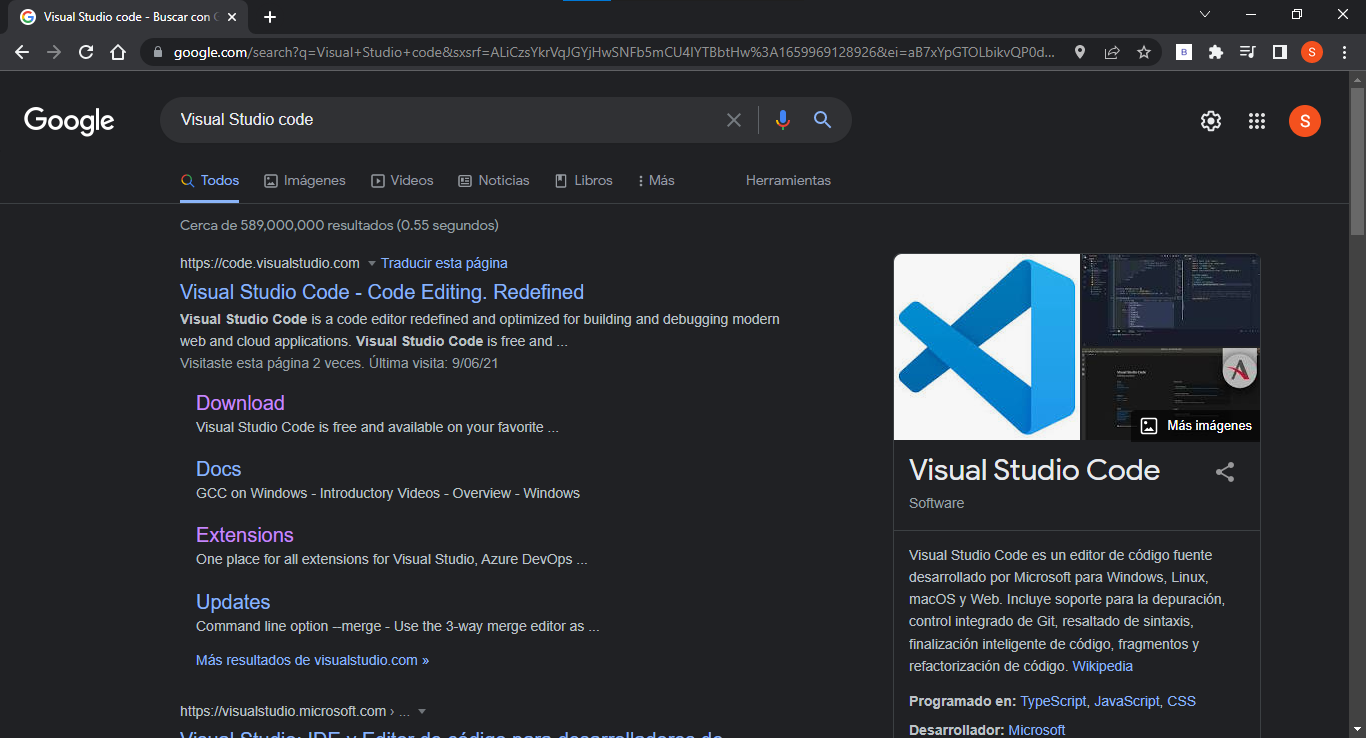
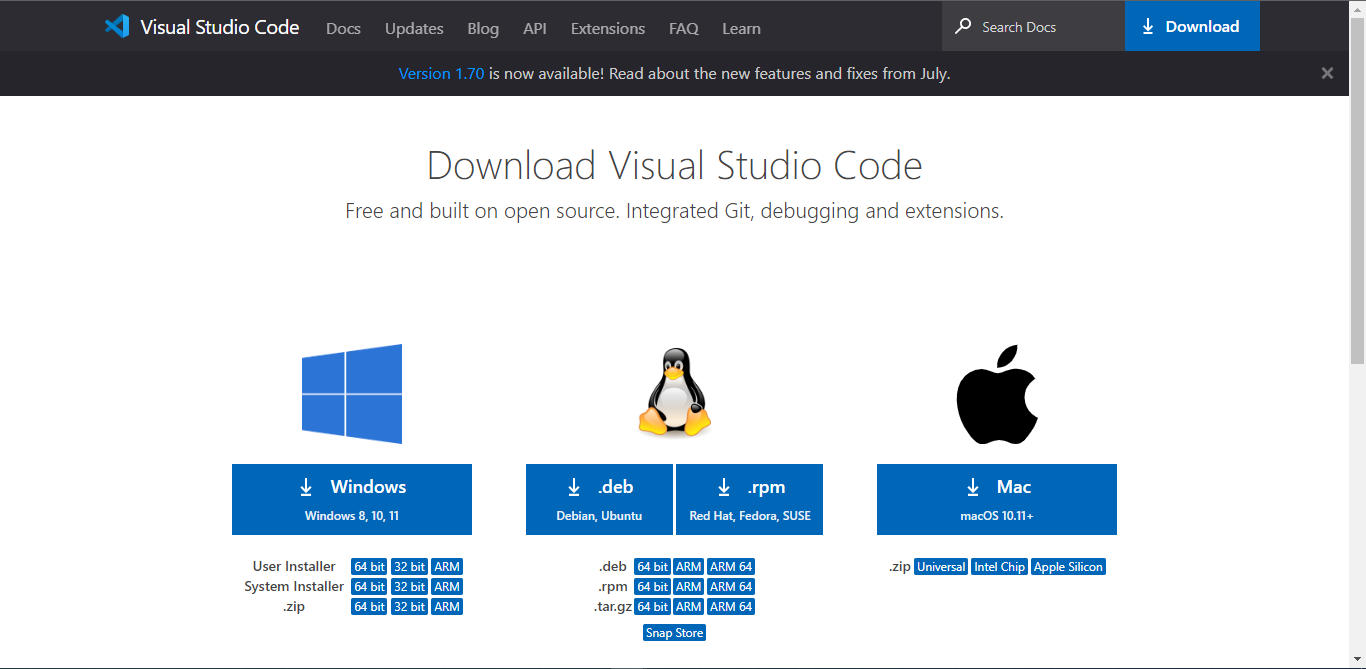
**Paso 1:** Escribimos en Google Visual Studio Code y seleccionamos donde dice “Download”.



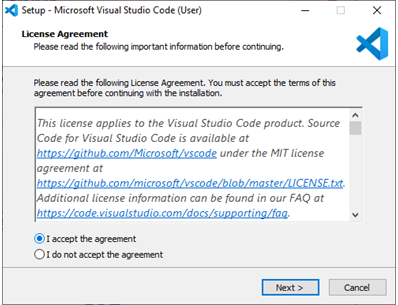
**Paso 2:** Seleccionamos el sistema operativo que tenemos y lo descargamos.



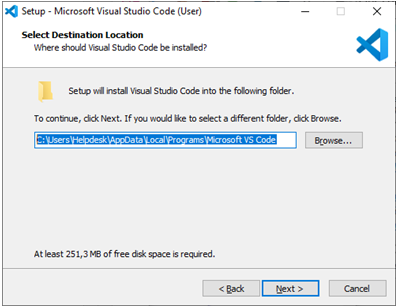
**Paso 3:** Al darle clic nos descargará un .exe, al cual le daremos clic encima.



**Paso 4:** Lee y acepta el acuerdo de licencia. Haz clic en Next para continuar.



**Paso 5:** Puedes cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada. Haz clic en Next para continuar.



**Paso 6:** Elige si deseas cambiar el nombre de la carpeta de accesos directos en el menú Inicio o si no deseas instalar accesos directos en absoluto. Haz clic en Next.



**Paso 7:** Selecciona las tareas adicionales, por ej. crear un icono en el escritorio o añadir opciones al menú contextual de Windows Explorer. Haz clic en Next.Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

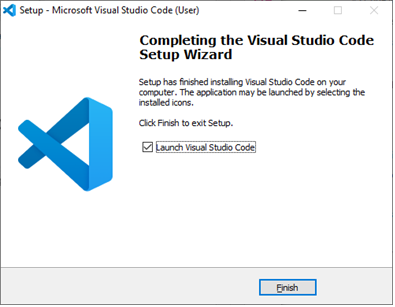
Descripción generada automáticamente

**Paso 8:** Haz clic en Install para iniciar la instalación.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 9:** El programa está instalado y listo para usar. Haz clic en Finish para finalizar la instalación y lanzar el programa.



**Paso 10**: ahora vamos a descargar node js, entramos a Google y buscamos node js download.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Paso 11**: escogemos la primera opción

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Imagen que contiene Escala de tiempo

Descripción generada automáticamentePaso 12**: ahora elegimos nuestro sistema operativo y los bits

**Paso 13**: abrimos el instalador y se nos mostrara una pestaña, apretamos “next”

Texto

Descripción generada automáticamente

**Paso 15**: tendremos que aceptar las licencias de node js marcamos el cuadrado de abajo a la derecha y next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 15**: elegimos el lugar del archivo y next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 16**: en esta ventana podremos customisar el node js con diferentes paquetes apretamos next

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 17**: aquí nos piden permiso para instalar diferentes herramientas del node js apretamos next

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 18**: finalmente apretamos install para comenzar la instalación

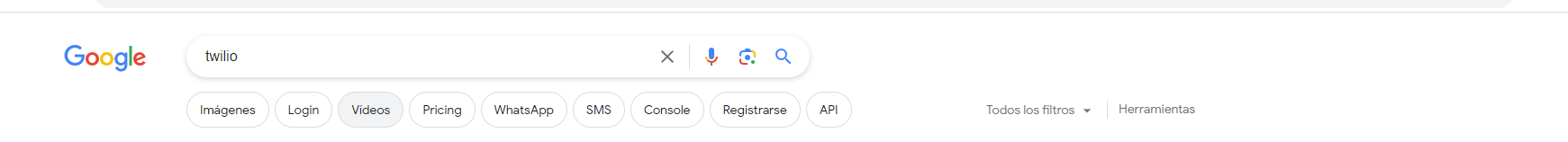
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

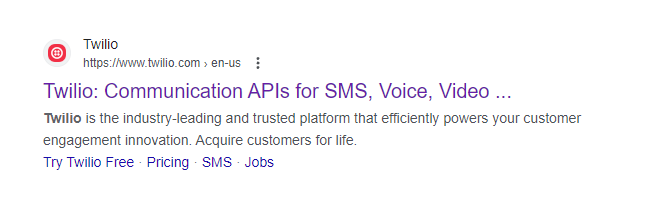
**Paso 19**: esperar la instalación

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

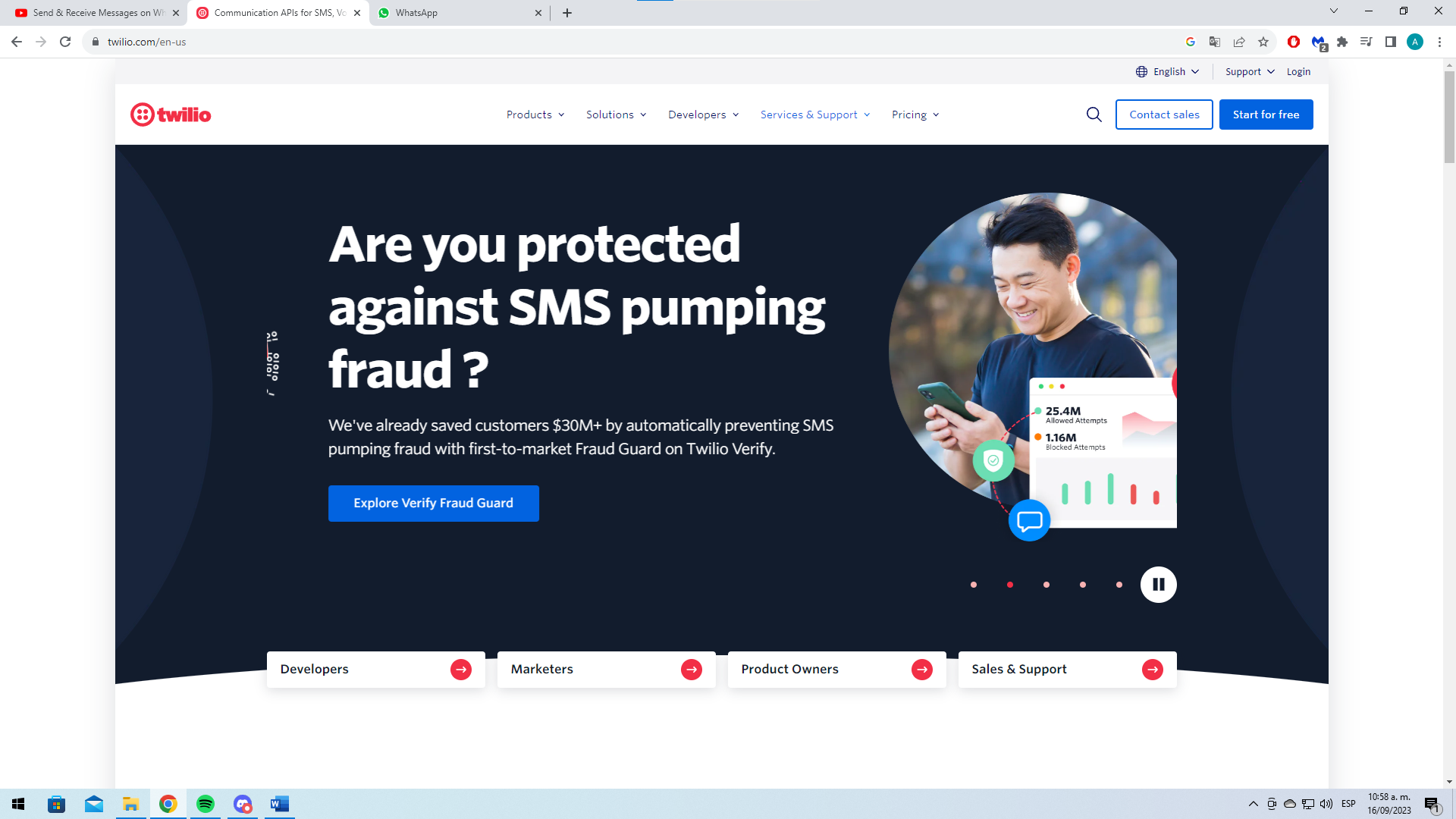
Descripción generada automáticamente

**Paso 20:** en el buscador de Google escribiremos twilio que es de donde sacaremos la api para enviar los correos

**Paso 21:** Tendremos que bajar hasta encontrar esta opción que es para las apis de twilio

****

**Paso 22:** nos llevara a esta pestaña

****

**Paso 23:** arriba a la derechabuscaremos el login

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Paso 24:** nos llevara en esta pestaña donde pideque ingresemos un correo pero como aun no tenemos cuenta hay que registrarse

**Interfaz de usuario gráfica, Aplicación

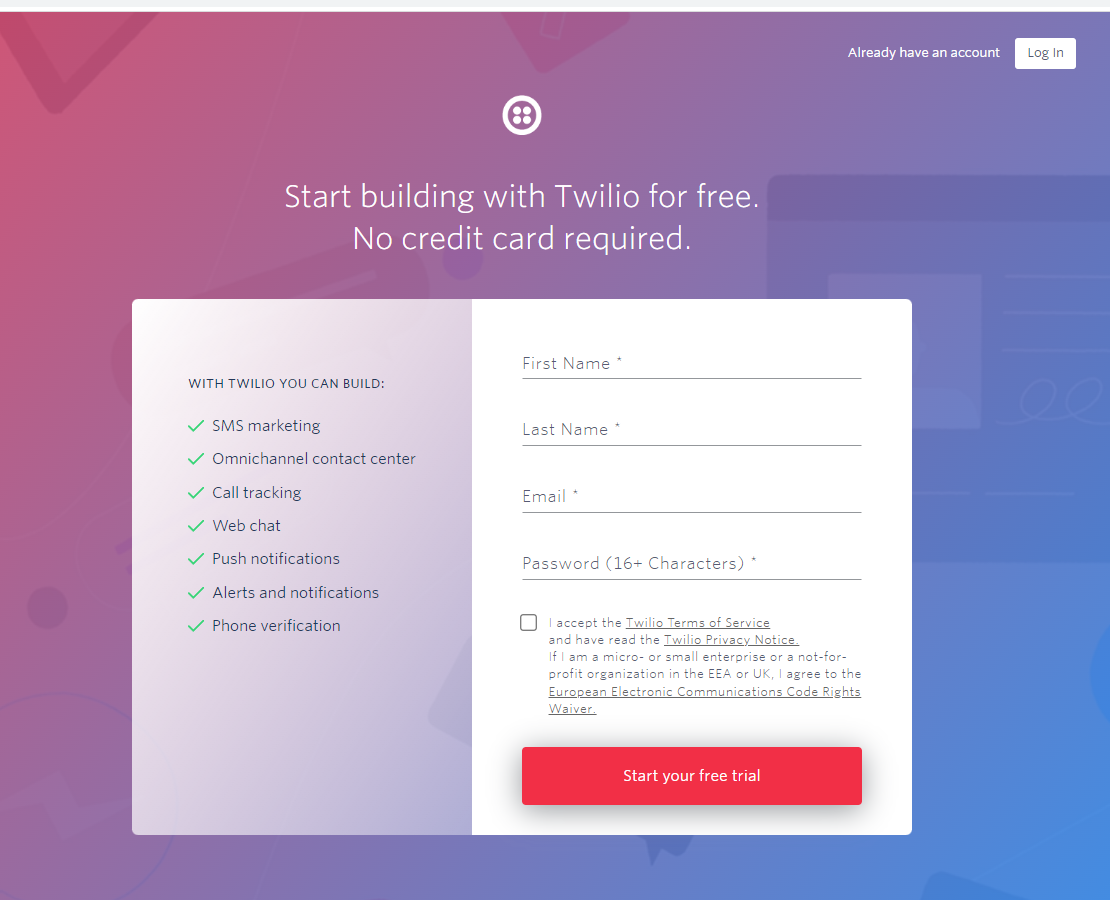
Descripción generada automáticamente**

**Paso 25:** le damos en sign up para registrarse

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Paso 26:** ahora llenamos los datos para el registro

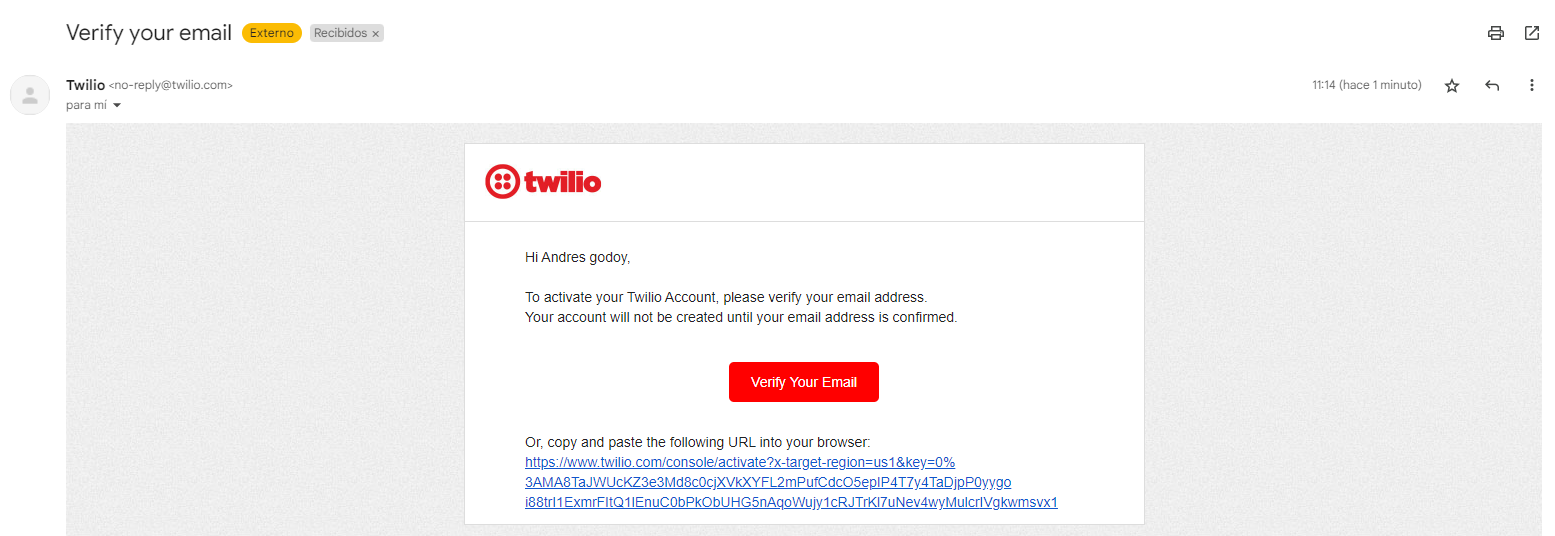
****

**Paso 27: ahora verificamos el correo**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

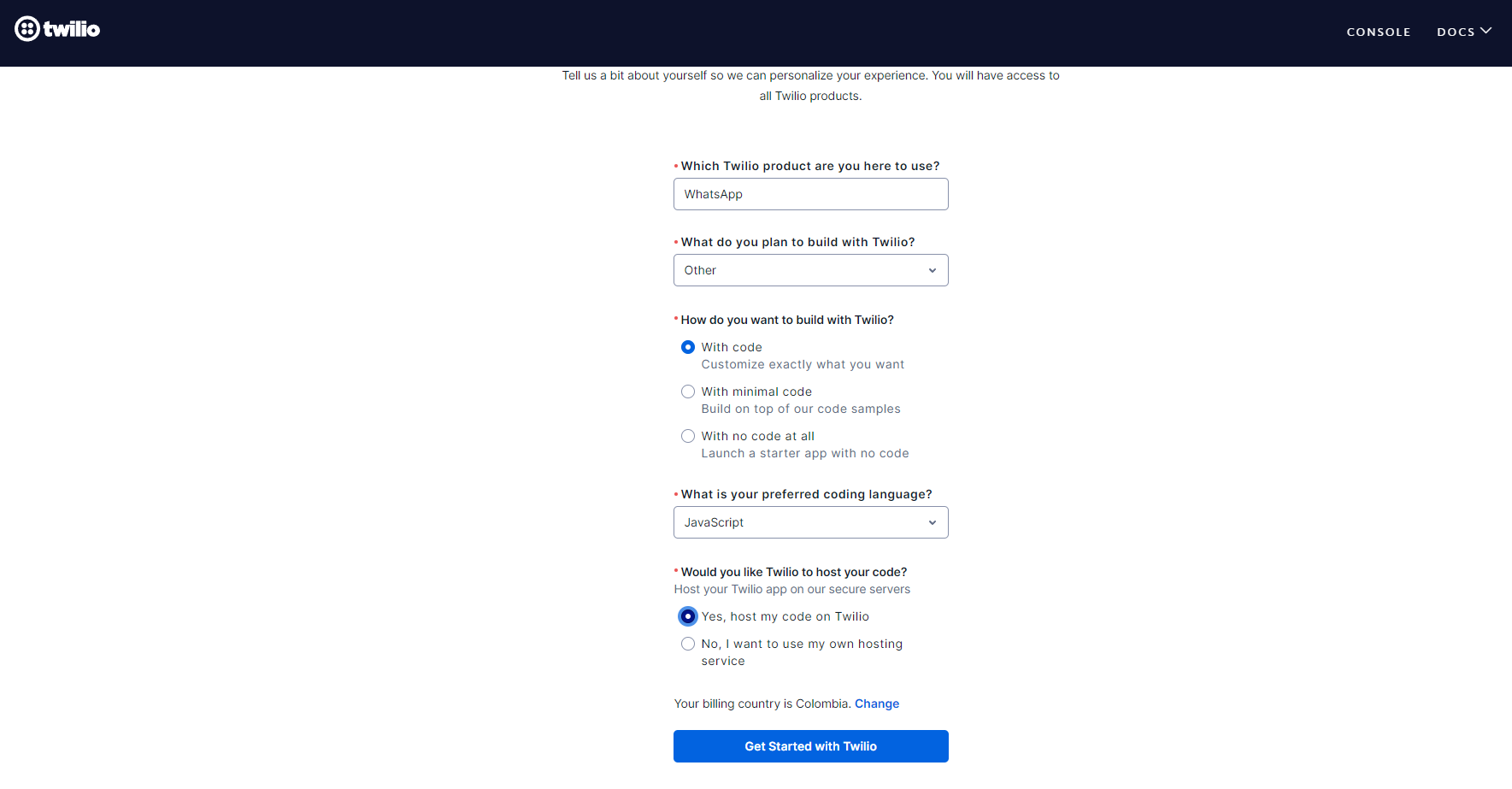
**Paso 28:** nos vamos al correo que registramos, buscamos el correo de twilio y le damos verificar email

****

**Paso 29: una vez verificado el correo, nos llevara a verificar nuestro numero**

**Interfaz de usuario gráfica, Aplicación, Correo electrónico

Descripción generada automáticamente**

**Paso 30**: nos darán un pequeño cuestionario para mejorar el uso de twilio****

**Paso 31**:una vez en el dashboard iremos a WhatsApp, a al izquierda en messaging > try it out > send a WhatsApp message

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Código QR

Descripción generada automáticamentePaso 32**: tendremos que agregar el numero que nos muestra en nuestro WhatsApp

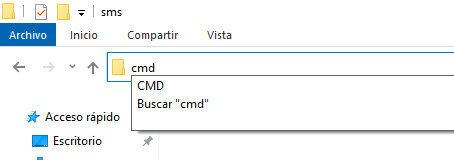
**Paso 33:** en el chat de Twilio escribimos el comando “join halfway-hunter” y recibiremos un mensaje diciendo que todo esta listo



**Paso 34:** ahora creamos una carpeta llamada “sms”



**Paso 35**: en el buscador de archivos dentro de la carpeta sms ponemos cmd para abrir la consola



**Paso 36**: en la consola ponemos “code .” para abrir el visual studio

Texto

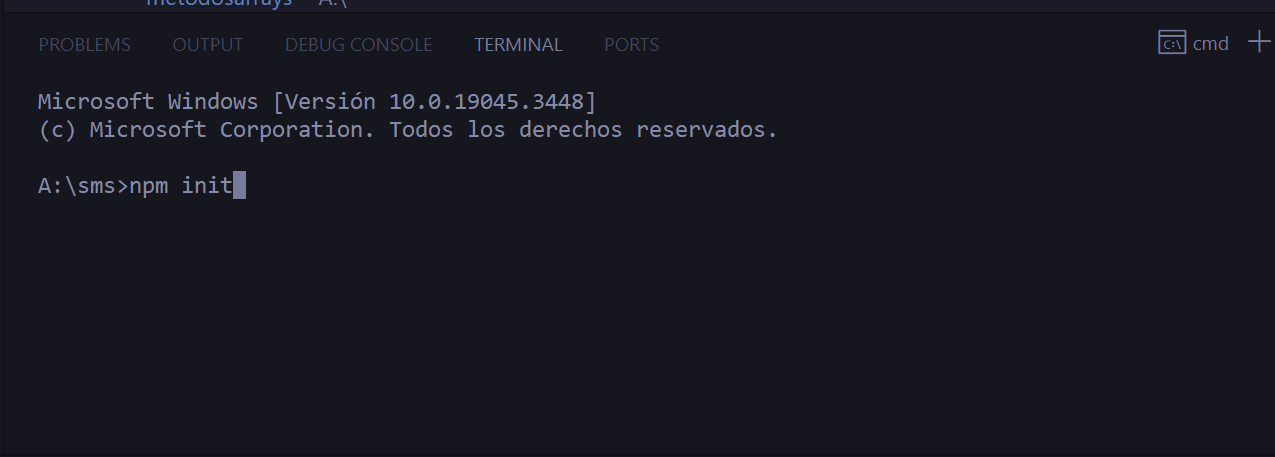
Descripción generada automáticamente

**Paso 37:** se abrirá esta pestaña del vs

**Captura de pantalla de computadora

Descripción generada automáticamente**

**Paso 38:**abrimos la consola del visual y ponemos el comando “npm init” para iniciar el node js

****

**Paso 39:**nos aparecerá una serie de datos para llenar aprieta enter para seguir una vez termines escribes “yes” para finalizar con los datos

**Texto

Descripción generada automáticamente**

**Paso 40:** al lado izquierdo en la carpeta sms se ha creado el archivo package.json

**Interfaz de usuario gráfica, Aplicación

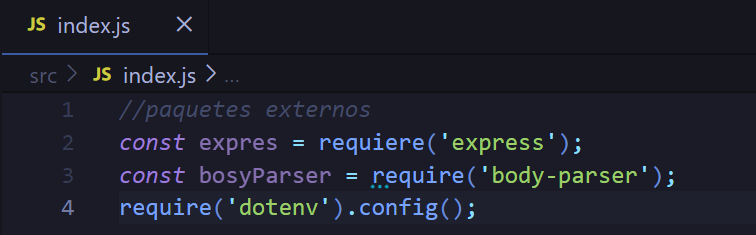
Descripción generada automáticamente**

**Paso 41:** en la carpeta sms creamos otra carpeta llamada “src” y dentro de esta crearemos el archivo llamado “index.js”

**Interfaz de usuario gráfica, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Paso 42:** en el archivo index.js importamos los paquetes y configuramos el paquete dotenv para cargar variables de entorno.

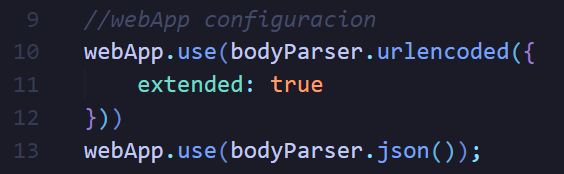
****

**Paso 43:** se inicializa la constante “webApp”

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

**Paso 44:** Este middleware se utiliza para analizar datos codificados en la URL (línea 10), permite analizar datos codificados en la URL con una sintaxis más rica(línea 11), se utiliza para analizar el cuerpo de las solicitudes en formato JSON (línea 13)(El middleware es un software con el que las diferentes aplicaciones se comunican entre sí)



**Paso 45**: configurando el puerto del servidor mediante una variable de entorno llamada PORT

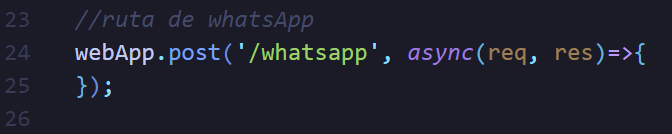


**Paso 46**: Esto configura un manejador de rutas para las solicitudes HTTP GET en la ruta raíz (línea 19), Este es el manejador de la ruta, una función que se ejecutará cuando se reciba una solicitud GET en la ruta raíz. Toma dos argumentos, req (la solicitud) y res (la respuesta). En este caso, simplemente envía la respuesta "hello world.!" al cliente (línea 20).

Texto

Descripción generada automáticamente

**Paso 47:** estás configurando una ruta en tu aplicación Express que maneja solicitudes POST en la ruta '/whatsapp'



**Paso 48**: inicia tu servidor Express y lo hace escuchar en el puerto definido por la variable PORT. El segundo argumento de listen es una función de devolución de llamada que se ejecutará una vez que el servidor esté listo para recibir conexiones.

Texto

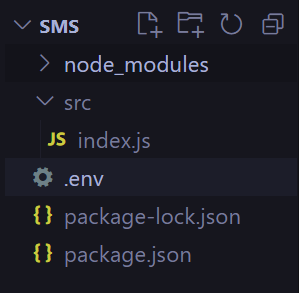
Descripción generada automáticamente

**Paso 49:** ahora en la consola instalamos los paquetes

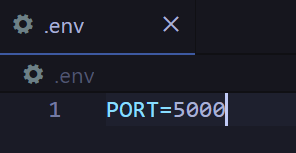
Texto

Descripción generada automáticamente

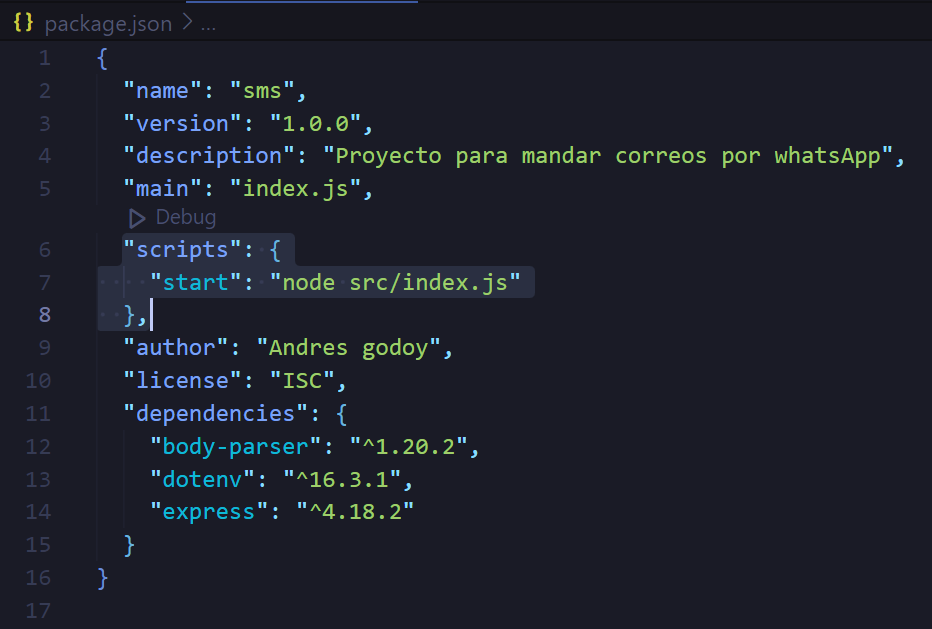
**Paso 50**: se nos creara una carpeta llamada node\_modules y un archivo package\_lock.json, en la carpeta raíz crearemos un archivo llamado “.env”



**Paso 51**: dentro de este creamos una variable PORT y le asignamos el valor 5000, este será el puerto



**Paso 52**: nos vamos al archivo packege.json y en scripts “test” lo cambiamos por “start” y en su interior escribimos “node src/index.js” que es la ruta del archivo que usaremos para iniciar el proyecto



**Paso 53:** ahora en la consola escribimos “npm install -g nodemon” el cual realizara los cambios que detecte automáticamente en el proyecto

Texto

Descripción generada automáticamente

**Paso 54:** en la consola escribimos “nodemon src/index.js”Texto

Descripción generada automáticamente

**Paso 55**: ahora en el buscador escribimos “localhost:5000” y nos debería salir “hello world”

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

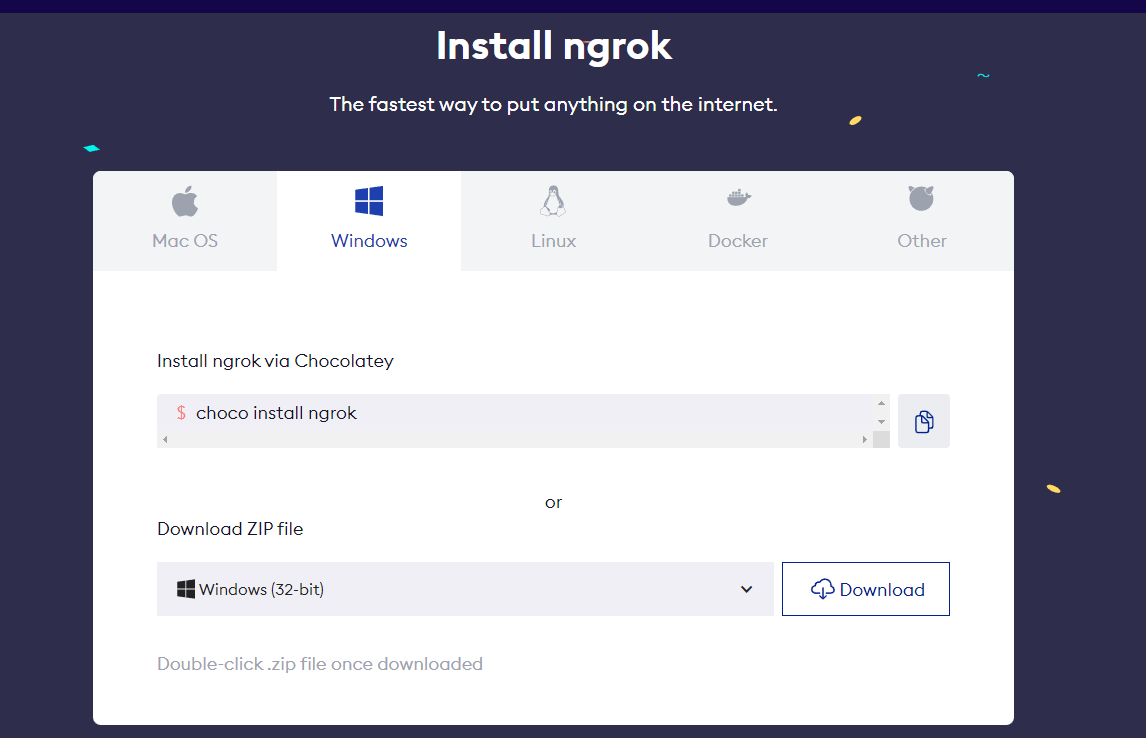
**Paso 56:** en Google buscamos ngroky le damos a la primera opción

****

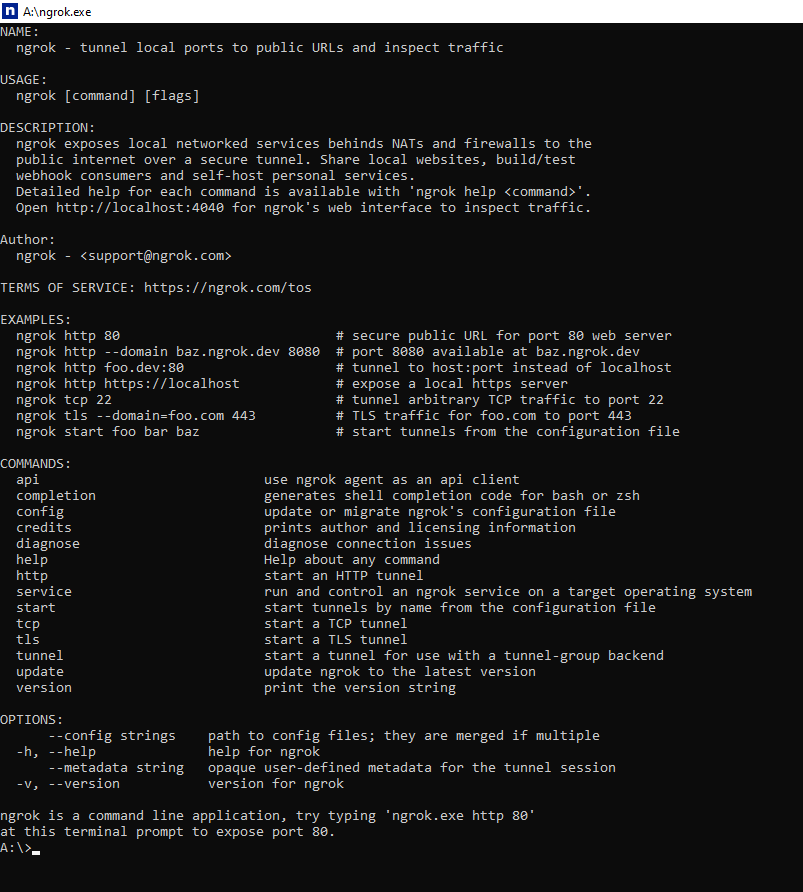
**Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamentePaso 57**: se nos abre esta pestaña, nos vamos arriba en download

**Paso 58**: elegimos nuestro sistema operativo y los bits y descargamos



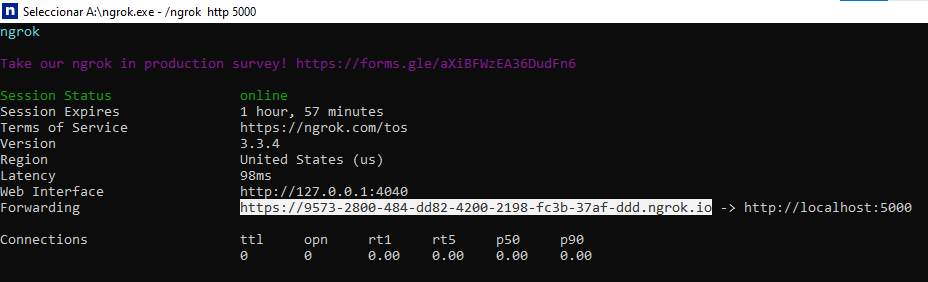
**Paso 59:** nos dará un archivo comprimido zip, los extraemos y abrimos, nos mandará a una consola



**Paso 60:** ahora en la consola de ngrok escribimos “/ngrok http 5000”



**Paso 61:** nos mandara esta serie de datos, nosotros tomaremos el link del forwarding



**Paso 62:** ahoranos vamos a twilio

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**Paso 63:** a la izquierda en develop le damos a messaging

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Paso 64:** le damos en “try it out” y nos desplegara este menú

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Paso 65:** en el menú le damos a “send a WhatsApp message”, en la opción “Sandbox settings”

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Aplicación, Correo electrónico

Descripción generada automáticamente**

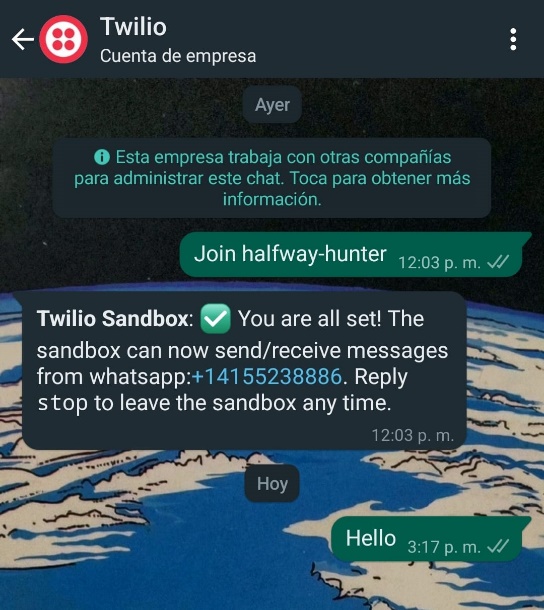
**Paso 66:** pegamos el link que nos dio ngrok y al final le agregamos la ruta de WhatsApp “/whatsapp”, le damos al botón de save

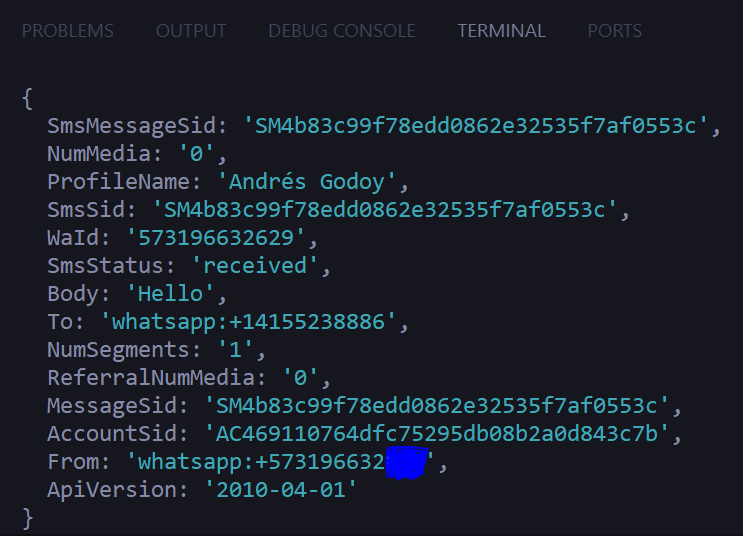
**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente**

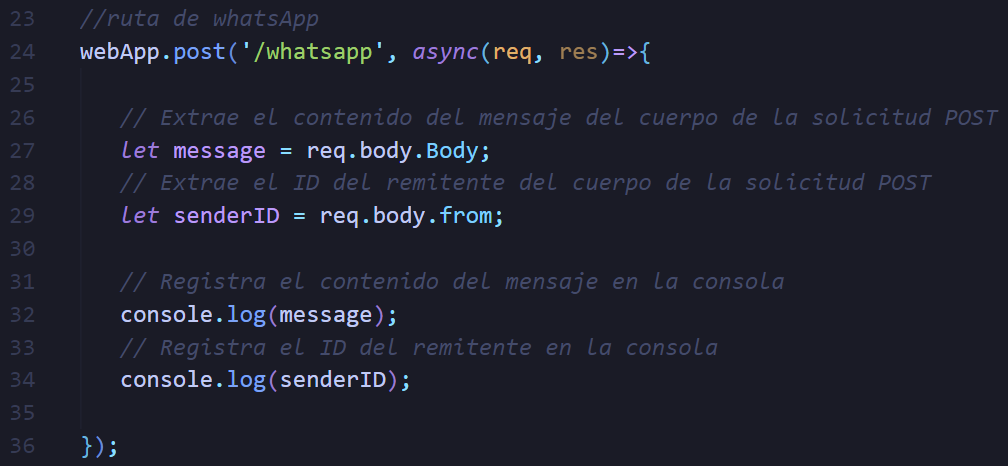
**Paso 67:** ahora en el archivo index.js en la ruta de whatsapp cuando se realice una solicitud POST a la ruta los datos enviados en el cuerpo de la solicitud se mostrarán en la consola****

**Paso 68:** ahora en el chat de WhatsApp que tenemos con twilio escribimos “hello” y nos debería parecer en la consola





**Paso 69:** extrae el contenido del mensaje recibido desde WhatsApp (línea 27), extrayendo el identificador del remitente del mensaje y lo asignas a la variable (línea 29), muestran el contenido del mensaje y el identificador del remitente en la consola (línea 32 y 34)



**Paso 70**: ahora crearemos una función que envié mensajes devuelta, primero crearemos en la carpeta raíz una carpeta llamada “funciones” dentro de esta crearemos un archivo llamado “mensaje-whatsapp.js”

Texto

Descripción generada automáticamente con confianza media

**Paso 71**: extrae las credenciales de autenticación de Twilio desde las variables de entorno (línea 3 y 4), importando y configurando el cliente Twilio. importa el módulo twilio y se le pasan las credenciales de autenticación (accountSid y authToken) como argumentos. El objeto de opciones { lazyLoading: false } se pasa como tercer argumento. La opción lazyLoading: false significa que el cliente Twilio se inicializará inmediatamente en lugar de esperar hasta que se realice la primera llamada a la API de Twilio.



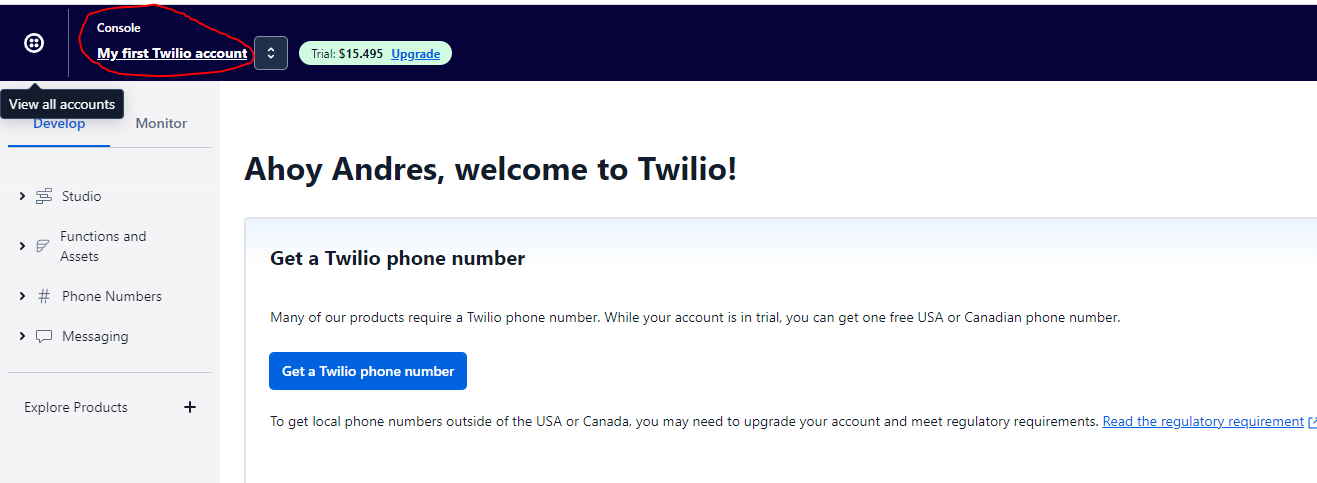
**Paso 72**: en el archivo “.env “ ponemos TWILIO\_ACCOUNT\_SID y

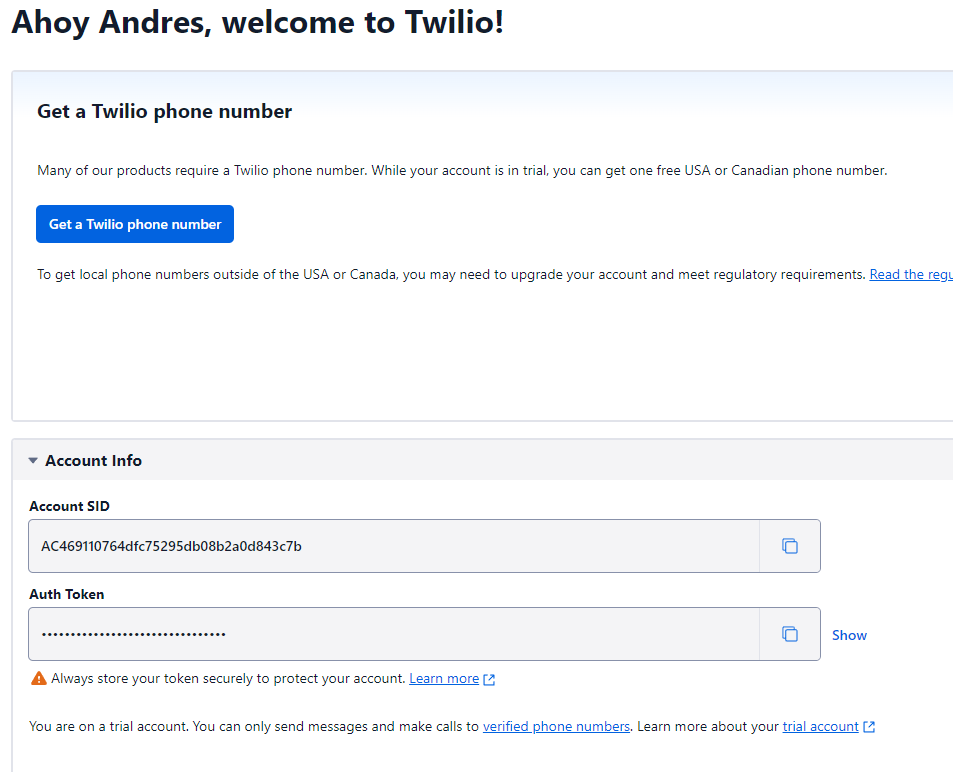
TWILIO\_AUTH\_TOKEN

Interfaz de usuario gráfica, Aplicación

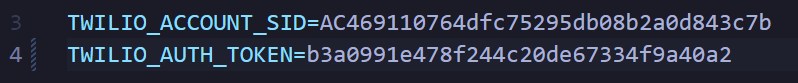
Descripción generada automáticamente

**Paso 73**: ahora nos vamos a twilio y arriba a la izquierda le damos en “my first Twilio account”.



**Paso 74:** en account info vamos copiar el account SID y el auth Token ****

**Paso 75:** en el archivo “.env” los pegamos



**Paso 76**: en el archivo mensaje-whatsapp creamos esta función llamada “sendMessage” y le damos dos parámetros “messge” y “senderID”

Texto

Descripción generada automáticamente

**Paso 77**: llama al método create del cliente Twilio para crear un mensaje (línea15), senderID es la variable que contiene el número de teléfono del destinatario. El contenido del mensaje de WhatsApp, El número de teléfono de WhatsApp desde el cual se enviará el mensaje. En este caso, se utiliza un número de teléfono de Twilio.

Texto

Descripción generada automáticamente

**Paso 78:**exportamos el modulo sendMessage

**Texto

Descripción generada automáticamente**

**Paso 79:** en el index.js hacemos referencia a la función exportada sendMessage y se le asigna a la constante “WA”

**Logotipo

Descripción generada automáticamente**

**Paso 80:** la función sendMessage exportada desde el módulo mensaje-whatsapp. La función toma dos argumentos: el contenido del mensaje y el número de teléfono de WhatsApp del destinatario

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

**Paso 81:**ahora en la consola instalamos el paquete de twilio y activamos nuevamente le nodemon (“nodemon src/index.js”)

****

**Paso 82:** en la consola escribimos nuevamente “nodemon src/index.js”



**Paso 83**: nos vamos al chat de WhatsApp con twilio y escribimos hola y nos debe responder



**Paso 84:** ahora en la consola nos debe aparecer el numero y el body del mensaje

**Texto

Descripción generada automáticamente**