

Memoria final grupo 2

10/02/2026

El objetivo de esta memoria es documentar todas las fases del proyecto, desde la planeación inicial y organización, hasta el diseño final y la recopilación de errores.

Contenido

Memoria final grupo 2.....	0
Información del equipo y roles	2
Nombre del equipo/Empresa	2
Miembros del equipo y roles	2
Introducción al PITCH.....	3
Propuesta final.....	3
Entrega de documentación final.....	4
Estructura lógica del proyecto	5
Como levantamos la infraestructura (ASO)	6
Balanceador (SAD)	7
Auto escalado (SAD).....	8
RDS(ASGBD).....	8
RDS Proxy (SAD)	10
Securización del entorno (SAD, IAW)	11
Entorno WEB (IAW, SRI)	15
Área paciente	16
Acceso	16
Funciones.....	16
Área Medico	19
Acceso	19
Funciones.....	19
Área Administrativo.....	20
Acceso	20
Funciones.....	21
App móvil	23
Costos y comparaciones	23
Gestión del proyecto	25
Trello.....	25
GitHub	¡Error! Marcador no definido.

Teams	26
Errores	26
Limitaciones de AWS	26
Incompatibilidad de base de datos	27
Problema de sticky session	27
Fallo de DNS	27
Acceso al dashboard de monitorización	27
Manual de mantenimiento	28

Información del equipo y roles

Nombre del equipo/Empresa

Nuestro grupo se llama **Health2you** esto debido a que nuestra empresa se centra en el sector médico, aparte que pensamos que el nombre sería el más adecuado al ser el grupo 2.

Miembros del equipo y roles

Valentín Gutiérrez: Coordinador del equipo, encargado de la comunicación con docentes y el cumplimiento de los hitos.

Daniel Lorenzo: Responsable de infraestructura, encargado del diseño de la VPC, despliegue de servidores y servicios web

Raquel Rodríguez: Responsable de ciberseguridad y automatización, scripting, segmentación de red y monitorización

Samuel Calleja: Responsable de documentación y viabilidad, memoria técnica, análisis de costes y soporte visual del Pitch.

Durante la mayor parte del reto hemos mantenido estos roles y en base a ellos hemos ido asignando las tareas acordes a ellos, pero debido a evitar el que los miembros del equipo se cansen de su parte y parte de que todos los miembros del grupo deban tener conocimiento sobre todas las partes del grupo, los roles se fueron cambiando.

Durante la última semana se cambió la asignación de tareas de Raquel, que comenzó a realizar tareas de seguridad y doble factor de autenticación, Daniel realizó la VPN y pruebas de caja negra, Samuel modificó el código de infraestructura y siguió documentando y Valentin pasó a realizar cambios en la página web.

Introducción al PITCH

El objetivo de entrega del pitch final constará del proyecto terminado, con todos sus apartados terminados y documentados en esta memoria, también se tendrán en cuenta errores sucedidos a lo largo de proyecto y proceso tanto de las tareas realizadas como de la planeación que hemos llevado a cabo para facilitar nuestro progreso.

Propuesta final

El trabajo final como se ha documentado desde el principio lo hemos llamado **Health2you**, dicho de forma sencilla, es una Web App que permite a sus usuarios registrados solicitar cita médica de diferentes maneras y a los médicos ver las citas que les han solicitado los pacientes.



El proyecto consta tanto de la web WordPress como de la infraestructura y configuraciones necesarias para levantar el servicio, para esta hemos optado por levantar toda la infraestructura en la nube, concretamente con los servicios de **AWS**, esto por el ahorro económico, la opción de automatizar ciertas tareas como las copias de seguridad y la disponibilidad de sus propios servicios para mejorar la calidad del proyecto.

Entrega de documentación final

Para comenzar a hablar sobre nuestro proyecto tenemos que presentar el dónde lo hemos alojado y por qué, nuestro trabajo está totalmente alojado en la nube, concretamente utilizando **AWS**.

Esto lo hicimos puesto a las ventajas que ofrecía frente a utilizar servidores on premises, concretamente una mayor viabilidad económica en un caso real de uso, la mejor capacidad de automatización de tareas, como puede ser la propia infraestructura como código y la gran oferta de servicios que ofrece para ofrecer una mejor calidad de producto al cliente.

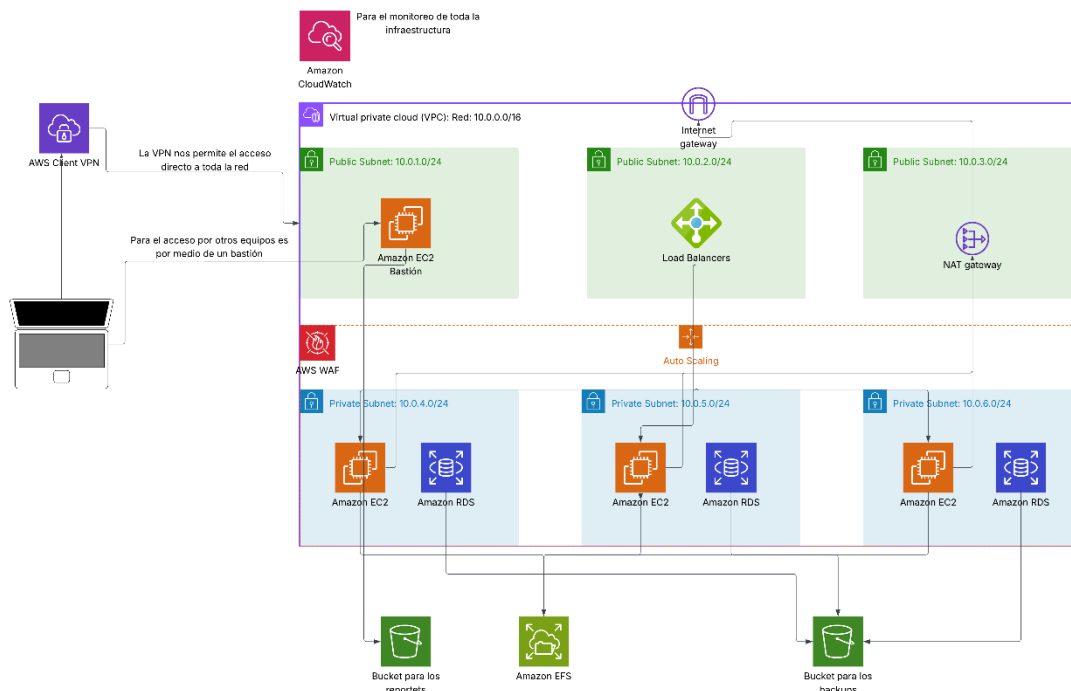


Como limitaciones encontradas al utilizar **AWS** sería el uso de la cuenta **AWS Academy**, la cual nos limita una serie de servicios que nos abrían facilitado el proyecto, como pueden ser algunos motores de base de datos y el gasto de dólares de los laboratorios, por el cual hemos tenido que racionar el uso de los laboratorios y reiniciarlos una vez acabado el día para no gastar de más.

Estos errores están mejor documentados en el apartado de errores del documento junto a las alternativas que hemos aplicado.

Estructura lógica del proyecto

En la siguiente imagen se visualiza de forma sencilla la estructura de nuestro despliegue en la nube dentro de una VPC.

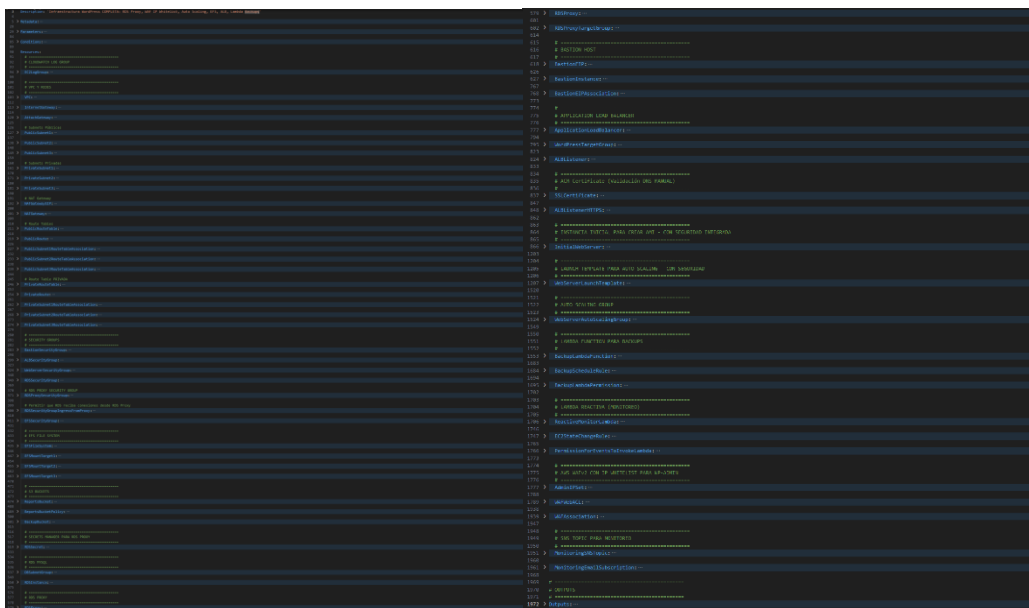


Como podemos observar tenemos 3 subredes tanto públicas como privadas en, las públicas tenemos un bastión que utilizamos para monitorizar y ejecutar scripts luego en la subnet pública 2 tenemos el balanceador de carga principal desde el accedemos a las instancias web y equilibramos el tráfico web, en la parte privada establecemos una RDS que por causas externas (Limitaciones de la cuenta la cuenta academy de AWS) no se ha podido crear de forma distribuida aunque si el proyecto tuviera acceso a los recursos necesarios la estructura lo soportaría ya que está planteada con ello en mente, por otro lado tenemos las instancias todas ellas están dentro de un grupo de auto escalado en el que más adelante profundizaremos, cada instancia tiene como base Ubuntu24.04, para poder hacer funcionar la web (WordPress) hemos creado un volumen EFS al que tienen acceso todas las instancias en él se encuentra todo el contenido referente al funcionamiento de la web, para monitoreo y control de toda esta infraestructura hemos establecido una serie de políticas de filtrado, por ultimo fuera de la VPC tenemos 2 buckets, uno para almacenamiento de backups y otro para almacenar los reportes generados por la monitorización.



Como levantamos la infraestructura (ASO)

Toda nuestra infraestructura esta [levantada como código \(IAC\)](#) esto nos facilita el trabajo a la hora de realizar pequeños cambios en caliente o nuevas implementaciones ya que todos los miembros del equipo parten desde el mismo punto, gracias a esto el flujo de trabajo ha sido el siguiente, un compañero buscaba y documentaba como implementar una función y cuando ya conocíamos cómo funcionaba y como configurarlo lo implementamos sobre el IAC y así hemos conseguido que toda la infraestructura vista en el esquema anterior se levante por completo. Además, esto reduce costes al poder reiniciar todo el laboratorio y lanzarlo de nuevo en unos minutos.

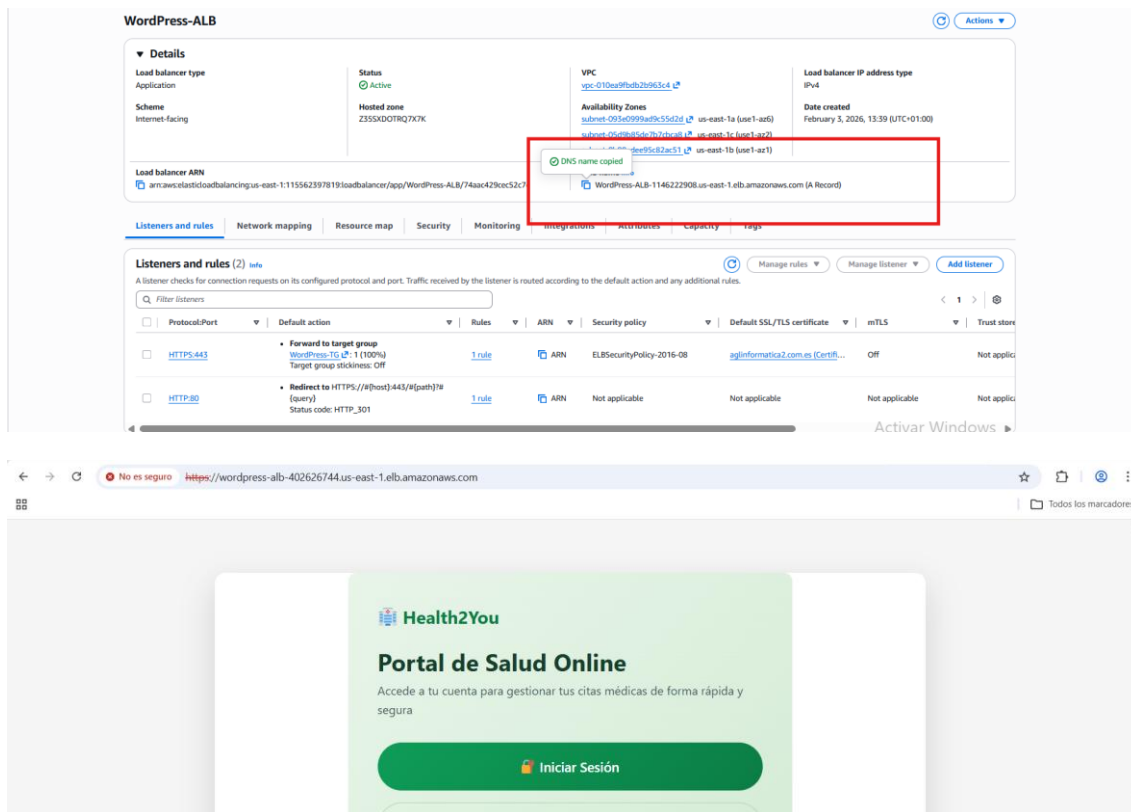


Para poder utilizar el código requerimos configurar una serie de parámetros definidos al comienzo del archivo como el nombre del stack, un correo y una ip o varias para filtrado que se utilizara en un “ipset” de WAF. Por otro lado, un gran inconveniente es que de nuevo limitados por los permisos de nuestra cuenta nos vemos obligados a tener que usar un rol por defecto llamado “LabRole” esto nos permite lanzar el código, aunque lo óptimo en condiciones reales seria crear un rol dedicado para ello.

Balanceador (SAD)

Como hemos explicado, nuestra infraestructura utiliza un balanceador de carga encargado de gestionar las instancias dentro de las subredes privadas, esto nos ofrece una capa extra de seguridad puesto que, a la web al acceder por el balanceador, las instancias no están expuestas a internet, aparte que nos da una mayor disponibilidad al repartir el tráfico entre las 3 subredes.

Aquí mostramos el uso del ALB del balanceador para acceder desde el navegador a la web.



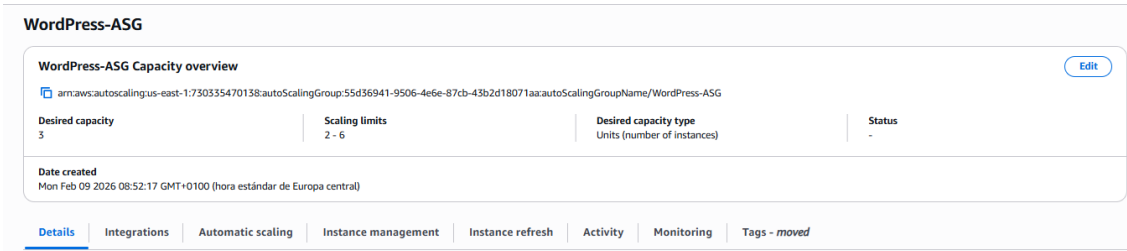
The image shows two screenshots. The top screenshot is from the AWS Management Console, displaying the configuration for a 'WordPress-ALB'. The 'Details' tab is active, showing the load balancer is 'Active' and has an 'Internet-facing' scheme. A red box highlights the 'Availability Zones' section, which lists three subnets in the 'us-east-1a' and 'us-east-1b' availability zones. Below this, the 'Listeners and rules' tab is visible, showing two listeners: 'HTTPS-443' and 'HTTP-80'. The 'HTTPS-443' listener has a rule that 'Forward to target group' (WordPress-TG) with a weight of 1 (100%). The bottom screenshot shows a web browser at the URL 'https://wordpress-alb-402626744.us-east-1.elb.amazonaws.com'. The page displays the 'Health2You' logo and the title 'Portal de Salud Online'. Below the title, it says 'Accede a tu cuenta para gestionar tus citas médicas de forma rápida y segura' and features a green button labeled 'Iniciar Sesión'.

Mientras lo configurábamos nos comenzó a dar un error relacionado a la opción de sticky session del grupo objetivo del balanceador el cual este documentado en el apartado de errores.

Auto escalado (SAD)

Junto al balanceador de carga configuramos además un auto escalado, el cual en caso de recibir tanto poco tráfico como un tráfico superior al esperado, este permite la eliminación o creación de nuevas instancias clonadas para mantener el servicio disponible, o en caso de recibir poco tráfico y se elimine una subred, ahorrar el costo de esta.

Como podemos ver en la imagen, tenemos configurado el auto escalado para que el mínimo de subredes sea 2 y el máximo sea 6.



WordPress-ASG

WordPress-ASG Capacity overview [Edit](#)

arn:aws:autoscaling:us-east-1:730335470138:autoScalingGroup:55d36941-9506-4e6e-87cb-43b2d18071aa:autoScalingGroupName/WordPress-ASG

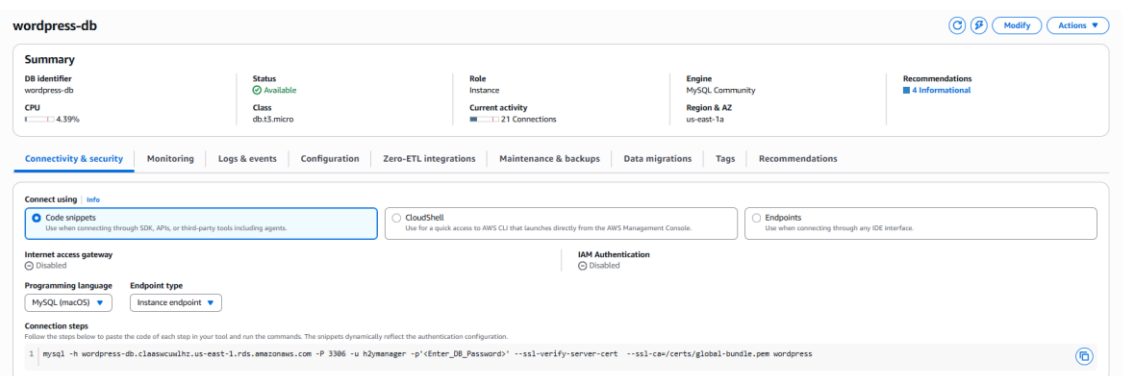
Desired capacity 3	Scaling limits 2 - 6	Desired capacity type Units (number of instances)	Status -
------------------------------	--------------------------------	-------------------------------------------------------------	--------------------

Date created
Mon Feb 09 2026 08:52:17 GMT+0100 (hora estándar de Europa central)

[Details](#) | [Integrations](#) | [Automatic scaling](#) | [Instance management](#) | [Instance refresh](#) | [Activity](#) | [Monitoring](#) | [Tags - moved](#)

RDS(ASGBD)

Como hemos explicado, nuestra base de datos utiliza como motor **MySQL**, está la tenemos alojada dentro de una **RDS**, para gestionar el acceso a este utilizamos sus credenciales dentro de los ficheros de configuración de las instancias, en nuestro caso dentro de los ficheros **wp-config.php**.



wordpress-db [Copy](#) [P](#) [Modify](#) [Actions](#)

Summary

DB identifier wordpress-db	Status Available	Role Instance	Engine MySQL Community	Recommendations 4 informational
CPU 4.39%	Class db.t3.micro	Current activity 21 Connections	Region & AZ us-east-1a	

[Connectivity & security](#) | [Monitoring](#) | [Logs & events](#) | [Configuration](#) | [Zero-ETL integrations](#) | [Maintenance & backups](#) | [Data migrations](#) | [Tags](#) | [Recommendations](#)

Connect using [Info](#)

☒ **Code snippets**
Use when connecting through SDK, API, or third-party tools including agents.

☐ **CloudShell**
Use for a quick access to AWS CLI that launches directly from the AWS Management Console.

☐ **Endpoints**
Use when connecting through any IDE interface.

Internet access gateway
☐ Disabled

IAM Authentication
☐ Disabled

Programming language
MySQL (macOS)

Endpoint type
Instance endpoint

Connection steps
Follow the steps below to paste the code of each step in your tool and run the commands. The snippets dynamically reflect the authentication configuration.

```
1 mysql -h wordpress-db.classicurl.us-east-1-rds.amazonaws.com -P 3306 -u h2ymanager -p'Enter_DB_Password' --ssl-verify-server-cert --ssl-ca=/certs/global-bundle.pem wordpress
```

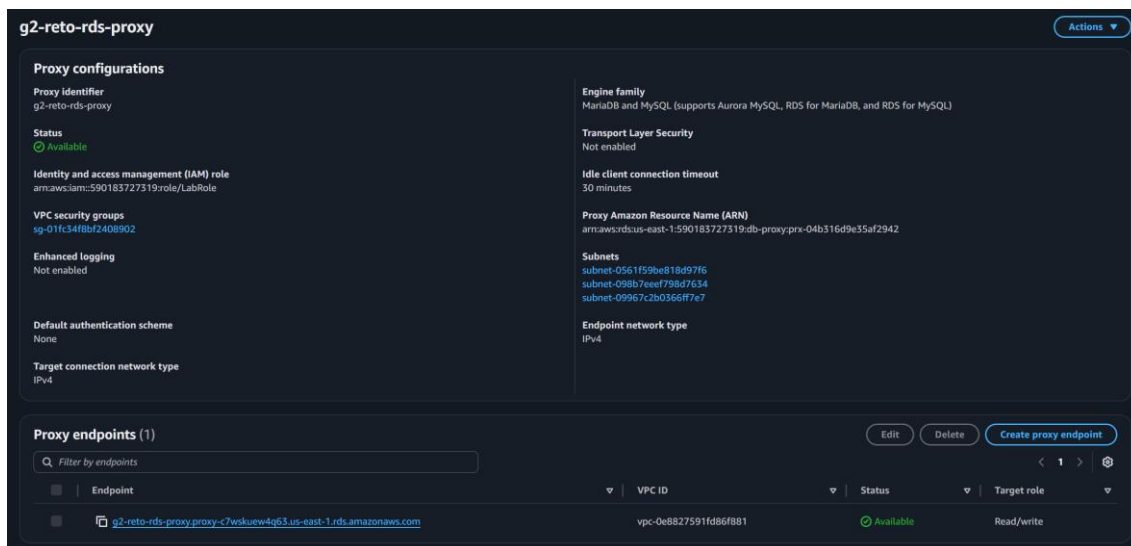


Como otras funciones a destacar un trigger que detecta el estado de las citas, una vez estas sean marcadas como atendidas por los médicos el trigger permite que se pueda tanto descargar como enviar el justificante de la cita.

Durante el proceso del proyecto la base de datos fue sufriendo modificaciones, puesto que comenzamos con una base de datos que solo guardaba los usuarios, médicos y las citas, fuimos agregando nuevos datos a lo largo que añadíamos servicios dentro de la web.

RDS Proxy (SAD)

Para cumplir el requisito de la implementación de un proxy nos hemos decidido por la implementación de un RDS proxy que hace de intermediario entre nuestra aplicación y la base de datos gestionando nuestras conexiones de forma eficiente, el Proxy mantiene un grupo de conexiones abiertas y las "presta" a las solicitudes según las necesitan. Esto ahorra muchísimo tiempo y recursos. Además, al usar como punto de conexión el endpoint del proxy y no de la base de datos en caso de un fallo como el proxy tiene acceso a la base de datos de las demás subredes puede redirigir a una instancia activa sin tener que cambiar el punto de acceso en las maquinas lo que refuerza la alta disponibilidad de nuestra estructura.



The screenshot displays the AWS Management Console for an RDS Proxy instance. The instance is named 'g2-reto-rds-proxy' and is in an 'Available' state. The configuration details are as follows:

- Proxy configurations:**
 - Proxy Identifier:** g2-reto-rds-proxy
 - Status:** Available
 - Identity and access management (IAM) role:** arn:aws:iam::590183727319:role/LabRole
 - VPC security groups:** sg-01fc34f8b2408902
 - Enhanced logging:** Not enabled
 - Default authentication scheme:** None
 - Target connection network type:** IPv4
- Engine family:** MariaDB and MySQL (supports Aurora MySQL, RDS for MariaDB, and RDS for MySQL)
- Transport Layer Security:** Not enabled
- Idle client connection timeout:** 30 minutes
- Proxy Amazon Resource Name (ARN):** arn:aws:rds:us-east-1:590183727319:db-proxy:prx-04b316d9e35af2942
- Subnets:** subnet-0561f59be818d97f6, subnet-098b7eeef738d7634, subnet-09967c2b0366f7e7
- Endpoint network type:** IPv4

Below the configuration details, there is a section for 'Proxy endpoints (1)'. It shows a single endpoint with the following details:

Endpoint	VPC ID	Status	Target role
g2-reto-rds-proxy-c7wskuew4q63.us-east-1.rds.amazonaws.com	vpc-0e827591fd86f881	Available	Read/write

Securización del entorno (SAD, IAW)

En el desarrollo de Health2You, teníamos claro que la seguridad no podía ser un añadido final, sino una base del proyecto. Sabiendo que WordPress es uno de los CMS más atacados del mundo, hemos implementado una estrategia de "cebolla" (capas de seguridad) para que, si una barrera falla, la siguiente detenga la amenaza.

Hemos actuado en varios frentes:

AWS WAF

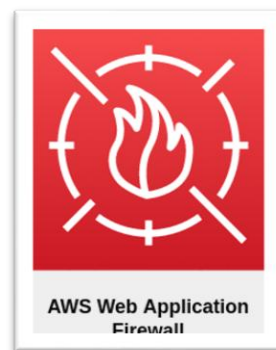
Implementamos un Web Application Firewall (WAF) configurado mediante código (IAC) para que actúe como un filtro inteligente antes de que el tráfico toque nuestros servidores.

En lugar de usar reglas genéricas, diseñamos una defensa a medida:

Zona para Administradores: Configuramos una "Lista Blanca" (AllowWpAdminFromTrustedIPs). Esto significa que el panel de administración de WordPress es invisible para el resto del mundo; si no entras desde nuestras IPs, el firewall te bloquea automáticamente.

Freno a los Bots: Para evitar que intenten adivinar nuestras contraseñas, impusimos un límite de velocidad (RateLimitLogin). Si una IP hace más de 100 peticiones en 5 minutos, el sistema asume que es un ataque de fuerza bruta y la bloquea temporalmente.

Escudo contra Hackers: Activamos reglas específicas (SQLiRules y XSSRules) que detectan y bloquean intentos de inyectar código malicioso en la base de datos o en la web.



AWS VPN

Uno de los mayores riesgos en la nube es dejar puertos de gestión (como el 22 para SSH) abiertos a internet. Nosotros decidimos cerrar esa puerta por completo.

Implementamos una AWS Client VPN con autenticación mutua mediante certificados. Lo innovador de nuestra configuración es que la seguridad se basa en la identidad, no solo en la red.



Invisibilidad: Nuestros servidores no aceptan conexiones SSH de ninguna dirección IP pública.

El "Carné" de Acceso: Configuramos los Grupos de Seguridad para que solo permitan el tráfico si viene "etiquetado" con el ID del Grupo de Seguridad de la VPN.

Resultado: Para un atacante externo, nuestros servidores parecen estar desconectados (puertos cerrados), pero para nosotros, al conectar la VPN, el acceso es transparente y seguro.

Fortificación (Hardening) de WordPress

Finalmente, endurecimos la propia aplicación para minimizar riesgos si un atacante lograra pasar el firewall. Aplicamos medidas de seguridad preventiva directamente en el código y la configuración:



Desactivamos el editor de archivos interno añadiendo `define('DISALLOW_FILE_EDIT', true);` en el código. Esto impide que, si alguien nos roba una contraseña de administrador, pueda inyectar virus o modificar la web desde el panel.

Ocultamos la versión de WordPress para que los escáneres automáticos no sepan qué vulnerabilidades buscar y bloqueamos el archivo `xmlrpc.php`, que es una puerta trasera habitual para ataques masivos.

Protección de Archivos: Configuramos permisos estrictos en la carpeta de "Subidas" para que nadie pueda subir y ejecutar un script malicioso en nuestro servidor.

Fortificación (hardening) de servidores

Se han aplicado directivas de seguridad modificando los parámetros del archivo `sshd_config` para proteger nuestros sistemas al conectarnos por SSH.

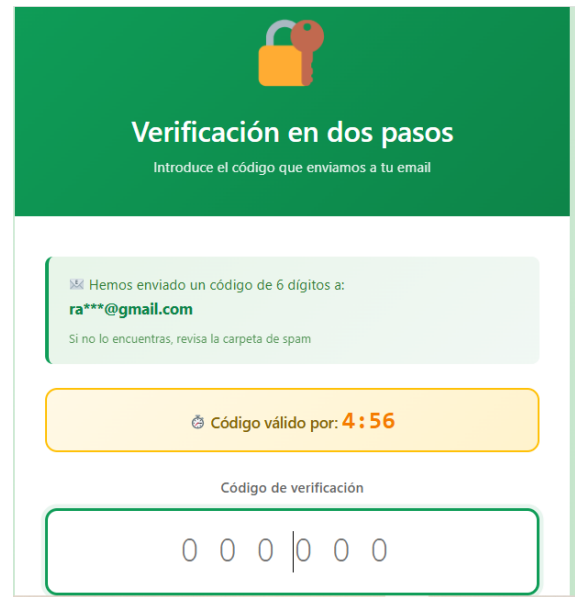
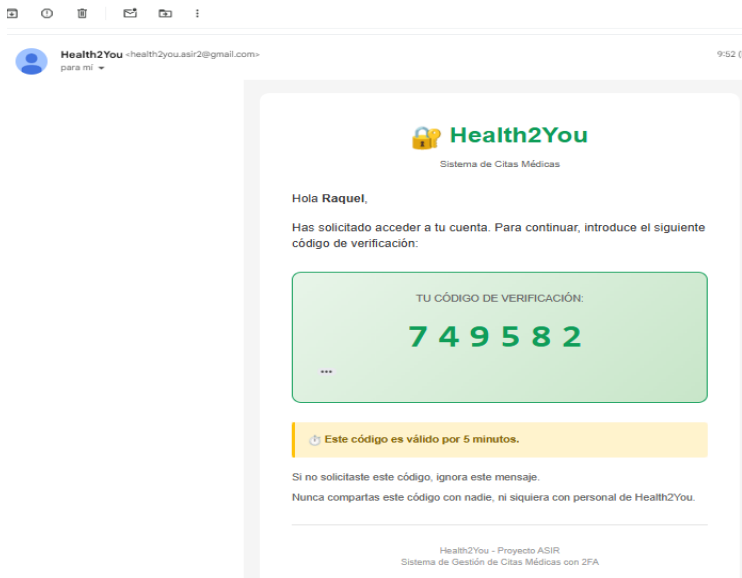
```
ubuntu@ip-10-0-1-155: ~  
GNU nano 6.2 /etc/ssh/sshd_config *  
#LogLevel INFO  
  
# Authentication:  
  
#LoginGraceTime 2m  
PermitRootLogin no  
#StrictModes yes  
MaxAuthTries 3  
#MaxSessions 10  
  
PubkeyAuthentication yes  
Protocol 2  
  
# Expect .ssh/authorized_keys2 to be disregarded by default in future.  
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2  
  
#AuthorizedPrincipalsFile none  
  
#AuthorizedKeysCommand none  
#AuthorizedKeysCommandUser nobody  
  
# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts  
#HostbasedAuthentication no  
# Change to yes if you don't trust ~/.ssh/known_hosts for  
# HostbasedAuthentication  
#IgnoreUserKnownHosts no  
# Don't read the user's ~/.rhosts and ~/.shosts files  
#IgnoreRhosts yes  
  
# To disable tunneled clear text passwords, change to no here!  
PasswordAuthentication no  
#PermitEmptyPasswords no
```

Así como desactivar los servicios que no son necesarios, cerrar puertos abiertos que no necesitamos, endurecer políticas de contraseñas y configurar las reglas del firewall de cada servidor para permitir únicamente la entrada por los puertos 22 para SSH y 80 y 443 para web.

Doble factor de autenticación (2FA)

Hemos creado una verificación en dos pasos, para que cuando un paciente se registre, le llegue a su correo personal un código de 6 dígitos que deberá introducir para poder iniciar sesión (si el paciente no tiene correo no pasa nada, puede iniciar sesión igualmente).

Para ello creamos un Gmail profesional y generamos una contraseña de aplicación. después modificamos distintos archivos `.php` para añadir las líneas de código necesarias.



Certificados

Para nuestro despliegue de certificados hemos utilizado también los de **AWS** con el servicio de **Certificate Manager**, esto lo hemos configurado primero creando los certificados en el servicio, después por medio del panel de control de hosting, agregar nuestros certificados como registros CNAME y después nuestro nombre de dominio apuntando a el ALB de nuestro balanceador de carga.



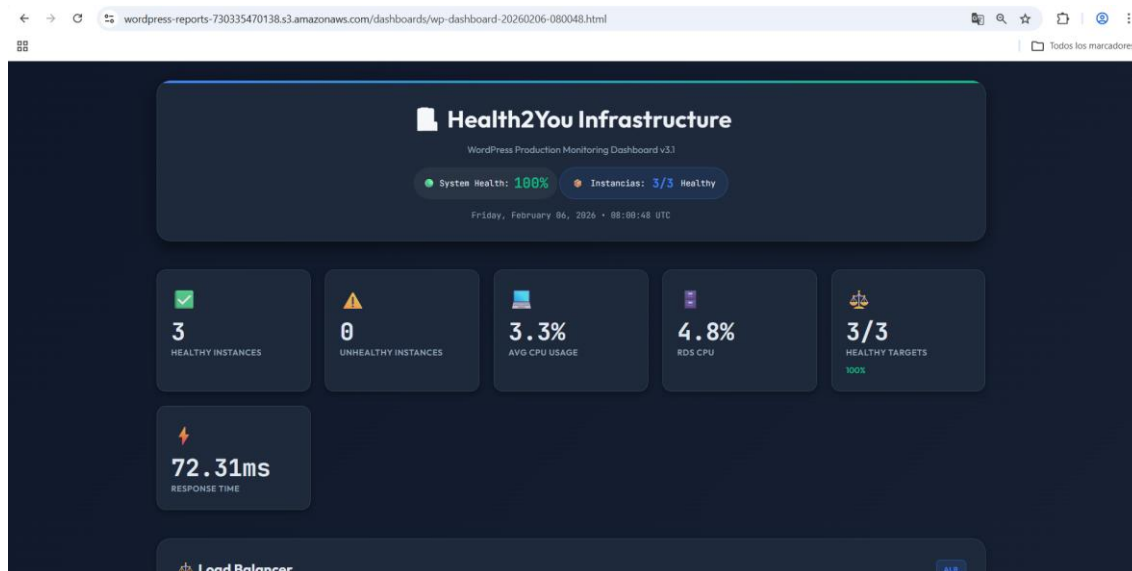
Eliminar			
Entrada DNS	Tipo	Valor	Modificar
<input type="checkbox"/> _13d246d4eaa22f6852297d82de8f.www.aginformatica2.com.es	CNAME	_d9b69e3f9e7b07c134e38d3da26c7c.jkddztzsm.acm-validations.aws	
<input type="checkbox"/> _c0d1083208575a1a152d3de18c821.aginformatica2.com.es	CNAME	_70f15971f13c79b857a9cd7dac2f8af.jkddztzsm.acm-validations.aws	
<input type="checkbox"/> aginformatica2.com.es	A	217.76.128.34	
<input type="checkbox"/> control.aginformatica2.com.es	CNAME	pdcs.servidoresdro.net	
<input type="checkbox"/> www.aginformatica2.com.es	CNAME	wordpress-alb-545811703.us-east-1.elb.amazonaws.com	

Resultados por página 10 1 - 5 de 5

Exportar a Excel Exportar a PDF

Monitorizado

Como herramientas hemos utilizado **AWS CloudWatch**, la cual permite crear paneles en el que se muestren los datos de nuestra infraestructura, además hemos creado un script de **PowerShell**, el cual por medio de otro script que recoge los datos de nuestra infraestructura, genere un dashboard personalizado con los datos de nuestra infraestructura.



Entorno WEB (IAW, SRI)

Nuestro entorno web esta levantado por completo sobre el CMS de WordPress en un tema personalizado por nosotros con ello tenemos los beneficios de los plugin y customización de WordPress y también la posibilidad de crear nuestro propio código.

Todo el código de la web está en el [repositorio grupal de GitHub](#) y este se clona automáticamente a nuestro volumen EFS gracias a el IAC esto nos permite que nada más ser lanzada la instancia ya tendremos accesible la web y nuestro tema activado.

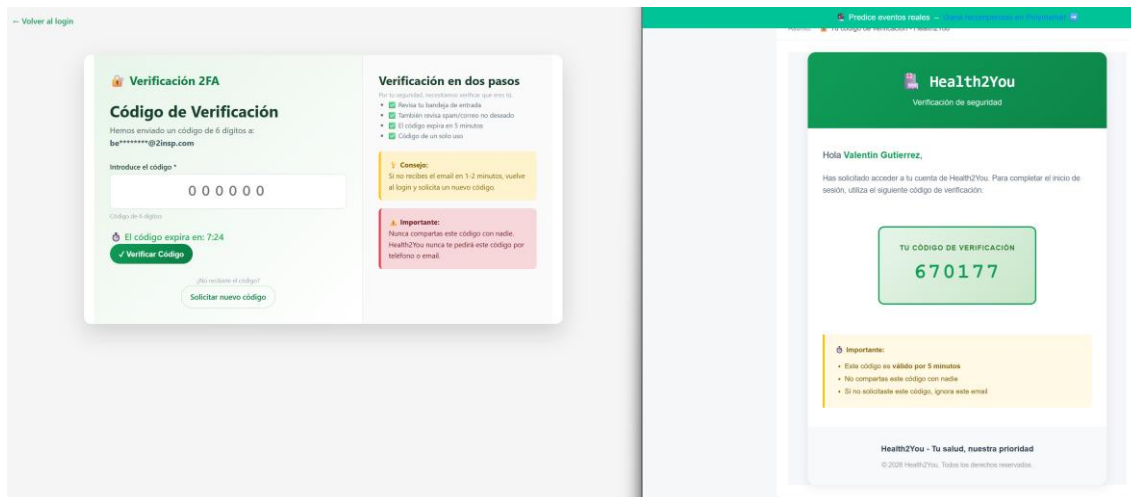
Cabe mencionar que también se instalan de forma autónoma los plugin que utilizamos para securización y customización.

A continuación, describiremos los principales servicios que encontramos dentro de ella.

Área paciente

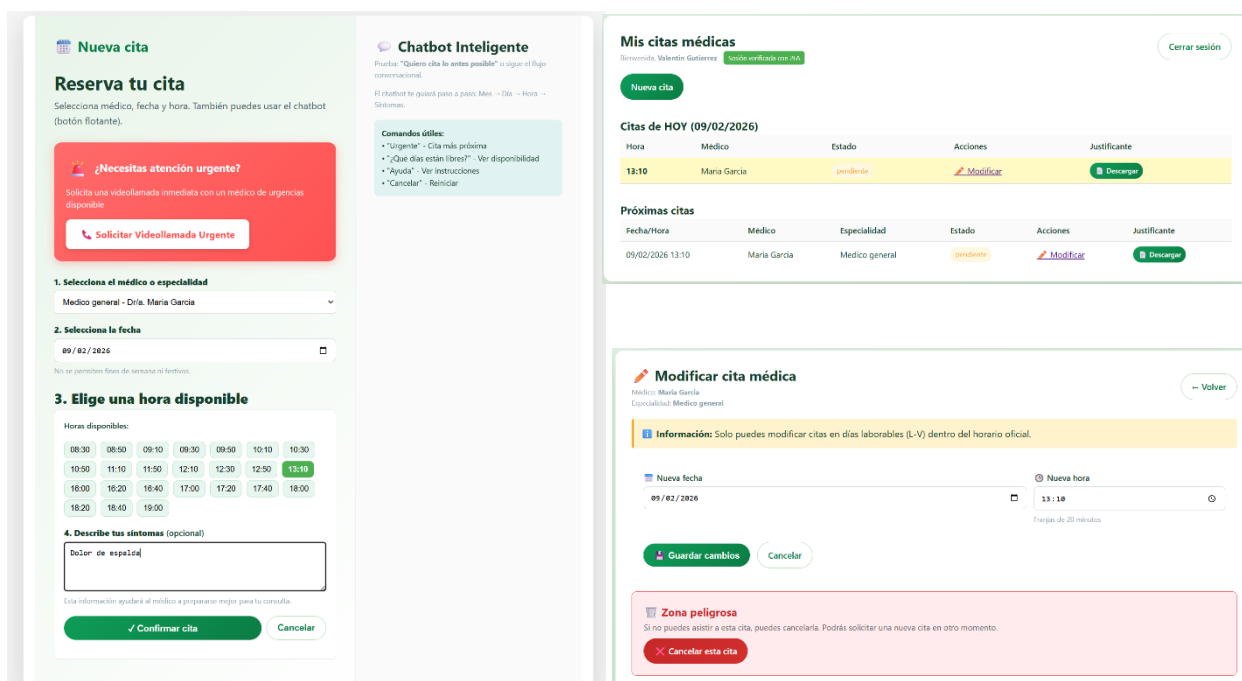
Acceso

El paciente tiene la opción de acceder mediante usuario y contraseña o si se ha registrado con email se le enviará un código para verificar su identidad.

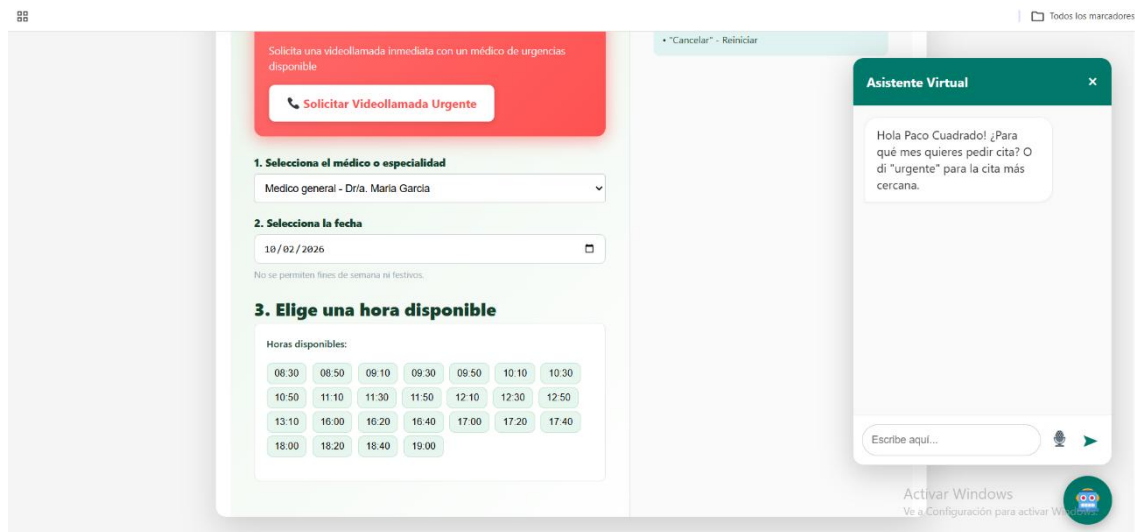


Funciones

Una vez ha accedido a la web el paciente tiene la posibilidad de solicitar una cita médica, visualizar las citas que ya haya atendido o solicitado y modificarlas. Tenemos 3 posibles vías para obtener una cita, rellenando el formulario en el que escoges médico, día, hora y una breve descripción de tus síntomas.

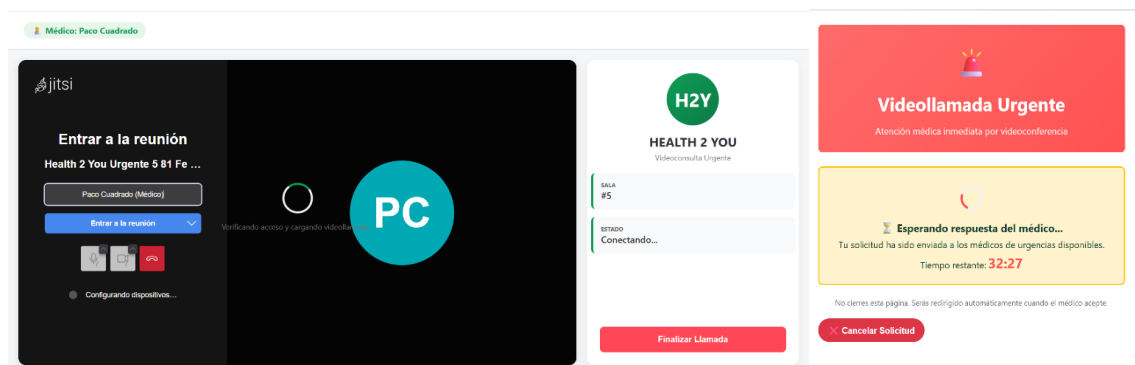


Por otro lado, hemos implementado un chatbot en el que a través de texto o audio podemos responder a sus preguntas para solicitar una cita una vez respondamos a todo rellenará los campos del formulario y nos redirigirá al dashboard.



Como opcion final tenemos la posibilidad de solicitar una cita urgente al dar a este boton nos redirige a una sala de espera para entrar a una video llamada con un medico.

Esto genera una peticion al medico de urgencia y si este acepta se generará una sesión esta está securizada con el uso de tokens de sesion, estos se generan de forma unica para cada sesion y son destruidos al finalizar la llamada ademas adicionalmente para que un paciente no pueda generar citas urgentes sin limite esta establecido un cooldown entre cada petición.



Como última función para pacientes tenemos la posibilidad de que generen un justificante para cada cita y enviarlo por correo o descargarlo.

JUSTIFICANTE DE CITA MEDICA

Health2You - Sistema de Gestion de Citas

DATOS DEL PACIENTE

Nombre completo: Valentin Gutierrez
Numero TSI: CANT19032002

DATOS DE LA CITA

Medico: Maria Garcia
Especialidad: Medico general
Fecha: 09/02/2026
Hora inicio: 13:10
Hora fin: 13:30
Duracion: 20 minutos
Estado: PENDIENTE
ID Cita: 000003

Se certifica que el/la paciente arriba mencionado/a tiene programada una cita medica para el dia 09/02/2026 a las 13:10 horas en las instalaciones de Health2You con el Dr./Dra. Maria Garcia.

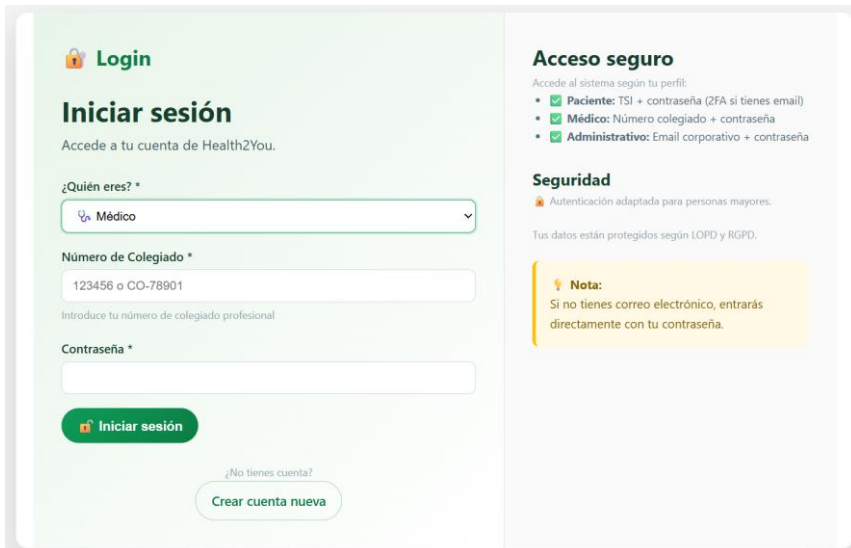
NOTA: Este documento es un justificante informativo generado por el sistema Health2You. Para validar su autenticidad, puede verificar el ID de cita en nuestro sistema o contactar con nuestras oficinas.

Health2You - Sistema de Gestion de Citas Medicas
Documento generado el: 09/02/2026 12:38:50
Codigo de verificacion: 636403F0

Área Medico

Acceso

Los médicos acceden desde el login relleno los campos de numero de colegiado como identificador y su contraseña.



Login

Iniciar sesión

Accede a tu cuenta de Health2You.

¿Quién eres? *

Número de Colegiado *

Introduce tu número de colegiado profesional

Contraseña *

[Iniciar sesión](#)

[¿No tienes cuenta? Crear cuenta nueva](#)

Acceso seguro

Accede al sistema según tu perfil:

- ✓ **Paciente:** TSI + contraseña (2FA si tienes email)
- ✓ **Médico:** Número colegiado + contraseña
- ✓ **Administrativo:** Email corporativo + contraseña

Seguridad

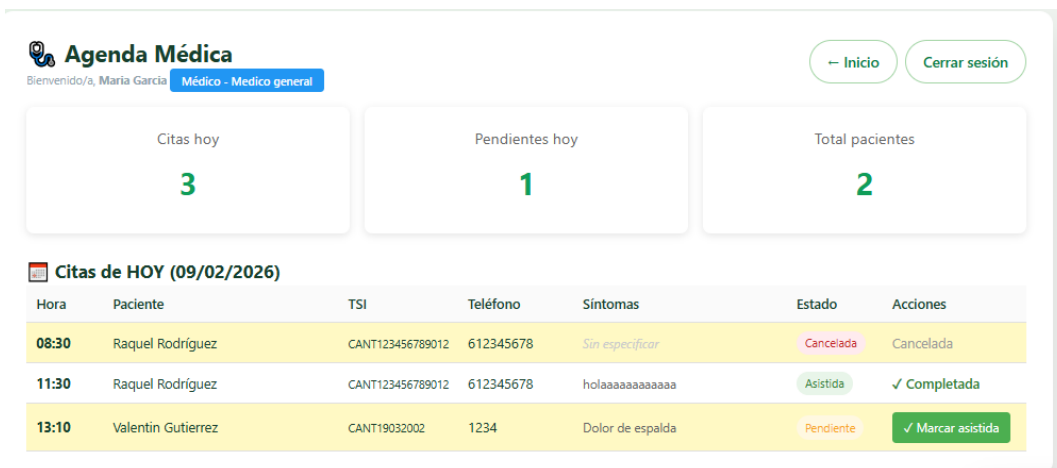
Autenticación adaptada para personas mayores.

Tus datos están protegidos según LOPD y RGPD.

Nota:
Si no tienes correo electrónico, entrarás directamente con tu contraseña.

Funciones

Una vez acceda tendrá a la vista las citas a atender durante la jornada de hoy y podrá marcarlas como asistida para que el paciente pueda descargar su pdf.



Agenda Médica

Bienvenido/a, Maria García [Médico - Medico general](#)

[Inicio](#) [Cerrar sesión](#)

Citas hoy

3

Pendientes hoy

1

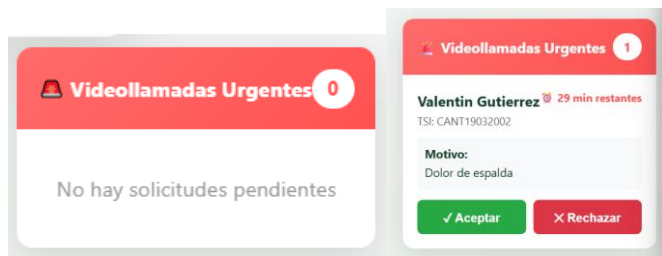
Total pacientes

2

Citas de HOY (09/02/2026)

Hora	Paciente	TSI	Teléfono	Síntomas	Estado	Acciones
08:30	Raquel Rodríguez	CANT123456789012	612345678	Sin especificar	Cancelada	Cancelada
11:30	Raquel Rodríguez	CANT123456789012	612345678	holaaaaaaaaaaaa	Asistida	✓ Completada
13:10	Valentin Gutierrez	CANT19032002	1234	Dolor de espalda	Pendiente	✓ Marcar asistida

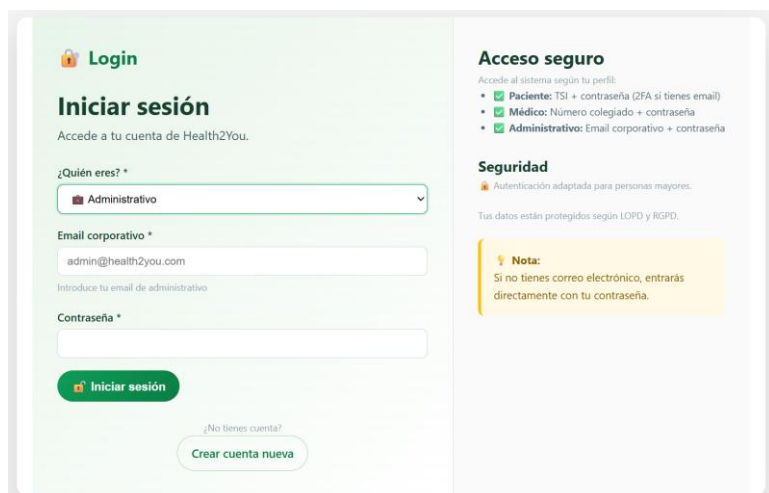
Luego en caso de que el medico sea asignado de urgencia podrá ver el siguiente panel en el que aparecen las solicitudes de llamadas con la información del paciente el tiempo de espera restante y la opción de aceptar o rechazar la petición.



Área Administrativo

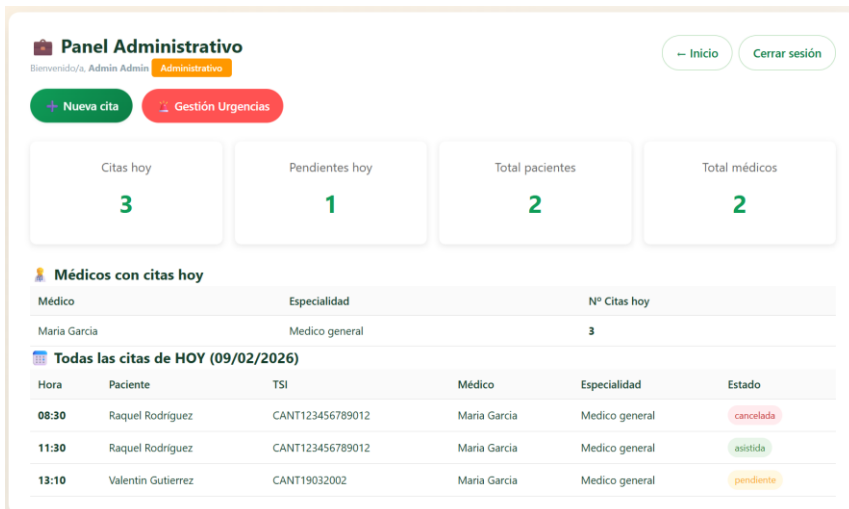
Acceso

Los administrativos se identifican con un correo corporativo y su contraseña.



Funciones

Los administrativos en su dashboard tienen cuatro funciones diferentes, visualizar la cantidad de citas asignadas para cada médico, observar las citas totales del día;



Panel Administrativo

Bienvenido/a, Admin Admin **Administrativo**

[Inicio](#) [Cerrar sesión](#)

[+ Nueva cita](#) [Gestión Urgencias](#)

Citas hoy	Pendientes hoy	Total pacientes	Total médicos
3	1	2	2

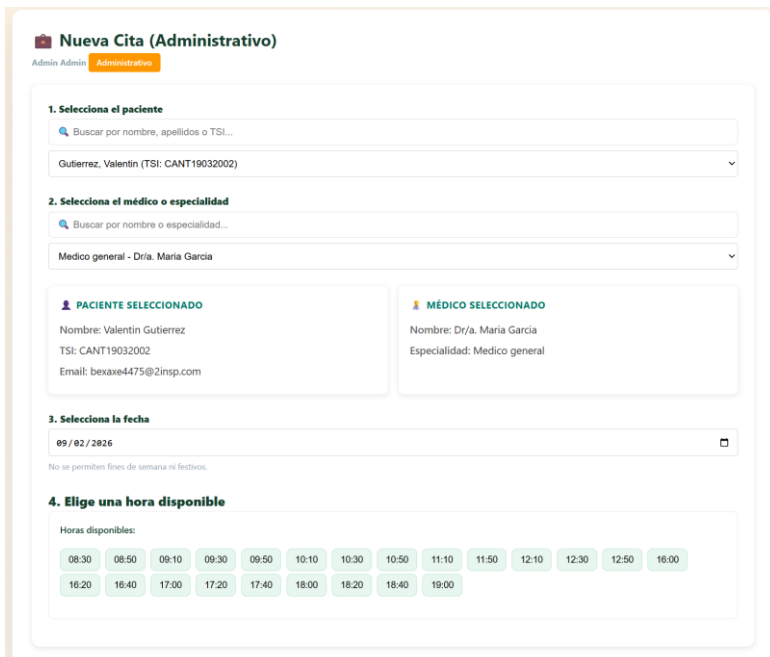
Médicos con citas hoy

Médico	Especialidad	Nº Citas hoy
Maria Garcia	Medico general	3

Todas las citas de HOY (09/02/2026)

Hora	Paciente	TSI	Médico	Especialidad	Estado
08:30	Raquel Rodriguez	CANT123456789012	Maria Garcia	Medico general	cancelada
11:30	Raquel Rodriguez	CANT123456789012	Maria Garcia	Medico general	asistida
13:10	Valentin Gutierrez	CANT19032002	Maria Garcia	Medico general	pendiente

La posibilidad de generar una nueva cita eligiendo al paciente, al médico fecha y hora.



Nueva Cita (Administrativo)

Admin Admin **Administrativo**

1. Selecciona el paciente

Buscar por nombre, apellidos o TSI...

Gutierrez, Valentin (TSI: CANT19032002)

2. Selecciona el médico o especialidad

Buscar por nombre o especialidad...

Medico general - Dr/a. Maria Garcia

PACIENTE SELECCIONADO

Nombre: Valentin Gutierrez
TSI: CANT19032002
Email: bexaxe4475@2insp.com

MÉDICO SELECCIONADO

Nombre: Dr/a. Maria Garcia
Especialidad: Medico general

3. Selecciona la fecha

09 / 02 / 2026

No se permiten fines de semana ni festivos.

4. Elige una hora disponible

Horas disponibles:

08:30 08:50 09:10 09:30 09:50 10:10 10:30 10:50 11:10 11:50 12:10 12:30 12:50 16:00
16:20 16:40 17:00 17:20 17:40 18:00 18:20 18:40 19:00

Y por último tienen la capacidad de asignar los médicos que asistirán las urgencias.

Gestión de Médicos de Urgencias

Configura qué médicos pueden atender videollamadas urgentes

[← Volver](#)

Estadísticas de Videollamadas (Últimos 30 días)

2	0	2	0
Total Solicitudes	Atendidas	Rechazadas	Expiradas

Selecciona Médicos para Urgencias

Los médicos seleccionados recibirán notificaciones de videollamadas urgentes en su dashboard

Dr./Dra. Maria Garcia

Medico general

 2 pacientes  3 citas  raquel.colin2796@gmail.com

Activo en urgencias ☒

Dr./Dra. Monolo Manos Largas

Medio general

 0 pacientes  0 citas  bexaxe4475@2insp.com

Activo en urgencias ☒

 **Guardar Configuración**

App móvil

Para mayor comodidad y así facilitar el uso de nuestra plataforma a los usuarios, hemos querido que nuestro proyecto se pueda descargar en una app web tanto para móvil como para ordenador, creando diferentes archivos .json y modificando algunos archivos .php para para generar nuestra PWA (Progressive Web App).



Costos y comparaciones

Análisis de Inversiones: Justificación final del ROI (Retorno de Inversión) y comparación detallada de costes: On-premises vs Cloud.

Este apartado evalúa la viabilidad financiera de la infraestructura diseñada en el **Diagrama de Red**, comparando el desembolso necesario para una solución física tradicional (**On-premises**) frente al modelo de servicios gestionados en la nube (**AWS Cloud**).

Comparativa Detallada de Costes (Ciclo de Vida: Año 1)

Para asegurar la **Alta Disponibilidad Multi-AZ** y la redundancia de datos (RDS, EFS, S3) que muestra el diseño, la solución física requiere una inversión inicial significativa en hardware redundante y sistemas de respaldo.

Concepto	On-Premises (Físico Estimado)	AWS Cloud (Tu Presupuesto)
Inversión Inicial (CAPEX)	18.500,00 € (Servidores, Rack, SAI, FW)	0,00 €
Coste Operativo Anual (OPEX)	3.600,00 € (Luz, Fibra, Mantenimiento)	5.600,00 € (\$6.033,60 USD)
Escalabilidad	Manual y costosa (comprar RAM/CPU)	Instantánea e incluida
Alta Disponibilidad	Compleja (duplicar hardware físico)	Nativa (Multi-AZ incluida)
COSTO TOTAL (Año 1)	22.100,00 €	5.600,00 €

Justificación del ROI (Retorno de Inversión)

Para cuantificar el beneficio económico de haber optado por la arquitectura Cloud, aplicamos la fórmula de eficiencia de costes basada en el ahorro generado frente a la inversión física en el primer año:

$$ROI = \frac{(\text{Coste On-prem} - \text{Coste Cloud})}{\text{Coste Cloud}} \times 100$$

Sustitución de valores (Datos Año 1):

- **Coste On-prem:** 22.100,00 €
- **Coste Cloud:** 5.600,00 €

$$ROI = \frac{(22.100 - 5.600)}{5.600} \times 100 = 294,64\%$$

Conclusión de Viabilidad

La implementación de la arquitectura en **AWS** se justifica como la opción más eficiente por los siguientes motivos técnicos y económicos:

Eficiencia Financiera Inmediata: Un ROI del **294,64%** en el primer año indica que por cada euro invertido en la suscripción de AWS, la organización evita gastar casi 3 euros en infraestructura física y su mantenimiento asociado.

Alineación con el Diseño de Red: El diagrama de infraestructura utiliza servicios como **Auto Scaling** y **Load Balancers**. Replicar estas funcionalidades en un entorno físico no solo subiría el CAPEX por encima de los 18.500 €, sino que aumentaría la complejidad operativa para el equipo de sistemas.

Eliminación del Riesgo de Obsolescencia: Al no existir inversión en activos físicos (CAPEX), la empresa mantiene su liquidez y puede migrar o evolucionar la infraestructura sin tener que amortizar hardware antiguo.

Soberanía de Gestión: El uso de **Amazon RDS** y **EFS** permite que el equipo técnico se centre en la optimización del servicio y no en el mantenimiento preventivo de discos, fuentes de alimentación o sistemas de climatización del CPD.

Gestión del proyecto

Durante el proyecto fuimos utilizando diferentes herramientas para facilitarnos tanto la gestión de los archivos y documentos como la propia comunicación entre los miembros del grupo, aquí explicamos las que hemos utilizado.

Trello

Para la asignación de tareas, está por medio de un tablero nos permite la asignación de tareas agregando a la propia tarea al usuario, como el registro de los avances realizados por los miembros, esto lo hacemos utilizando estados de tareas (en proceso, finalizado, etc....) y checklist dentro de las propias tareas para marcar objetivos para cumplir la tarea.



GitHub

Para guardar el código utilizado por el reto, este nos ha facilitado varias tareas, en primer lugar, como punto de guardado para todo nuestro código, este se encuentra dentro de una rama **main**, a su vez también nos permite crear ramas individuales en la que cada

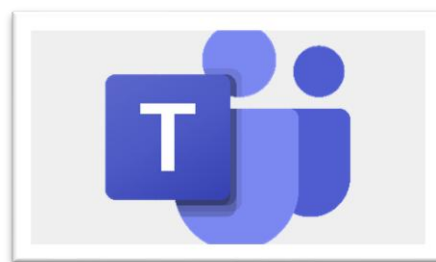


miembro va realizando sus propios cambios, para al final del día volver toda la rama sobre la principal.

Aparte como se mencionó anteriormente, hemos utilizado comandos de GitHub dentro de las instancias para que se importe automáticamente la rama main dentro de las instancias para tener la web disponible al levantar la infraestructura.

Teams

Esta la hemos utilizado aparte de para subir documentos de texto o imágenes como lo son los hitos o el logo de la empresa, esta cumple con la función de permitir la comunicación con los miembros del grupo para comunicar cambios en el proyecto o faltas de asistencia, aparte de permitir la propia entrega de las tareas.



Errores

Limitaciones de AWS

Como se comentó en la explicación del uso de AWS, uno de los problemas principales problemas fue la limitación de ciertos servicios, el primero que nos encontramos fue a la hora de diseñar la base de datos que pensamos utilizar **Oracle** y después **Postgre**, pero ambas están bloqueadas por lo que optamos por utilizar **MySQL**.

Dentro del ámbito de base de datos también pensábamos utilizar una base de datos distribuida, pero al estar también bloqueada optamos por crear una **RDS** en una subred y modificamos la estructura para que en caso de tener una base de datos distribuida esta la pueda soportar.

también nos sucedió a la hora de realizar la auditoria de seguridad, puesto que pensamos utilizar la herramienta **AWS Audit Manager**, pero al estar bloqueado por el plan de estudiantes optamos por utilizar la herramienta de **Inventory** y un script lanzado desde el bastión que realiza una auditoria de las maquinas.

Incompatibilidad de base de datos

Como mencionamos anteriormente una de nuestras opciones de motor de base de datos fue utilizar **Postgre**, pero este nos dio problemas puesto que no tiene compatibilidad con WordPress, la alternativa a este fue utilizar **MySQL**.

Problema de sticky session

Como mencionamos en el apartado de balanceador de carga, al no tener configurada esta opción la web comenzó a dar error puesto a que, al tener una instancia apagada, el balanceador seguía redirigiendo a ella, dando fallo, la solución a ello fue simplemente activando la opción para que solo nos redirija entre las subredes activas.

Fallo de DNS

Durante la semana, después de implementar el certificado para la web tuvimos problemas al acceder a los laboratorios puesto que al acceder por medio del DNS accedíamos al laboratorio del compañero que tenía configurada sus certificados en el panel de control de DNS. Aparte nos dio un problema de que se nos bloqueaba el acceso aun teniendo todo bien configurado y por el momento solo lo solucionamos reiniciando la IP de la maquina local, el error se da debido al firewall de la clase que está cacheando nuestra web.

Acceso al dashboard de monitorización

Durante las pruebas al intentar utilizar el script para extraer los datos del stack, como el laboratorio de uno de los compañeros estaba utilizando los certificados en el hosting, el stack de otros compañeros no termina de completarse al no estar siendo usados, lo que hace que el script no lo detecte. Como solución hemos encontrado dos, la primera seria utilizar los certificados propios para que marque el stack como completado, y otra seria que el archivo que se genera automáticamente crearlo a mano con nuestras credenciales, esto hará que el segundo script pueda generar el dashboard con las credenciales especificadas.

Manual de mantenimiento



MANUAL DE
MANTENIMIENTO.docx