

SOFTWARE INSTALADO EN LA PLATAFORMA DEL CLIENTE.

AUTOR:

BRAYAN ALEXANDER PUENTES MARTINEZ

INSTRUCTOR:

ANDRES RUBIANO CUCARIAN

SENA

ANALISIS Y DESARROLLO DE SOFTWARE(ADSO)

CENTRO DE SERVICIOS FINANCIEROS

BOGOTÁ D.C. 14 DE MAYO DE 2024

Tabla de contenido

INTRODUCCIÓN.....	4
CARACTERÍSTICAS DEL SISTEMA OPERATIVO	5
DEFINICIÓN DE SISTEMA OPERATIVO.....	5
IMPORTANCIA DE SELECCIONAR EL SISTEMA OPERATIVO ADECUADO PARA EL ENTORNO DE REDES Y NETWORKING.....	5
REQUISITOS MÍNIMOS DE HARDWARE Y SOFTWARE.....	6
COMPATIBILIDAD CON DISPOSITIVOS DE RED Y PROTOCOLOS ESTÁNDAR PARA PÁGINA WEB.....	6
SEGURIDAD Y CAPACIDAD DE GESTIÓN DE REDES PARA UNA PÁGINA WEB... ..	7
ESCALABILIDAD Y RENDIMIENTO PÁGINA WEB.	7
ORGANIZACIONES ESTÁNDARES EN REDES Y NETWORKING	8
¿QUÉ ES UN ESTANDAR?	8
¿QUIÉN DEFINE LOS ESTÁNDARES EN TELECOMUNICACIONES?	8
¿PARA QUÉ SIRVE UN ESTÁNDAR EN TELECOMUNICACIONES?	8
¿QUÉ SON: AMPS, CDMA, TDMA, GPRS Y PCS, ¿TÉRMINOS TECNOLOGICOS UTILIZADOS A MENUDO EN PROPAGANDAS Y COMERCIALES?	9
¿QUÉ ES WIFI?	10
¿QUÉ ES WIMAX?	11
¿QUÉ NORMAS DEL MINISTERIO DE TECNOLOGIAS DE INFORMACION Y COMUNICACIONES REGLAMENTAN LAS TECNOLOGIAS GSM, CDMA Y WIMAX ENTRE OTRAS?	12
¿QUÉ ES BANDA ANCHA Y CUÁL ES LA VELOCIDAD MINIMA PARA BANDA ANCHAS EN COLOMBIA?	12
GRANDES FAMILIAS DE PROTOCOLOS DE TRANSMISIÓN Y RECEPCIÓN DE DATOS.....	13
PROTOCOLOS DE RED	13
MEDIOS DE TRANSMISIÓN GUIADOS Y NO GUIADOS	15
MEDIOS DE TRANSMISIÓN GUIADOS.....	15
CABLE DE COBRE	15
FIBRA ÓPTICA	15
MEDIOS DE TRANSMISIÓN NO GUIADOS.....	15
ONDAS DE RADIO	15
MICROONDAS.....	15
DESPLIEGUE DE LA APLICACIÓN	16

PASOS DEL DESPLIEGUE LOCAL	16
CAPTURAS DE PANTALLA DEL DESPLIEGUE LOCAL	22
PASOS DEL DESPLIEGUE.....	27
CAPTURAS DE PANTALLA DEL DESPLIEGUE.....	¡Error! Marcador no definido.
CONCLUSIONES.....	55
Bibliografía.....	55

INTRODUCCIÓN

En el contexto actual de la tecnología de la información y las comunicaciones, la selección adecuada de componentes y tecnologías es esencial para el despliegue exitoso de sistemas y servicios informáticos. Dentro de este panorama, el sistema operativo, los medios de transmisión y otros aspectos de la infraestructura de red juegan un papel fundamental en el funcionamiento y la eficacia de las aplicaciones y servicios en línea.

El presente trabajo tiene como objetivo abordar diversos aspectos relacionados con la selección y evaluación de componentes clave para la implementación y operación de sistemas informáticos y servicios en línea. En particular, se enfocará en la importancia de seleccionar el sistema operativo adecuado, comprender las familias de protocolos de transmisión, y analizar los medios de transmisión disponibles en el contexto de una página web.

CARACTERÍSTICAS DEL SISTEMA OPERATIVO

DEFINICIÓN DE SISTEMA OPERATIVO.

El sistema operativo es un conjunto de programas y software que actúan como intermediarios entre el hardware de una computadora y sus usuarios. Su función principal es facilitar la comunicación entre el hardware y el software de la computadora, proporcionando una interfaz para que los usuarios interactúen con el sistema y ejecuten programas de manera eficiente.

IMPORTANCIA DE SELECCIONAR EL SISTEMA OPERATIVO ADECUADO PARA EL ENTORNO DE REDES Y NETWORKING.

RAZONES POR LAS QUE SON IMPORTANTES SELECCIONAR UN SISTEMA OPERATIVO ADECUADO.

- **Compatibilidad:** El sistema operativo debe ser compatible con los dispositivos de red y protocolos estándar utilizados en la infraestructura de red. Esto garantiza una comunicación fluida y sin problemas entre los diferentes componentes de la red.
- **Seguridad:** El sistema operativo debe contar con características de seguridad robustas para proteger la integridad de los datos y la privacidad de la red. Esto incluye capacidades de firewall, detección de intrusos y gestión de acceso.
- **Gestión de Redes:** Un sistema operativo adecuado para entornos de redes debe ofrecer herramientas de gestión de redes integradas que permitan monitorear y administrar eficientemente todos los dispositivos conectados a la red.
- **Rendimiento:** El sistema operativo debe ser capaz de gestionar eficientemente el tráfico de red y los recursos del sistema para garantizar un rendimiento óptimo de la red. Esto incluye capacidades de optimización de ancho de banda y administración de recursos.

REQUISITOS MÍNIMOS DE HARDWARE Y SOFTWARE.

COMPATIBILIDAD CON DISPOSITIVOS DE RED Y PROTOCOLOS ESTÁNDAR PARA PÁGINA WEB.

Hardware: Se requiere un hardware básico que pueda ejecutar el sistema operativo y los servicios necesarios para alojar y servir la página web. Esto incluye un procesador lo suficientemente potente, suficiente memoria RAM y espacio de almacenamiento para los archivos del sitio web.

Software: El sistema operativo debe ser compatible con los servidores web y las tecnologías de desarrollo utilizadas para crear la página web, como PHP, Python, o Node.js. Además, se deben considerar los requisitos de software para la gestión de bases de datos, si la página web utiliza bases de datos para almacenar información.

Ejemplo de requisitos mínimos para la página web hipertrophy fitness software y hardware

REQUISITOS DE HARDWARE Y SOFTWARE	
HARDWARE	SOFTWARE
<ul style="list-style-type: none">• Procesador: Intel Core i3 de cualquier generación o equivalente de otro fabricante.• Memoria RAM: 4 GB.• Almacenamiento: Disco duro con al menos 128 GB de espacio disponible.• Conectividad: Conexión a Internet (por cable o inalámbrica).• Pantalla: Cualquier pantalla que permita visualizar contenido web de forma legible.	<ul style="list-style-type: none">• Sistema operativo: Windows 7 o superior, macOS 10.12 o superior, o una distribución de Linux compatible.• Navegador web: Cualquier navegador moderno y actualizado, como Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, entre otros.

SEGURIDAD Y CAPACIDAD DE GESTIÓN DE REDES PARA UNA PÁGINA WEB.

En la mayoría de los casos las páginas web deben contar con las siguientes opciones en seguridad y gestión de redes. Todo esto puede variar de acuerdo con un presupuesto o la experiencia del programador al realizar un proyecto.

- El sistema operativo debe proporcionar características de seguridad robustas, como firewalls, filtrado de paquetes, y detección de intrusiones, para proteger la página web y los datos de los usuarios contra amenazas en línea.
- Debe ofrecer herramientas de gestión de redes que faciliten la configuración, monitorización y mantenimiento de la red, permitiendo a los administradores gestionar de forma efectiva la seguridad y el rendimiento de la página web.

ESCALABILIDAD Y RENDIMIENTO PÁGINA WEB.

- El sistema operativo debe ser escalable para adaptarse al crecimiento de la página web y a un aumento en la cantidad de tráfico. Debe ser capaz de manejar un gran número de solicitudes simultáneas sin degradación del rendimiento.
- Además, debe ofrecer un rendimiento óptimo para garantizar tiempos de carga rápidos y una experiencia de usuario fluida en la página web, lo que contribuye a la retención de usuarios y al éxito del sitio.

ORGANIZACIONES ESTÁNDARES EN REDES Y NETWORKING

¿QUÉ ES UN ESTANDAR?

Un estandar (como lo define la ISO) "son acuerdos documentados que contienen especificaciones técnicas u otros criterios precisos para ser usados consistentemente como reglas, guias o definiciones de caracteristicas para asegurar que los materiales, productos, procesos y servicios cumplan con su proposito". (mintic, s.f.)

¿QUIÉN DEFINE LOS ESTÁNDARES EN TELECOMUNICACIONES?

Básicamente, existen dos tipos de organizaciones que definen estándares: Las organizaciones oficiales y los consorcios de fabricantes. Por otra parte, también existen los "estándares propietarios" que son propiedad absoluta de una corporacion u entidad. El primer tipo de organismo está integrado por consultores independientes, integrantes de departamentos o secretarias de estado de diferentes paises y otras entidades particulares autorizadas por los estados. Ejemplos de este tipo de organizaciones son la ITU (UIT), ISO, ANSI, IEEE, IETF, IEC, entre otras. (mintic, s.f.)

¿PARA QUÉ SIRVE UN ESTÁNDAR EN TELECOMUNICACIONES?

Los estándares facilitan el acceso, la interconexion, la integracion y la convergencia de equipos, redes y servicios de telecomunicaciones alrededor del mundo. Los estándares facilitan el intercambio internacional de bienes y servicios y desarrollan la cooperacion en la esfera de la actividad intelectual, científica, tecnologica y economica. Los estándares crean nichos de mercado, proyectan economias de escala y promueven la competencia. (mintic, s.f.)

¿QUÉ ES INFORMACIÓN ANÁLOGA Y DIGITAL?

La informacion analogica es aquella que puede tomar infinitos valores, codificándose mediante combinaciones de las cifras que van del 0 al 9, para transmitirlos a través de cualquier sistema de comunicacion. La Informacion digital es, en cambio, aquella en que la codificacion toma únicamente dos valores: "0" o "1". Los datos alfanuméricos son originariamente señales digitales; pero no es el caso de la voz y de las imágenes, que necesitan digitalizarse para convertirse en formato digital. Se denomina digitalizacion, el proceso de conversion de una señal analoga en digital. Cada tipo de señal (analogica o digital) requiere distinta

capacidad de proceso caracterizándose por el ancho de banda y la velocidad de transmisión que requiere su transporte: a mayor cantidad de información, mayor ancho de banda y velocidad se precisan. (mintic, s.f.)

¿QUÉ SON: AMPS, CDMA, TDMA, GPRS Y PCS, ¿TÉRMINOS TECNOLOGICOS UTILIZADOS A MENUDO EN PROPAGANDAS Y COMERCIALES?

- AMPS (Advanced Mobile Phone System). Sistema de Teléfono Móvil Avanzado: Fue la especificación estándar original o la primera generación de los sistemas analógicos de la telefonía móvil.
- CDMA (Code Division Multiple Access). Acceso Múltiple por División de Código: Tecnología para la transmisión digital de señales de radio entre, por ejemplo, un teléfono móvil y una estación base de radio. En CDMA, una frecuencia se divide en una cantidad de códigos.
- TDMA (Time Division Multiple Access). Acceso Múltiple de División de Tiempo: Es la actualización digital estándar analógico AMPS. Formalmente conocido como D-AMPS (Digital Advanced Multiple Access / Acceso Múltiple Avanzado Digital).
- GSM (Global System for Mobile Communications). Sistema Global para Comunicaciones Móviles: Originalmente desarrollado como estándar europeo para la telefonía móvil digital. GSM opera en las frecuencias de 900 MHz, 1800 MHz y 1900 MHz.
- PCS (Personal Communications Services). Servicios de Comunicaciones Personales: Término colectivo que se refiere a los servicios de telefonía móvil en América, en la banda de frecuencia de 1900 MHz.
- PDC (Personal Digital Cellular). Celular Digital Personal: Estándar japonés para la telefonía móvil digital en las bandas anchas 800 MHz y 1500 MHz.
- CDPD (Cellular digital Packet Data). Datos de Paquetes Digitales Celulares: es el primer paso para brindar servicios de Internet móvil en el estándar de telefonía móvil TDMA. Los datos CDPD se transfieren en paquetes por canales que no están siendo utilizados por canales de voz. Es un sistema compartido por voz y datos que utiliza una capacidad que de lo contrario sería desaprovechada por la transmisión de voz.
- GPRS (General Packet Radio Service). Servicio de Radio de Paquetes Generales: Actualización de las redes móviles existentes que posibilita estar

siempre "en linea" y brinda una Internet mucho más rápida en el teléfono móvil

- CDMA 2000: Tecnología de radio transmisión que favorece la evolución cdmaOne/IS-95 de banda angosta a la tercera generación, sumando múltiples operadores.
- SMS (Short Message Service). Servicio de Mensajes Cortos: servicio disponible en redes digitales que permite enviar y recibir hasta 160 caracteres en un teléfono móvil, a través del centro de mensajes del operador de la red.
- UMTS (Universal Mobile Telecommunications System). Sistema Universal de Telecomunicaciones Móviles: Designación del estándar de telefonía móvil de tercera generación en Europa, normalizado por ETSI.
- ADSL (Asymmetrical Digital Subscriber Line). Línea de Abonado Digital Asimétrica: Método para aumentar la velocidad de transmisión en un cable de cobre. ADSL facilita la división de la capacidad en un canal con velocidad mayor hacia el abonado, en general para la transmisión de video, y un canal con velocidad mucho menor en la otra dirección.
- SDH (Synchronous Digital Hierarchy). Jerarquía Digital Síncrona: Estándar para la transmisión de señales digitales dentro de redes de transporte. (mintic, s.f.)

¿QUÉ ES WIFI?

Es una tecnología que permite la conexión inalámbrica a internet y a redes locales (LAN) de dispositivos electrónicos. La palabra "Wi-Fi" es una abreviatura de "Wireless Fidelity". Funciona utilizando ondas de radio para transmitir datos entre dispositivos, como computadoras, teléfonos inteligentes, tabletas, impresoras y otros dispositivos compatibles.

La tecnología Wi-Fi se basa en estándares definidos por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), específicamente en la familia de estándares IEEE 802.11. Estos estándares establecen las especificaciones técnicas para la comunicación inalámbrica, incluyendo la frecuencia de operación, la velocidad de transferencia de datos y los protocolos de seguridad.

¿QUÉ ES WIMAX?

El organismo internacional de estandarización IEEE (Institute of Electrical and Electronic Engineers) aprobó en abril del 2003 el estándar IEEE 802.16a que delimita las reglas tecnológicas que deben seguir los fabricantes de la infraestructura inalámbrica en banda ancha (Air Interface for Fixed Broadband Wireless Access Systems), y que permite operar redes en grandes superficies WMAN, entre los 2 GHz a los 11 GHz, mediante un esquema Fijo Punto a Multipunto y que soporta modos de operación tanto TDD (Time Division Duplex) como FDD (Frequency Division Duplex). El estándar IEEE 802.16a, utiliza una tecnología basada en OFDM con 256 subcarreras, permite conexiones de hasta 50 kilómetros, sin necesidad de línea de visión directa (NLOS), con capacidad para transmitir datos a una velocidad de 70 Mbps con una tasa máxima de 5.0 bps/Hz y podrá soportar usuarios en una escalabilidad de canales de 1.5 MHz a 20 MHz. Este estándar soporta niveles de servicio (SLA) y calidad de servicio (QoS). En junio de 2004, IEEE expidió el estándar IEEE 802.16d, conocido como la revisión 2004 que realiza mejoras al IEEE 802.16a. En diciembre de 2005 el IEEE aprobó el nuevo estándar IEEE 802.16e que dota de movilidad a los terminales de las redes de área metropolitana inalámbricas WMAN.

El nuevo estándar se centra en la adición de capacidades para soportar movilidad, incluye mejoras para uso en interiores con modems auto instalables por el usuario, la optimización del consumo de energía para los dispositivos móviles y la disminución del tamaño del modem de usuario CPE (Customer Premise Equipment). WiMAX Forum o Alianza por la Interoperabilidad para el Acceso en Microondas, por sus siglas en inglés (Worldwide Interoperability for Microwave Access), es una organización sin fines de lucro, creada con el objeto de promover y certificar la interoperabilidad de los productos inalámbricos de banda ancha de conformidad con los estándares IEEE 802.16. A la fecha, son miembros del WiMAX Forum más de 300 empresas. El Foro Wimax persigue la ambiciosa meta de que los estándares IEEE 802.16., sean la tecnología inalámbrica que unifique el mundo de la telefonía móvil y las redes de datos, en torno a bandas concretas del espectro de frecuencia, fundamentalmente en las bandas de 2.3 GHz, 2.5 GHz, 3.5 GHz y 5.8 GHz. (mintic, s.f.)

¿QUÉ NORMAS DEL MINISTERIO DE TECNOLOGIAS DE INFORMACION Y COMUNICACIONES REGLAMENTAN LAS TECNOLOGIAS GSM, CDMA Y WIMAX ENTRE OTRAS?

Las normas reglamentarias nacionales versan sobre las redes, sistemas y servicios de telecomunicaciones, más no sobre las tecnologías, las cuales en si mismas son neutrales y pueden ser aptas para la configuración de las redes y la prestación de los servicios de telecomunicaciones. Colombia mantiene una política de neutralidad tecnológica, donde la regulación es independiente de la tecnología. Sin embargo, los desarrollos normativos inalámbricos toman en consideración los estudios y análisis de los diferentes estándares tecnológicos y las Recomendaciones de la UIT, para la planeación y canalización del espectro radioeléctrico. (mintic, s.f.)

¿QUÉ ES BANDA ANCHA Y CUÁL ES LA VELOCIDAD MÍNIMA PARA BANDA ANCHA EN COLOMBIA?

La Resolución 2064 de 2005 define: BANDA ANCHA: Técnica de transmisión que mediante el uso de tecnologías digitales permite la telecomunicación, entre otras, de voz, sonidos, datos, imágenes y video, por un mismo canal, con velocidades que garantizan calidad de servicio, y que proporciona la integración de facilidades de telecomunicación y el acceso a la información. Por su parte, la Resolución 1740 de 2007 de la Comisión de Regulación de Telecomunicaciones CRT define como velocidad mínima para la Banda Ancha en Colombia 512 Kb. (mintic, s.f.)

GRANDES FAMILIAS DE PROTOCOLOS DE TRANSMISIÓN Y RECEPCIÓN DE DATOS

PROTOCOLOS DE RED

PROTOCOLO DE RED	LICENCIA	PLATAFORMAS	DESCRIPCIÓN	FUNCIONES	PUNTOS DÉBILES
IP (Internet Protocol), protocolo de Internet	Libre (RFC 791 / 2460)	Todas	Sin conexión; longitud de la dirección: 128 bits (IPv6) / 32 bits (IPv4)	Enrutamiento, direccionamiento	Pila de protocolos muy extensa; las funciones de seguridad no están implementadas desde el principio (IPv4)
ARP (Address Resolution Protocol), protocolo de resolución de direcciones	Libre (RFC 826)	Todas	Interfaz entre las capas 2 y 3 con función propia de memoria caché	Resolución de direcciones (encontrar la dirección MAC para la dirección IP correspondiente) en IPv4	No es posible comprobar si la resolución es correcta, lo que implica el riesgo de ARP spoofing o envenenamiento de tablas ARP
NDP (Neighbor Discovery Protocol)	Libre (RFC 4861)	Todas	Enlace entre las capas 2 y 3 con memoria caché propia	Resolución de direcciones para IPv6; identificación de direcciones IP dobles	La protección contra spoofing no está integrada de forma estándar, sino que necesita la extensión SEND
ICMP (Internet Control Message Protocol)	Libre (RFC 792)	Todas	Componente autónomo de IPv4	Intercambio de notificaciones de información y de errores	Puede ser usado para llevar a cabo ataques DoS/DDoS
SNA (Systems Network Architecture)	Propietario (IBM)	Dispositivos IBM	Antigua arquitectura de red jerárquica con diferentes protocolos	Conecta a los ordenadores y a sus recursos en redes SNA	La conexión con redes no SNA era muy complicada; comparativamente alto coste
NBF (NetBIOS Frames Protocol)	Propietario (Microsoft)	Windows (hasta 2000 incluida)	Antiguo protocolo para sistemas Windows	Comunicación con la capa de representación	No enrutable; indicado solo para redes pequeñas

				(NetBIOS) y la capa de seguridad (LLC)	(hasta 20 ordenadores)
IPX (Internetwork Packet Exchange)	Propietario (Novell)	NetWare (obsoleto), Linux, Windows	Protocolo sin conexión, funcionalmente parecido a IP; longitud de dirección: 80 bits (dirección de host de 48 bits, número de red de 32 bits)	Enrutamiento, direccionamiento	No indicado para redes WAN muy grandes
DDP (Datagram Delivery Protocol)	Propietario (Apple)	Dispositivos que soportan AppleTalk (hasta Mac OS X 10.6 incluida)	Componente de la pila de protocolos de AppelTalk (ya no es soportado); longitud de dirección: máximo 13 bytes (encabezado) y 587 bytes (datos de uso)	Enrutamiento, direccionamiento	Riesgo de spoofing de AppelTalk; caudal útil o goodput (tasa de paquetes que llegan a destino por unidad de tiempo) más débil por el tamaño reducido de los paquetes
OSPF (Open Shortest Path First)	Libre (RFC 2328)	Todas	Protocolo de red basado en el algoritmo de Dijkstra, especialmente indicado para grandes redes corporativas	Optimiza el routing en relación con los costes de transmisión; distribución dinámica de la carga	Procesador con alta capacidad de carga y elevados requerimientos de almacenamiento; configuración y mantenimiento complejos

(Ionos, s.f.)

MEDIOS DE TRANSMISIÓN GUIADOS Y NO GUIADOS

MEDIOS DE TRANSMISIÓN GUIADOS

CABLE DE COBRE

El cable de cobre es uno de los medios de transmisión más comunes en las redes de computadoras. Se utiliza para transmitir datos a través de señales eléctricas que viajan a lo largo de los cables de cobre.

Los cables de cobre pueden clasificarse en diferentes tipos, como par trenzado (UTP, STP), cable coaxial y cable de par trenzado apantallado (FTP).

FIBRA ÓPTICA

La fibra óptica es un medio de transmisión que utiliza haces de luz para enviar datos a través de fibras de vidrio o plástico.

Ofrece ventajas como mayor ancho de banda, menor atenuación de la señal, mayor inmunidad al ruido electromagnético y mayor seguridad contra la interceptación de datos en comparación con los cables de cobre.

MEDIOS DE TRANSMISIÓN NO GUIADOS

ONDAS DE RADIO

Las ondas de radio se utilizan en tecnologías inalámbricas como Wi-Fi, Bluetooth, y redes celulares (2G, 3G, 4G, 5G).

Estas ondas son emitidas por antenas y viajan a través del aire, permitiendo la comunicación inalámbrica entre dispositivos.

MICROONDAS

Las microondas son ondas electromagnéticas de alta frecuencia que se utilizan para la comunicación inalámbrica en distancias cortas o medianas.

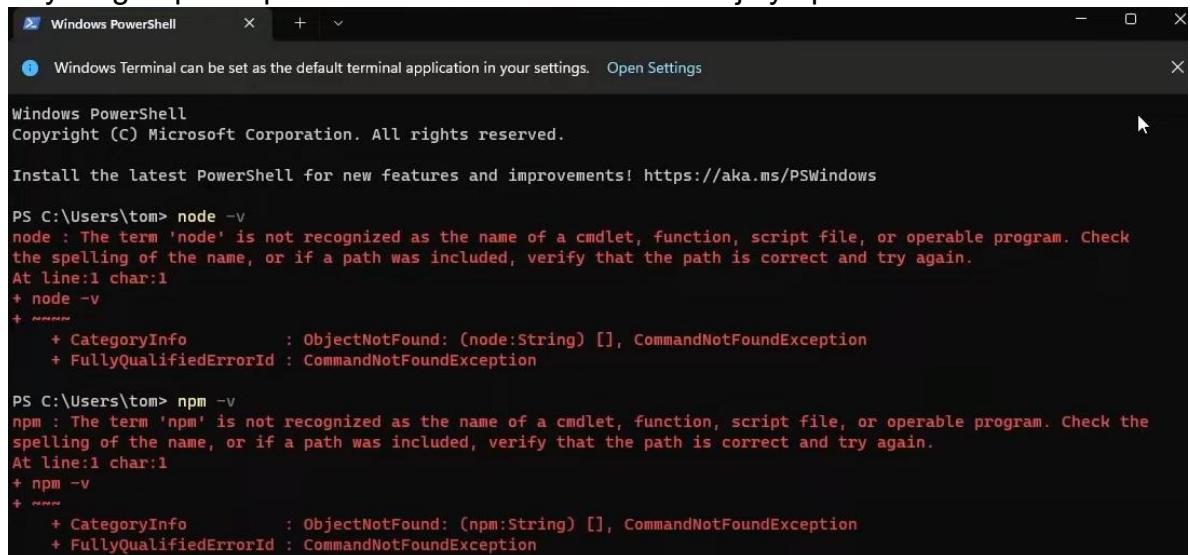
Se utilizan en tecnologías como enlaces punto a punto para comunicaciones de larga distancia y en sistemas de comunicación por satélite.

DESPLIEGUE DE LA APLICACIÓN

Para el despliegue de este proyecto, se optó por utilizar Node.js y Express como las herramientas principales. Estas tecnologías fueron empleadas para simular un servidor y facilitar el despliegue local. La elección de Node.js y Express se basó en su capacidad para proporcionar una combinación óptima de ventajas: facilidad de desarrollo, eficiencia en el manejo de solicitudes, flexibilidad y un amplio ecosistema de módulos. Esta combinación hace que sean una opción altamente atractiva para una amplia gama de desarrolladores web.

PASOS DEL DESPLIEGUE LOCAL

1. Abre el símbolo del sistema (cmd) y ejecuta los siguientes comandos: node -v y luego npm -v para Verificar la versión de Node.js y npm



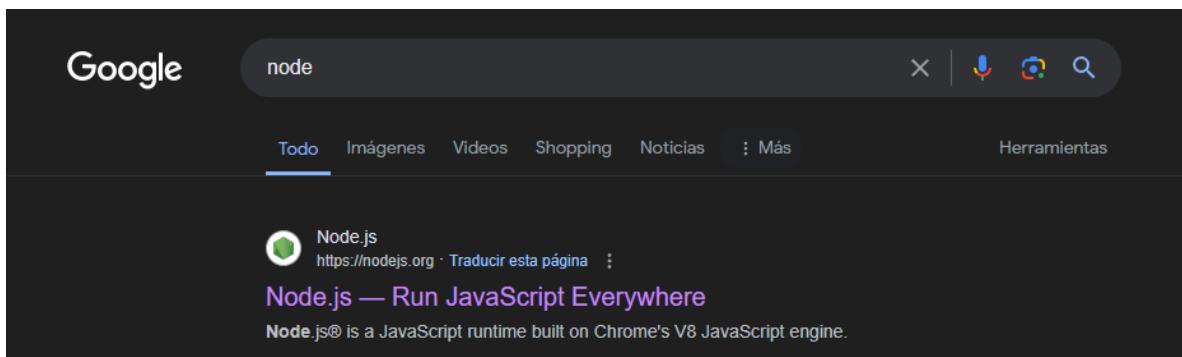
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

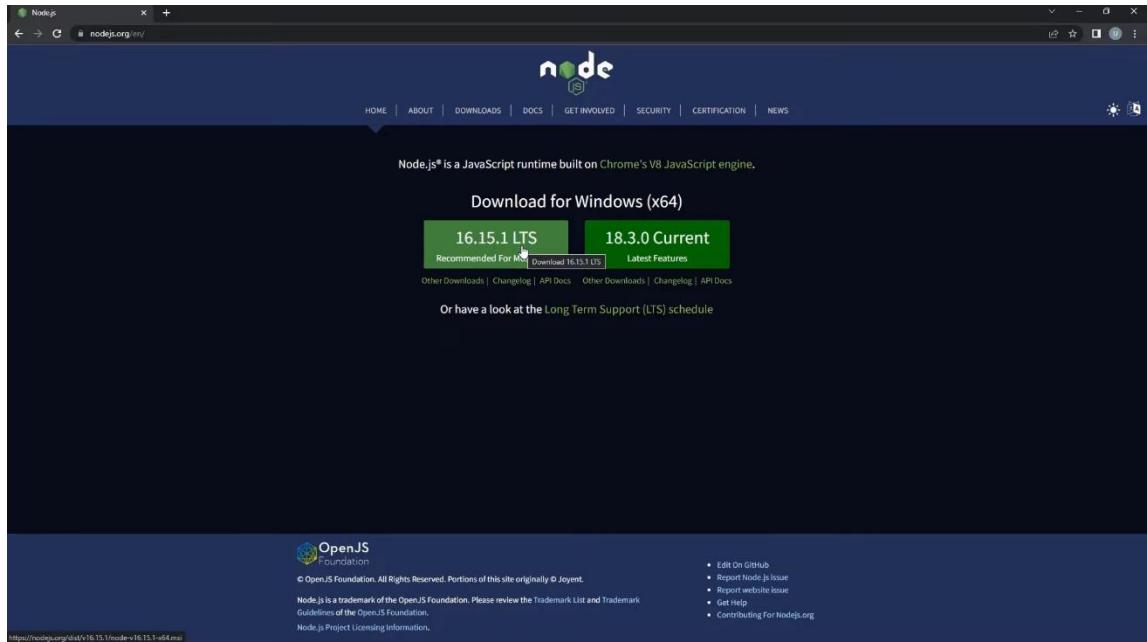
PS C:\Users\tom> node -v
node : The term 'node' is not recognized as the name of a cmdlet, function, script file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ node -v
+ ~~~
    + CategoryInfo          : ObjectNotFound: (node:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\tom> npm -v
npm : The term 'npm' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ npm -v
+ ~~~
    + CategoryInfo          : ObjectNotFound: (npm:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

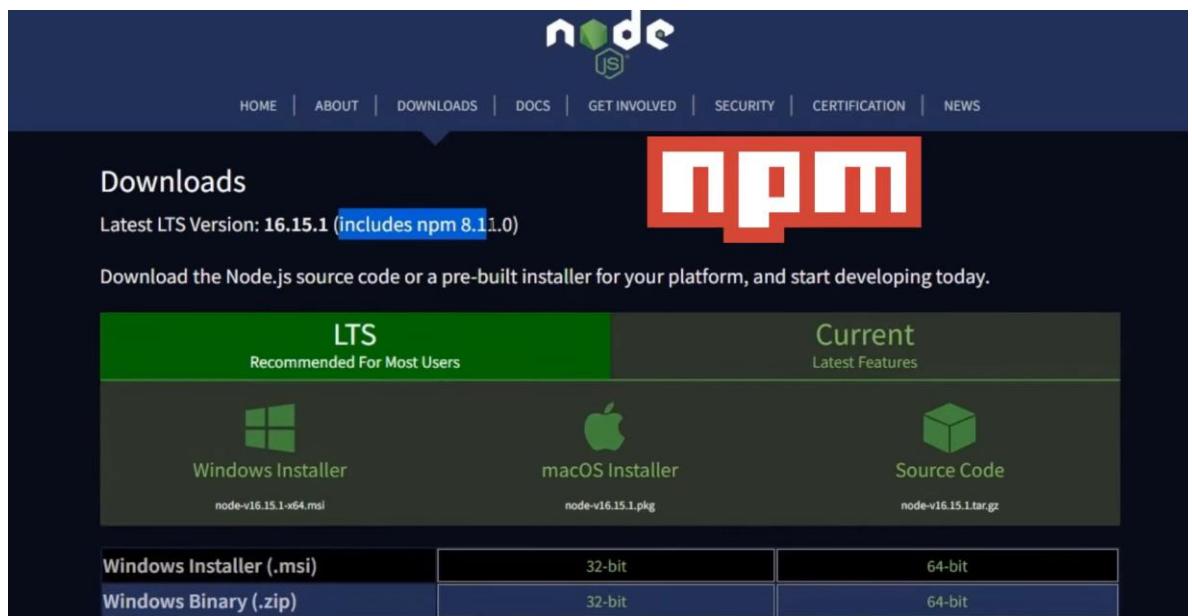
2. Para instalarlo en el navegador se busca la palabra node y se selecciona la opción que dice Node.js – run JavaScript Everywhere



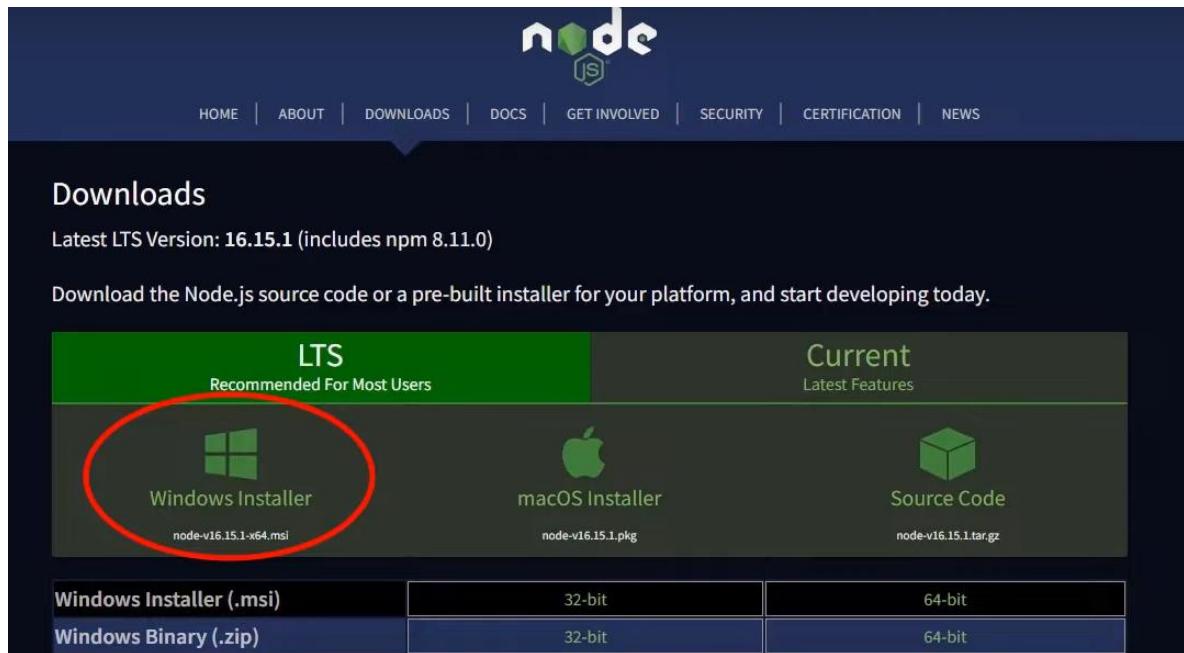
3. A continuación, se abre esta pagina de node : <https://nodejs.org/en/> donde se seleccionara la opción que dice LTS porque tiene soporte a largo plazo



4. En la página oficial de nodejs.org, no solo se instalará el entorno de ejecución sino también se instalara el instalador de dependencias npm, que es el que se evidencia en la siguiente imagen



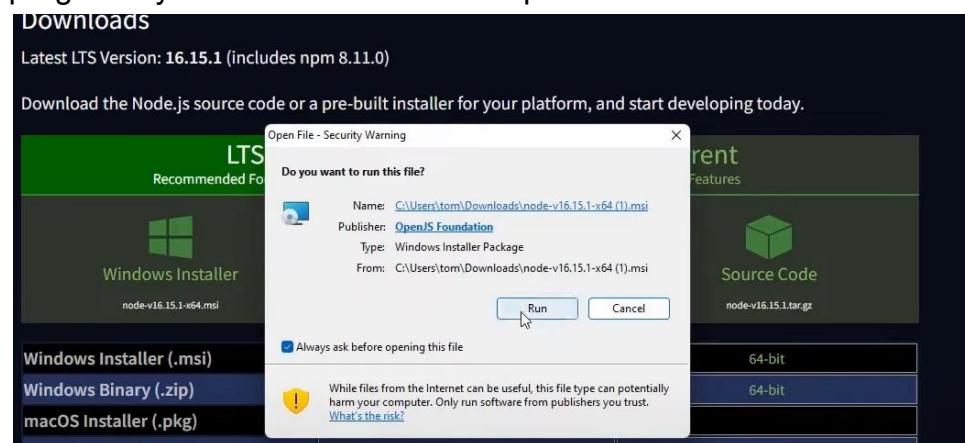
5. Se le da click donde dice Windows installer como se evidencia en la siguiente imagen



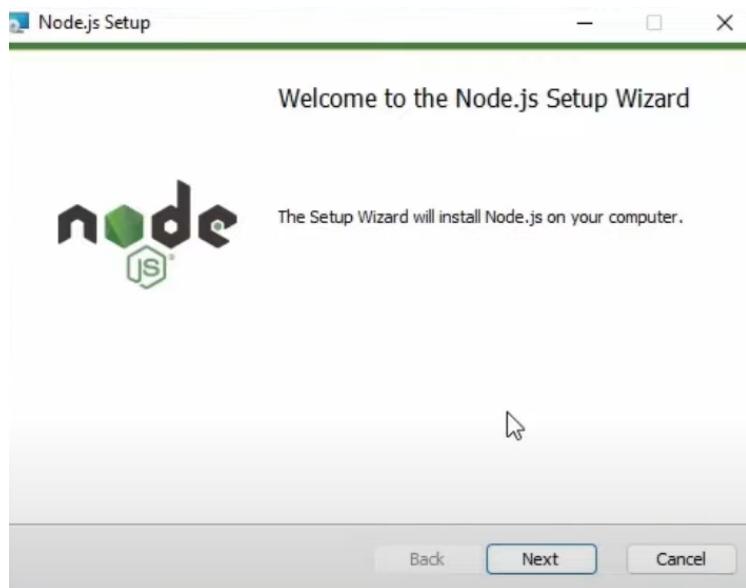
6. Cuando termine de descargarse hacemos click al archivo que descargamos



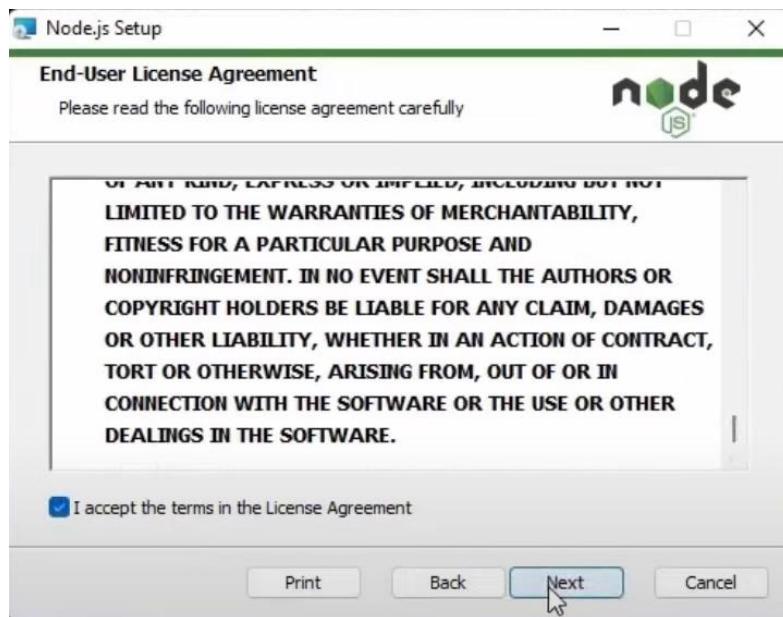
7. Se abrirá esta ventana donde nos preguntara si esta seguro de correr este programa y le damos click al botón que dice Run



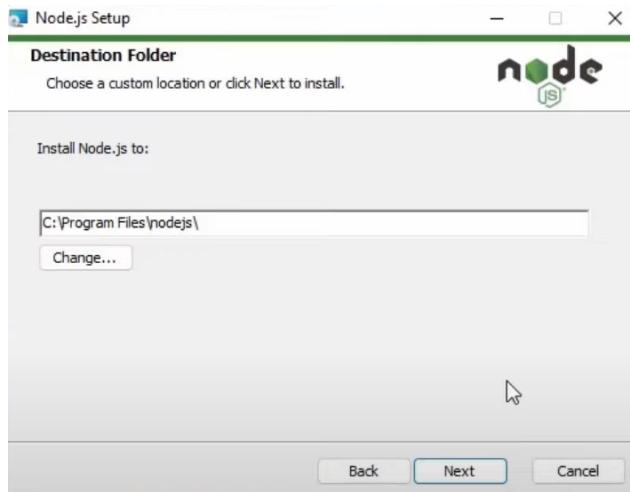
8. Aparece el wizard de node.js donde nos da la bienvenida y hacemos click sobre el botón que dice next



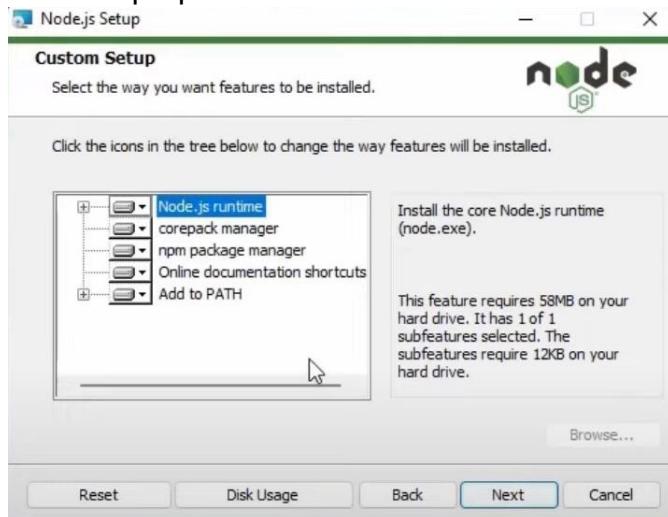
9. Luego vemos los términos de la licencia, leemos toda la licencia para después dar click sobre el checklist y oprimir el botón next



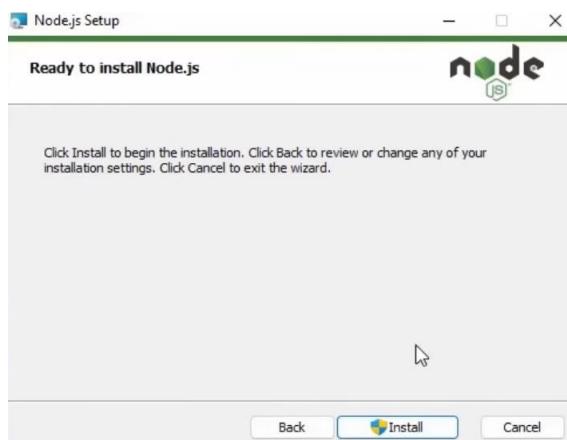
10. A continuación, nos dice donde se va a instalar node.js, en este caso la ruta por defecto. De acuerdo con el usuario se cambia la ruta dependiendo de las necesidades del usuario. Se le da click sobre el botón next



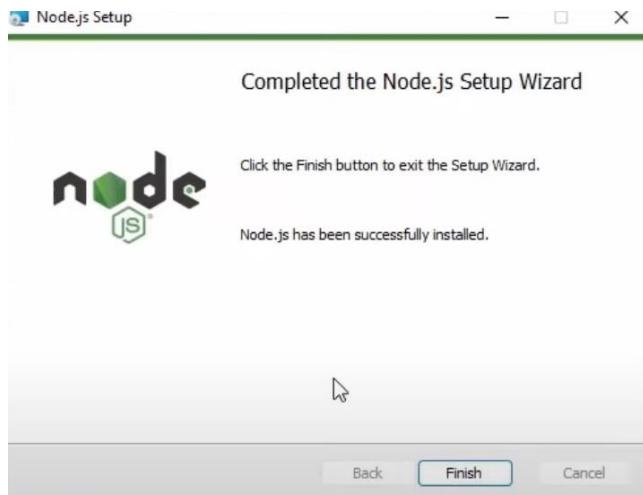
11. En la siguiente imagen aparecen los paquetes que se van a instalar a parte de node.js y le damos click sobre el botón next después de leer la descripción de cada paquete



12. Finalmente, le damos click sobre la opción install



13. Se abrirá una ventana emergente para que node.js haga cambios a nuestro ordenador, seleccionamos la opción si
14. Luego de esto, se va a instalar correctamente y le damos a la opción de terminar



15. Se vuelve a abrir un cmd o powerShell para verificar si node.js quedo instalado correctamente poniendo en la consola los comandos node -v y npm -v donde deben aparecer las versiones de cada una como se evidencia en la imagen

A screenshot of a Windows PowerShell window titled "Windows PowerShell". The window shows the following command-line session:

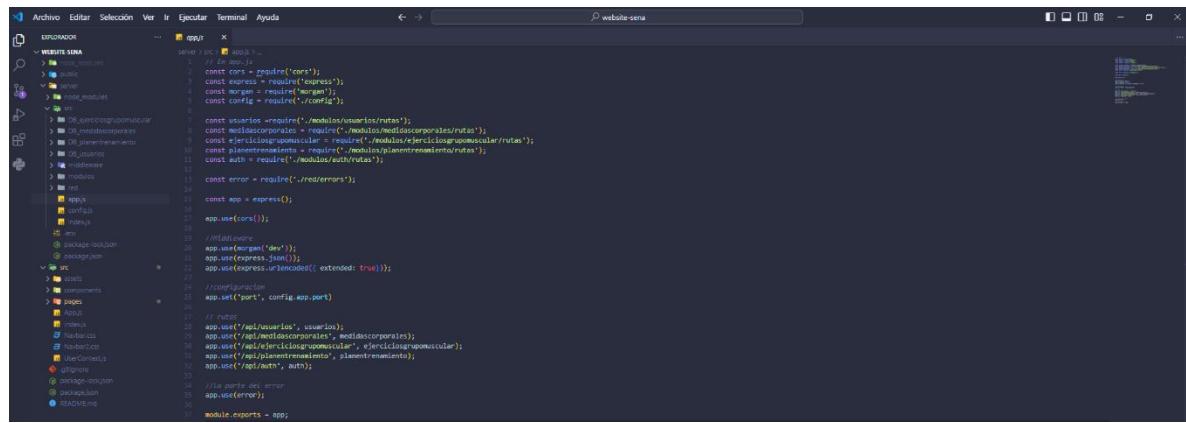
```
PS C:\Users\tom> node -v
v16.15.1
PS C:\Users\tom> npm -v
npm [WARN config] global '--global', '--local' are deprecated. Use '--location=global' instead.
8.11.0
```

16. El siguiente paso es instalar Express en tu proyecto Node.js. Puedes hacerlo utilizando el administrador de paquetes npm, que viene integrado con Node.js. Lo primero que hay que hacer es abrir Visual Studio Code y asegúrate de que estás en el directorio de tu proyecto Node.js.
17. Abre una terminal integrada en Visual Studio Code. Puedes hacerlo seleccionando "Terminal" en la barra de menú y luego "Nueva terminal", o simplemente presionando Ctrl+Shift+`.
18. En la terminal, ejecuta el siguiente comando para instalar Express como una dependencia de tu proyecto: **npm install express**

19. Esto descargará e instalará Express en tu proyecto, junto con todas sus dependencias. Una vez que la instalación esté completa, podrás empezar a utilizar Express en tu aplicación Node.js.

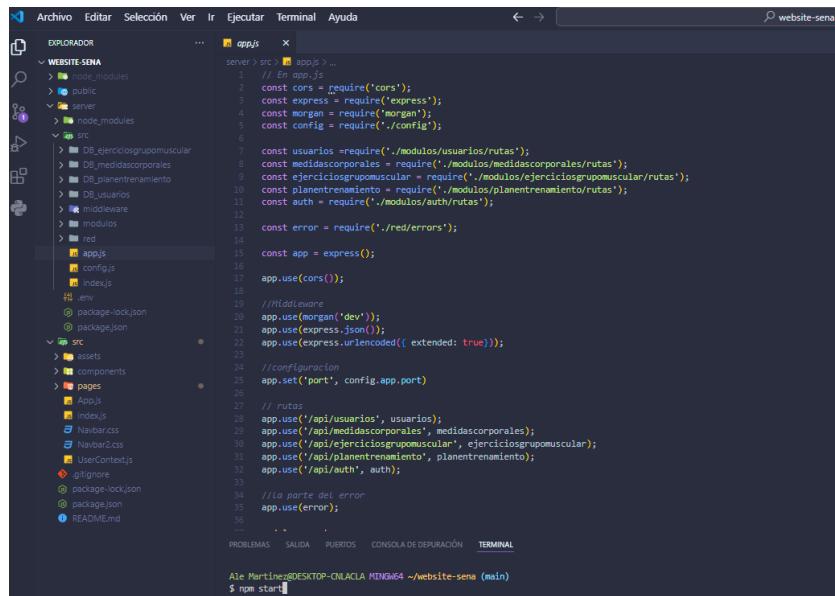
CAPTURAS DE PANTALLA DEL DESPLIEGUE LOCAL

1. Para utilizar Express en tu código, simplemente necesitas importarlo en tu archivo JavaScript principal. En el contexto de este proyecto, se creó un servidor web con Express, y el archivo principal se ve de la siguiente manera:



```
server > src > app.js > ...
1 // En app.js
2 const cors = require('cors');
3 const express = require('express');
4 const morgan = require('morgan');
5 const config = require('./config');
6
7 const usuarios = require('../modulos/usuarios/rutas');
8 const medidascorporales = require('../modulos/medidascorporales/rutas');
9 const ejerciciosgrupomuscular = require('../modulos/ejerciciosgrupomuscular/rutas');
10 const planentrenamiento = require('../modulos/planentrenamiento/rutas');
11 const auth = require('../modulos/auth/rutas');
12
13 const error = require('../red/errors');
14
15 const app = express();
16
17 app.use(cors());
18
19 //Middleware
20 app.use(morgan('dev'));
21 app.use(express.json());
22 app.use(express.urlencoded({ extended: true }));
23
24 //rutas
25 app.use('/api/usuarios', usuarios);
26 app.use('/api/medidascorporales', medidascorporales);
27 app.use('/api/ejerciciosgrupomuscular', ejerciciosgrupomuscular);
28 app.use('/api/planentrenamiento', planentrenamiento);
29 app.use('/api/auth', auth);
30
31 //La parte del error
32 app.use(error);
33
34 module.exports = app;
```

2. A continuación, se usará el comando **npm start**, que se utiliza comúnmente para iniciar una aplicación. Cuando ejecutas npm start en el directorio de un proyecto Node.js, npm buscará en el archivo package.json de tu proyecto la propiedad "scripts" y ejecutará el script asociado con la clave "start".



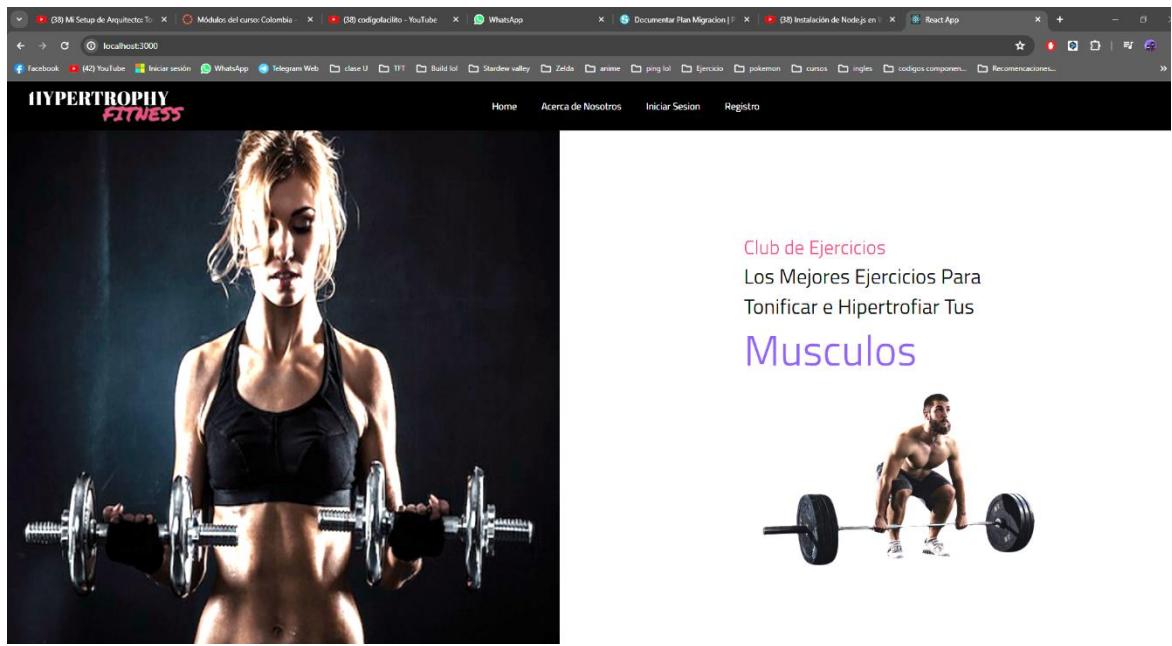
```
Ale Martinez@DESKTOP-ONLACIA MINGW64 ~/website-sena (main)
$ npm start
[1] 11888 ? website-sena
[1]+ 11888 ? website-sena`npm start`
```

3. Si todo está configurado correctamente, se mostrará lo siguiente, tal como se muestra en la imagen adjunta.

```

  Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
  🔍 Explorador ... app.js
  WEBSITE-SENA
  > node_modules
  > public
  < server
  > node_modules
  < src
    > DB_ejerciciosgrupomuscular
    > DB_medidascorporales
    > DB_planentrenamiento
    > DB_usuarios
    > middleware
    > modulos
    > red
      > app.js
      > config.js
      > index.js
    < .env
    < package-lock.json
    < package.json
  < src
    > assets
    > components
    > pages
      > App.js
      > index.js
      < Navbar.css
      < Navbar2.css
      < UserContext.js
    < .gitignore
    < package-lock.json
    < package.json
    < README.md
  < ESQUEMA
    < cors
    < express
    < morgan
    < config
    < usuarios
    < medidascorporales
    < ejerciciosgrupomuscular
  > LÍNEA DE TIEMPO
  X main.js ⌂ ⌂ 0 △ 0 ⌂ 0
  🔍 website-sena
  server > src > app.js > ...
  2 const cors = require('cors');
  3 const express = require('express');
  4 const morgan = require('morgan');
  5 const config = require('./config');
  6
  7 const usuarios = require('../modulos/usuarios/rutas');
  8 const medidascorporales = require('../modulos/medidascorporales/rutas');
  9 const ejerciciosgrupomuscular = require('../modulos/ejerciciosgrupomuscular/rutas');
  10 const planentrenamiento = require('../modulos/planentrenamiento/rutas');
  11 const auth = require('../modulos/auth/rutas');
  12
  13 const error = require('../red/errors');
  14
  15 const app = express();
  16
  17 app.use(cors());
  18
  19 //Middleware
  20 app.use(morgan('dev'));
  21 app.use(express.json());
  22 app.use(express.urlencoded({ extended: true }));
  23
  24 //configuracion
  25 app.set('port', config.app.port)
  26
  27 // rutas
  PROBLEMAS SALIDA PUERTOS CONSOLA DE DEPURACIÓN TERMINAL
  Search for the keywords to learn more about each warning.
  To ignore, add // eslint-disable-next-line to the line before.
  Note that the development build is not optimized.
  To create a production build, use npm run build.
  Compiled successfully!
  You can now view website-sena in the browser.
  Local: http://localhost:3000
  On Your Network: http://192.168.20.23:3000
  Compiled successfully!
  You can now view website-sena in the browser.
  Local: http://localhost:3000
  On Your Network: http://192.168.20.23:3000
  Note that the development build is not optimized.
  To create a production build, use npm run build.
  webpack compiled successfully
  
```

4. Cuando ejecutas el comando `npm start`, se abrirá automáticamente una ventana en tu navegador. En esta ventana, podrás ver la página web en la que estás trabajando, la cual pertenece al proyecto denominado "hypertrophy fitness". Al observar la URL en la barra de direcciones, notarás que el proyecto se encuentra en un entorno local, ya que la dirección comienza con "localhost", seguido de dos puntos y el puerto 3000.



- Además, en Visual Studio Code, en la terminal, se utiliza el comando cd server para acceder a la carpeta donde se estableció la conexión con la base de datos.

```

archivo: app.js
server.js
index.js
node_modules
public
src
  - assets
  - components
  - pages
  - services
  - utils
  - api
  - config
  - middleware
  - models
  - routes
  - server
  - static
  - tests
  - types
  - utils
  - views
  - web
  - webpack
  - .env
  - .gitignore
  - package-lock.json
  - package.json
  - README.md

archivo: index.js
module.exports = (config) => {
  const express = require('express');
  const cors = require('cors');
  const morgan = require('morgan');
  const config = require('./config');

  const usuarios = require('../modules/usuarios/rutas');
  const medioscorporales = require('../modules/medioscorporales/rutas');
  const ejercicios = require('../modules/ejercicios/entrenamiento/rutas');
  const planentrenamiento = require('../modules/planentrenamiento/rutas');
  const auth = require('../modules/auth/rutas');

  const error = require('../red/errors');

  const app = express();
  app.use(cors());
  app.use(morgan("dev"));
  app.use(express.json());
  app.use(express.urlencoded({ extended: true }));

  //rutas
  app.use('/api/usuarios', usuarios);
  app.use('/api/medioscorporales', medioscorporales);
  app.use('/api/ejercicios/espomocular', ejercicios);
  app.use('/api/planentrenamiento', planentrenamiento);
  app.use('/api/auth', auth);

  Compiled successfully!
  You can now view website-sena in the browser.
  Local: http://localhost:3000
  On Your Network: http://192.168.20.23:3000
  Compiled successfully!

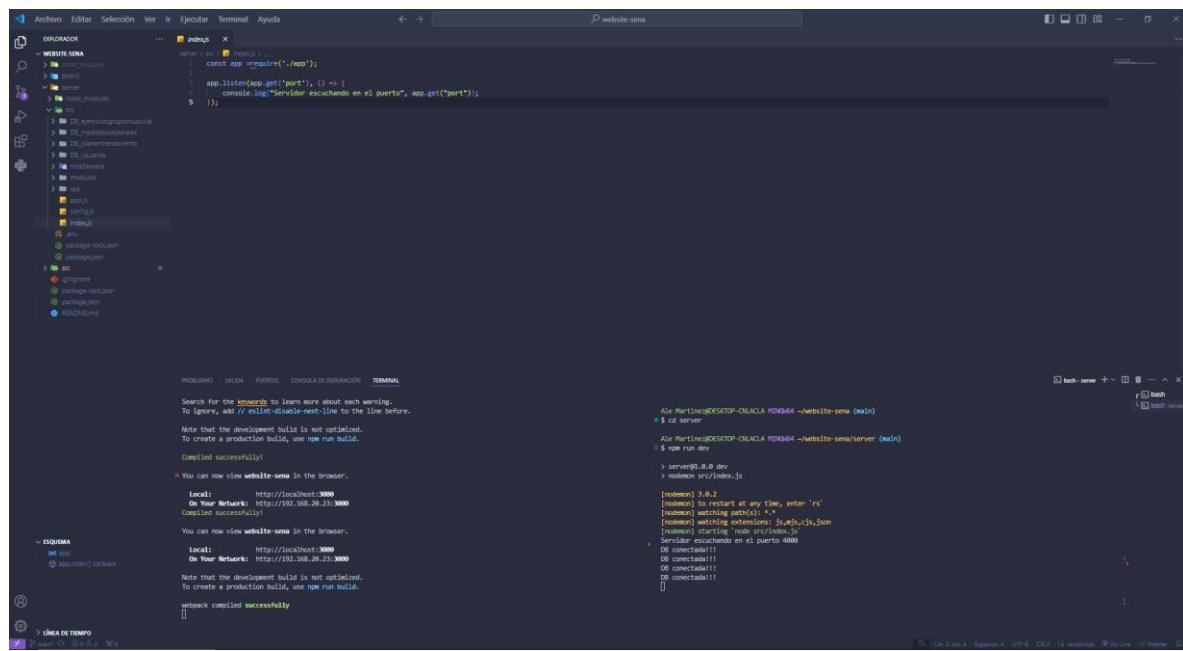
  module.exports = app;
}

archivo: webpack.config.js
module.exports = {
  entry: './src/index.js',
  output: {
    path: __dirname + '/public',
    filename: 'bundle.js'
  },
  module: {
    rules: [
      {
        test: /\.js$/,
        exclude: /node_modules/,
        use: {
          loader: 'babel-loader',
          options: {
            presets: ['@babel/preset-env']
          }
        }
      }
    ]
  }
};

archivo: package.json
{
  "name": "website-sena",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "start": "node server"
  },
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.17.1",
    "morgan": "^1.9.1"
  },
  "devDependencies": {
    "@babel/core": "^7.1.2",
    "@babel/preset-env": "^7.1.2",
    "babel-loader": "^8.0.4",
    "webpack": "^4.41.2",
    "webpack-cli": "^3.3.12"
  }
}
  
```

- En el archivo index.js, se ponen estas líneas de código ya que este código importa la configuración de la aplicación desde app.js, inicia un servidor Express en el puerto especificado en la configuración de la aplicación y luego

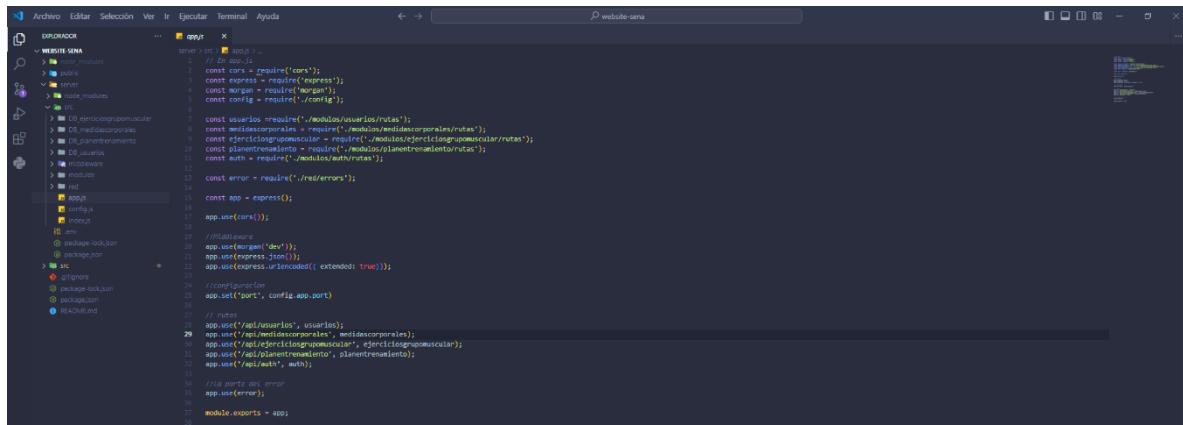
imprime un mensaje en la consola indicando que el servidor está escuchando en ese puerto.



```
ale Martínez@DESKTOP-ONLAIA MINGW64 ~/website-sena (main)
$ cd server
ale Martínez@DESKTOP-ONLAIA MINGW64 ~/website-sena/server (main)
$ npm run dev
> server@0.0.0 dev
> node main.js

(node:1016) [DEP080] To restart at any time, enter 'rl'
(node:1016) [DEP080] Watching path(s): '*'
(node:1016) [DEP080] Watching file(s): 'src/app.js,json'
(node:1016) [DEP080] Watching file(s): 'src/app.js'
  Servidor escuchando en el puerto 4000
[0]
[0]
[0]
[0]
[0]
[0]
[0]
[0]
[0]
```

7. Luego, en una carpeta llamada app.js se pone este código porque es el archivo principal de configuración de una aplicación Node.js utilizando Express como marco web. En resumen, este código configura la aplicación Express, incluyendo la configuración de middleware, la definición de rutas y la gestión de errores, preparándola para ser ejecutada como un servidor web completo.



```
ale Martínez@DESKTOP-ONLAIA MINGW64 ~/WEBSITE-SENA (main)
$ cd server
ale Martínez@DESKTOP-ONLAIA MINGW64 ~/WEBSITE-SENA/server (main)
$ npm run dev
> server@0.0.0 dev
> node main.js

(node:1016) [DEP080] To restart at any time, enter 'rl'
(node:1016) [DEP080] Watching path(s): '*'
(node:1016) [DEP080] Watching file(s): 'src/app.js,json'
(node:1016) [DEP080] Watching file(s): 'src/app.js'
  Servidor escuchando en el puerto 4000
[0]
[0]
[0]
[0]
[0]
[0]
[0]
[0]
```

8. Despues, se pone el comando **npm run dev** porque es una convención común utilizada en el desarrollo de proyectos Node.js para iniciar un servidor de desarrollo local.

```

server > npm run dev
...
const cors = require('cors');
const express = require('express');
const mongoose = require('mongoose');
const config = require('./config');

const usuarios = require('./modulos/usuarios/rutas');
const medicoscorporales = require('./modulos/medicoscorporales/rutas');
const ejerciciosgrupomuscular = require('./modulos/ejerciciosgrupomuscular/rutas');
const planentrenamiento = require('./modulos/planentrenamiento/rutas');
const auth = require('./modulos/auth/rutas');

const error = require('./red/errors');

const app = express();
app.use(cors());
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// configuracion
app.set('port', config.app.port);

// rutas
app.use('/api/usuarios', usuarios);
app.use('/api/medicoscorporales', medicoscorporales);
app.use('/api/ejerciciosgrupomuscular', ejerciciosgrupomuscular);
app.use('/api/planentrenamiento', planentrenamiento);
app.use('/api/auth', auth);

Compiled successfully!

```

You can now view [website-sena](#) in the browser.

Local: <http://localhost:3000>
On Your Network: <http://192.168.20.23:3000>

Compiled successfully!

```

You can now view website-sena in the browser.
Local: http://localhost:3000
On Your Network: http://192.168.20.23:3000
Note that the development build is not optimized.
To create a production build, use npm run build.

wscompile compiled successfully

```

9. Por lo general, el comando `npm run dev` se define en el archivo `package.json` dentro de la sección "scripts" de esta manera:

```
{
  "scripts": {
    "dev": "node server.js"
  }
}
```

10. Finalmente, en la terminal, debería aparecer el mensaje: "Servidor escuchando en el puerto 4000", indicando que el despliegue local se ha realizado correctamente para este proyecto.

```

{
  "scripts": {
    "dev": "node server.js"
  }
}

You can now view website-sena in the browser.
Local: http://localhost:3000  
On Your Network: http://192.168.20.23:3000
```

```

Compiled successfully!

```

```

You can now view website-sena in the browser.
Local: http://localhost:3000
On Your Network: http://192.168.20.23:3000
Note that the development build is not optimized.
To create a production build, use npm run build.

wscompile compiled successfully

```

```

Alc Martínez@DESKTOP-ONLAUCA MINGW64 ~/website-sena (main)
$ cd website-sena
Alc Martínez@DESKTOP-ONLAUCA MINGW64 ~/website-sena/server (main)
$ npm run dev
> server@0.0.2 dev
> node server.js
[nodemod] 0:6.2
[nodemod] 0: restart at any time, enter 'r'
[nodemod] 0: listening on port 4000
[nodemod] 0: listening on address: 192.168.20.23:4000
[nodemod] 0: conectado!!!
[nodemod] 0: conectado!!!
[nodemod] 0: conectado!!!
[nodemod] 0: conectado!!!

```

PASOS DEL DESPLIEGUE

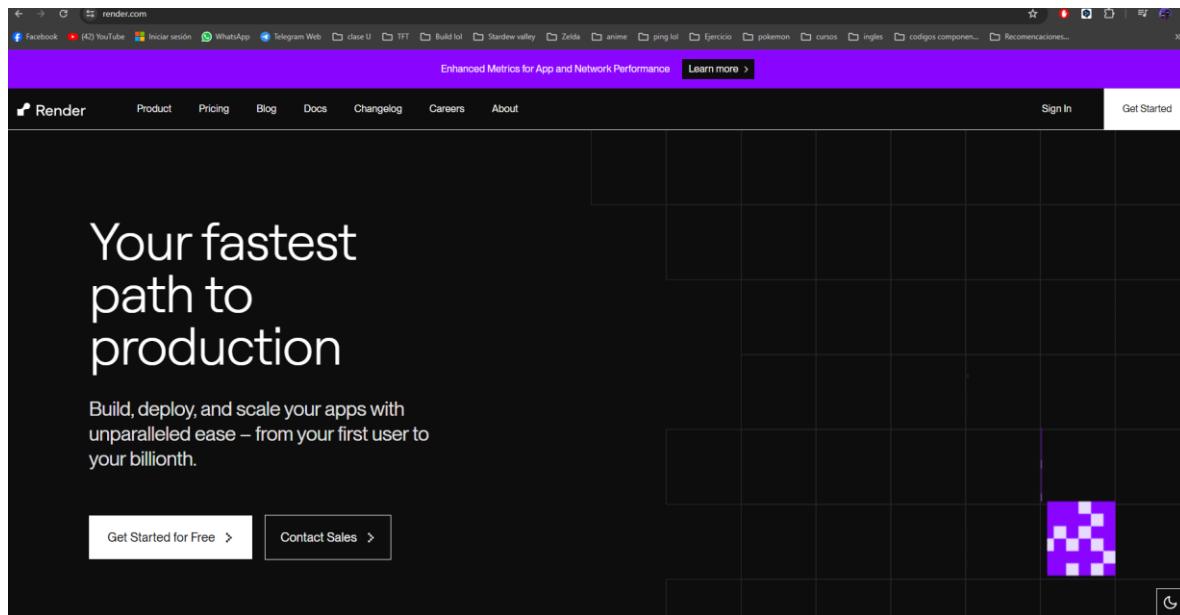
HOSTING

Para realizar el hosting, es decir, cambiar del modo desarrollador a producción, donde las personas puedan ver la página web mediante un enlace, link o URL, se utilizaron dos herramientas. A continuación, se detalla el proceso paso a paso utilizando Render para desplegar tanto el frontend como el backend.

RENDER

FRONT END

Inicialmente se entra en la página principal de render



Posteriormente, se realiza un registro en Render, donde en este caso se seleccionó la opción de GitHub, ya que allí se subió todo el proyecto desarrollado en Visual Studio Code.

Render

Create an account

or

Email

⚠ Required

Password

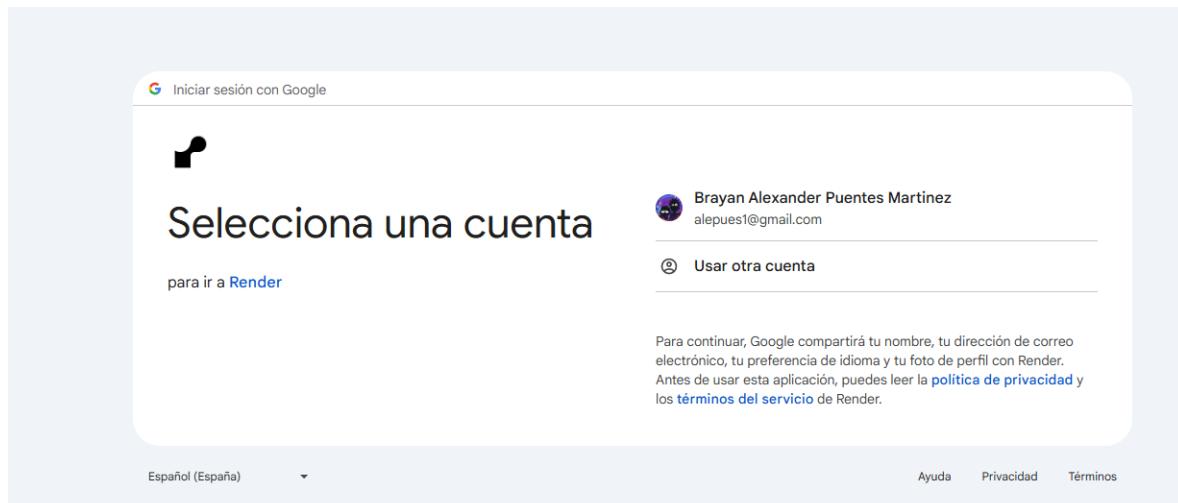
⚠ Required

By signing up you agree to our [terms of service](#).
Already have an account? [Sign in](#)

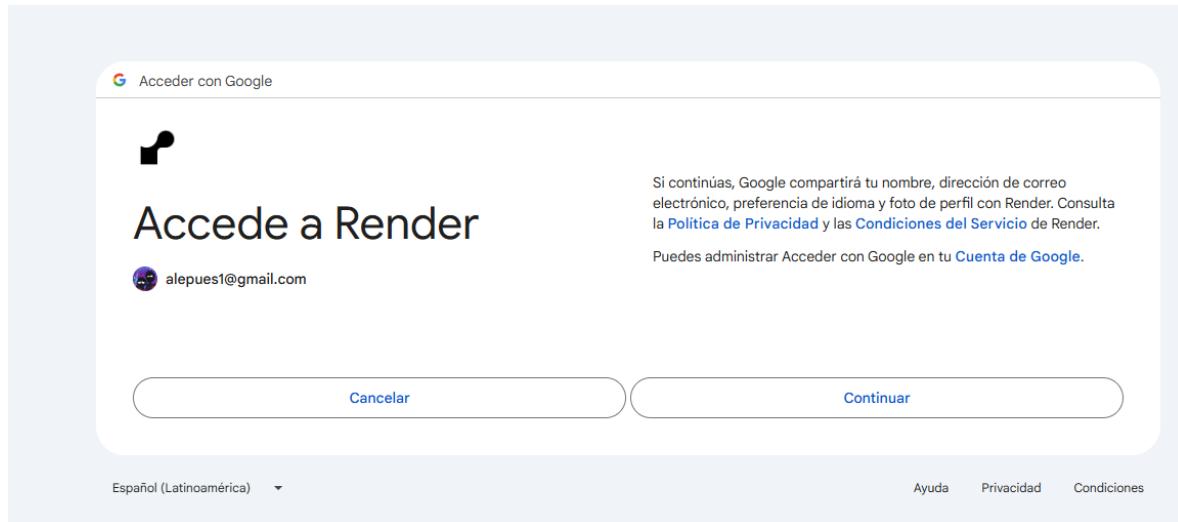
This site is protected by [Cloudflare](#). Its [Privacy Policy](#) and [Terms of Service](#) apply.

AVI ITSKOVICH, CO-FOUNDER AT WATERSHED 

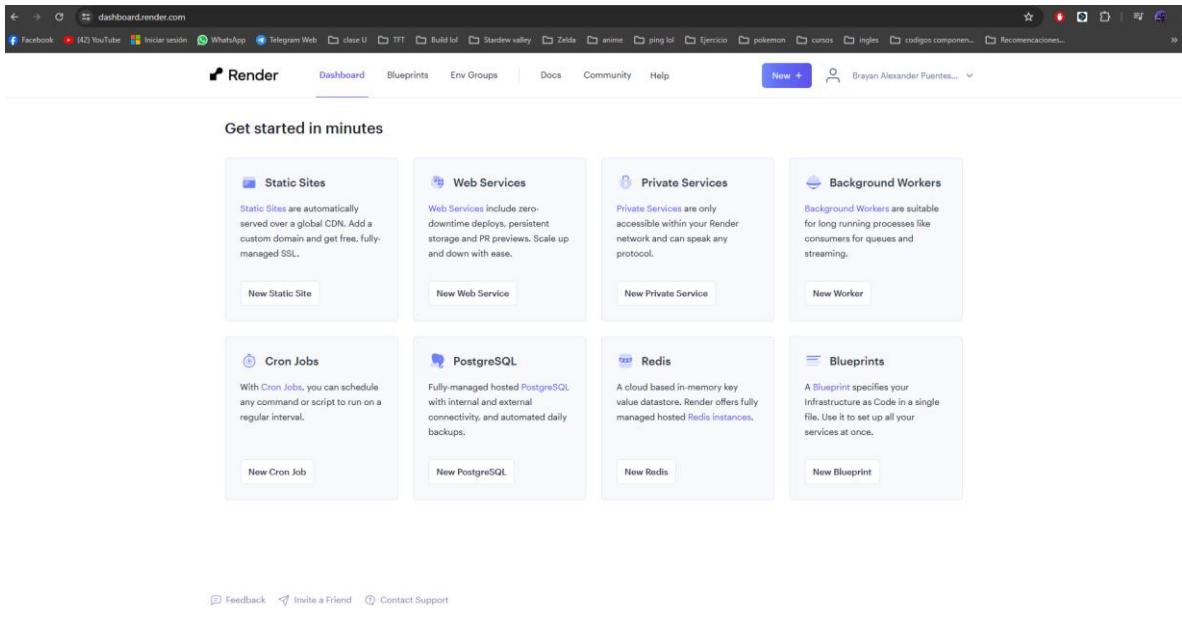
Se selecciona la cuenta para seguir haciendo el registro correspondiente en render.



Se aceptan los termino y condiciones de render para poder usarlo sin problemas.



Se inicia sesión de forma exitosa en render y se evidencia en la imagen la pagina de inicio donde aparecen varias opciones.



En este caso, se selecciona web services porque ofrece ventajas significativas en términos de tiempo, costo, escalabilidad y eficiencia, especialmente para proyectos grandes o complejos. Además, el proyecto es una página web, por esa razón se eligió esta opción.

A detailed view of the 'Web Services' card from the Render dashboard. The card features a blue icon of a globe with a grid pattern. The title 'Web Services' is in bold. The description reads: 'Web Services include zero-downtime deploys, persistent storage and PR previews. Scale up and down with ease.' Below the description is a large 'New Web Service' button with a rounded rectangle border.

En este caso se desea implementar el servicio web mediante GitHub ya que previamente se subió el proyecto a un repositorio.

Render Panel Planos Grupos ambientales Documentos Comunidad Ayuda Nuevo + Brayán Alejandro Puentes ... ▾

Crear un nuevo servicio web

Conecte un repositorio de Git o utilice una imagen existente.

¿Cómo le gustaría implementar su servicio web?

Construya e implemente desde un repositorio Git
Conecte un repositorio de GitHub o GitLab.

Implementar una imagen existente desde un registro AVANZADO
Extraiga una imagen pública de cualquier registro o una imagen privada de Docker Hub, GitHub o GitLab.

Próximo

Luego, se debe crear un nuevo servicio web, seleccionando la opción "Conectar Cuenta" en el apartado de GitHub.

Render Panel Planos Grupos ambientales Documentos Comunidad Ayuda Nuevo + Brayán Alejandro Puentes ... ▾

Crear un nuevo servicio web

Conecte su repositorio Git o utilice una URL de repositorio público existente.

Conectar un repositorio

Conecte su cuenta de Render a GitHub, GitLab o Bitbucket para comenzar a usar sus repositorios existentes para nuevos servicios.

GitHub
[+ Conectarcuenta](#)

GitLab
[+ Conectarcuenta](#)

Bitbucket
[+ Conectarcuenta](#)

Conectar GitHub

Conectar GitLab

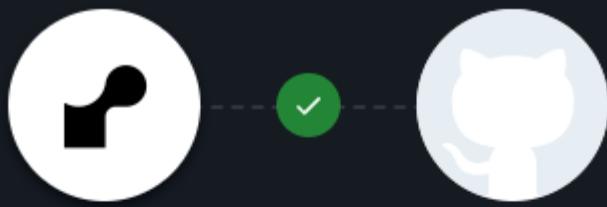
Conectar Bitbucket

Repositorio público de Git

Utilice un repositorio público ingresando la URL a continuación. Funciones como [vistas previas de PR](#) e [implementación automática](#) no están disponibles si el repositorio no se ha configurado para renderizar.

Continuar

Como se evidencia en la imagen se le da permiso a render de poder acceder al GitHub



Render by **Render** would like permission to:



Verify your GitHub identity (AlexanderxB)



Know which resources you can access



Act on your behalf

[? Learn more](#)

Resources on your account



Email addresses (read)

[View your email addresses](#)

[Learn more about Render](#)

[Cancel](#)

[Authorize Render](#)

Authorizing will redirect to

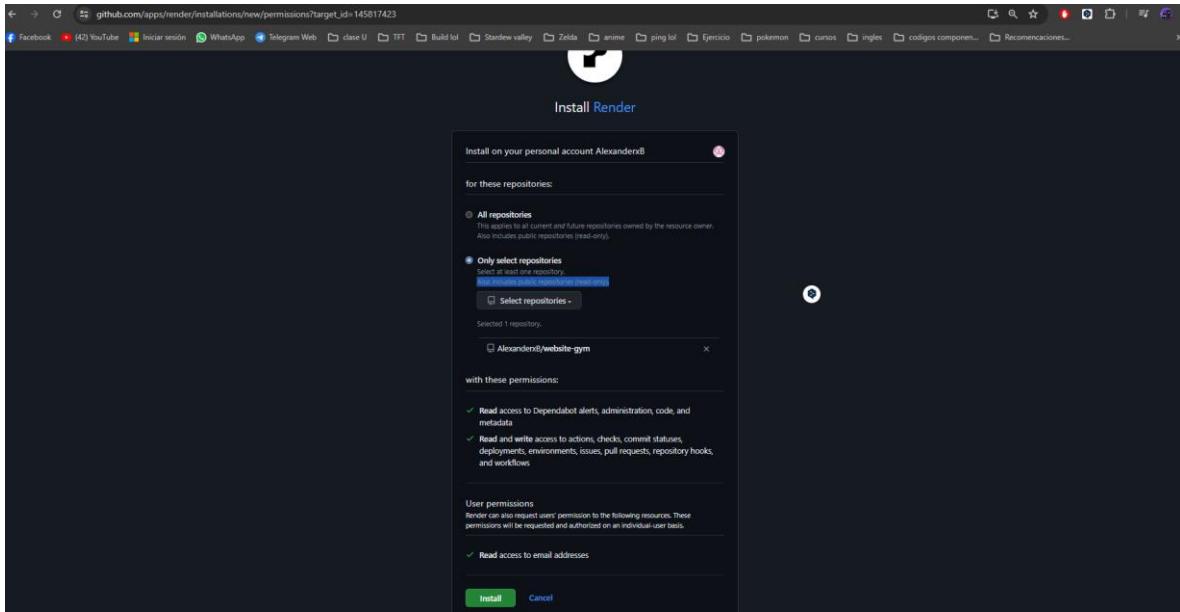
<https://dashboard.render.com>

Not owned or
operated by GitHub

Created 6 years ago

More than 1K
GitHub users

Se le da permiso únicamente para que acceda render al repositorio que dice website-gym



Se ha creado con éxito la creación de un web service

Create a new Web Service

Connect your Git repository or use an existing public repository URL.

Connect a repository

GitHub

GitLab

Bitbucket

Public Git repository

Al dar click al web service creado, se empieza a realizar la configuración de este para poder desplegar el front end del proyecto “Hypertrophy Fitness”

The screenshot shows the Render dashboard with the following configuration:

- Name:** website-gym
- Region:** Oregon (US West)
- Branch:** main
- Root Directory:** e.g., src (Optional)
- Runtime:** Node

Lo primordial es seleccionar la opción de Node. En el campo "build command", se ingresa el comando `npm install`, y en el campo "start command", se ingresa el comando `npm run start`. Además, se elige la opción "free" para utilizar Render de forma gratuita para realizar el despliegue.

The screenshot shows the Render settings page for the 'website-gym' service, with the following configurations:

- Build Command:** \$ npm install
- Pre-Deploy Command:** \$ (Optional)
- Start Command:** \$ npm run start
- Auto-Deploy:** Yes
- Deploy Hook:** https://api.render.com/deploy/srv-cp06nj7jltc73do6ulg?key=UoOoPRhOrME

Instance Type

For hobby projects	Free \$0 / month	512 MB (RAM) 0.1 CPU	Upgrade to enable more features Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks. Select any paid instance type to enable these features.
For professional use For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:	Starter \$7 / month	512 MB (RAM) 0.5 CPU	Standard \$25 / month
<ul style="list-style-type: none"> Zero Downtime SSH Access Scaling One-off jobs Support for persistent disks 	Pro \$85 / month	4 GB (RAM) 2 CPU	Pro Plus \$175 / month
	Pro Max \$225 / month	16 GB (RAM) 4 CPU	Pro Ultra \$450 / month
		32 GB (RAM) 8 CPU	

Need a custom instance type? We support up to 512 GB RAM and 64 CPUs.

Finalmente, se da click sobre el botón que dice “Create web service”

For professional use
For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

Starter	512 MB (RAM) 0.5 CPU	Standard	2 GB (RAM) 1 CPU
Pro \$85 / month	4 GB (RAM) 2 CPU	Pro Plus \$175 / month	8 GB (RAM) 4 CPU
Pro Max \$225 / month	16 GB (RAM) 4 CPU	Pro Ultra \$450 / month	32 GB (RAM) 8 CPU

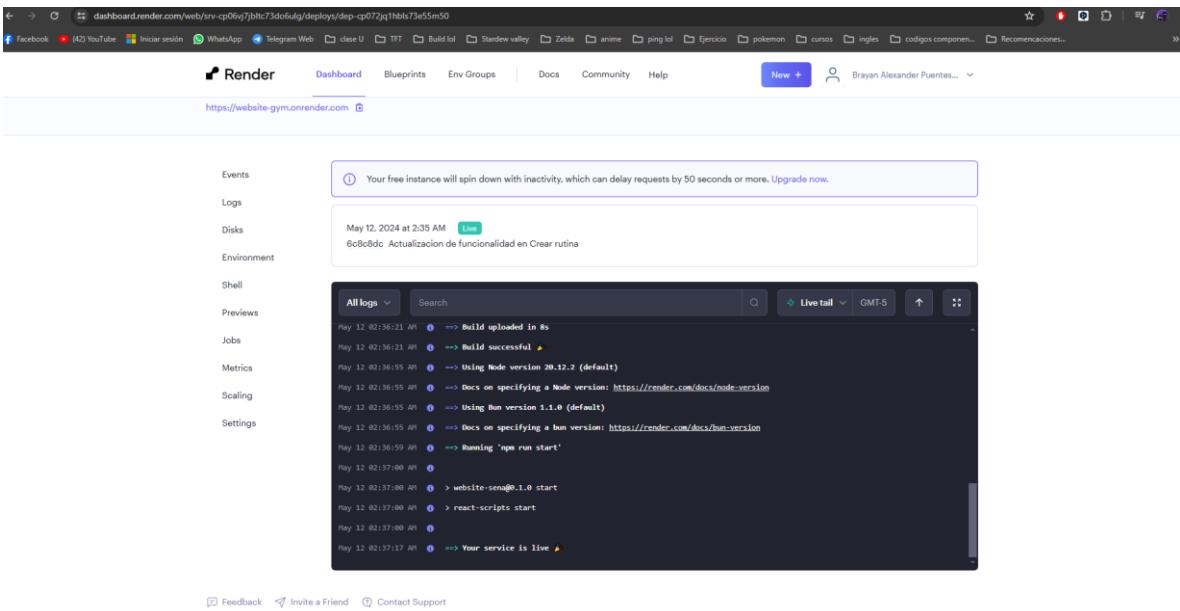
Need a custom instance type? We support up to 512 GB RAM and 64 CPUs.

Environment Variables Optional
Set environment-specific config and secrets (such as API keys), then read those values from your code. Learn more.

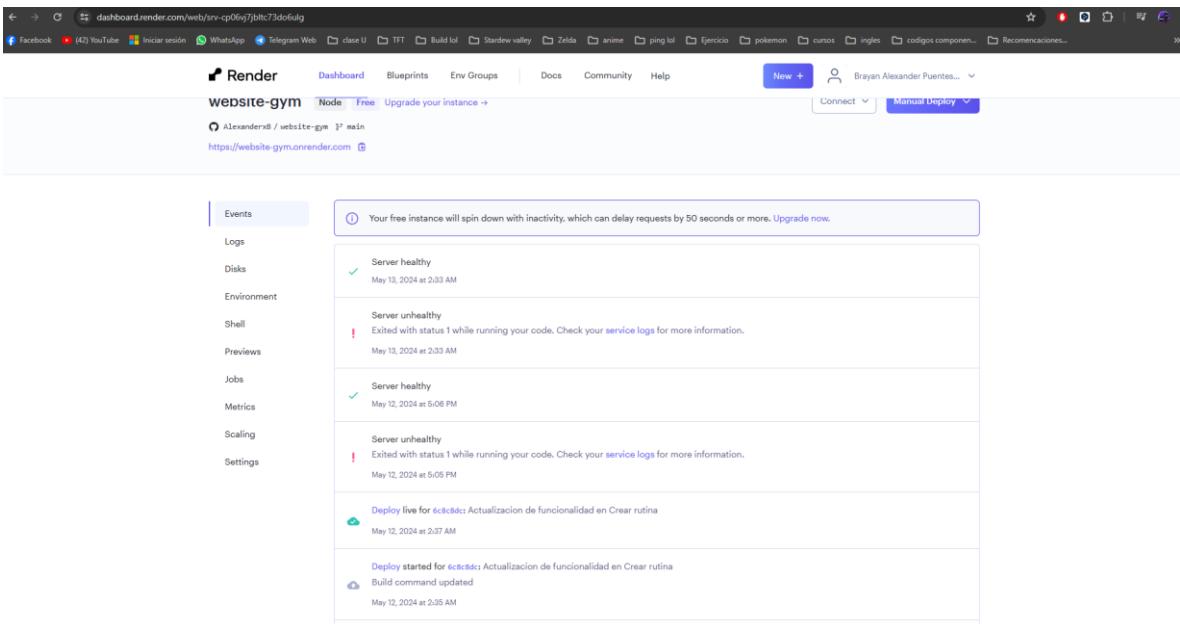
NAME_OF_VARIABLE value

+ Add Environment Variable

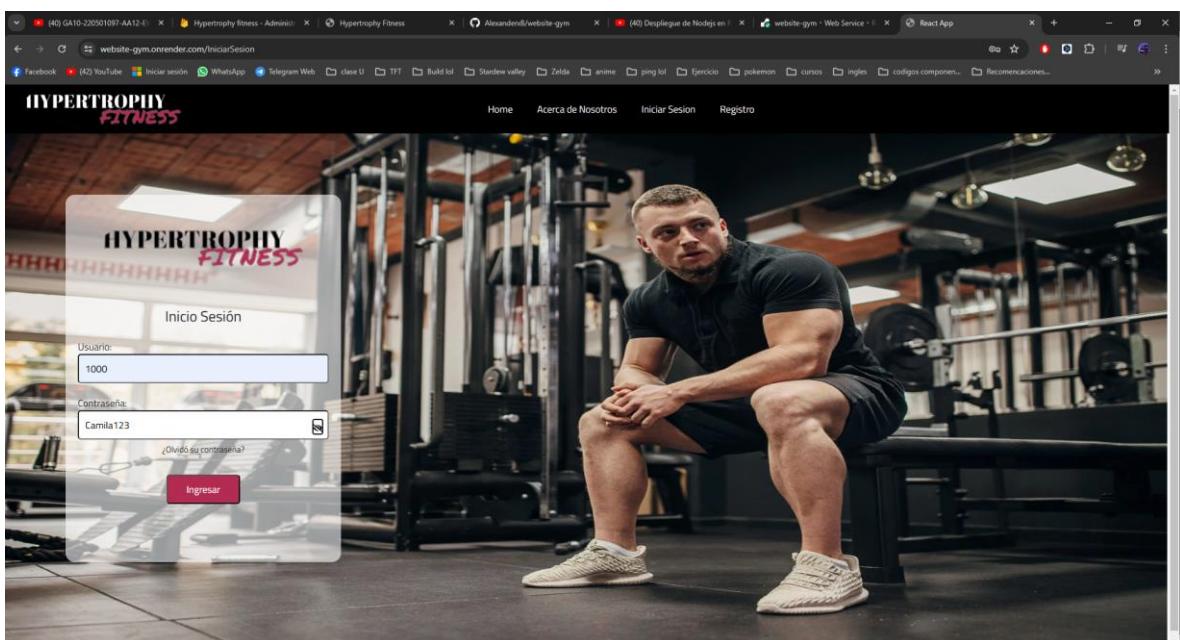
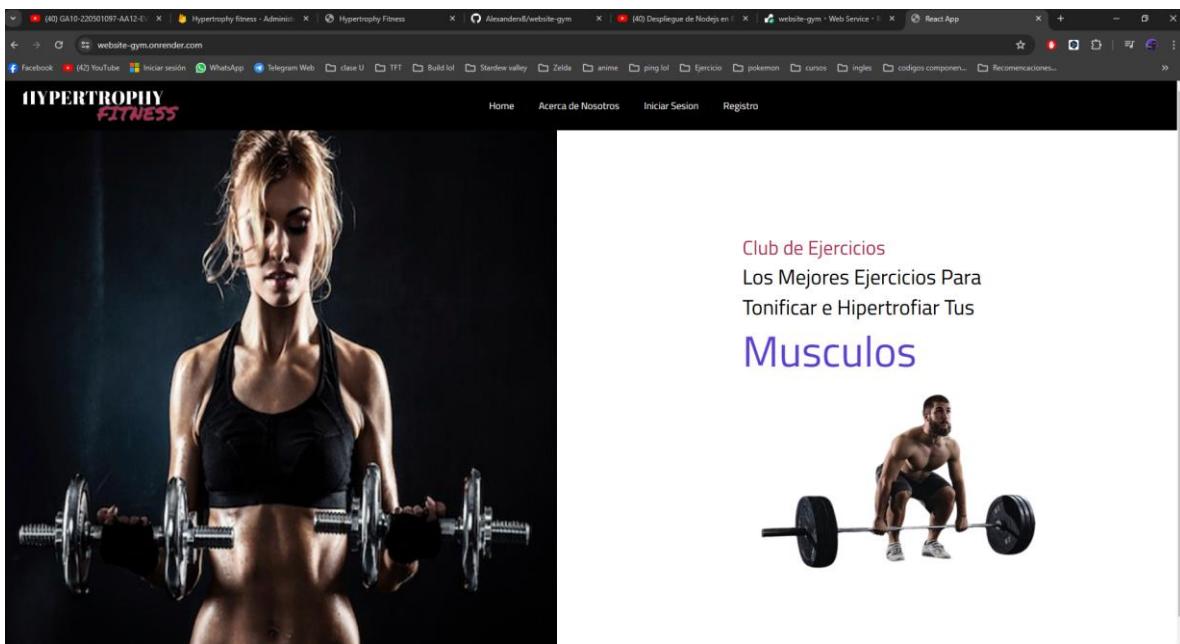
En el lado izquierdo, se selecciona la opción "Logs". Aquí se puede observar el proceso de build, el despliegue, posibles errores y otros detalles relevantes.

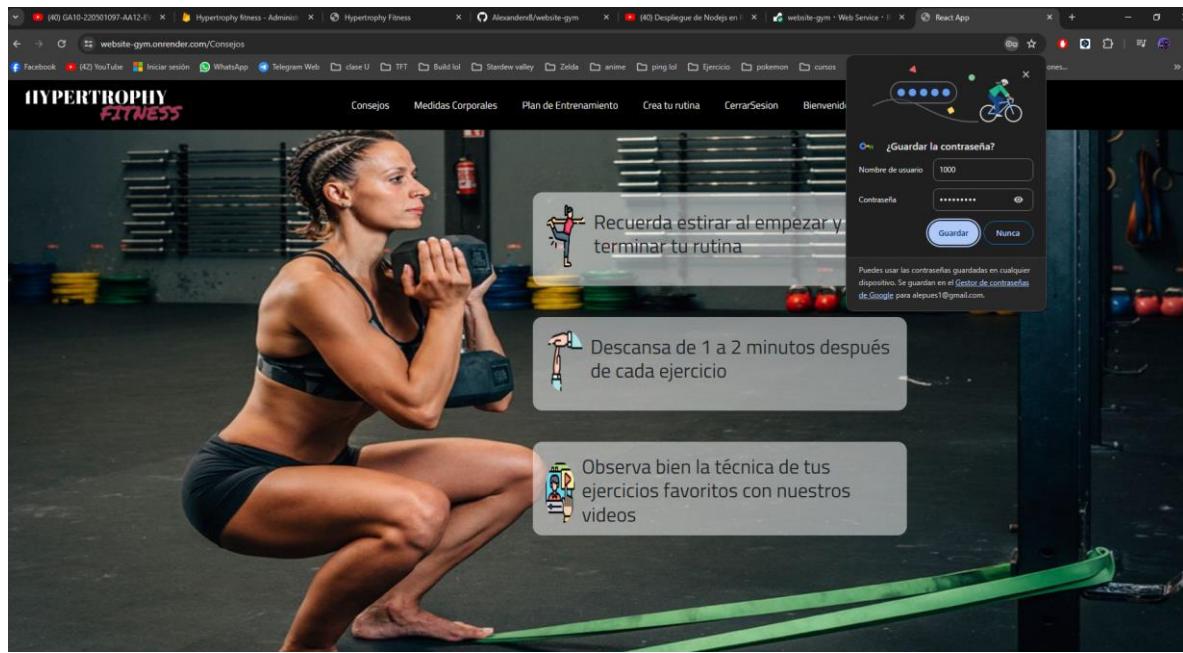


En esta imagen se evidencia con el icono de una nube verde que el despliegue salió exitoso.



En las imágenes se muestra que al usar la URL generada por Render, se puede verificar que el despliegue del front end fue exitoso.





En esta imagen se evidencia la URL que Render genera

A screenshot of the Render web interface. At the top, there's a navigation bar with "Render", "Dashboard", "Blueprints", "Env Groups", "Docs", "Community", and "Help". On the right, there are "New +", a user profile icon, and "Brayan Alexander Puentes...". Below the navigation, it says "WEB SERVICE" and shows "website-gym" as a "Node" service. It indicates the service is "Free" and provides a link to "Upgrade your instance →". It also shows "AlexanderxB / website-gym" and "main". A "Connect" button and a "Manual Deploy" button are visible. The URL "https://website-gym.onrender.com" is shown at the bottom.

URL DESPLIEGUE FRONT END: <https://website-gym.onrender.com>

BACK END

Ahora, para realizar el despliegue del back end, se sigue el mismo procedimiento descrito anteriormente. En esta imagen se evidencia el despliegue exitoso del front end.

The screenshot shows the Render dashboard at dashboard.render.com. The 'Overview' section displays a single service named 'website-gym' with a status of 'Deployed'. The service is a 'Web Service' running on Node.js in the Oregon region, last deployed a day ago. A search bar and filter buttons for 'Active', 'Suspended', and 'All' are visible. At the bottom, there are links for 'Feedback', 'Invite a Friend', and 'Contact Support'.

Al hacer clic en el botón que dice "New", se despliegan varias opciones. En este caso, se selecciona la opción "Web Service".

The screenshot shows the Render dashboard at dashboard.render.com. The 'Overview' section shows the same 'website-gym' service. A modal window is open over the 'New +' button, listing service types: 'Static Site', 'Web Service' (which is selected and highlighted in blue), 'Private Service', 'Background Worker', 'Cron Job', 'PostgreSQL', 'Redis', and 'Blueprint'. A detailed description of 'Web Service' is provided, mentioning SSL and HTTPS support. At the bottom of the modal, there are filter buttons for 'Active', 'Suspended', and 'All'.

Se vuelve a crear el web service para realizar el despliegue del back, conectando el GitHub

Create a new Web Service

Connect a Git repository, or use an existing image.

How would you like to deploy your web service?

Build and deploy from a Git repository
Connect a GitHub or GitLab repository.

Deploy an existing image from a registry ADVANCED
Pull a public image from any registry or a private image from Docker Hub, GitHub, or GitLab.

Next

En la sección de "Public Git repository", se introduce la URL del repositorio de GitHub.

Connect a repository

Search...

AlexanderxB / website-gym · a day ago Connect

GitHub @AlexanderxB · 1 repo Configure account

GitLab Connect account

Bitbucket Connect account

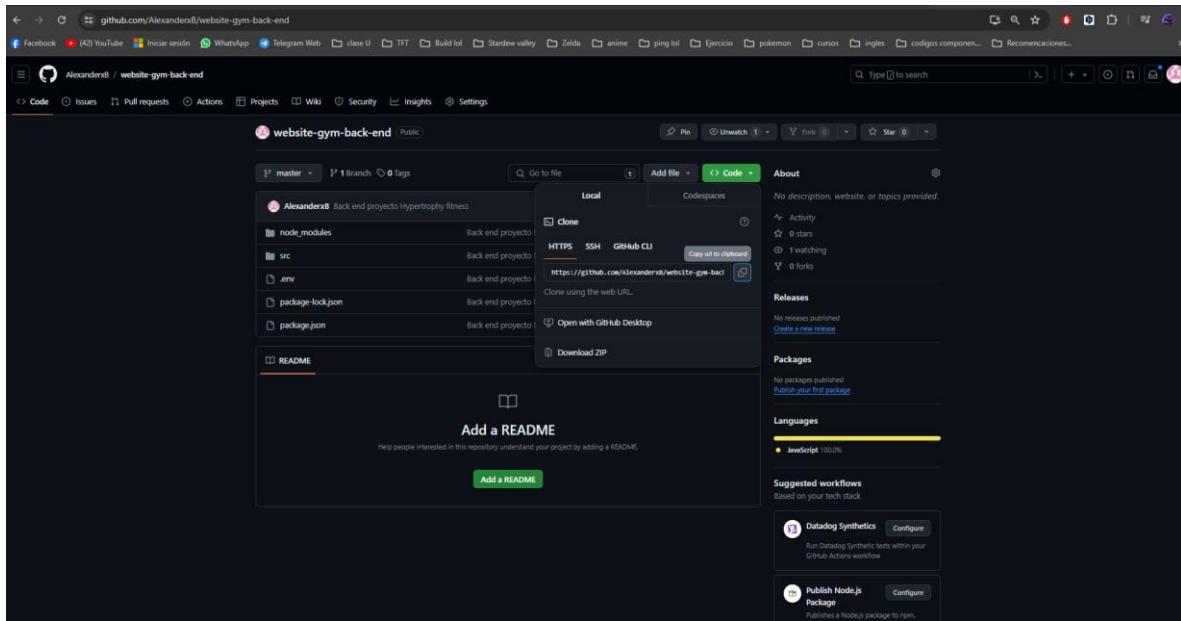
Public Git repository

Use a public repository by entering the URL below. Features like PR Previews and Auto-Deploy are not available if the repository has not been configured for Render.

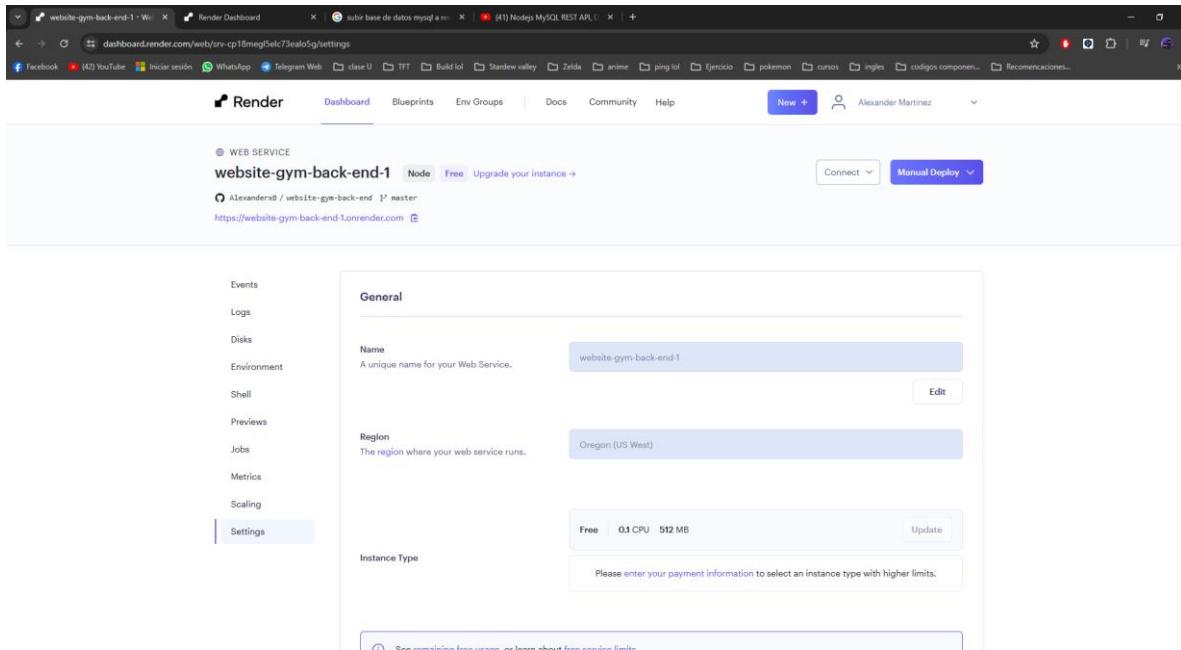
https://github.com/AlexanderxB/website-gym-back-end.git Continue

Feedback Invite a Friend Contact Support

De aquí se genera la URL mencionada en el paso anterior.



Se vuelve a realizar la configuración para realizar el despliegue y el build



The screenshot shows the Render Dashboard with the service "website-gym-back-end-1" selected. The "Build & Deploy" section is open, showing the following configuration:

- Repository:** https://github.com/Alexandersd/website-gym-back-end
- Branch:** master
- Root Directory:** Optional (Defaults to repository root)
- Build Filters:** Filter for files and paths to monitor for automatic deploys.
- Included Paths:** Changes that match these paths will trigger a new build.

En la sección del build se ingresa el comando `npm install` y en el apartado de start comando se ingresa el comando `node src/index.js`

The screenshot shows the Render Dashboard with the service "website-gym-back-end-1" selected. The "Settings" section is open, showing the following configuration:

- Build Command:** \$ npm install
- Pre-Deploy Command:** \$ (Optional)
- Start Command:** \$ node src/index.js
- Auto-Deploy:** Yes
- Deploy Hook:** https://api.render.com/deploy/srv-cp18meg1Selc7Sealo5g?key=Jhs7kt6oalE

Se evidencia en Events que el despliegue se realizo de forma correcta

The screenshot shows the Render Dashboard for the service 'website-gym-back-end-1'. On the left, a sidebar lists options like Events, Logs, Disks, Environment, Shell, Previews, Jobs, Metrics, Scaling, and Settings. The main area displays deployment events:

- Deploy live for 474739d: Se agrego el get en index.js (May 13, 2024 at 4:54 PM)
- First deploy started for 474739d: Se agrego el get en index.js (May 13, 2024 at 4:50 PM)

At the bottom, there are links for Feedback, Invite a Friend, and Contact Support.

Y en la imagen se evidencia el despliegue tanto del front end como del back end

The screenshot shows the Render Dashboard's Overview page. It lists two services:

Service Name	Status	Type	Runtime	Region	Last Deployed
website-gym-back-end-1	Deployed	Web Service	Node	Oregon	4 hours ago
website-gym	Deployed	Web Service	Node	Oregon	4 hours ago

Aquí se puede evidenciar la URL del back end

The screenshot shows the Render Dashboard for the service 'website-gym-back-end-1'. The main area displays the following information:

WEB SERVICE
website-gym-back-end-1 Node Free Upgrade your instance →
AlexanderXB / website-gym-back-end master
<https://website-gym-back-end-1.onrender.com>

At the top right, there are 'Connect' and 'Manual Deploy' buttons.

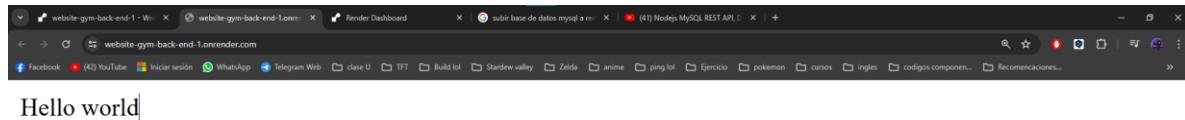
Se realizan pruebas en Visual Studio donde se coloca un mensaje como "hello world" para verificar que el despliegue del back end se haya realizado

correctamente.



```
const app = require('./app');
app.get('/', (req, res) => {
  res.send("Hello world");
});
app.listen(app.get('port'), () => {
  console.log("Servidor escuchando en el puerto", app.get("port"));
});
```

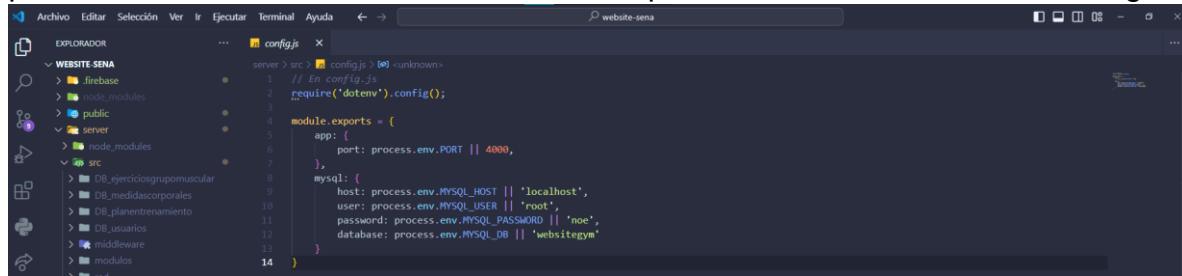
En esta imagen se evidencia el mensaje hello world, es decir, el despliegue esta funcionando correctamente.



URL DESPLIEGUE BACK END: <https://website-gym-back-end-1.onrender.com>

DESPLIEGUE BASE DE DATOS

Por el momento, tener las credenciales de la base de datos de forma local está bien para el desarrollo, como se puede ver en la imagen.

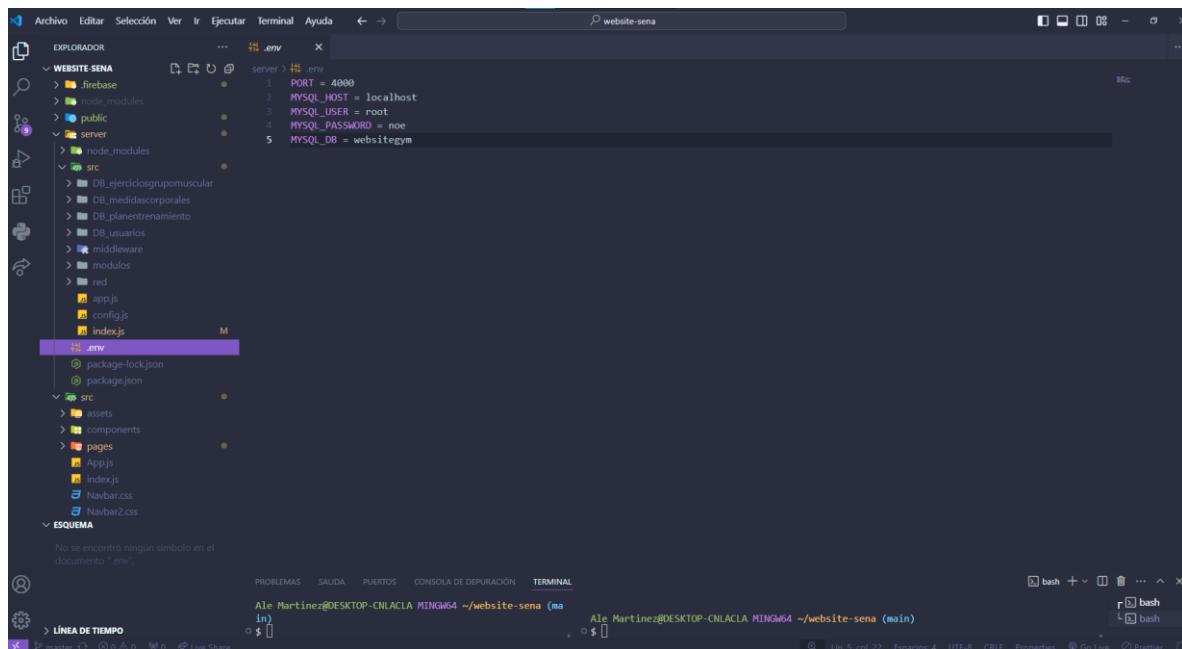


```
server > src > config.js > [o] <unknown>
1 // En config.js
2 require('dotenv').config();
3
4 module.exports = {
5   app: {
6     port: process.env.PORT || 4000,
7   },
8   mysql: {
9     host: process.env.MYSQL_HOST || 'localhost',
10    user: process.env.MYSQL_USER || 'root',
11    password: process.env.MYSQL_PASSWORD || 'noe',
12    database: process.env.MYSQL_DB || 'websitegym'
13  }
14 }
```

Para llevar a cabo esto en producción, necesitamos un entorno más preparado. Vamos a utilizar variables de entorno, que son variables almacenadas en el sistema operativo y que nuestra aplicación puede leer y utilizar.

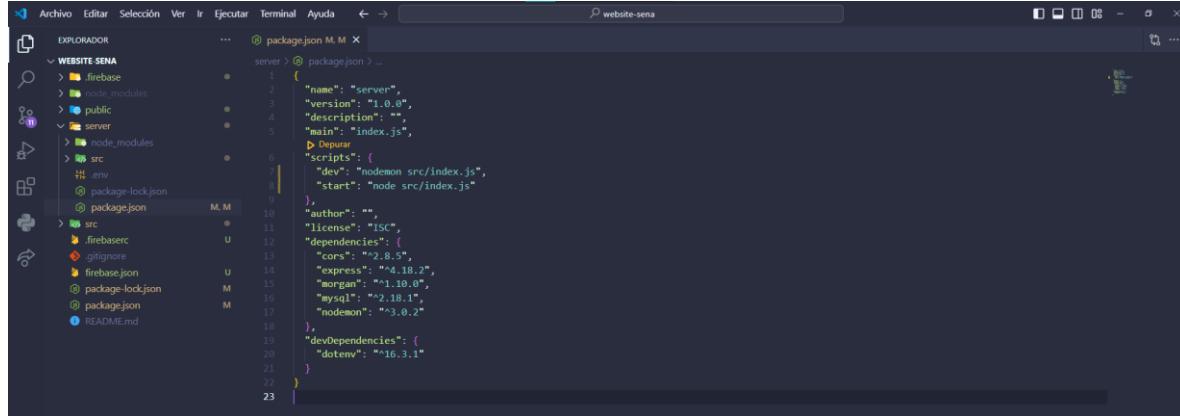
Primero, necesitamos instalar un módulo en la terminal de Visual Studio Code. Podemos hacerlo con el comando npm i dotenv.

Este comando nos permite leer un archivo llamado ".env", donde definiremos nuestras variables de entorno.



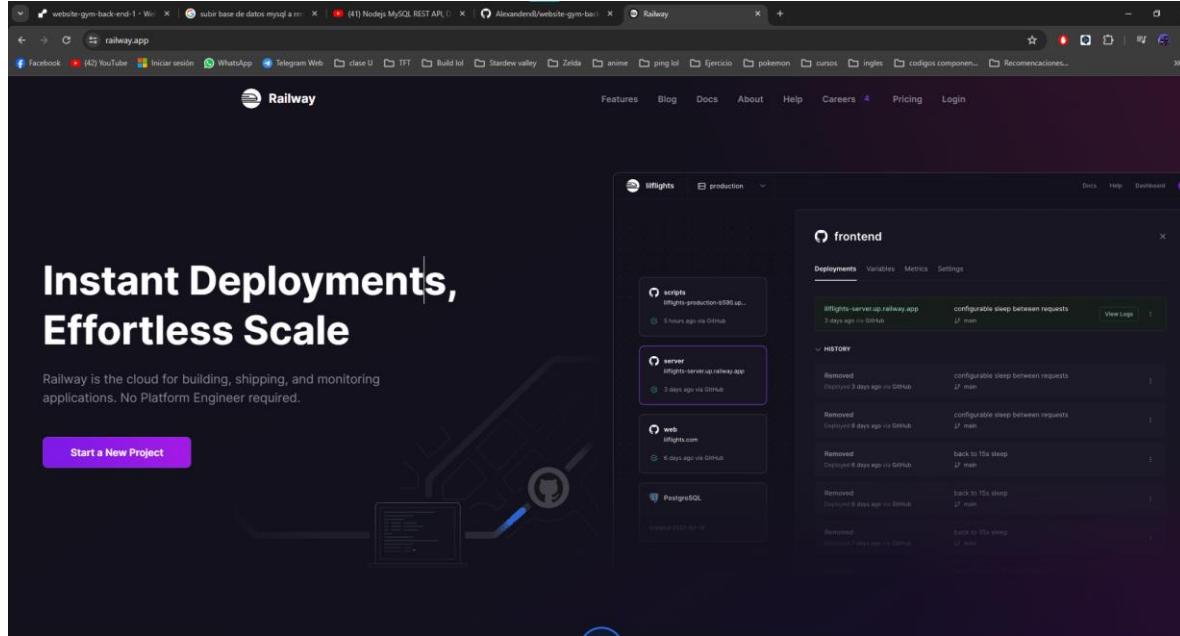
```
server > .env
1 PORT = 4000
2 MYSQL_HOST = localhost
3 MYSQL_USER = root
4 MYSQL_PASSWORD = noe
5 MYSQL_DB = websitegym
```

En el archivo package.json, se tendrán en cuenta dos comandos: "dev" y "start". En desarrollo, se utilizará el comando: "dev": "nodemon src/index.js". Por otro lado, en producción se usará el comando: "start": "node src/index.js", como se muestra en la siguiente imagen:

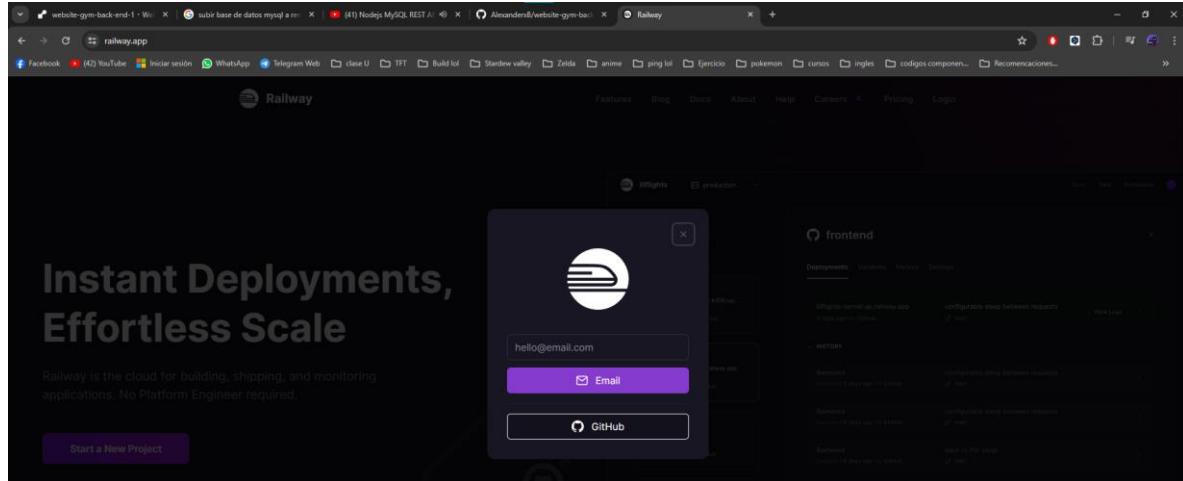


```
1  {
2   "name": "server",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "dev": "nodemon src/index.js",
8     "start": "node src/index.js"
9   },
10  "author": "",
11  "license": "ISC",
12  "dependencies": {
13    "cors": "^2.8.5",
14    "express": "^4.18.2",
15    "morgan": "^1.10.0",
16    "mysql": "^2.18.1",
17    "nodemon": "3.0.2"
18  },
19  "devDependencies": {
20    "dotenv": "^16.3.1"
21  }
22}
```

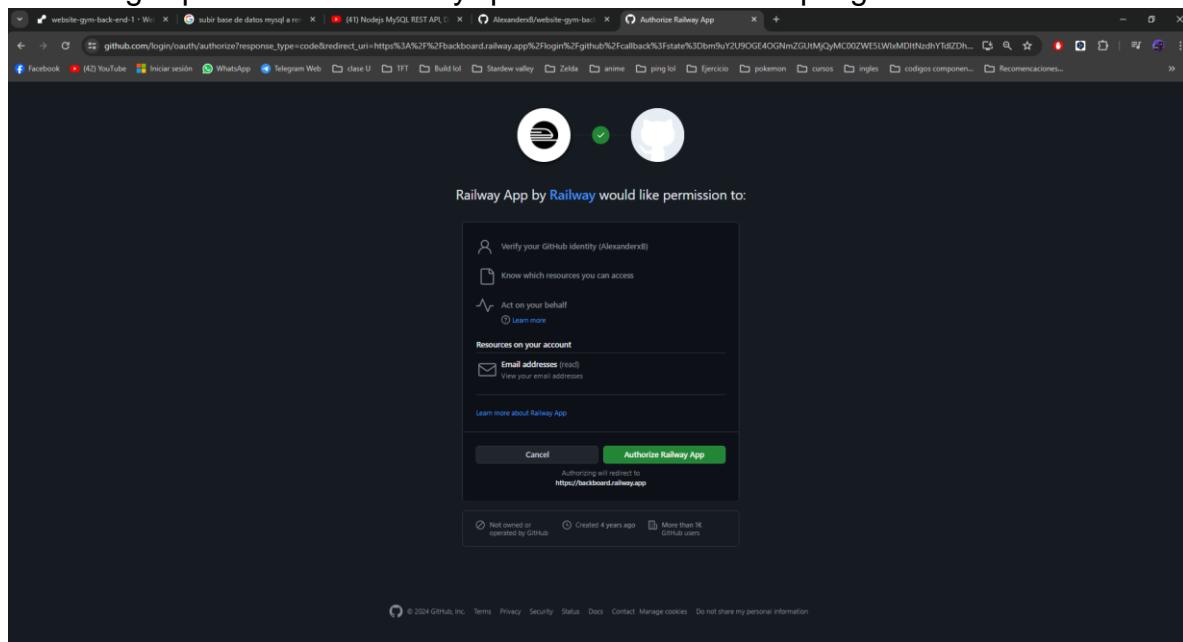
Para desplegar la base de datos se va a usar <https://railway.app/>



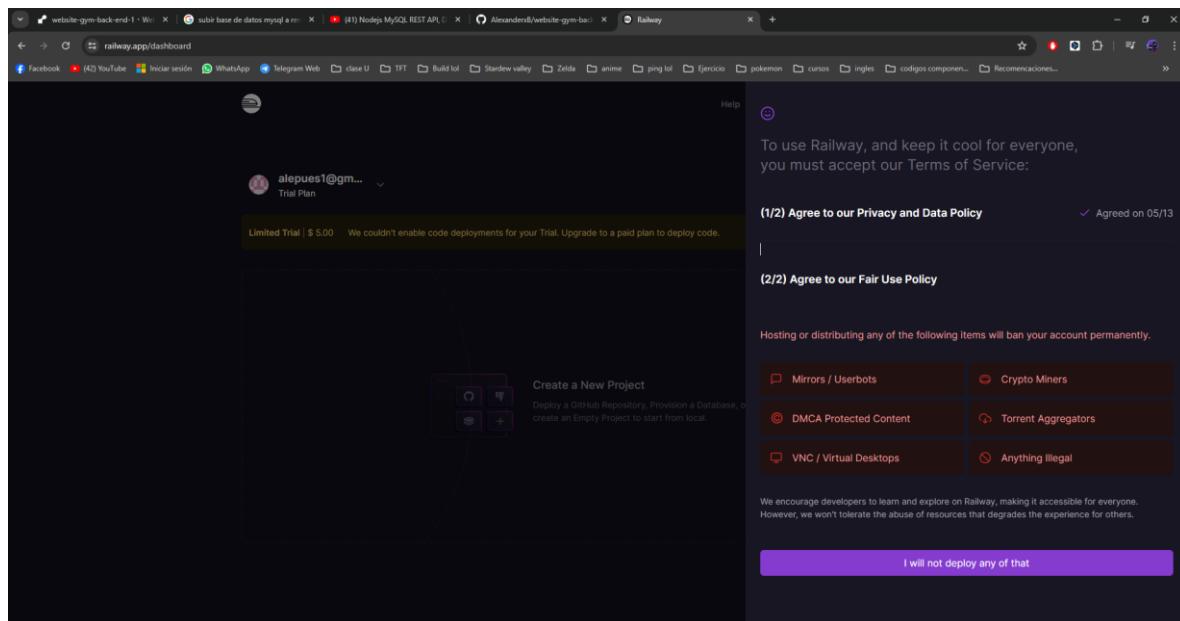
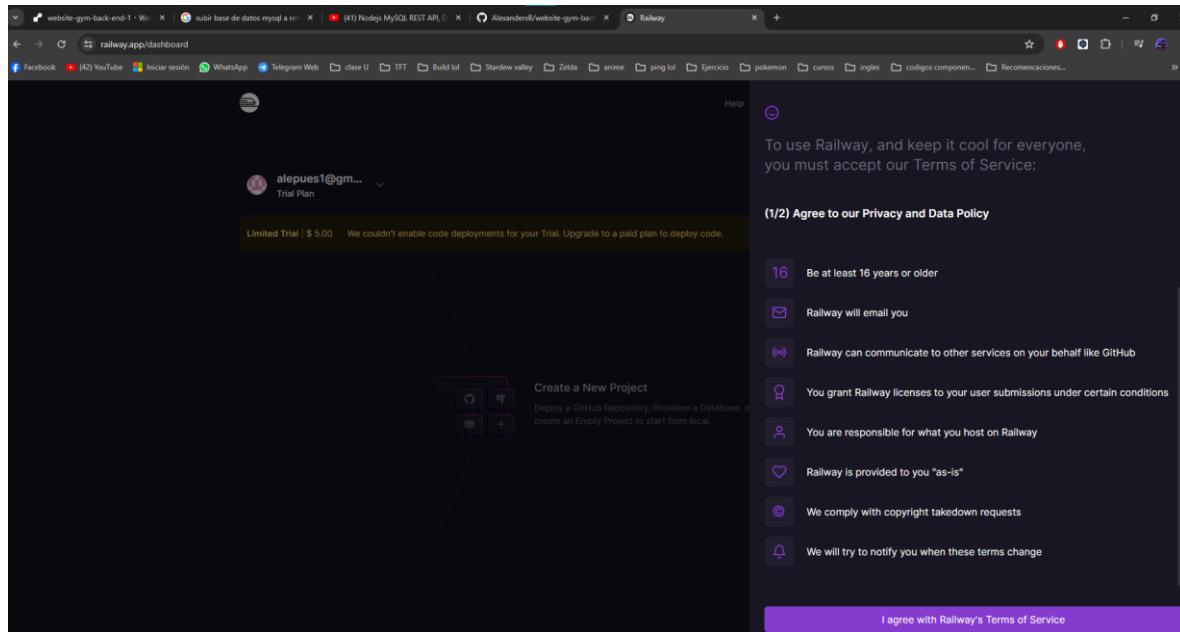
Para utilizar Railway, primero debes iniciar sesión. En este caso, lo haremos a través de GitHub.



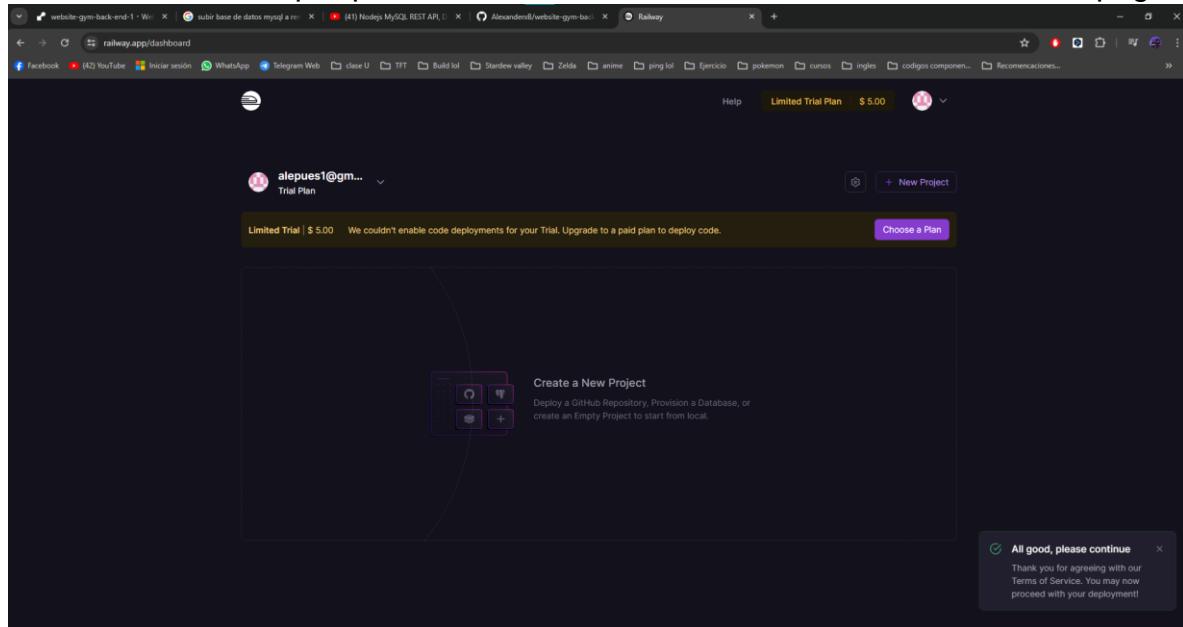
Se otorga permiso a Railway para realizar el despliegue mediante GitHub.



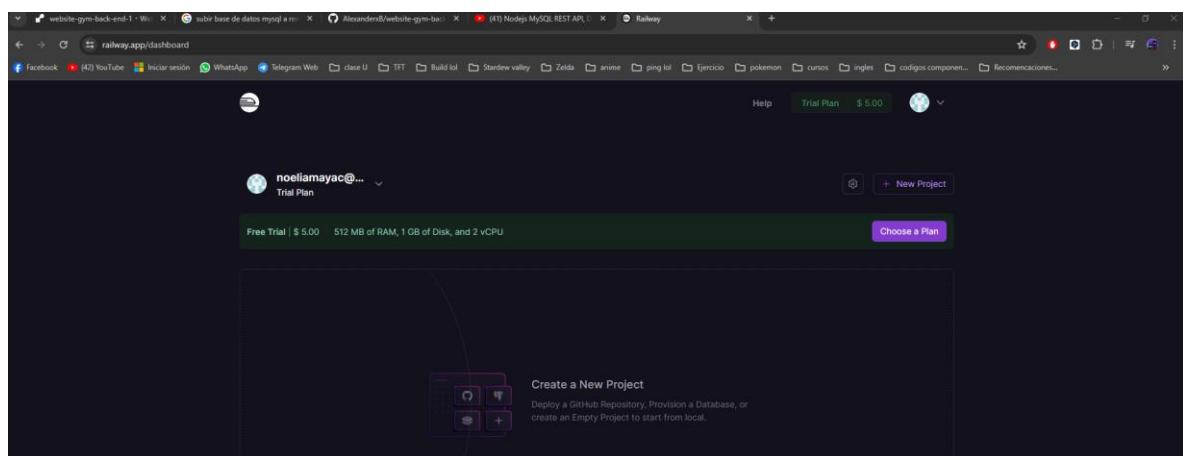
Se aceptan los términos y condiciones para poder usar railway



El despliegue se realizó con éxito en otra cuenta que permite el uso de Railway gratis ya que como se evidencia en la imagen en la cuenta que se creo no es posible usarla porque se debe pagar.



Esta es la cuenta donde se va a realizar el despliegue ya que es gratis



A continuación, se detallará el proceso para utilizar Railway:

Agregar una base de datos: Una vez dentro de tu proyecto, busca la opción para agregar un nuevo servicio o recurso. Allí encontrarás la opción para añadir una base de datos. Railway es compatible con varias bases de datos, como PostgreSQL, MySQL, MongoDB, entre otras. Selecciona la base de datos que mejor se ajuste a tus necesidades.

En este caso particular, se optó por MySQL debido a que el proyecto fue desarrollado con MySQL y Workbench. Por lo tanto, se crearon dos proyectos en Railway: uno para MySQL y otro para desplegar la base de datos a través del repositorio de GitHub del backend.

Estos pasos se ilustran en la siguiente imagen:

The image consists of two screenshots of the Railway platform's service management interface.

Top Screenshot (MySQL Service):

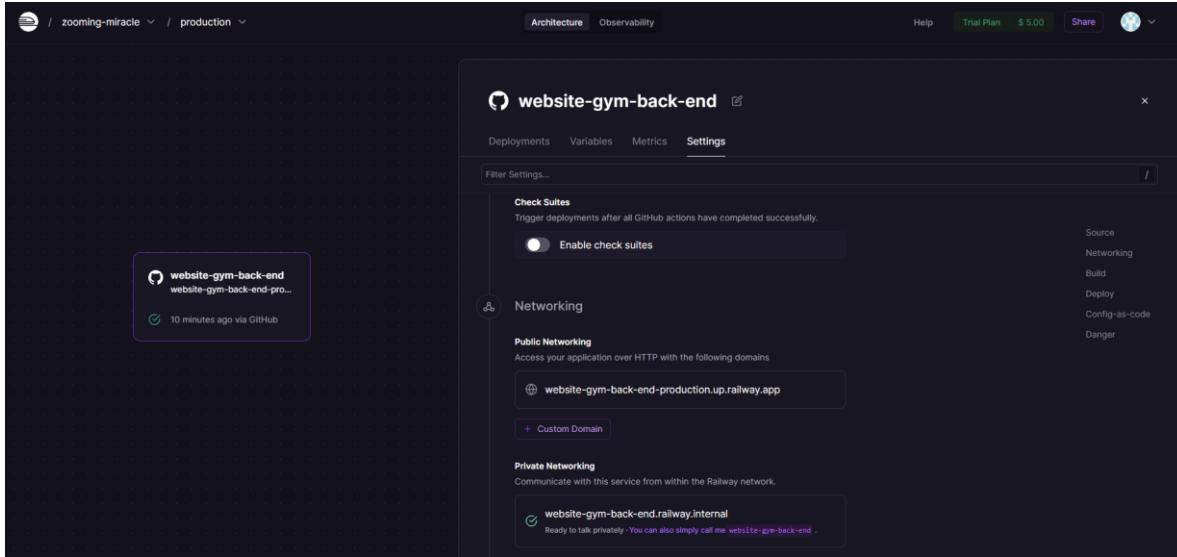
- Project: `charismatic-fulfillment` / `production`
- Service: MySQL
- Status: Active (55 minutes ago via Docker Image)
- Image: mysql (Dockerhub)
- Actions: View Logs, More options

Bottom Screenshot (website-gym-back-end Service):

- Project: `zooming-miracle` / `production`
- Service: website-gym-back-end
- Status: Active (9 minutes ago via GitHub)
- Script: start script añadido (master)
- Actions: View Logs, More options

Después de seleccionar el tipo de base de datos, Railway te guiará a través de los pasos necesarios para configurarla. Esto puede incluir la selección de la región del

servidor, el plan de precios (si corresponde), y la configuración de credenciales y otras opciones específicas de la base de datos.

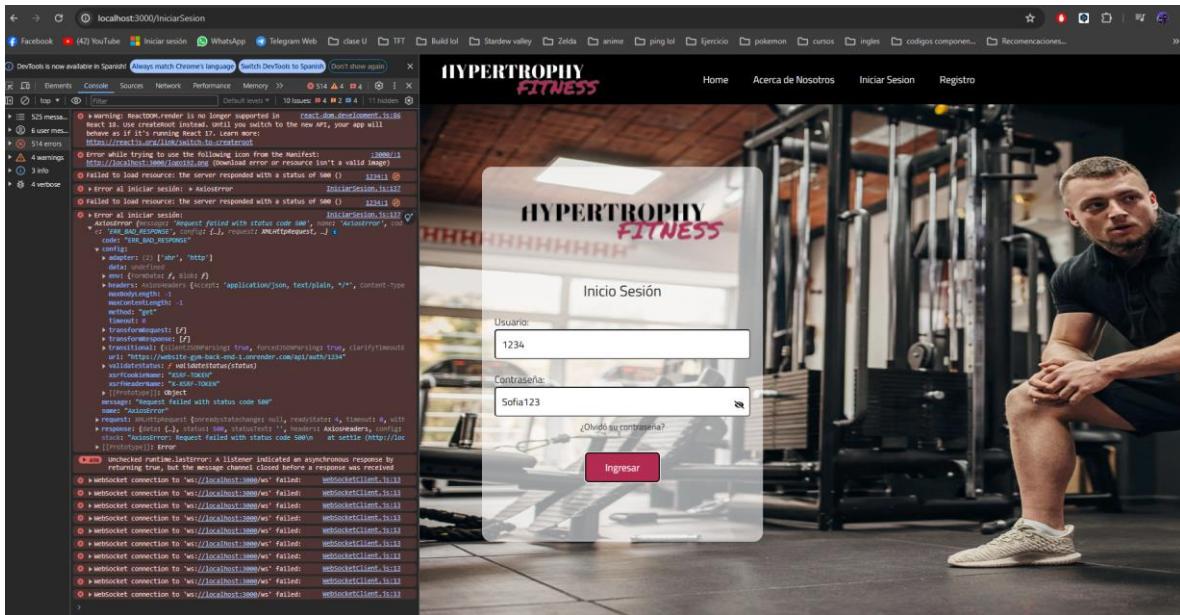


En esta imagen se evidencian las variables de entorno para hacer la conexión con la base de datos, por el momento se están usando los datos para acceder a la pagina web de forma local porque a pesar de que la base de datos tuvo un deploy exitoso, y se hicieron las configuraciones respectivas en el back end como se evidencia en la imagen

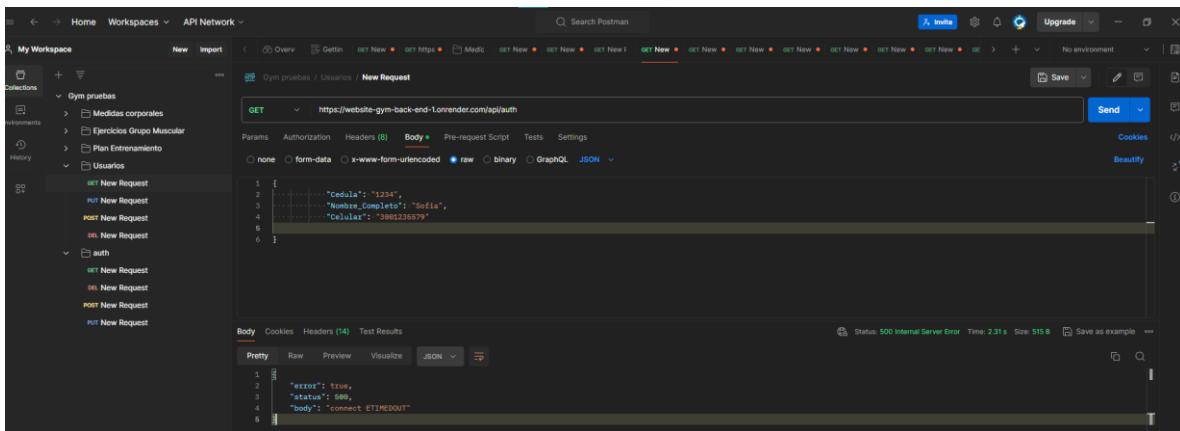
A screenshot of a terminal window with the title bar 'website-sena'. The window shows an 'EXPLORADOR' sidebar with project files like '.fbconfig', 'node_modules', 'public', 'server', 'src', 'assets', 'components', and 'pages'. The main pane displays the contents of a '.env' file under the 'server' directory. The file contains the following environment variables:

```
# PORT = 58859
# MYSQL_HOST = roundhouse.proxy.railway.net
# MYSQL_USER = root
# MYSQL_PASSWORD = JInnbBuiughTheSMBaX9Iu6DSKoZkPydGVm
# MYSQL_DB = railway
PORT = 4080
MYSQL_HOST = localhost
MYSQL_USER = root
MYSQL_PASSWORD = noe
MYSQL_DB = websitedegym
```

Sin embargo, al hacer pruebas se generan los siguientes errores como se muestra en la imagen



También aparece el mismo error en el postman, el error es el : ""connect ETIMEDOUT""



El error que estamos enfrentando parece estar relacionado con el peso de las imágenes que hemos incluido en el frontend, lo que está causando tiempos de carga más largos. Además, existe la posibilidad de que el problema esté relacionado con el hecho de que tanto el backend como el frontend se presentan en el mismo archivo. Sin embargo, la información que hemos encontrado en Internet no es lo suficientemente clara para resolver este error.

Aquí se evidencia el deploy exitoso mencionado de la base de datos

The screenshot shows the MySQL section of the Railway observability interface. At the top, there are tabs for Deployments, Data, Variables, Metrics, and Settings. The Data tab is selected. Below the tabs, there is a purple header bar with a 'Connect' button. The main area displays a list of tables: auth, ejerciciosgrup..., medidascorpor..., planentrenami..., and usuarios. The 'usuarios' table is highlighted with a dark gray background. A 'Create table' button is located at the bottom right of the table list.

This screenshot shows the same MySQL interface, but the 'usuarios' table is now fully visible. The table has three columns: Cedula, Nombre_Completo, and Celular. The data is as follows:

Cedula	Nombre_Completo	Celular
1000	Camila Sofia	3143175333
1013693067	Ana Maria Lozano Reyes	3182134459
1014	alex	3143175333
1015	Mara	31431752345
1016	brayan	3183456780
1017	alejandra	3183456780
1018	pablo	3183456780
1234	Sofia	3143175333
1235	Nani	312342345

MySQL Workbench

Schemas: railway, auth, ejerciciosgrupomuscular, medicoscorporales, plan營amento, users.

Table: users

Cedula	Nombre_Completo	Celula
1000	Carmen Sofia	3182134469
1013983067	Alejandra Letona Reyes	3143175333
1014	alex	3143175245
1015	Mora	3183455700
1016	bryan	3183455700
1017	Alejandra	3183455700
1018	pablo	3183455700
1235	Nari	312342345
3028		

Table: auth

Cedula	Nombre	Contrasena
1000		
1014		
1015		
1016		
1017		
1018		
1235		
3028		

SQL Additions: Automatic context help disabled. Use the toolbar manually get help for t current caret position or toggle automatic help.

LINK DESPLIEGUE BASE DE DATOS: website-gym-back-end-production.up.railway.app

CONCLUSIONES

En el mundo actual de la tecnología de la información y las comunicaciones, la selección cuidadosa de componentes y tecnologías es un factor determinante para el éxito en el despliegue de sistemas y servicios informáticos. Este trabajo ha destacado la importancia crítica de considerar aspectos clave como el sistema operativo, los protocolos de transmisión y los medios de transmisión al implementar y operar sistemas y servicios en línea.

La elección del sistema operativo adecuado es fundamental, ya que influye en el rendimiento, la seguridad y la compatibilidad de las aplicaciones. Comprender las familias de protocolos de transmisión es esencial para garantizar una comunicación eficiente y confiable entre los componentes de la red. Asimismo, el análisis de los medios de transmisión disponibles es crucial para determinar la velocidad, la estabilidad y la capacidad de escalabilidad de una página web.

Bibliografía

Protocolos de red, la base de la transmisión electrónica de datos. (2019, julio 30).

IONOS Digital Guide; IONOS.

<https://www.ionos.es/digitalguide/servidores/know-how/los-protocolos-de-red-en-la-transmision-de-datos/>

Estándares y Tecnologías - Estándares y Tecnologías. (s/f). MINTIC Colombia.

Recuperado el 8 de mayo de 2024, de

<https://mintic.gov.co/portal/inicio/Atencion-y-Servicio-a-la-Ciudadania/Preguntas-frecuentes/5236:Estandares-y-Tecnologias>