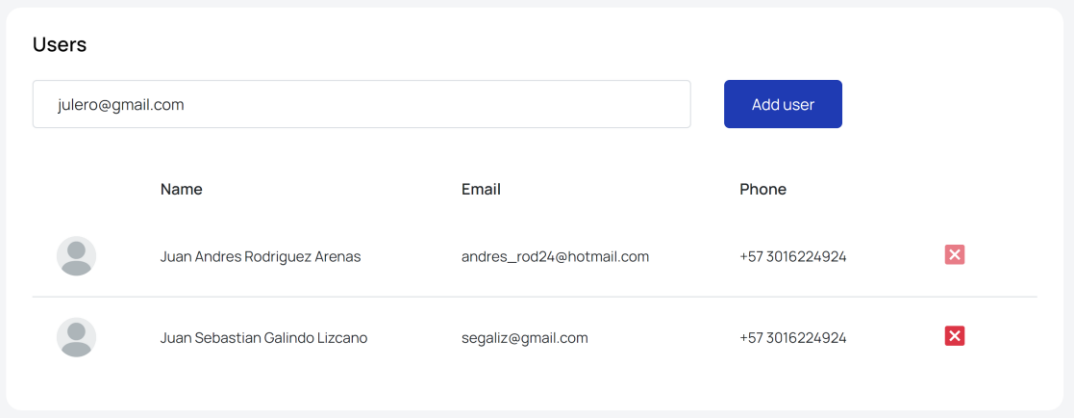


Figura 48. Opción de agregar usuario en el cliente

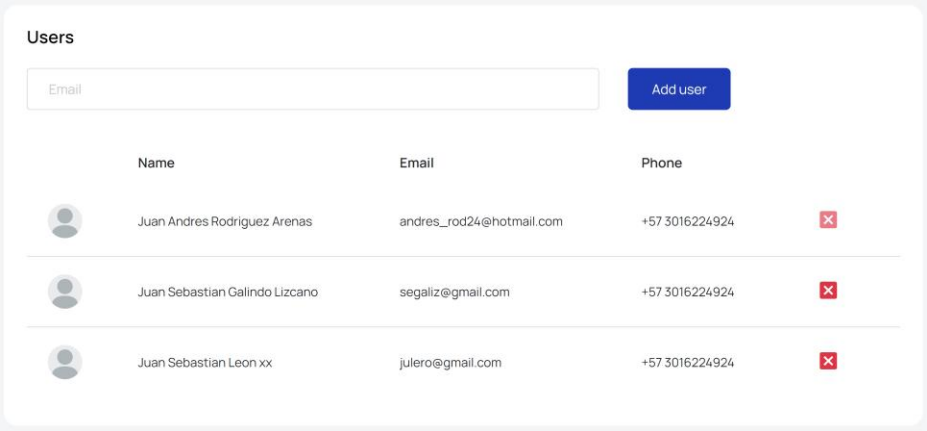


Figura 49. Usuario registrado en el cliente

En la vista de los servidores de la figura 50 se tienen datos como el nombre con su cliente y fecha de registro al sistema, datos de consumo en CPU y memoria que se van reportando con cada monitoreo que llegue a la base de datos, las alertas por usuario donde se puede seleccionar el tipo de alerta que se desea recibir, y por ultimo las maquinas registradas en este con su estado de monitoreo.

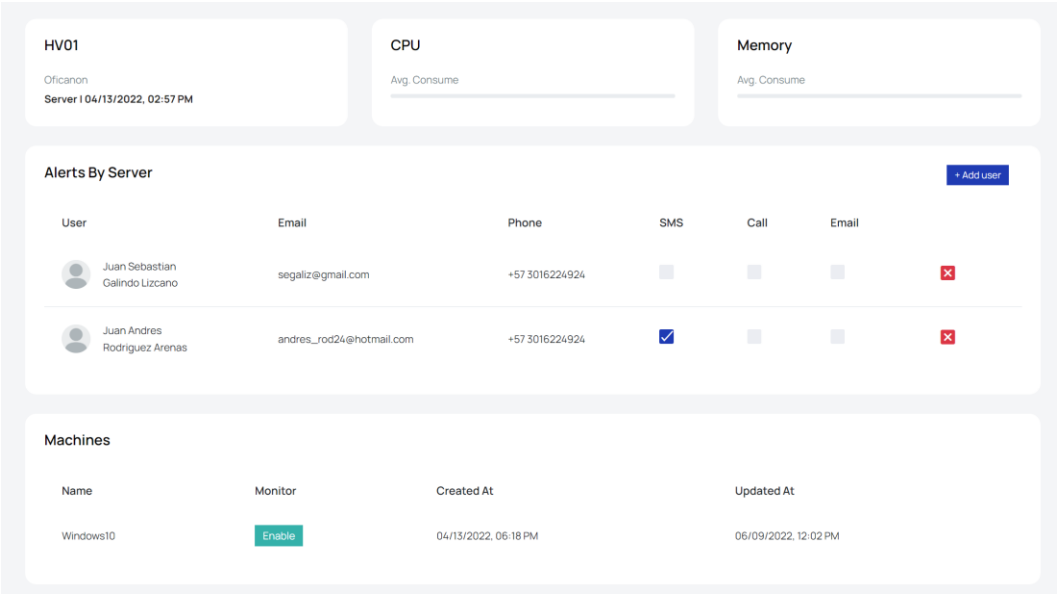


Figura 50. Vista administración para servidores

Para agregar un usuario a las alertas del servidor se presiona en la opción de agregar usuario, luego se selecciona del listado de los usuarios del cliente y se escoge que tipo de alerta que se desea tener para luego guardar al dar oprimir el icono correspondiente a esta funcion, el cual representa con un disquete de color azul como se puede observar en la figura 51.

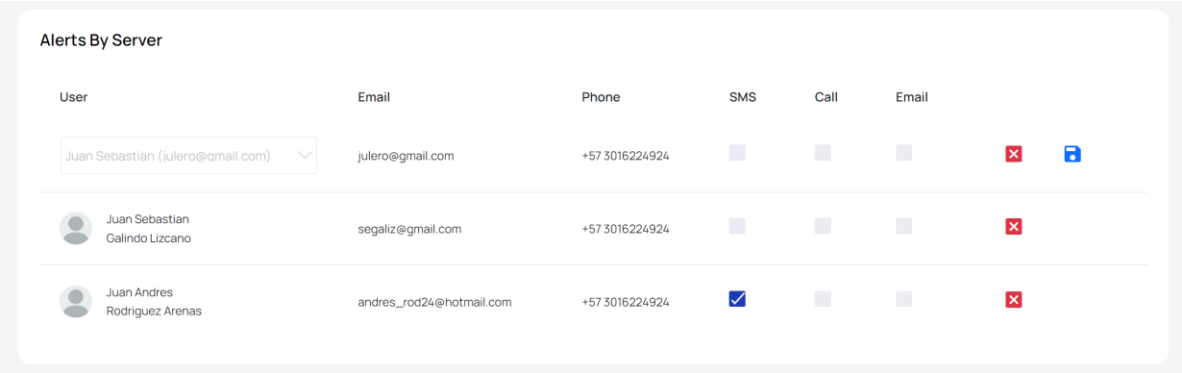


Figura 51. Agregar alertas al administrador de servidor

Para la vista de máquinas virtuales de la figura 52 se tiene una interfaz parecida a la del servidor con una tabla adicional con el listado de replicaciones.

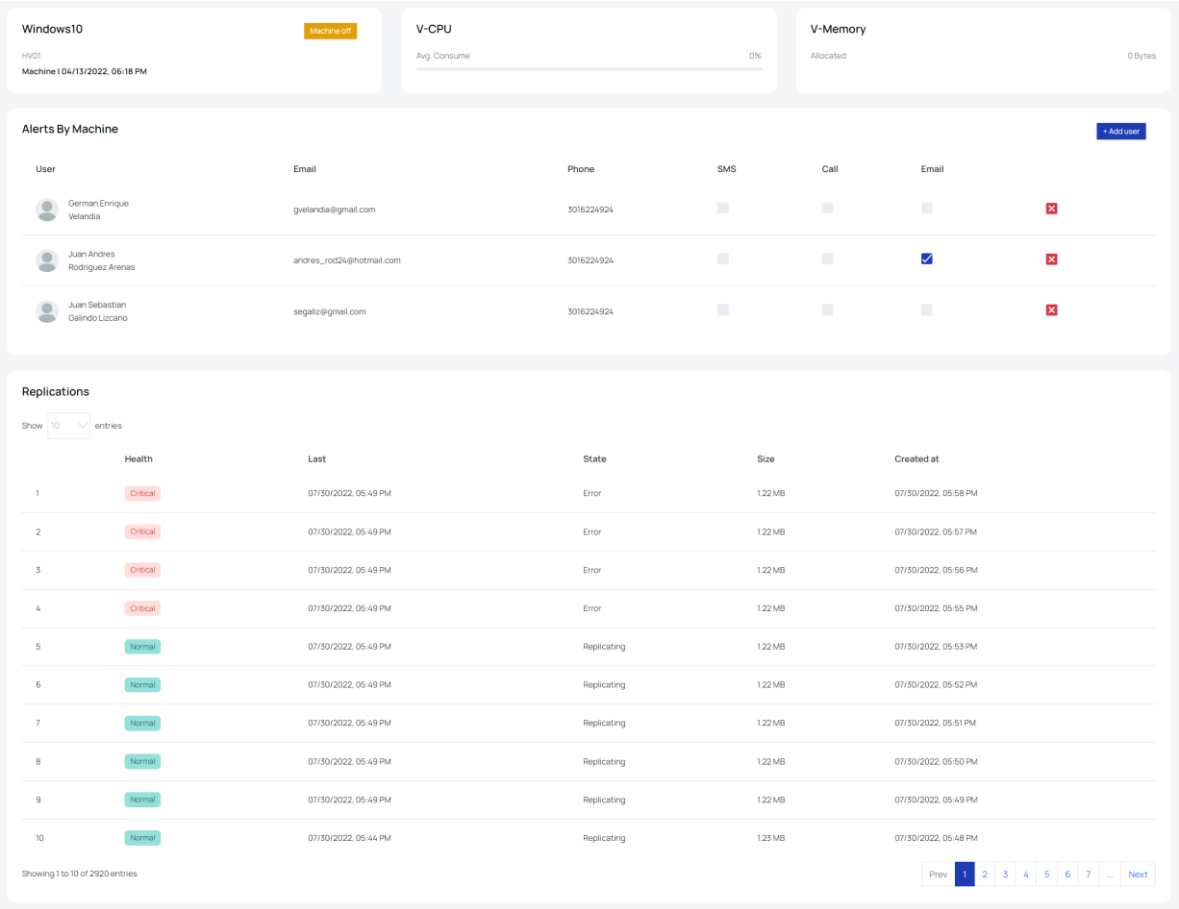


Figura 52. Vista administración para máquinas virtuales

En la tabla de replicaciones se tienen los datos como la fecha de replicación, el tamaño, el estado y la salud de la replicación en la maquina representada con colores.

4.2.4 Configuración VPS. El servidor de pruebas suministrado por Patiño y Contreras observado en la figura 53 se trata de una máquina virtual montado bajo un servidor ESXI de VMWare con un sistema operativo Linux de distribución Ubuntu que cuenta con 4GB de memoria RAM y una CPU con 8 núcleos.

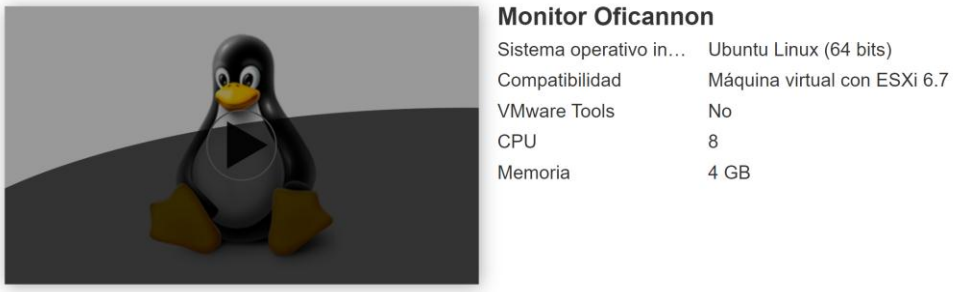


Figura 53. Datos del VPS suministrados por Patiño y Contreras

Figura 54. Conexión SSH servidor de la aplicación

Para la configuración del servidor es necesario instalar NodeJS por medio de una herramienta conocida como NodeJS versión manager (NVM), esta herramienta facilita la adquisición de cualquier versión de NodeJS de una manera sencilla, además de instalar NodeJS en conjunto con NPM esta permite cambiar entre versiones de una manera sencilla, rápida y limpia ya que esta sería una instancia diferente por versión.

Para instalar NVM en Ubuntu se ejecutarán los siguientes comandos de manera secuencial comenzando con adquisición del paquete desde el repositorio de descarga mediante curl:

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash

Ya con el paquete de instalación se debe activar la variable de entorno del NVM mediante el siguiente script:

source ~/.bashrc

Para verificar que NVM está instalado de manera correcta se usara el comando de versión en la terminal de Ubuntu:

nvm –version

Para instalar NodeJS en su versión más reciente se ejecuta el comando nvm install node

para luego solicitar la versión instalada mediante el comando node -v como se muestra en la figura 55.

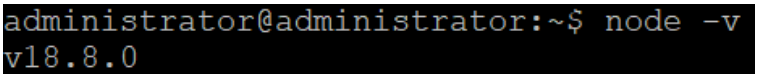


Figura 55. Vista NodeJS en el VPS

Ya con NodeJS corriendo satisfactoriamente en el servidor se instala PM2 el cual corre las aplicaciones escritas en NodeJS en segundo plano como un servicio que estará en línea 24/7 mediante el siguiente comando:

npm install pm2 -g.

Se necesita instalar la base de datos en MariaDB con PHPMyAdmin y para ello se usan los siguientes comandos empezando por actualizar los paquetes del servidor:

apt update -y

apt upgrade -y

Se instala el servidor Web Apache y se verifica en la dirección IP del VPS que esté funcionando observando el mensaje de la figura 56:

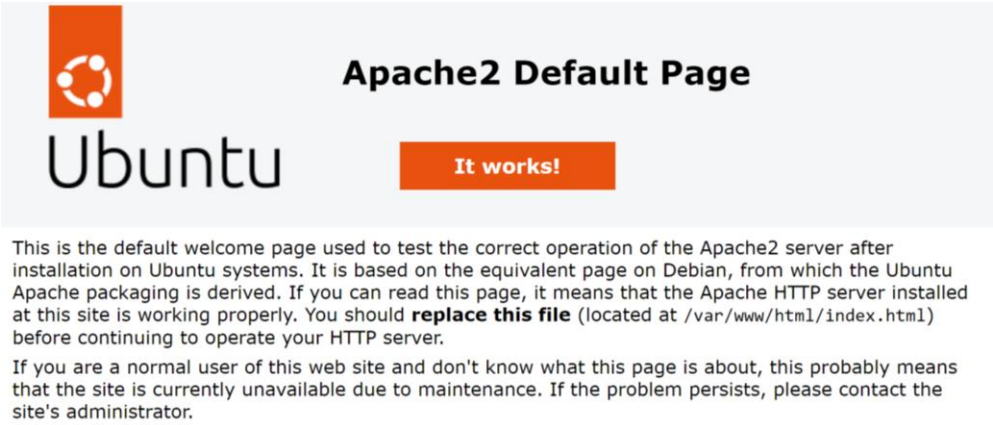


Figura 56. Apache corriendo en el servidor

Para instalar y configurar MariaDB se ejecutan los siguientes comandos:

apt install mariadb-server mariadb-client -y

mysql_secure_installation

Durante este proceso de instalación y configuración en la terminal se preguntará opciones de seguridad donde se responderán de la siguiente manera:

Set a root password? [Y/n]: y

Remove anonymous users? : y

Disallow root login remotely? : y

Remove test database and access to it? : y

Reload privilege tables now? : y

Ya terminado el proceso se verifica el funcionamiento de esta por medio del comando

“mysql -u root -p” terminal de la figura 57:

```
root@administrator:/home/administrator# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 217
Server version: 10.6.7-MariaDB-2ubuntu1.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Figura 57. Terminal corriendo MariaDB

Solo faltaría instalar PHP en el apache para tener acceso a la base de datos mediante PHPMyAdmin, ahora se procede a ejecutar de manera secuencial los siguientes comandos:

apt install php php-common php-mysql php-gd php-cli -y

apt install phpmyadmin -y

systemctl restart apache2

Una vez completado el proceso de instalar los paquetes se abre un navegador de internet y con la dirección ip del VPS en conjunto con "/phpmyadmin" y se muestra la página de login como se observa en la figura 58.

Figura 58. Inicio de sesión PHPMyAdmin instalado en el VPS

Es necesario crear un usuario para poder administrar la base de datos mediante PHPMyAdmin, este usuario se crea en la terminal de MariaDB mediante los siguientes comandos:

mysql -u root -p

CREATE USER ‘admin’@’localhost’ IDENTIFIED BY ‘PASSWORD

GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost';

FLUSH PRIVILEGES;

Una vez configurado el VPS se descarga el repositorio git mostrado en la figura 59 y se configuran las variables de entorno del proyecto:

Figura 59. Repositorio git del proyecto almacenado en github

4.2.5 Migración de la base de datos. En el desarrollo del proyecto se generaron copias de la base de datos como se muestra en la figura 60 las cuales son las versiones de la estructura del modelo, para migrar solo es necesaria la que tenga la última fecha de copia que se encuentra en el nombre del archivo.

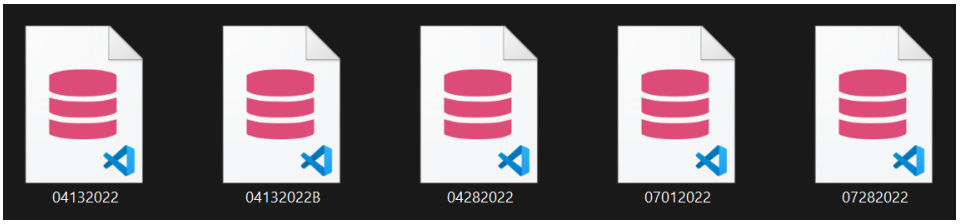


Figura 60. Archivos de las copias de la base de datos

Para importar la base de datos solo basta desde PHPMyAdmin en la opción de importar seleccionar el archivo y ejecutarlo como se observa en la figura 61.

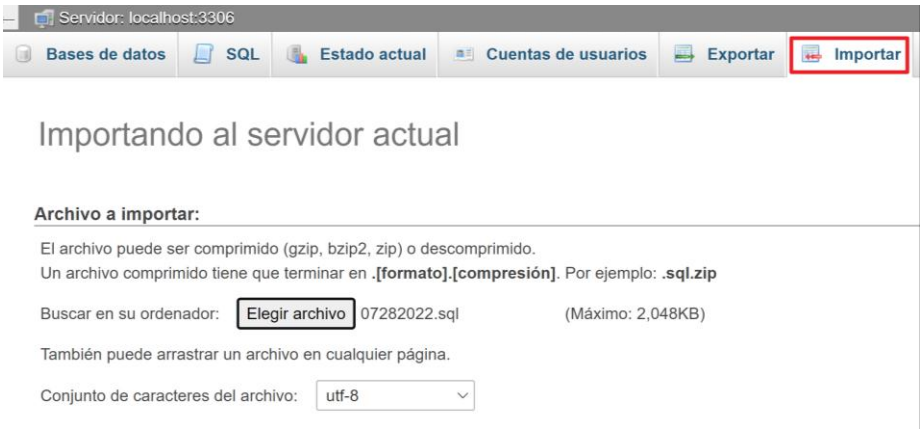


Figura 61. Importación del archivo copia de la base de datos en PHPMyAdmin

Para verificar que todo funcionó correctamente en el menú lateral izquierdo de PHPMyAdmin se deben observar que la base de datos hace parte de la lista como se muestra en la figura 62.

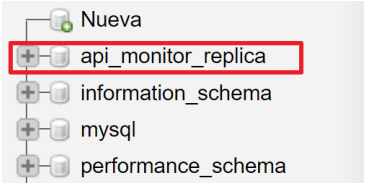
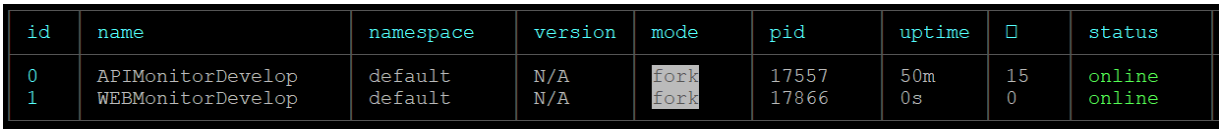


Figura 62. Base de datos de la aplicación mostrada en el PHPMyAdmin

Con la ayuda de PM2 se lanzan las aplicaciones en segundo plano y sé verifican que estén funcionando correctamente terminal de la, figura 63.



id	name	namespace	version	mode	pid	uptime	□	status
0	APIMonitorDevelop	default	N/A	fork	17557	50m	15	online
1	WEBMonitorDevelop	default	N/A	fork	17866	0s	0	online

Figura 63. Panel del PM2 corriendo las aplicaciones del servidor

4.3 Crear una Aplicación móvil Android para Obtener el Historial de las Máquinas y Recibir Notificaciones de Alerta

Una aplicación móvil es básicamente un cliente del API ejecutándose en el teléfono, donde se tienen que crear múltiples vistas para interactuar con los datos obtenidos por peticiones al servidor o acciones establecidas para cálculos que sean necesarios, además estas aplicaciones pueden tener datos almacenados tales como credenciales del usuario en una variable JSON Web TOKENS (JWT) o configuraciones del usuario.

Para el presente proyecto se crea una aplicación que consta de un Login y una Dashboard con navegación por pestañas entre clientes, servidores y maquinas. Esta aplicación se desarrolló en Flutter ya que permite ahorrar tiempo y recursos por su curva fácil de aprendizaje.

Para instalar Flutter en Windows 11 es necesario instalar Android Studio en conjunto a los paquetes de desarrollo necesarios por el Android SDK Manager aplicación como se observa en la figura 64, para saber el estado de la instalación de Flutter se ejecuta desde la terminal de Windows el comando “flutter doctor” como se muestra en la figura 65.

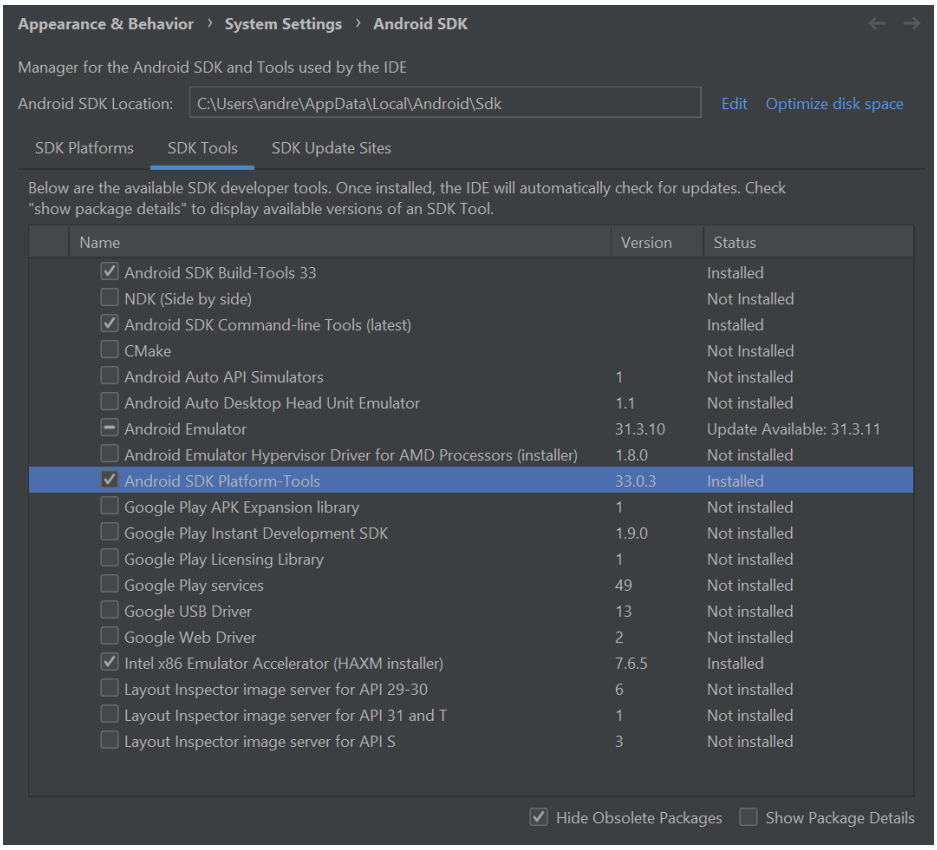


Figura 64. Android SDK Manager

```
C:\Users\andre>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.3.2, on Microsoft Windows [Version 10.0.22000.978], locale es-CO)
[√] Android toolchain - develop for Android devices (Android SDK version 33.0.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop for Windows (Visual Studio Community 2022 17.3.4)
[√] Android Studio (version 2021.2)
[√] VS Code (version 1.71.2)
[√] Connected device (3 available)
[√] HTTP Host Availability

• No issues found!
```

Figura 65. Flutter doctor para verificar la instalación de Flutter

Es necesario crear un emulador de Android para depurar las aplicaciones, con el Virtual Device Manager de Android Studio se creó una maquina con Android 11 que cuenta con los requisitos óptimos para este tipo de aplicación que son 1.5 GB de memora RAM, 10 GB de memoria ROM y una pantalla de 5.5 pulgadas como se muestra en la figura 66.

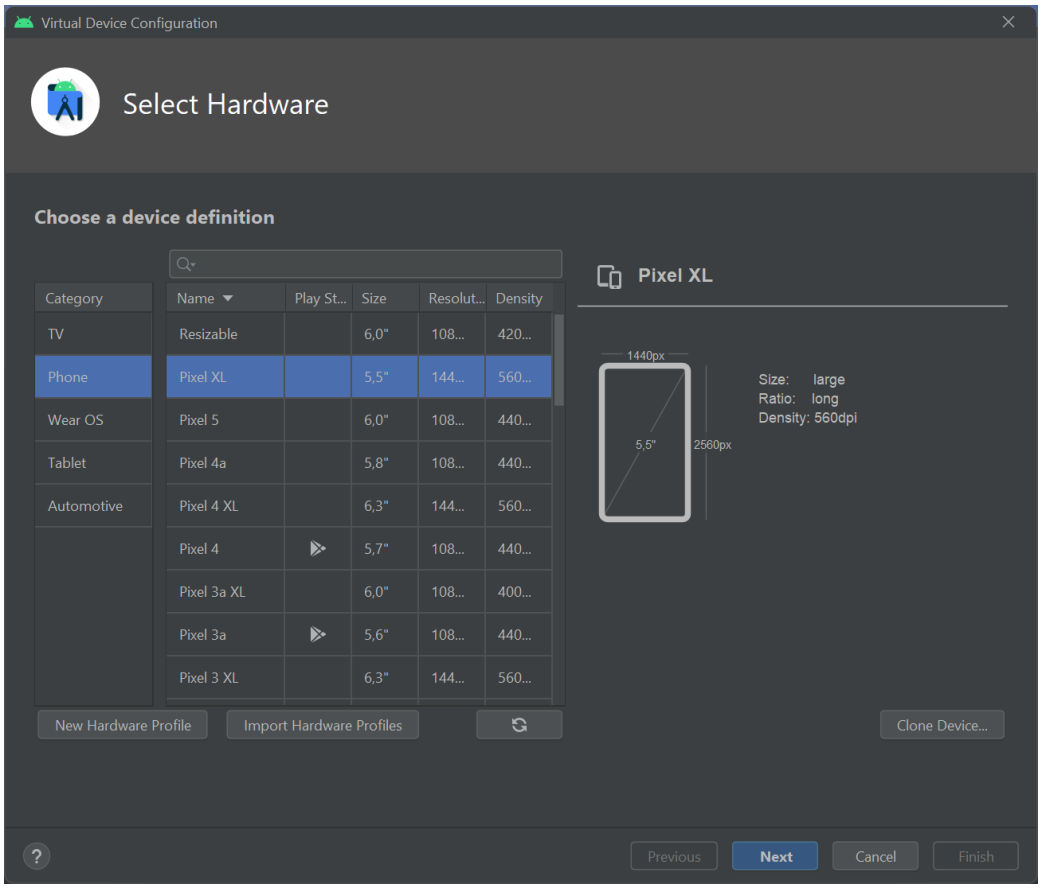


Figura 66. Creación máquina virtual Android con Virtual Device Manager

El entorno de desarrollo integrado (IDE) que se usó para el desarrollo de la aplicación Android es Visual Studio Code con el plugin verificado de Flutter como se muestra en la figura 67, en este se tienen opciones de creación, depuración e incluso de lanzar el emulador Android como se observa en la figura 68 para ser conectado mediante ADB.

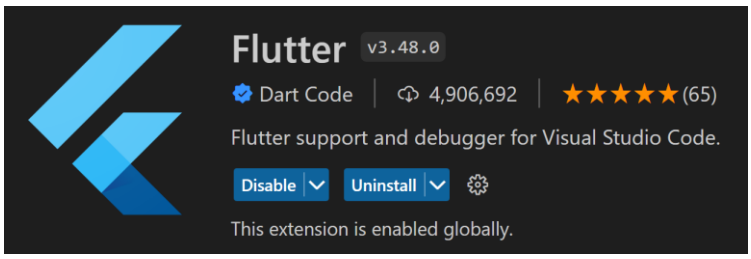


Figura 67. Plugin Flutter Visual Studio Code

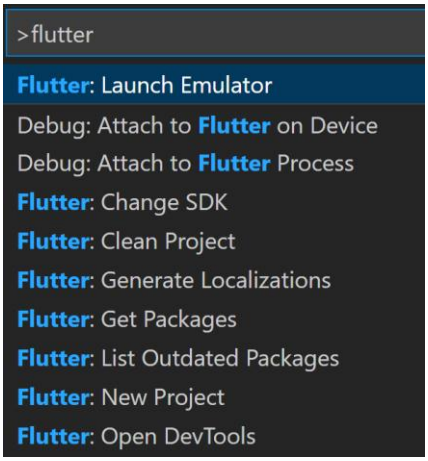


Figura 68. Opciones del plugin Flutter en Visual Studio Code

Al crear un proyecto en Flutter mediante Visual Studio Code se observan los archivos del proyecto como se muestra en la Figura 69 tales como pubspec.yaml el cual es el manejador de paquetes de Flutter denominados dependencias, donde se añaden las que necesarias para la aplicación como se observa en la Figura 70.

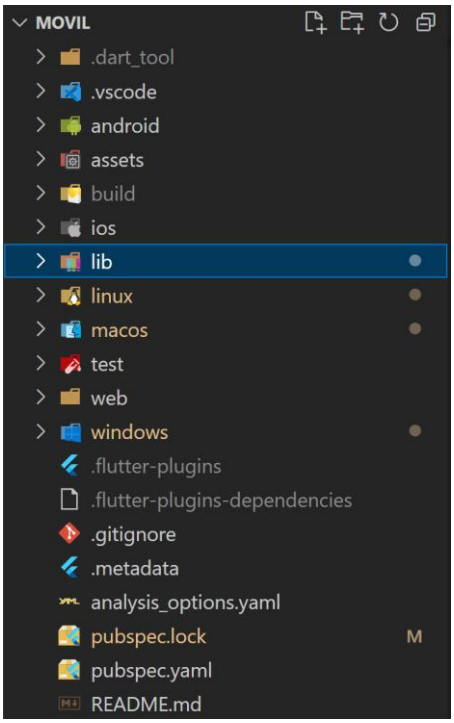


Figura 69. Proyecto Flutter en Visual Studio Code

Figura 70. Dependencias del proyecto

Las dependencias necesarias para este proyecto son:

- Cupertino icons: Soporte para usar vectores en formato SVG como iconos.
- Flutter svg: Librería para el cargue de los vectores SVG del logo.
- Http: Librería para peticiones HTTP para consumir los servicios del API.
- Shared preferences: Provee del acceso a la base de datos de preferencias de la aplicación que se usara para guardar el login JWT del usuario.
- Provider: Manejo de estados para los datos de la aplicación entre los componentes.
- Font Awesome Flutter: Librería de iconos para personalizar nuestra aplicación.

4.3.1 Diseño de las vistas aplicación móvil. Se crean 2 widgets de Flutter en la carpeta lib señaladas en la figura 71 que se le cargan al sistema de rutas en el archivo maint.dart observadas en la figura 72, uno de estos widgets para el login (LoginScreen) y el otro para la dashboard (HomeScreen) el cual cuenta con unas pestañas para el cambio de su contenido mediante vistas como se observa en la figura 73 que serían clientes (ClientView) , servidores (ServersView) y maquinas (MachinesView).

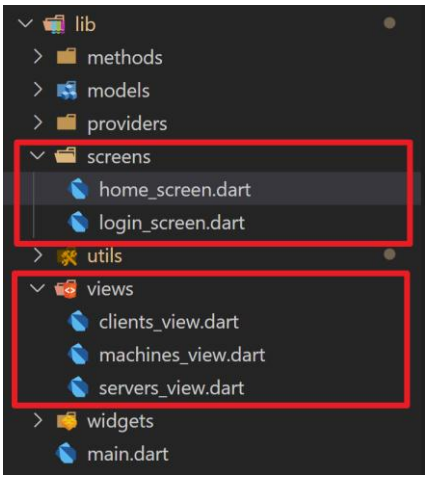


Figura 71. Archivo vistas Flutter en lib

```
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: Provider.of<ThemeProvider>(context).currentTheme,
      initialRoute: accessToken == null  ? 'login' : 'home',
      routes: {
        'login': (context) => const LoginScreen(),
        'home': (context) => const HomeScreen(),
      }
    );     // MaterialApp
  }
}
```

Figura 72. Rutas Flutter main.dart

```
body: const TabBarView(
  children: [
    ClientsView(),
    ServersView(),
    MachinesView(),
  ],
), // TabBarView
```

Figura 73. Vistas de las pestañas de la vista principal aplicación móvil

Para la vista de inicio de sesión de usuarios de la figura 74, se creó un formulario con 3 entradas, una de tipo texto correo, texto contraseña y un botón para enviar los datos con un

elemento GIF que crear el efecto de carga mientras se ejecutan las peticiones. Además de los elementos decorativos como el logo de la aplicación y la versión en su parte inferior con el nombre comercial de la empresa.

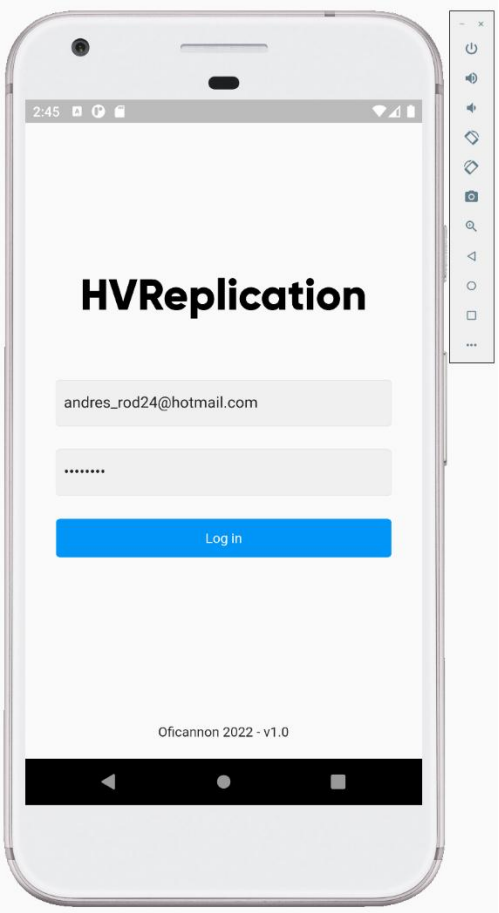


Figura 74. Vista login aplicación Android

La vista principal cuenta con un menú superior como se observa en la figura 75 con el nombre de la aplicación a su izquierda y a la derecha con la opción de cambiar el tipo de vista de clara a oscura como un valor agregado de la figura 76 además junto a esta se podrá hallar la opción para cerrar la sesión del usuario; inmediatamente debajo de este se encuentra una navegación por pestañas para cambiar entre clientes, servidores y máquinas virtuales (VMs).

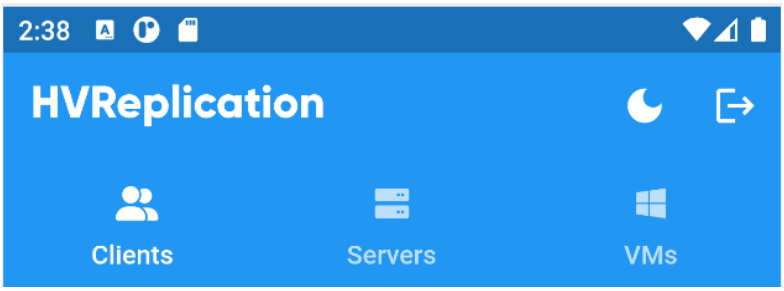


Figura 75. Superior vista principal aplicación Android

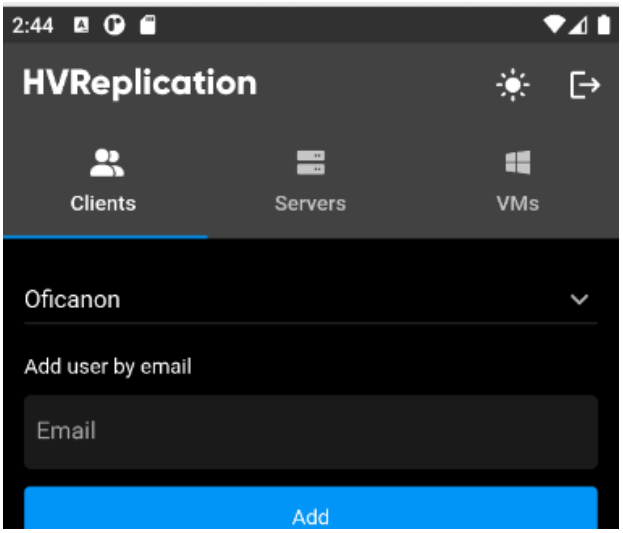


Figura 76. Modo oscuro

En la pestaña de cliente de la figura 77 se lista en un selector los diferentes registros donde el usuario pertenezca como se aprecia en la figura 78, dependiendo del cambio de este se actualizarán los listados de servidores y de máquinas virtuales, siendo las maquinas a listar relacionadas a el primer servidor en la lista y cuando este cliente no tenga servidores o maquinas no se mostrarán en el menú de navegación.

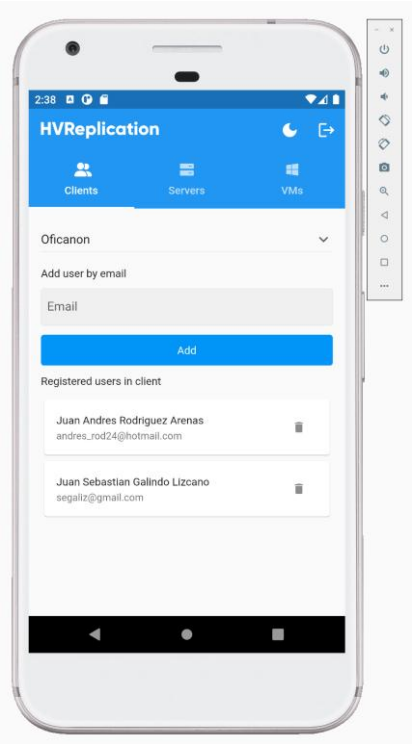


Figura 77. Vista de los clientes aplicación Android

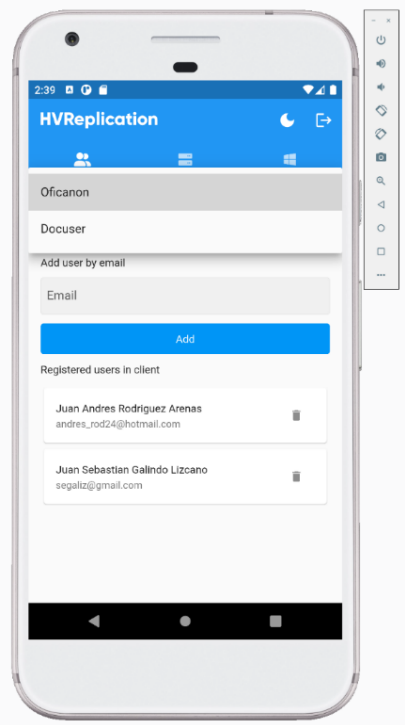


Figura 78. Cambio de cliente en el selector

Para agregar usuarios a los clientes estos deben estar previamente registrados en el sistema de monitoreo, con solo ingresar el correo electrónico en la entrada de agregar usuario y presionar en la opción de agregar este usuario quedara registrado en el cliente como se puede apreciar en la figura 79.

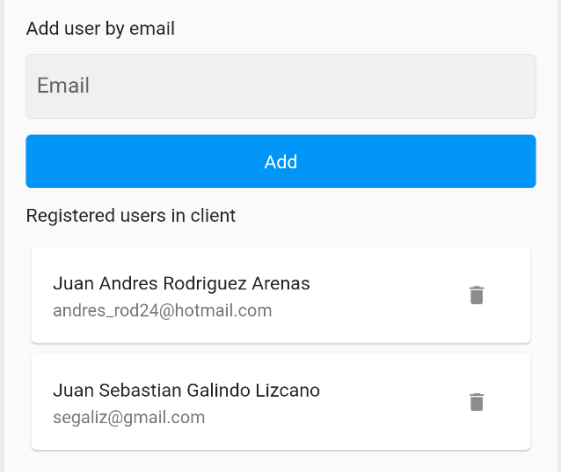


Figura 79. Entrada de usuarios para registrar en los clientes aplicación Android

Al igual que la vista de clientes se tiene un selector con el listado de servidores que pertenecen al cliente seleccionado en la pestaña clientes como se observa en la figura 80, se cuenta la opción de seleccionar los usuarios para registrar alarmas como se muestra en la figura 81.

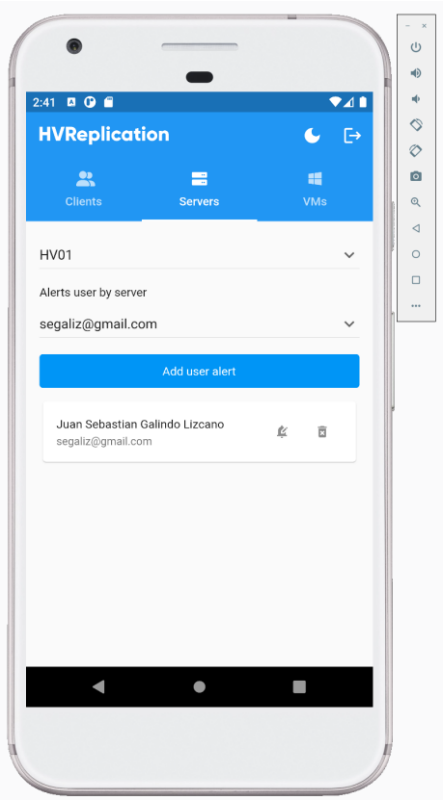


Figura 80. Vista de los servidores aplicación Android

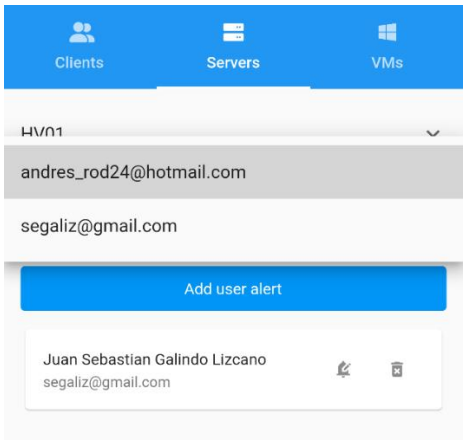


Figura 81. Seleccionador de usuario para crear la alarma

Cuando se crea una alarma esta por defecto viene con todas las opciones de notificación desactivadas, basta con solo seleccionar la edición de notificación para observar un menú flotante en el centro de la pantalla como se muestra en la figura 82 con una lista seleccionable con los 3

tipos de notificaciones, los 3 tipos de notificaciones son SMS, llamado y correo electrónico.

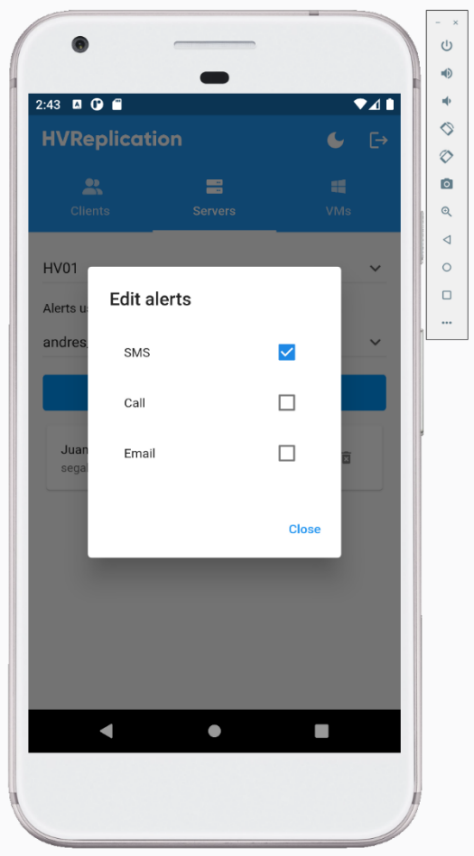


Figura 82. Editar alertas menú flotante

Por último, la vista de máquinas virtuales, esta vista al igual que la de servidores cuenta con un selector para cambiar la máquina virtual a modificar y un selector para crear las alarmas del usuario con su respetivo menú de edición flotante centrado en la pantalla, figura 83.

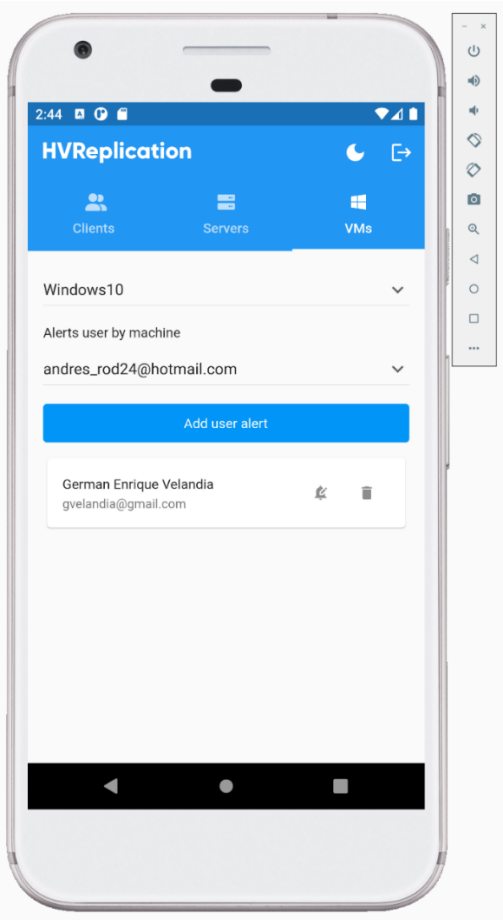


Figura 83. Vista de las máquinas virtuales en la aplicación Android

4.3.2 Programación consumo API. Para el manejo de datos se aplica un patrón "provider" en el cual guardan en memoria los datos con una estructura definida por un modelo y a medida que estos cambian un manejador que está escuchando el cambio modifica la zona o región del componente que uso los datos, los providers que se crearon son:

- User
- Servers
- Theme
- Config
- Machines

- Clients

Los modelos necesarios para el funcionamiento de la aplicación son:

- User {id, username, email, provider, confirmed, blocked, createdAt, updatedAt, firstname, lastname, phone, countrycode}
- Server {id, name, client, interval, machines, alerts, createdAt, updatedAt, publishedAt}
- Replication {id, health, machine, last, size, state, createdAt, updatedAt, publishedAt}
- Machine {id, name, server, interval, replication, monitor, alerts, createdAt, updatedAt, publishedAt}
- Client {id, name, createdAt, updatedAt, publishedAt, users, alerts}
- Alert {id, user, mail, sms, call, machine, server, createdAt, updatedAt, publishedAt}

4.4 Evaluar el Desempeño del Sistema de Monitoreo

A medida del avance del proyecto cada parte que conforma el sistema de monitoreo se fue probando con una pequeña aplicación que recreaba los datos obtenidos del sistema, esta aplicación enviaba mediante peticiones HTTP en intervalos aleatorios los datos que detonaran una alerta del sistema o una confirmación que esta funcionaba correctamente teniendo como resultado que cumplía con lo requerido, la empresa Patiño y Contreras S.A.S solicito como pruebas del sistema un conjunto de peticiones HTTP en Postman y un sistema en físico de 2 Hipervisores con la replicación y monitoreo configurado en una cuenta de prueba.

4.4.1 Elaboración pruebas del sistema. La empresa Patiño y Contreras se encargó de instalar el agente monitor en 2 de sus servidores de pruebas los cuales cuentan con 2 distribuciones de Windows Servers diferentes, observaron que el comportamiento de este agente es óptimo e incluso resaltaron el poco consumo que este ocupaba en la máquina.

Para probar el API se cuenta con la colaboración del ingeniero Juan Sebastián Galindo codirector del proyecto y encargado del área de desarrollo, con el cual se crea un espacio de trabajo en Postman con todas las peticiones que se podrían ejecutar en el API, se ordenan las peticiones según el modelo creado en el Strapi mediante un conjunto de carpetas de manera ordenada como en la figura 84.

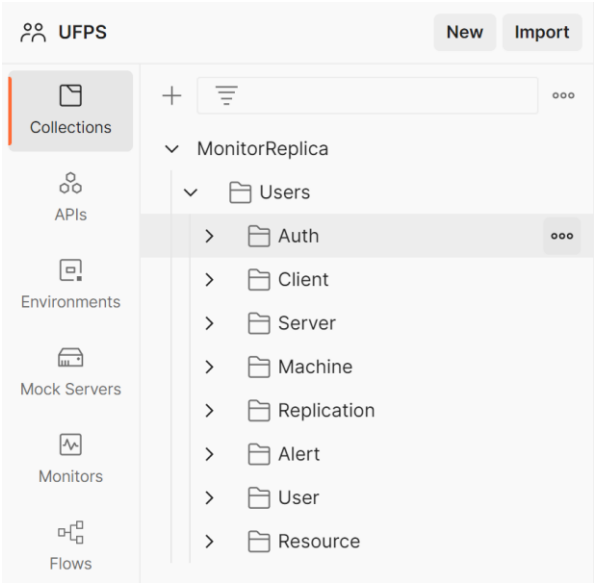


Figura 84. Espacio de trabajo en Postman

Para las peticiones que requieren un token de autenticación Postman permite agregar una cabecera de autenticación, en este proyecto se optó por ponerle el token a la carpeta padre ya que casi todas las peticiones que se realizan al Api son con un usuario con sesión iniciada a excepción de la autenticación, basta con solo seleccionar la carpeta padre y en la pestaña de Autorización seleccionar el tipo como un “Bearer Token” y agregarlo en el capo de texto token, figura 85.

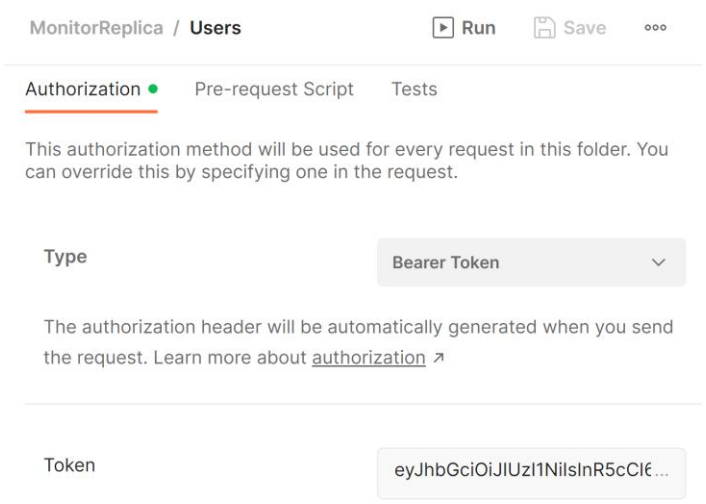


Figura 85. Configuración token autenticación Postman

Las peticiones HTTP que se crearon en el espacio de trabajo en Postman son:

- Auth/Login (POST): {identifier, password}
- Auth/Me (GET) Header Token
- Client/Create (POST) Header Token: {name}
- Client/Details/ (GET) Header Token: id
- Client/All (GET) Header Token: {filters: {name}}
- Client/Update (PUT) Header Token: id, {client, name}
- Client/AddUserByMail (POST) Header Token: {client, email}
- Server/Create (POST) Header Token: {client, name}
- Server/Details (GET) Header Token: id, {filters: {client, name}}
- Server/Get (GET) Header Token: id, {filters: {client}}
- Server/Update (PUT) Header Token: id, {client, name}
- Machine /Create (POST) Header Token: {client, server, name}
- Machine /Details (GET) Header Token: id, {filters: {client, name}}

- Machine /Get (GET) Header Token: id, {server, filters: {client}}
- Machine/Update (PUT) Header Token: id, {client, server, name, replication, monitor}
- Replication /Create (POST) Header Token: {health, state, last, size, latency, machine, server, client}
- Replication /All machine (GET) Header Token: id, {filters: {client, server, machine}}
- Alert /UpdateServer (PUT) Header Token: {mail, sms, call, enable, client, server, machine}
- Alert /ByMachine (GET) Header Token: id, {filters: {client, server, machine}}
- Alert /ByServer (GET) Header Token: id, {filters: {client, server}}
- Alert /Create (POST) Header Token: {mail, sms, call, enable, client, server, machine}
- Alert /Delete (DELETE) Header Token: id, {filters: {client, name}}

Al ejecutar cada petición se observa que el tiempo de respuesta no es constante ya que este depende de la red donde se transporte y el estado de la maquina donde se ejecute el API, pero estaba entre lo normal de procesamiento y entrega del mismo.

En las pruebas de la aplicación Web Nuxt el ingeniero Juan Sebastián Galindo dejó corriendo un Script en PuppeteerJS que a lo largo de una semana abrió y cerró sesión en múltiples ocasiones en intervalos aleatorios de tiempo, además de emular un internet lento y negación por toda la aplicación. Se observa que la aplicación respondió bien sin presentar algún error o falla en estructura y datos traídos desde el API.

4.4.2 Socialización resultados del sistema con la empresa Patiño y contreras SAS. Por medio de una reunión con el equipo técnico encargado de los servicios relacionados, se explicó la instalación, el funcionamiento y mantenimiento del sistema de monitor en máquinas virtuales Hyper-V.

5. Conclusiones

El sistema de monitoreo le suma credibilidad a el servicio ofrecido por Patiño y Contreras S.A.S, ya que este permite saber si ocurre algún inconveniente en la copia de seguridad de las máquinas virtuales de sus clientes en un tiempo prudente, el cual puede variar según el tráfico de la red de mensajería de correo electrónico o móvil, se observó en las pruebas del sistema donde se crearon 50 alertas a diferentes intervalos de tiempos y con varios tipos de notificación tales como SMS, llamadas o correos electrónicos a múltiples usuarios creados con teléfonos de diferentes operadores de telefonía móvil como Claro, Movistar o Tigo y correos de varios proveedores tales como Gmail, Outlook y Zoho.

Se observó que el tiempo de entrega de la notificación es de 15 a 120 segundos en un estado óptimo de la red, recalcando que en producción este tipo de notificación depende de los servicios de terceros tales como los proveedores de mail que usen los clientes o los operadores de telefonía móvil lo cual no garantiza que estos tiempos sean valores exactos, se determinó que él envió de la petición a Plivo para enviar llamadas o mensajes de texto fue de inmediata al igual que el correo electrónico donde además se deben tener cuenta valores como la latencia de la red de donde se esté haciendo el servicio, lo cual en el momento de las pruebas se garantizó que llegaban las notificaciones evitando así que crezca un fallo que conlleve a la paralización de un negocio o perdida de información de este, se pudo observar que en la práctica es más fiable la llamada ya que esta no puede ser ignorada con facilidad, además por costos es recomendable usar la llamada debido a que el mensaje de texto tendría el mismo valor, y cabe señalar que los correos electrónicos son vulnerables a recibir mucho contenido que en su mayoría es considerado basura y ocultaría la alerta.

Con respecto al desarrollo, sé determino que un sistema desacoplado en pequeñas aplicaciones permite a futuro mejoras o correcciones sin depender de un solo trabajador o herramienta informática, funcionando esta manera más rápido y de fácil acceso por su escalabilidad, durante cada una de las etapas de este proyecto se usaron diferentes tecnologías de las cuales el estudiante y el equipo de Patiño y Contreras S.A.S aprendieron para ser usadas a futuro en su vida profesional.

El marco de desarrollo Strapi que usado en el proyecto permitió el diseño del Api de manera muy rápida, permitiendo además ser un manejador de contenido que se usara para el registro de los usuarios, además del paquete de desarrollo de aplicaciones Web NUXT que permitió crear un cliente Web reactivo con estados y de manera muy ordenada.

Gracias al marco de desarrollo Strapi se creó un Api de manera rápida y sencilla, proveyendo del manejador de contenido y administrador de usuarios, además con el paquete de desarrollo de aplicaciones web NUXT se creó un cliente web reactivo VUE con estados dinámicos VUEX de manera ordenada.

En el desarrollo Android con Flutter se creó una aplicación limitada por el entorno de su ejecución que poseía componentes limitados, los objetivos de esta aplicación fueron cumplidos en su totalidad, además de que esta se diseñó de manera que en una futura actualización esta sea escalable.

6. Recomendaciones

El sistema de monitoreo de máquinas virtuales en Hyper-V tiene como necesidad para poder notificar el poseer un saldo en la cuenta Plivo, evitando que esta llegue a 0 por ser prepagado para poder enviar mensajes de texto y llamadas, además se debe estar pendiente de que el sistema este levantado ya sea por una caída del VPS donde este hospedado o por la renovación del dominio al cual sea registrado.

Se recomienda estar pendientes de las actualizaciones del API por Strapi o del marco de desarrollo de aplicaciones móviles Flutter para futuras mejoras de este, además el de corregir fallas que no fueron detectadas durante este desarrollo.

Referencias Bibliográficas

Anchuelo, J. (2021). *Diseño e implementación* de *sistema de monitorización de máquinas virtuales.* Tesis de grado. Universidad de Alcalá. Alcalá, España.

Congreso de Colombia. (2008). *Ley 1266 de 2008. Por la cual se dictan las disposiciones generales del hábeas data y se regula el manejo de la información contenida en bases de datos personales, en especial la financiera, crediticia, comercial, de servicios y la proveniente de terceros países y se dictan otras disposiciones.* Bogotá: Diario Oficial No. 47.793

Congreso de Colombia. (2019). *Ley 1951 de 2019.* Por la cual crea el Ministerio de Ciencia, Tecnología e Innovación, se fortalece el Sistema Nacional de Ciencia, Tecnología e Innovación y se dictan otras disposiciones. Bogotá: Diario Oficial No. 50.846.

Costas, J. (2014). *Seguridad y Alta Disponibilidad.* Madrid, España: RA-MA, S.A Editorial y Publicaciones.

Garzón, C. (2021). *Implementación de una herramienta de monitoreo sobre la emulación de bienes de infraestructura tecnológica en una red LAN.* Tesis de pregrado. Universidad de Antioquia. Medellín, Colombia.

Gómez, P. (2006). Máquinas Virtuales en las clases de Informática. *Jornadas de Enseñanza Universitaria de la Informática,4*(2), 1-15.

Google Developers. (2020, 05 07). *Descripción general de la compatibilidad de dispositivos.* Recuperado de: https://developer.android.com/guide/practices/compatibility?hl=es-419

Google. (2021). *Dart programming language.* Recuperado de: https://dart.dev/

Google. (2021). *Flutter - Build apps for any screen*. Recuperado de: https://flutter.dev/

Google. (2021). *Qué es Android*. Recuperado de: https://www.android.com/intl/es_es/what-is-android/

Hernández, W. (2017). *Implementación de arquitectura de micro* servicios *utilizando virtualización por sistema operativo.* Guatemala: Universidad de San Carlos de Guatemala.

Hostingred. (2021). *¿Qué es un VPS?* Recuperado de: https://www.hostingred.com/servidores/informacion-servidores-virtuales-vps/

Hyper-V on Windows. (2019). *Microsoft*. Recuperado de: https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows /

Jayaseelan, G. & P. Charles, J. (2014, 03 01). Automated Secured Disaster Recovery with Hyper-V Replica and PowerShell. *Congreso Mundial sobre Tecnologías Informáticas y de Comunicación*, 2(1), 150-153. Recuperado de: https://ieeexplore.ieee.org/abstract/document/6755125

Luján, S. (2002). *Programación de aplicaciones Web: historia, principios básicos y clientes Web.* Madrid: San Vicente del Raspeig,

Manage Engine OpManager (2022). *Monitoreo de Microsoft Hyper V.* Recuperado de: https://www.manageengine.com/latam/network-monitoring/software-monitoreo-hyper-v.html#:~:text=La%20funci%C3%B3n%20de%20monitoreo%20de,usando%20plantillas%20exclusivas%20para%20dispositivos

MDN Contributors. (2021). *Soporte a navegadores antiguos.* Recuperado de: https://developer.mozilla.org/es/docs/Learn/CSS/CSS_layout/Supporting_Older_Browsers

Medina, V. (2013). *Alta disponibilidad en máquinas virtuales con un esquema de replicación de servicios TCP/IP*. Tesis de doctorado. Universidad Michoacana de San Nicolas de Hidalgo. Hidalgo, Mexico.

Microsoft Corporation. (2018). *Hyper-V Architecture*. Recuperado de: https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/reference/hyper-v-architecture

Microsoft Corporation. (2021). *What is PowerShell?* Recuperado de: https://docs.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.2

Nemnom, C. (2022, 01 24). *Advanced Hyper-V Replica Monitoring via #PowerShell #HyperV.* Recuperado de: https://charbelnemnom.com/advanced-hyper-v-replica-monitoring-via-powershell-ws12r2-hyperv-sapien/#How_to_use_it

OpenJS Foundation. (2021). *Node.js*. Recuperado de:://nodejs.org/es/

Plivo. (2021a). *Plivo SMS.* Recuperado de: https://www.plivo.com/docs/sms

Plivo. (2021b). *Plivo voice.* Recuperado de: https://www.plivo.com/docs/voice/

Postman, Inc. (2021). *Postman API Platform.* Recuperado de: https://www.postman.com/

Ramirez, H. (2021). *Control de acceso basado en roles (RBAC): Una forma de mejorar la seguridad del sistema*. Recuperado de: https://protecciondatos-lopd.com/empresas/control-de-acceso-basado-en-roles-rbac/

Red Hat. (2020). *¿Qué es una API de REST?* Recuperado de:

https://www.redhat.com/es/topics/api/what-is-a-rest-api

Sistema Unico de Informacion Normativa. (n.d.). *Ley 16 de 1968.* ecuperado de:

https://www.suin-juriscol.gov.co/viewDocument.asp?id=1809639

Strapi. (2021). *Strapi - Open source Node.js Headless CMS*. ecuperado de: https://strapi.io/

The Institute of Computer Science. (2020). *Monitoring, system performance, Performance Counters, Zabbix, virtualization, Hyper-V.* Polonia: FEIT/ICS.

Wikipedia. (2021). *Sistema de gestión de contenidos.* Recuperado de:

https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_contenidos