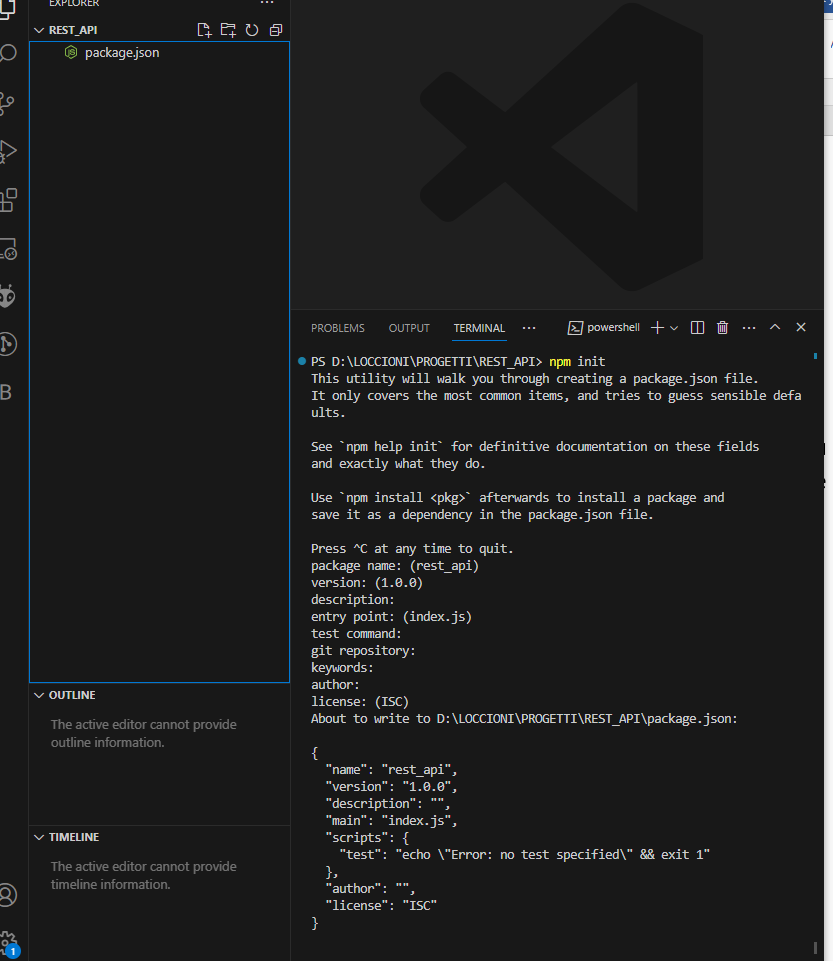
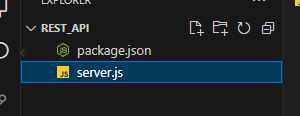
INFO SU C++ NODEJS E JAVASCRIPT REST API

1. CONFIGURAZIONE DI NODEJS: (vedi: https://www.youtube.com/watch?v=cUlih69K0dI)

* Creare una cartella ed inserirla su VScode;
* Poi aprire un terminare e scrivere : npm init



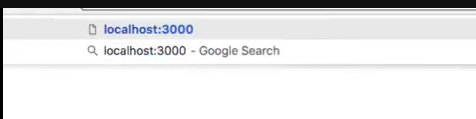
* Viene creato il file .json, ora creare un file con il nome “server.js” in cui scrivere il server:



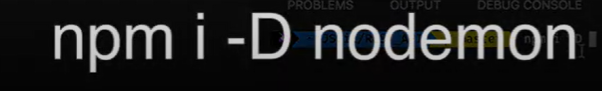
* Una volta scritto il serve per lanciare l’applicazione scrivere :



* Poi andare sul browser sulla porta 3000:

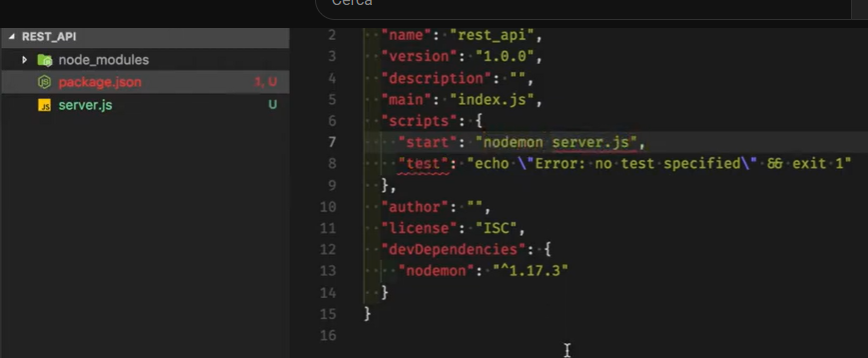


1. È possibile installare un tool:

In modo da rendere il server RT cioè evitare di stoppare il server per aggiornarlo.

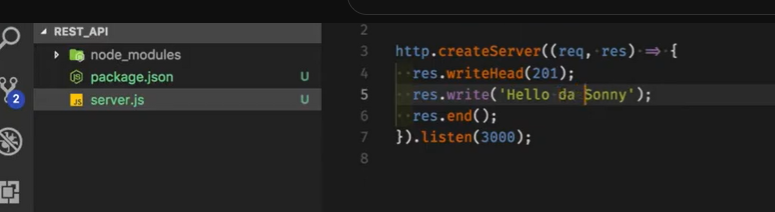
Quindi creiamo uno script per far funzionare questo:



Quindi riavvio il server con : npm start.

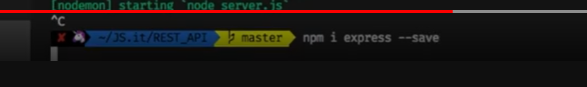


Ora se modifico il file sorgente con una page refresh:

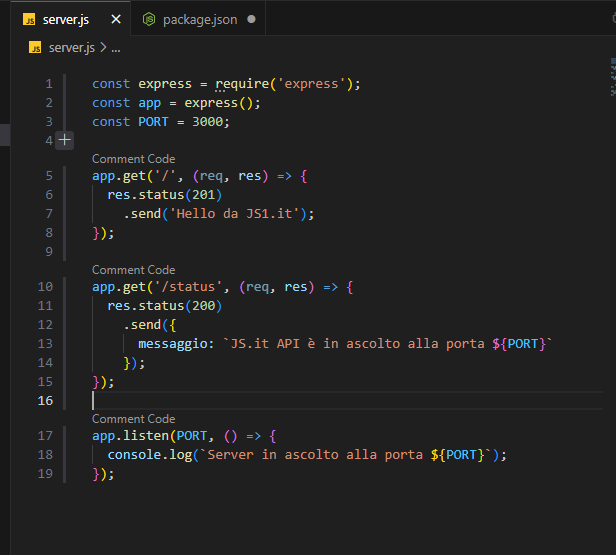


Ottengo un a modifica.

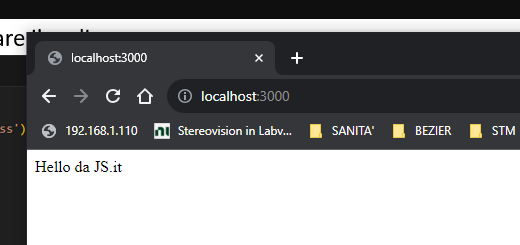
1. Uso di express:



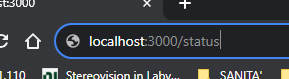
Fatto questo vado a modificare il codice come segue:



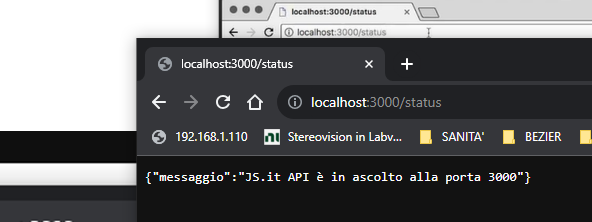
Ottengo questo:



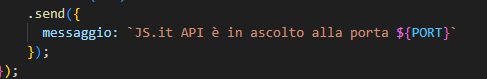
Se invece vado ad aggiungere:



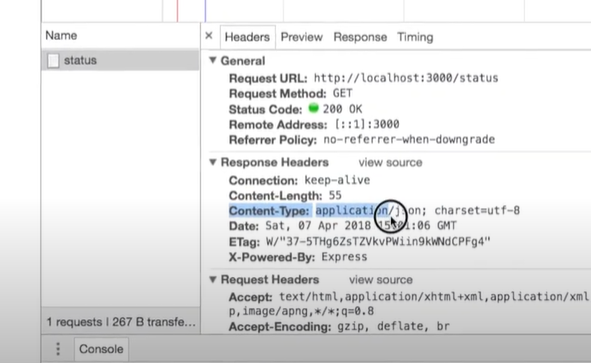
Ottengo:



Inoltre come vedi il messaggio è in messaggio json perché è scritto con parentisi qui:



Dal content type: (si vede che è in json).



CLIENT C++ CHE RICEVE LE INFORMAZIONI DAL SERVER

Seguiamo ora questo tutorial che ci permette di fondere le informazioni tra nodejs e C++:

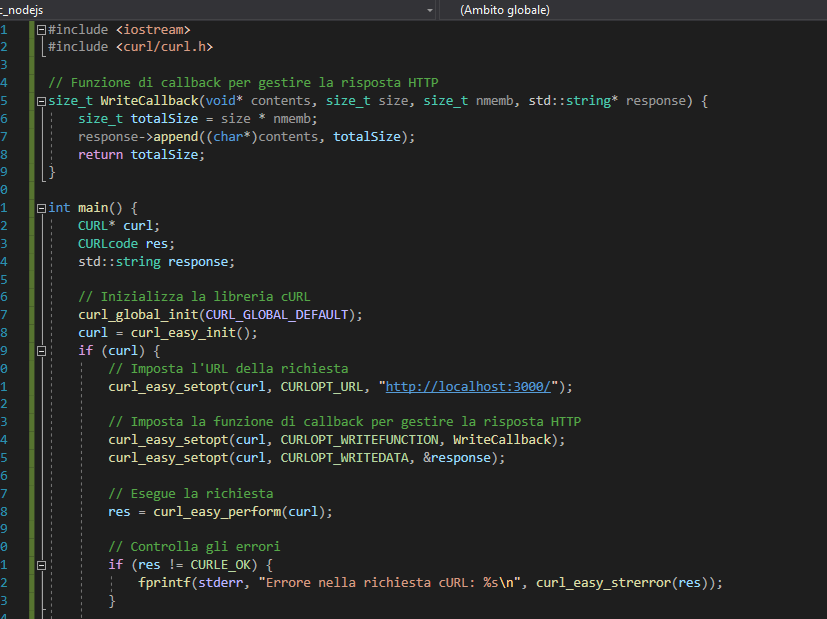
<https://www.tutorialspoint.com/how-to-communicate-json-data-between-cplusplus-and-node-js>

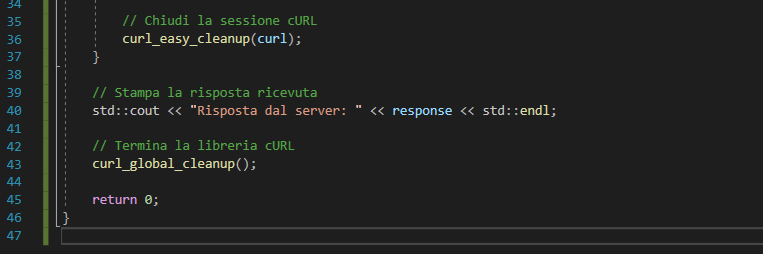
Effettua richieste HTTP da C++

Ci serve la libreria CURL:

Per fare richieste HTTP da un programma C++, puoi utilizzare una libreria esterna che semplifica l'invio e la gestione delle richieste HTTP. Un'opzione comune è utilizzare la libreria CURL, che è una libreria di trasferimento di dati che supporta una vasta gamma di protocolli, tra cui HTTP.

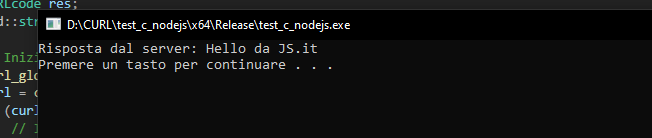
Ecco un esempio di come fare una richiesta GET utilizzando CURL in C++:



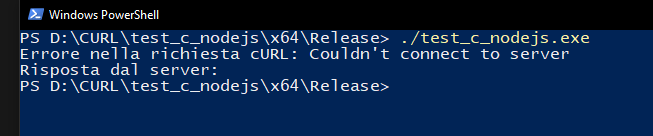


Questo esempio esegue una richiesta GET all'URL "http://localhost:3000/" e stampa la risposta ricevuta dal server Node.js nel terminale. Puoi personalizzare l'URL della richiesta secondo le tue esigenze.

Se il server nodejs è in esecuzione, abbiamo che la risposta del server la leggo sul mio programma C++:



Se vogliamo fermare il server che gira devo inviare sul VSCode: ctrl+c



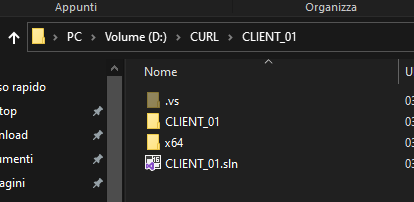
C++ -> NODEJS -> WEB-BROWSER

programma in C++ che crea dei dati per esempio acquisiti da un sensore e li invia al server scritto in NodeJS e poi li posso visualizzare tramite un browser web.

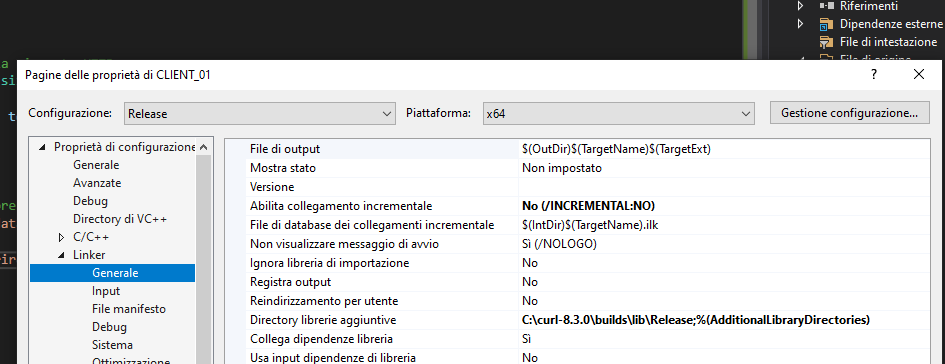
1. Programma C++ per acquisire dati dal sensore:

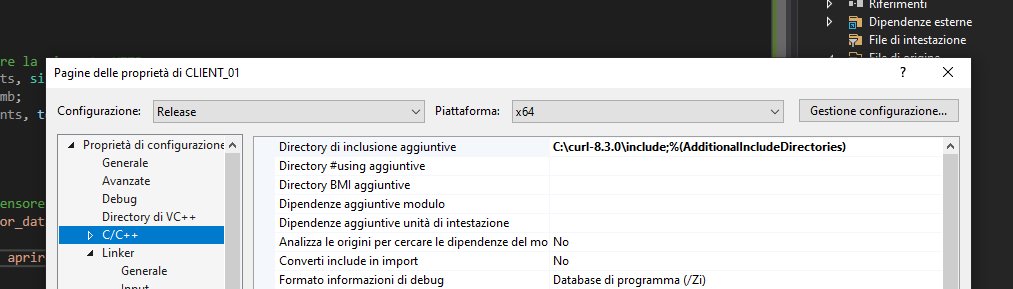
Scrivi un programma in C++ che acquisisce dati dal sensore. Puoi utilizzare le librerie adeguate per comunicare con il sensore e leggere i dati. Successivamente, puoi utilizzare la libreria CURL o altre librerie HTTP in C++ per inviare i dati al server Node.js tramite richieste HTTP POST o GET.

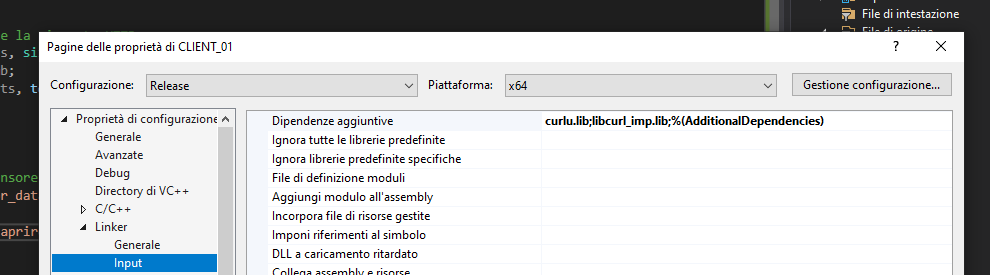
Per prima cosa ho creato un progetto nuovo su:



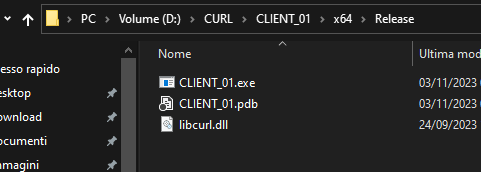
Inserendo tutte le dipendenze in modalità Release e x64:







Infine dopo la compilazione inserire la .dll di CURL nel percorso dell’eseguibile:



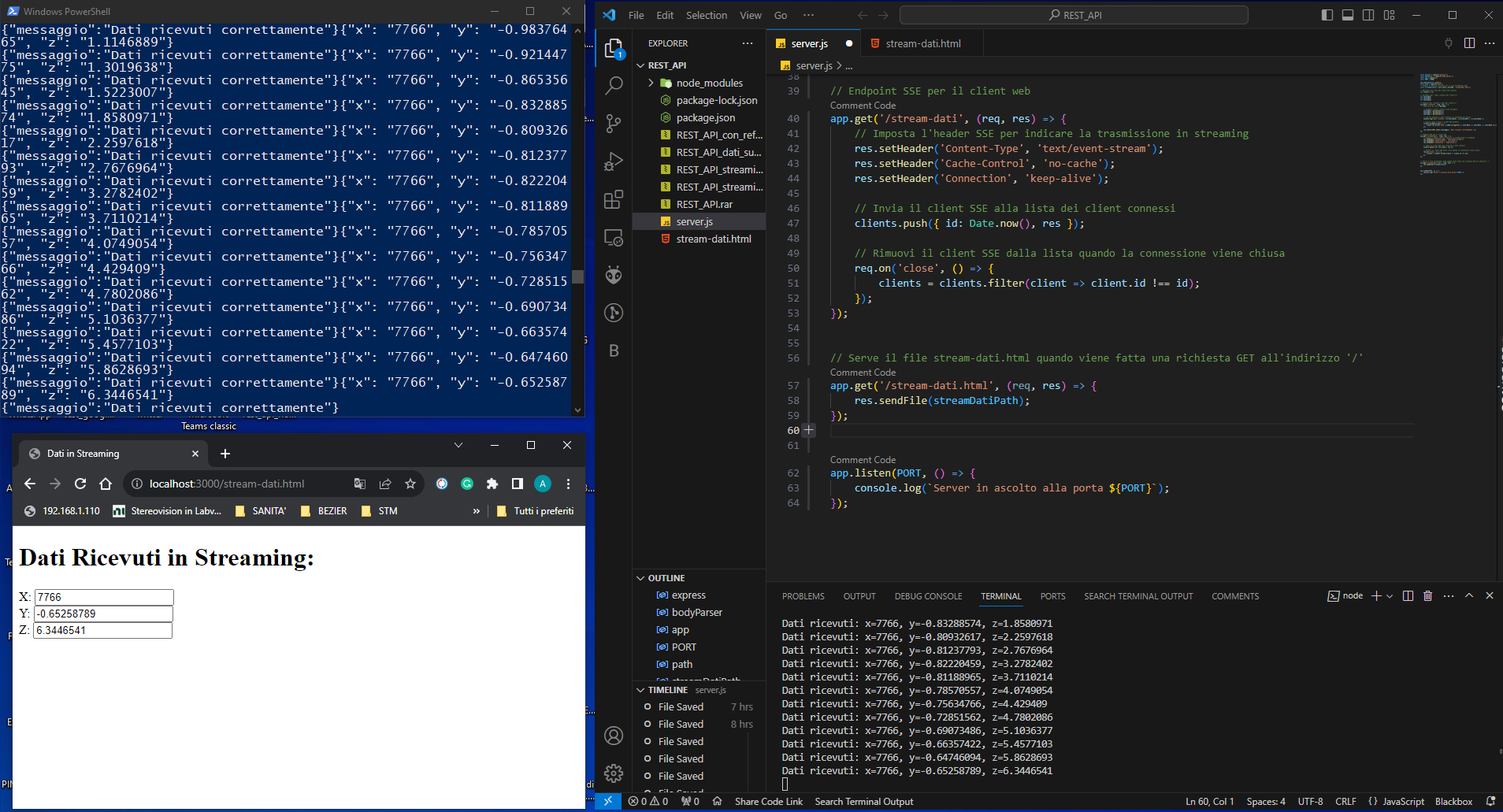
Descrizione del programma:

In questo esempio, il programma C++ legge le righe dal file "sensor\_data.txt" e invia ciascuna riga al server Node.js utilizzando una richiesta HTTP POST. Dopo ogni invio, il programma attende per 1 secondo prima di inviare la riga successiva.

Assicurati di sostituire l'URL nell'opzione CURLOPT\_URL con l'endpoint appropriato del server Node.js a cui desideri inviare i dati. E, naturalmente, assicurati che il server Node.js sia in esecuzione e pronto per ricevere le richieste.

Nota: Questo è solo un esempio di base per mostrarti come puoi inviare dati al server da un programma C++. Potresti voler aggiungere ulteriori controlli sugli errori, gestire le risposte del server e gestire eventuali eccezioni o situazioni impreviste nel tuo codice.

Da spiegare:



Per pubblicare: <https://www.youtube.com/watch?v=CquOvfqMWn4>

Hostare website: <https://www.youtube.com/watch?v=YZsQ8V16vpY>

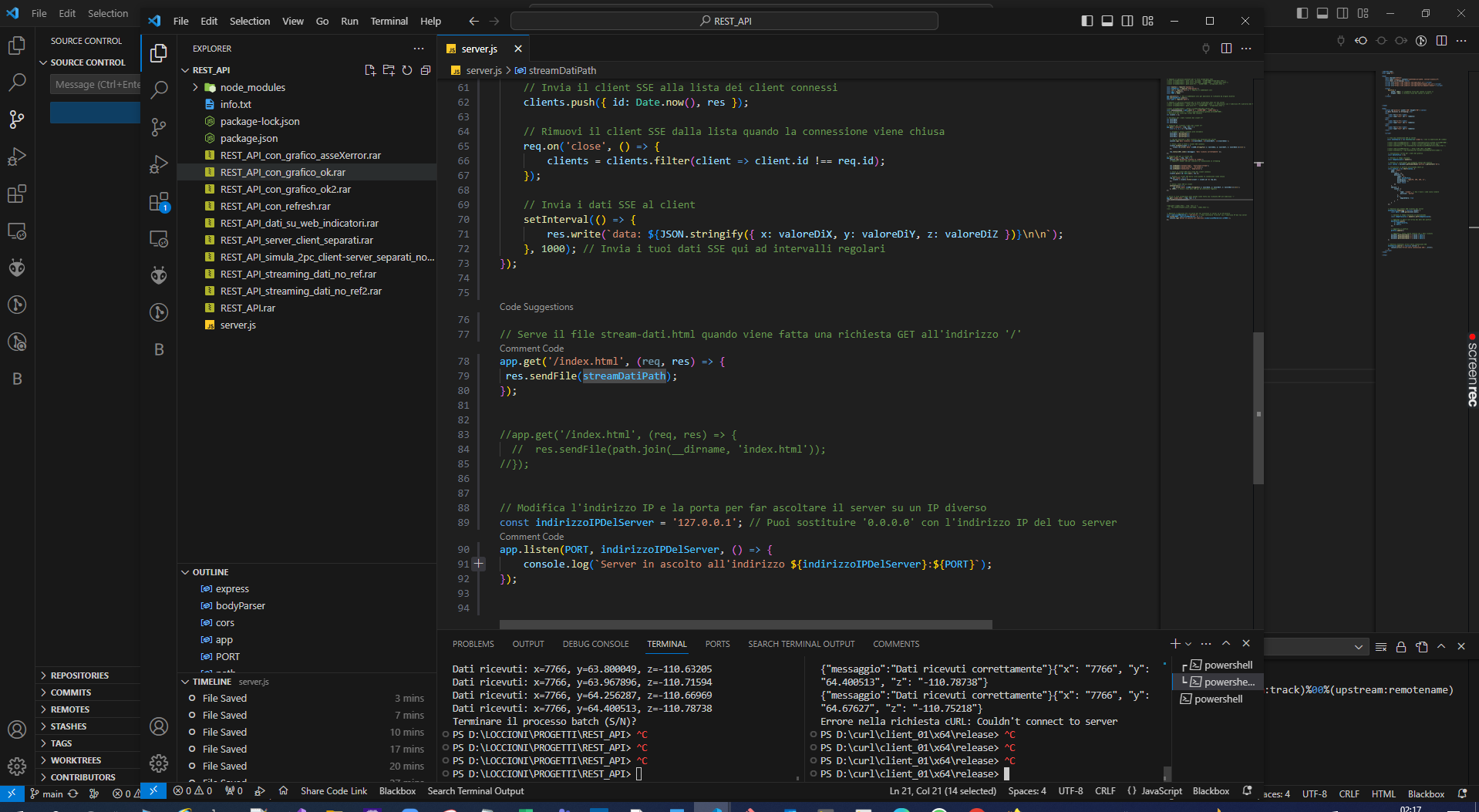
PS D:\curl\client\_01\x64\release>

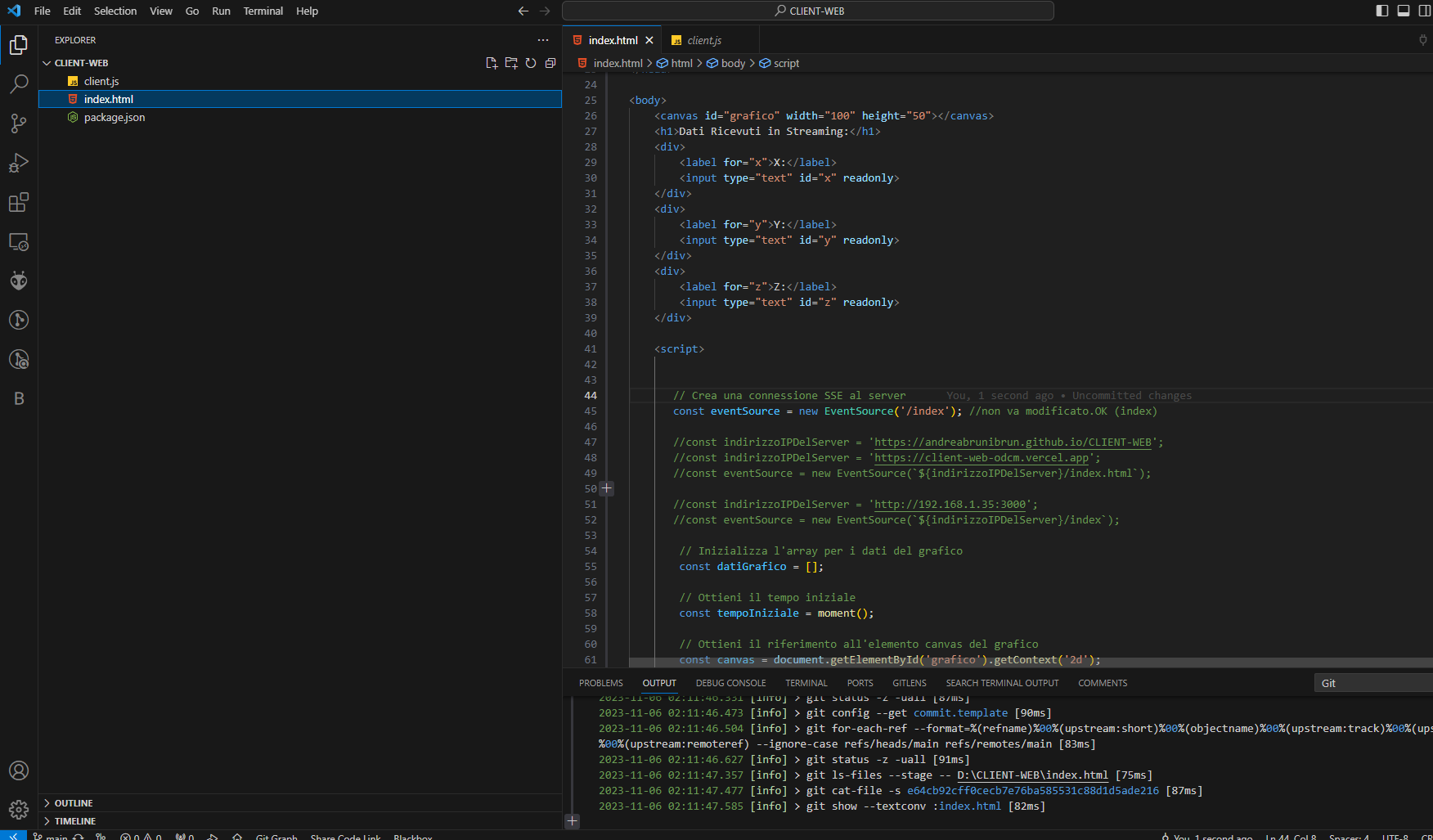
Vercel: <https://vercel.com/abrunis-projects/client-web-odcm>

<http://127.0.0.1:3000/index.html> (OK)

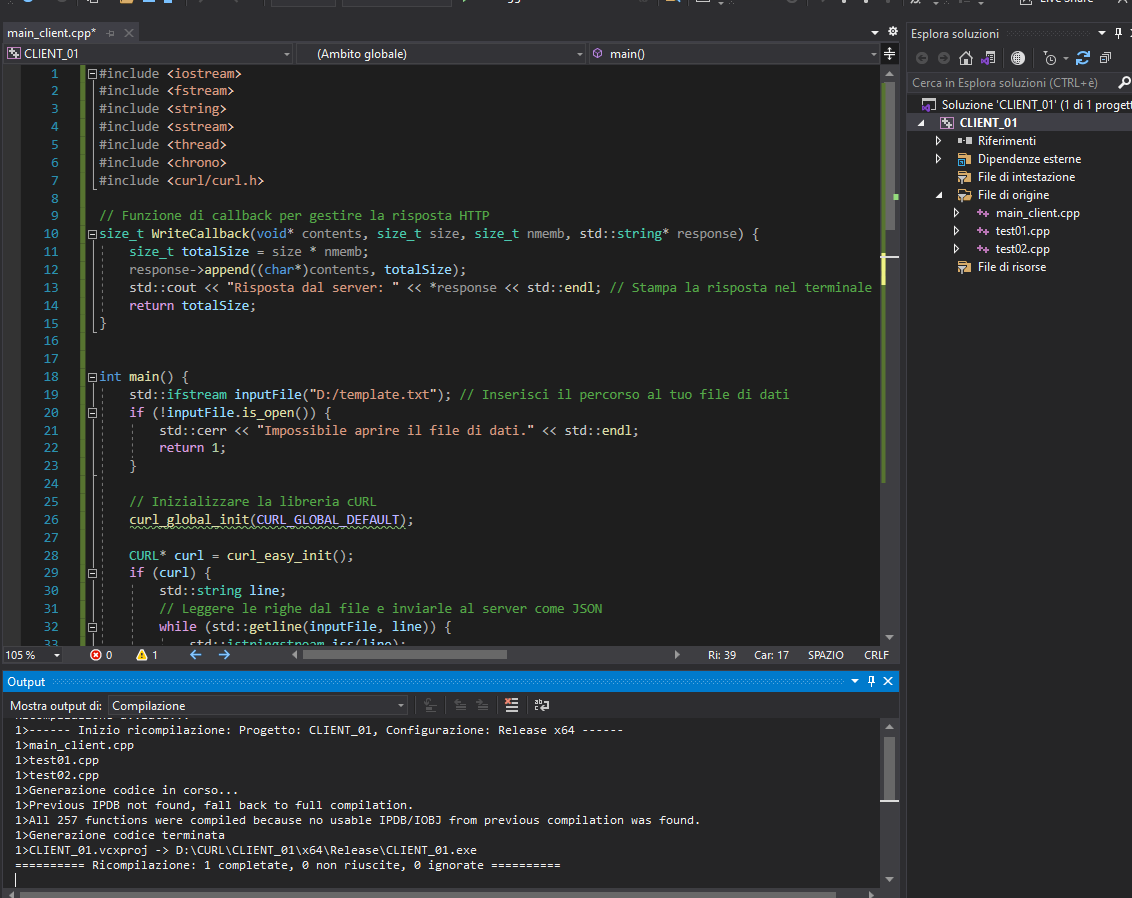
<https://client-web-odcm.vercel.app/> (ko)

<https://andreabrunibrun.github.io/CLIENT-WEB/index.html> (KO GIT).









D:\CLIENT-WEB

<http://127.0.0.1:3000/index.html>

<https://andreabrunibrun.github.io/CLIENT-WEB/index.html>

<https://client-web-odcm.vercel.app/>

<https://vercel.com/abrunis-projects/client-web-odcm>

<https://github.com/AndreaBruniBRUN/CLIENT-WEB/tree/main>

D:\LOCCIONI\PROGETTI\REST\_API

What is my IP

Da aprire per test:

<https://andreabrunibrun.github.io/CLIENT-WEB/index.html>

<https://whatismyipaddress.com/>

<https://github.com/AndreaBruniBRUN/CLIENT-WEB/tree/main>

<https://chat.openai.com/c/77c9b242-3873-4666-83e5-0ad832cbccb1>

corso node js: <https://www.youtube.com/watch?v=k5RoVFYICgA>

<https://www.youtube.com/playlist?list=PLGtYBmlIBbHSod7PSri_7nEERtBNuFzAb>

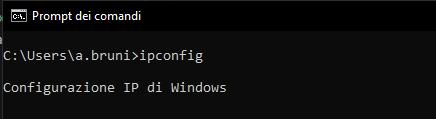
vps: <https://www.scalahosting.com/blog/installing-node-js-on-a-vps-server-2/>

<https://www.youtube.com/watch?v=LZiWOkGLuGM>

gratis: <https://www.youtube.com/watch?v=fYdrgRuWhT0>

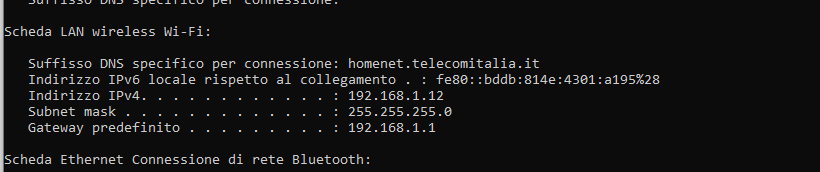
**FUNZIONAMENTO DEL PROGRAMMA COMPLETO**

Aprire il prompt dei comandi con CMD:



Digitare “ipconfig” e scorrere in fondo.

Nella voce schede LAN WIFI abbiamo l’indirizzo della porta a cui fa riferimento il PC connesso al modem:

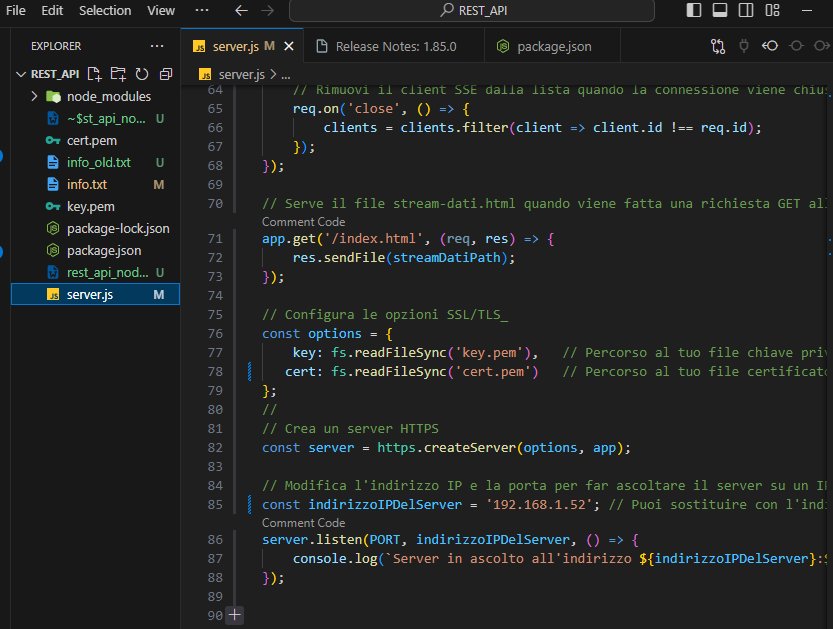


Ipv4: 192.168.1.12

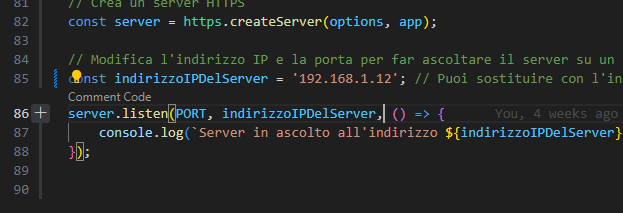
Aprire il SW del Server: D:\LOCCIONI\PROGETTI\REST\_API

Trascinare la cartella su VSCode.

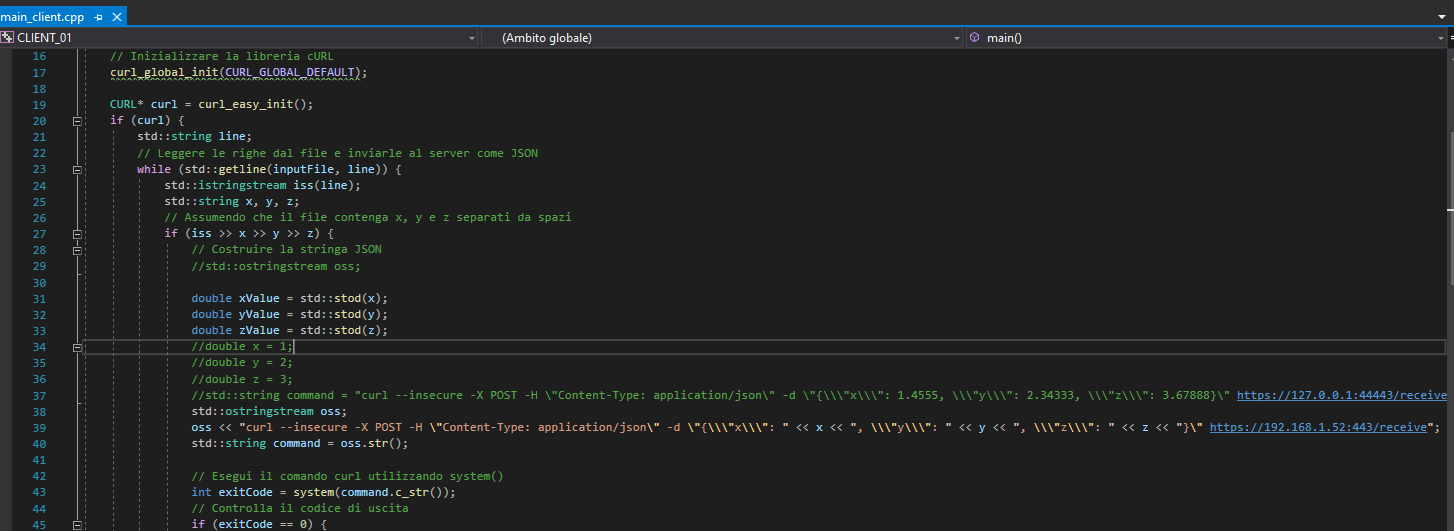
Abbiamo il seguente programma:



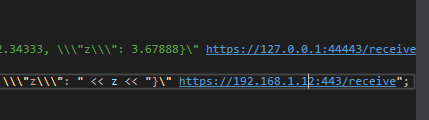
Alla riga 85 cambiare l’indirizzo con quello trovato sopra:



A questo punto aprire il seguente programma: D:\CURL\CLIENT\_01\CLIENT\_01.sln

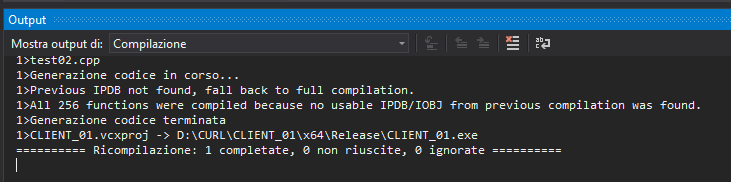


Andare alla riga 37 e cambiare anche qui l’indirizzo IP:



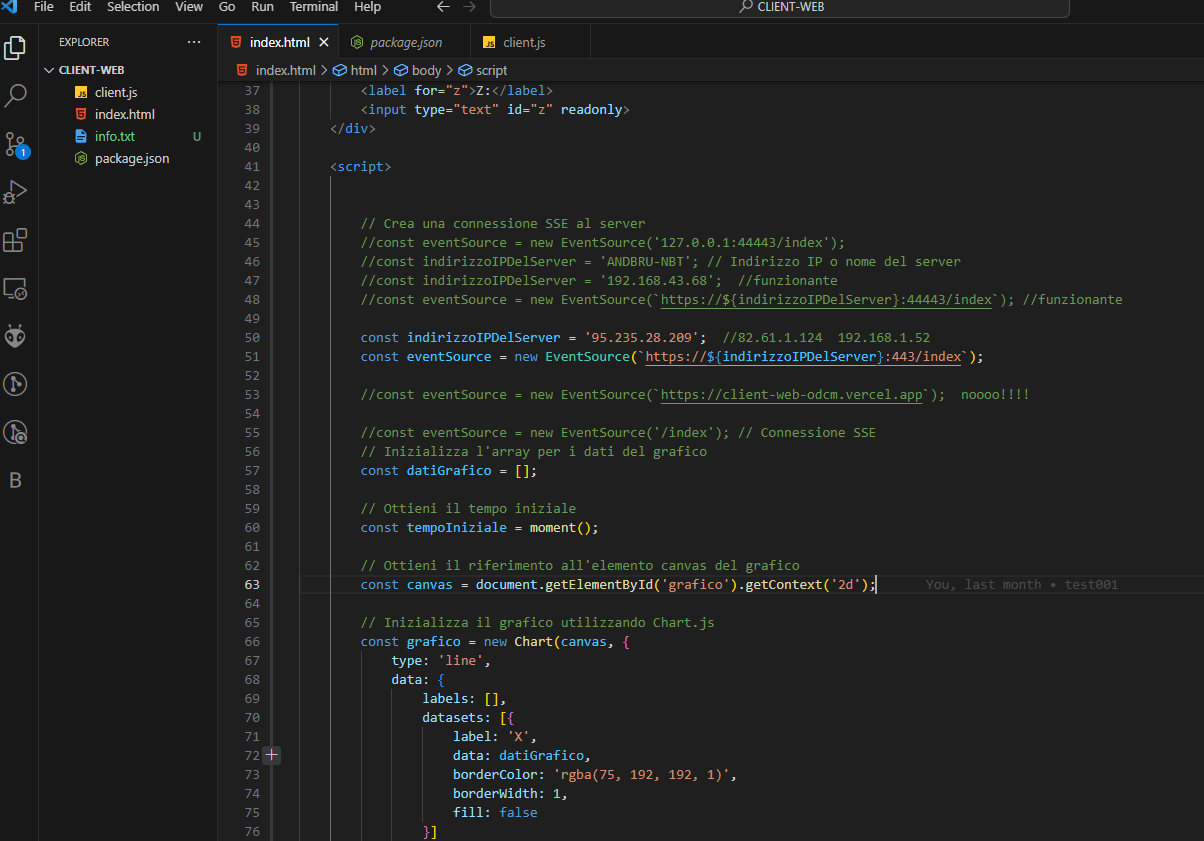
Questo è il programma scritto in linguaggio C++ che invia i dati acquisiti dallo strumento (per il momento è simulato con la lettura da file).

Fatto questo bisogna ricompilare il programma.



Ora andare sul seguente programma: D:\CLIENT-WEB

Aprire un nuovo VSCode e trascina la cartella:



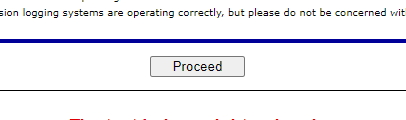
Ora bisogna modificare l’indirizzo per l’accesso da ip pubblico.

Per trovare l’ip pubblico procedere nel modo seguente.

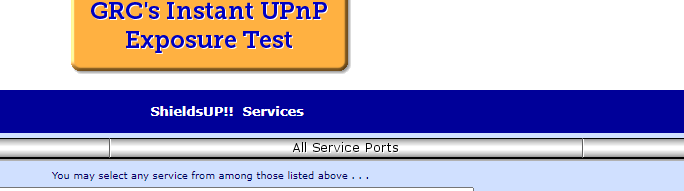
La cosa importante ora è trovare l’indirizzo pubblico sul quale la pagina web che risiede su VERCEL deve puntare per poter entrare nel rooter di casa in modo da prelevare i dati e visualizzarli da pagina web esterna.

Vai al seguente indirizzo: <https://www.grc.com/x/ne.dll?bh0bkyd2>

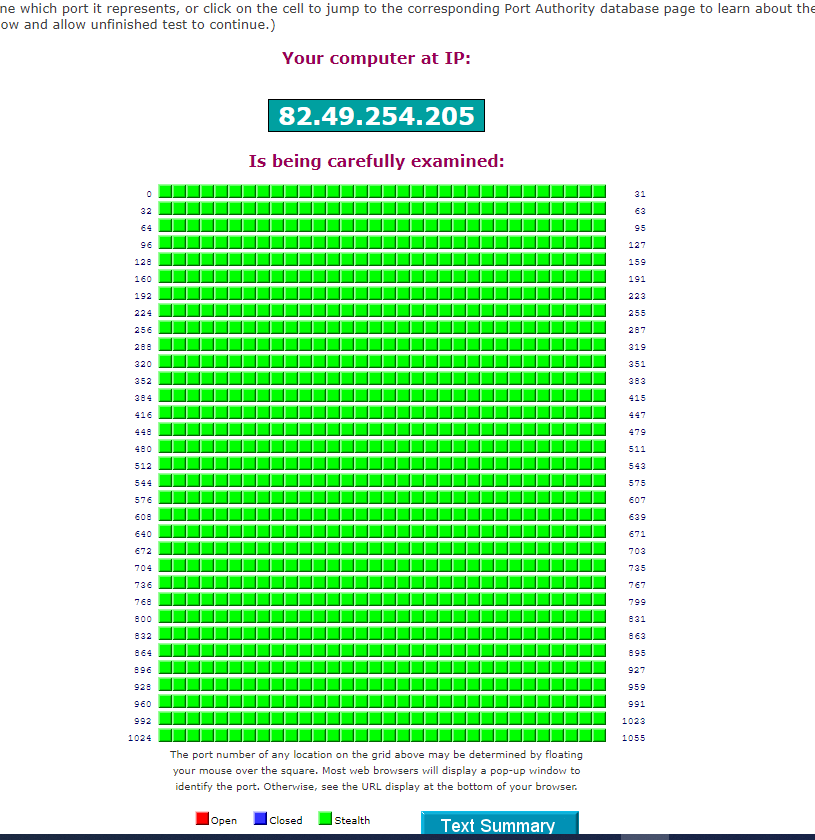
Premere su:



Vai ora su:



Vengono scansionate le prime 1056 porte del rooter:



Quindi IP pubblico è il seguente:

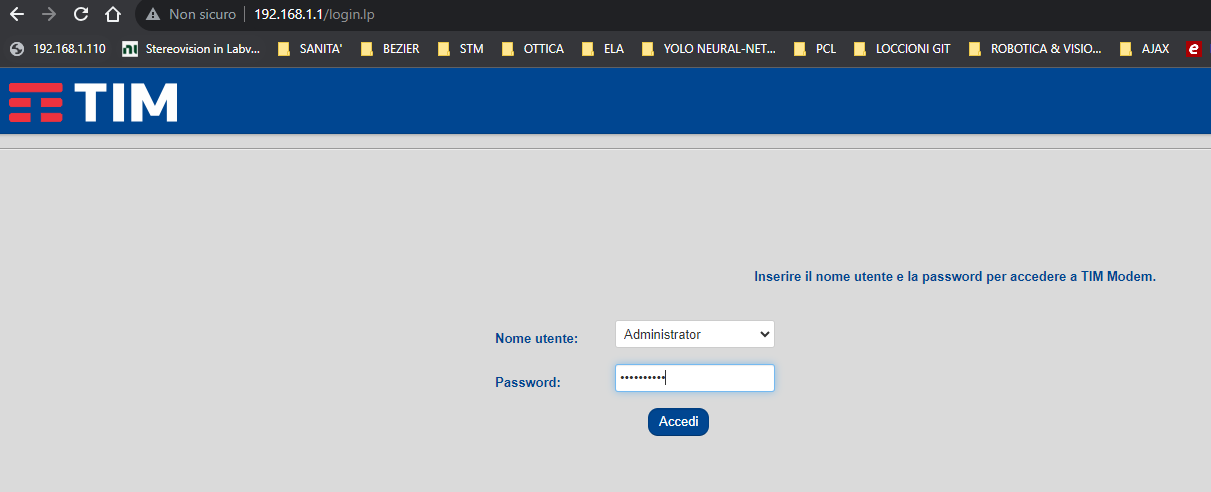


Come vediamo nessuna porta è aperta e quindi è impossibile per una pagina web esterna o per un server interno mandare fuori dei dati.

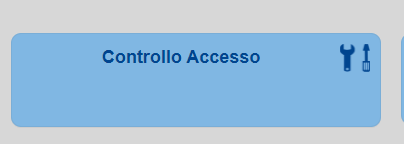
Quindi bisogna aprire una porta. In particolare siccome il server è creato con modalità SSL la porta da aprire è la 443.

Aprire una pagina web: <http://192.168.1.1/>

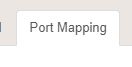
Accedere quindi al modem:



Vai su:



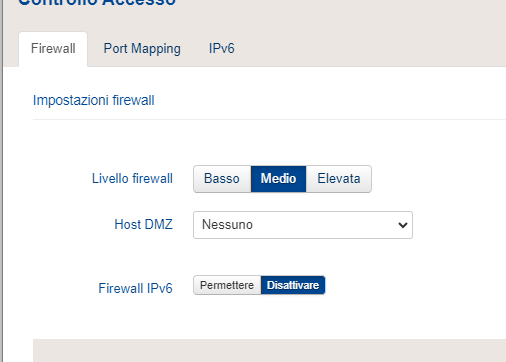
Poi vai su:



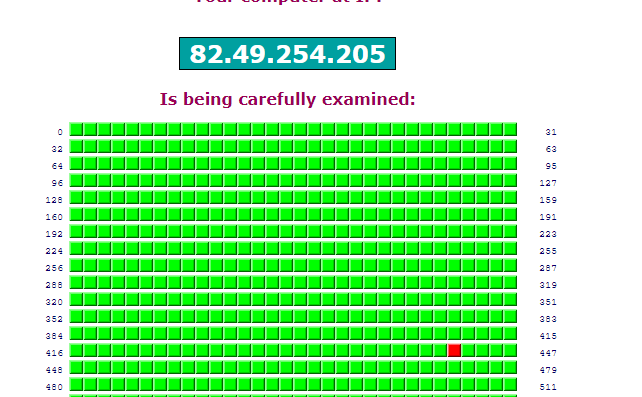
Crea una nuova regola o aggiorna quella esistente:



Oltre a questo disattiva il firewall ipv6:

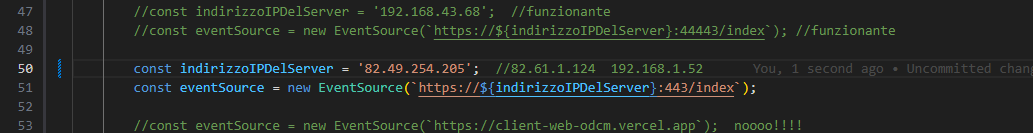


Ora rifare la scansione delle porte:



Ora andiamo a modificare la porta della CLIENT-WEB:

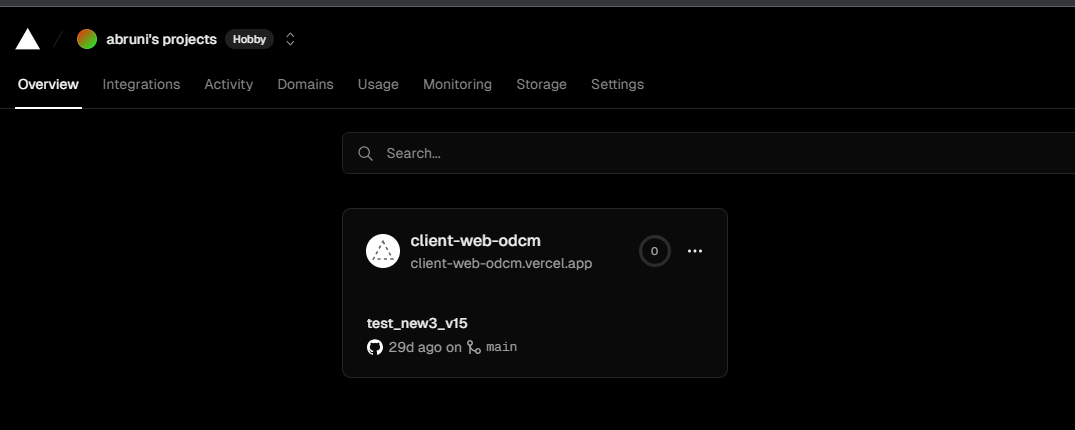
alla riga 50:



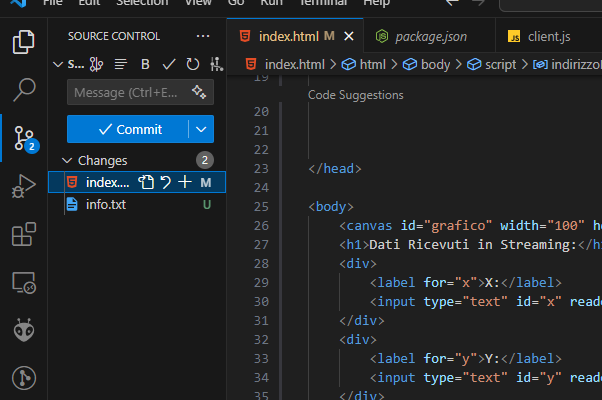
Fatto questo dobbiamo scaricare su VERCEL il programma modificato:

vai al seguente indirizzo: <https://vercel.com/>

una volta entrati siamo a questa pagina web:

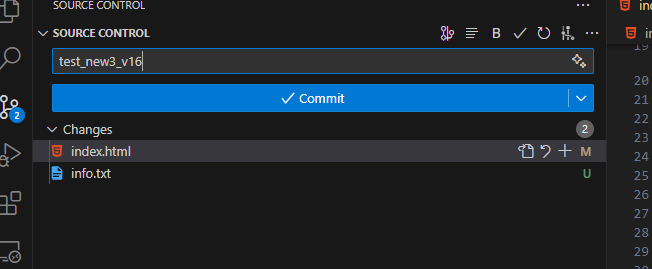


Ora andare sul progetto di CLIET-WEB su VSCode ed andare sulla repository:

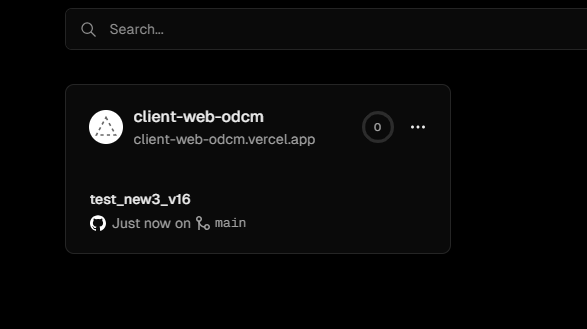


Scrivi sul messaggio un nuovo messaggio per ricordarsi la modifica.

Fai il commit ed il synk:



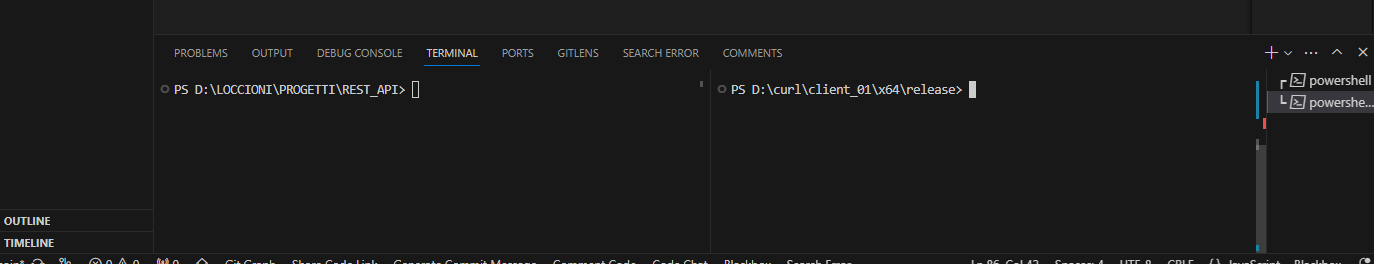
Ora sulla pagina di VERCEL troviamo aggiornato il repository:



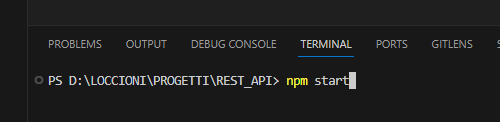
Ora avviare i programmi sul PC in locale:

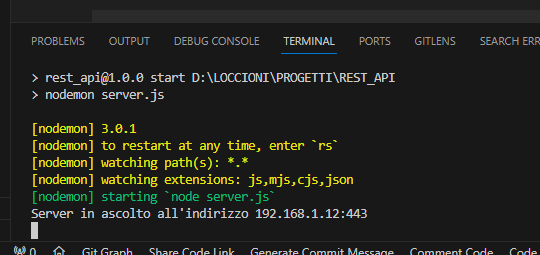
Andare su programma del SERVER:

aprire un secondo terminale:



Avviare il server:

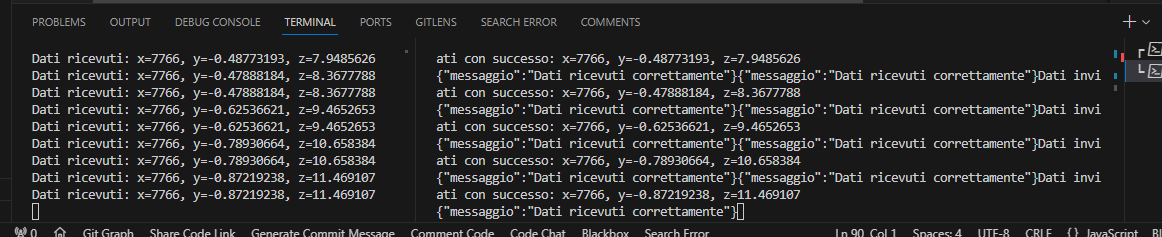




Avviare il programma che invia i dati:

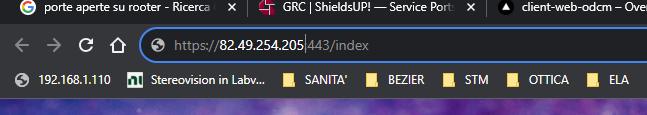


Il programma invia i dati ed il server li accetta e li manda all’esterno:

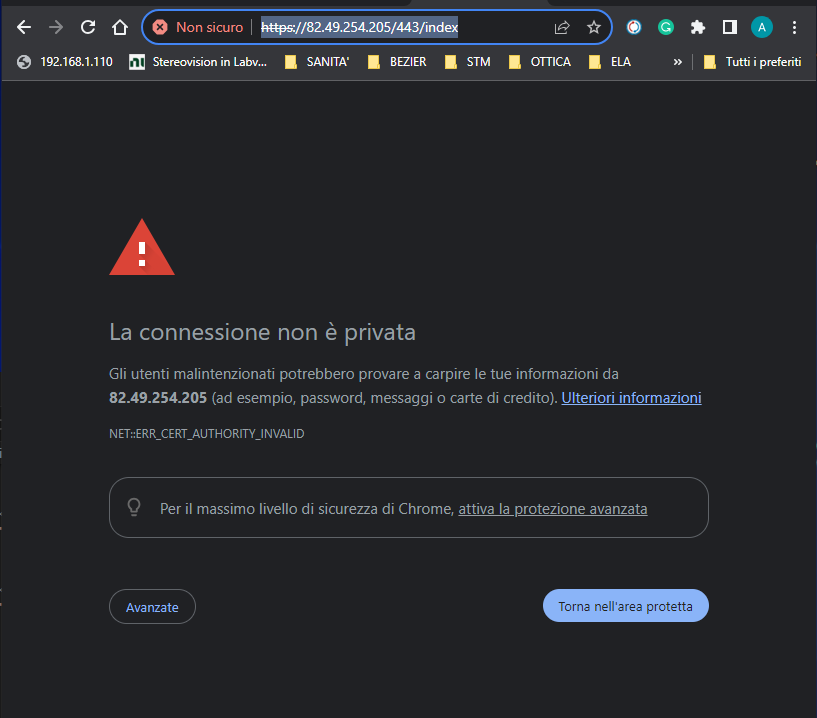


Ora vai su vercel.

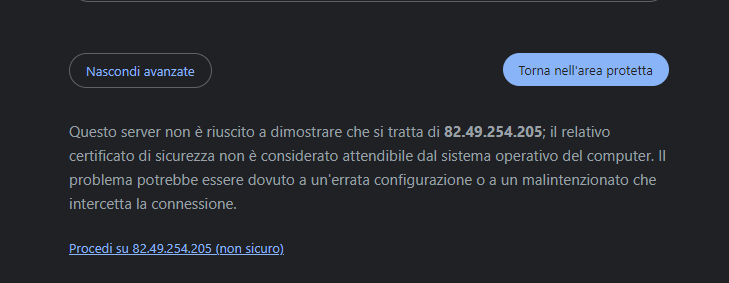
Per prima cosa apri la pagina web:



Compare la seguente schermata:

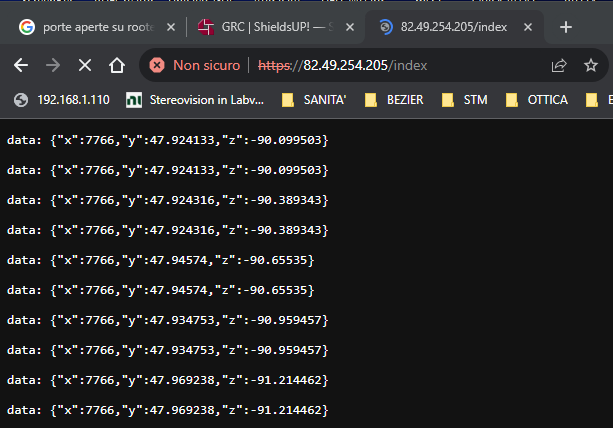


Vai su avanzate e procedi anche se non sicuro:

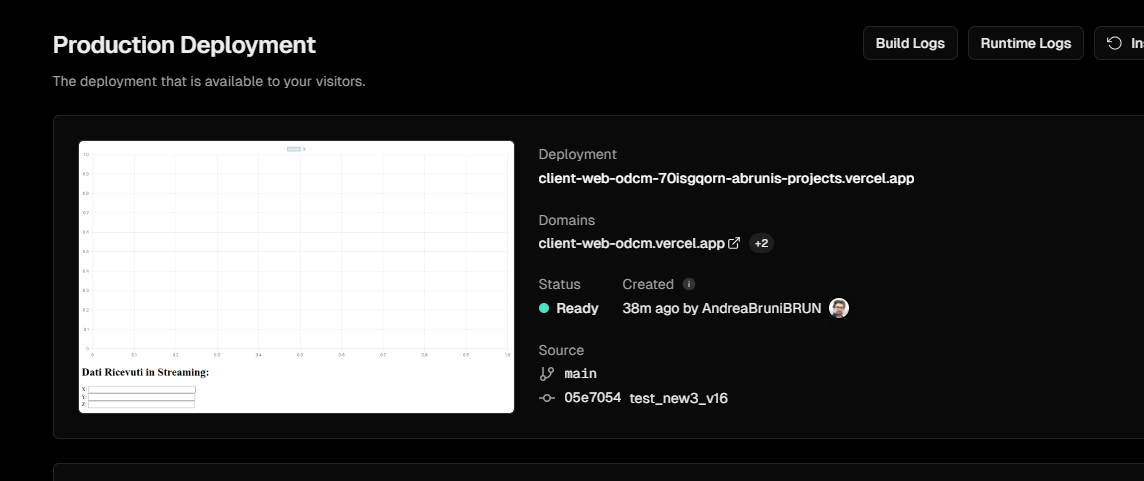


https://82.49.254.205:443/index

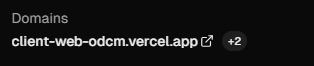
quindi otteniamo che la pagina risponde:



Ora andare su Vercel:

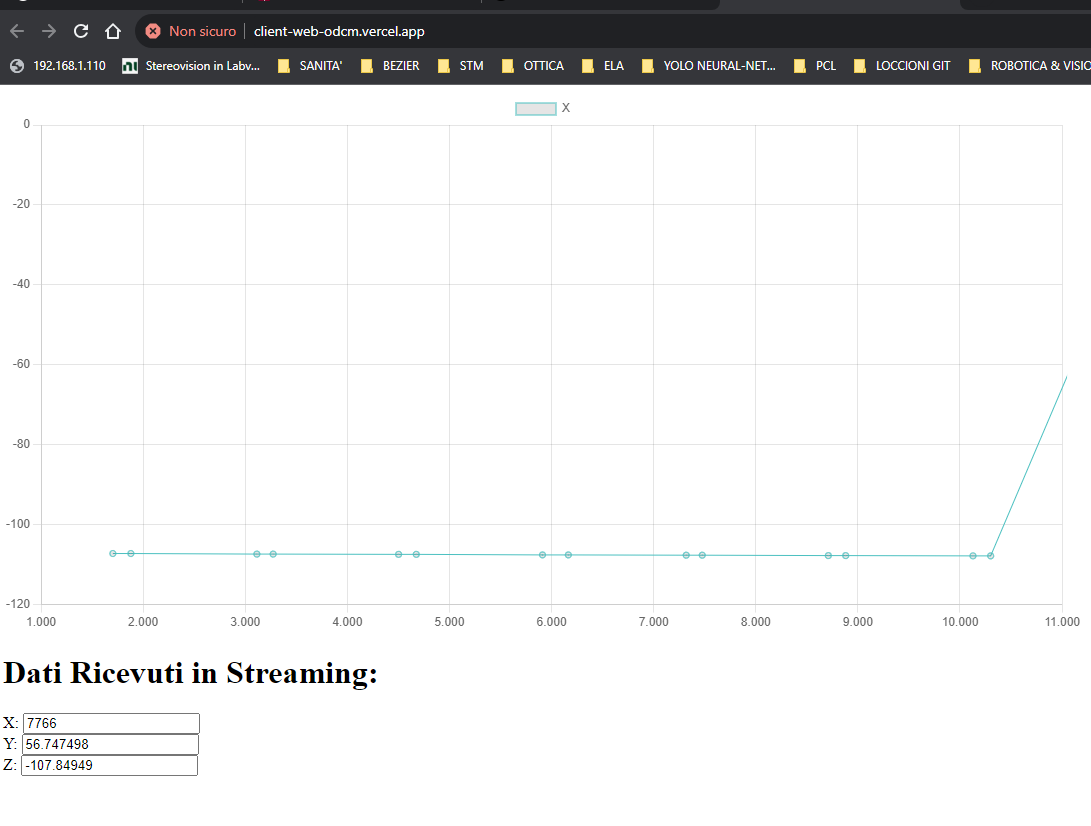


Aprire la pagina di Domains:



https://client-web-odcm.vercel.app/

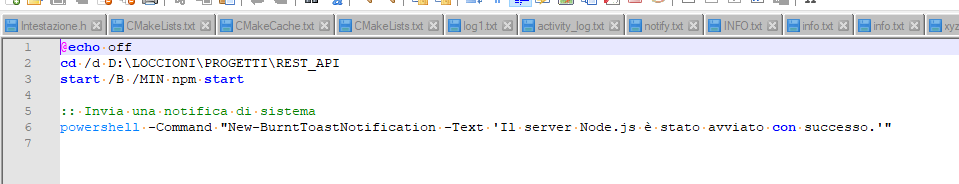
La pagina da in streaming i dati correttamente:

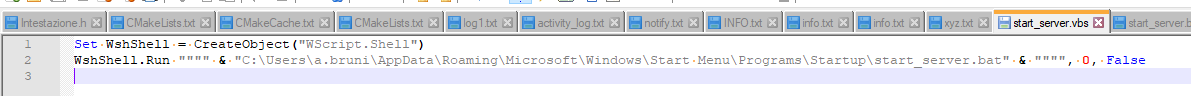


INSTALLAZIONE SU PC:

Se vogliamo far partire il servizio SERVER allo start up del PC si procede in questo modo:

1. Premi Windows + R per aprire la finestra di dialogo "Esegui".
2. Digita shell:startup e premi Invio. Si aprirà la cartella di avvio automatico di Windows.
3. Copia o sposta lo script batch start\_server.bat nella cartella di avvio automatico.



1. Crea un file VBS per avviare lo script batch: Puoi creare un file VBS (Visual Basic Script) per avviare lo script batch in background senza mostrare la finestra del prompt dei comandi. Ecco come puoi farlo:
2. Crea un nuovo file di testo e incolla il seguente codice:
3. 
4. Quando si avvia il PC esegue il file .VBS (possiamo spostare il file .bat da un'altra parte e non tenerlo su questa cartella di avvio).
5. Viene eseguito .vbs e questo chiama il file .bat.
6. Lanciare il programma : D:\CURL\CLIENT\_01\x64\Release\client\_01.exe e il programma non si blocca e viene eseguito correttamente.
7. Avviare una pagina web : <https://127.0.0.1:4443/index.html> (vediamo i risultati su un grafico).