**Il Video Streaming**

Il video streaming è basato sull’utilizzo di una tecnologia Open Source chiamata WebRTC (Web Real Time Comunication), una tecnologia che permette la comunicazione in tempo reale grazie all’utilizzo di un web browser.

Il Video Streaming è un unione di linguaggi quali: javascript(il principale) per le logiche di funzionamento, html per l’interfaccia e css per la grafica. La libreria della tecnologia WebRTC é scritta completamente in javascript.

Una buona guida che permette di capire come funziona javascript e come poterlo implementare è disponibile al seguente link:   
<https://www.html.it/guide/guida-javascript-di-base/>

Una buona guida che permette di capire come funziona tutta la tecnologia WebRTC e come farne un implementazione base è disponibile ai seguenti link:   
<https://webrtc.org/getting-started/overview>  
  
<https://codelabs.developers.google.com/codelabs/webrtc-web/#0>

Il video streaming al quale fa riferimento questo documento viene avviato su un server node js, per effettuare una gestione delle sessioni è stata utilizzata una socket. Una buona spiegazione di come poter gestire la socket con node js è disponibile al seguente link:   
<https://www.knowledgehut.com/tutorials/node-js/socket-services>

Il video streaming è sviluppato per essere contenuto all’interno del L4 della piattaforma SYENMAINT tramite l’ausilio di codice python. Una buona guida per apprendere il funzionamento del codice python è disponibile al seguente link:   
<https://www.html.it/guide/guida-python/>

**I File**

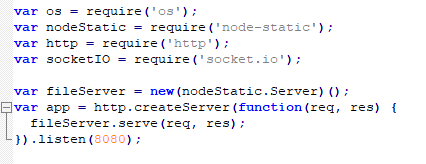
**Index.js e index per https.js**  
Il server node js è contenuto ed eseguito nei file “index.js” ed “index per https.js”

I due file sono simili nel funzionamento, l’unica differenza è che “index.js” crea un server http mentre “index per https.js” istanzia un server https, tutti e due i file vengono gestiti sulla porta 8080.

**Differenze tra server http ed https**

Il video streaming per poter funzionare ha bisogno dell’accesso alla video camera e al microfono del device sul quale deve operare.

**HTTP**



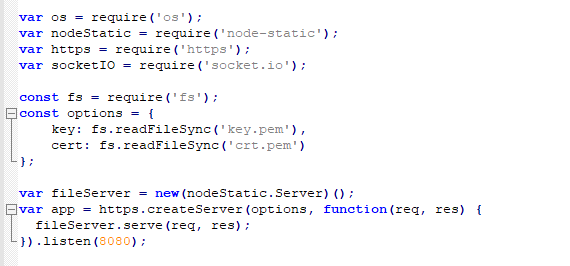
Nel momento in cui si prova ad accedere ad un indirizzo http il browser nega la possibilità di accesso alla video camera e al microfono a meno che l’indirizzo non venga aggiunto alle eccezioni.

Per aggiungere un indirizzo alle eccezioni con chrome bisogna

1. andare nella barra degli indirizzi e cercare chrome://flags/
2. individuare la sezione Insecure origins treated as secure
3. inserire nell’apposito campo gli indirizzi completi quindi <http://192.168.x.x:8080> o <http://indirizzopubblico:8080>
4. selezionare nel menu a tendina di fianco la voce enabled

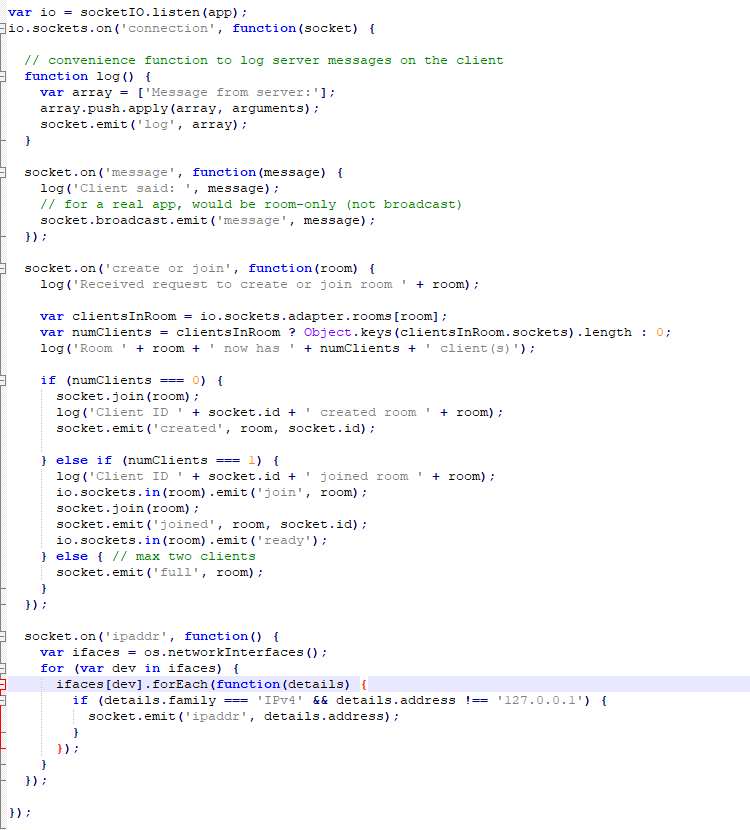


**HTTPS**



Per poter usufruire in maniera libera della video camera e del microfono il sito deve essere protetto quindi gestito tramite protocollo https. Il protocollo https per poter funzionare ha bisogno di una chiave e di un certificato, e come si puó notare dal codice è l’unica differenza tra i due index. Nella cartella certificates sono presenti un key.pem ed un crt.pem, una chiave ed un certificato che permettono il funzionamento di un indirizzo https ma sono presi gratuitamente da internet, i quali dovrebbero essere sostituiti con quelli rilasciati dagli enti specifici.

**Codice comune**



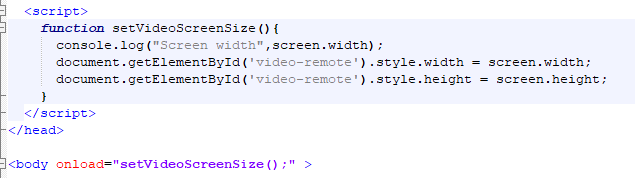
Nel codice vengono gestiti i vari stati della socket ovvero dall’accesso del primo utente, accesso del secondo utente ed infine viene privato l’accesso ad un terzo utente facendo un’indicazione di stanza piena

**Index.html**

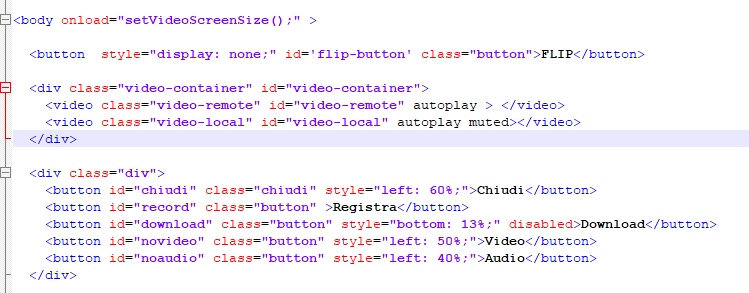
La UI dell’applicativo viene gestita all’interno del file index.html.



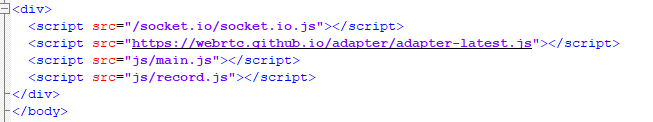
Codice:



All’apertura dell’indirizzo del server viene eseguita la funzione setVideoScreenSize() che si occupa di regolare l’immagine in base a quanto è grande lo schermo



In questo codice vengono creati i vari button ed il video sia locale che remoto.



In questo div vengono caricati tutti i file js che si occupano di gestire il backend.

È importante fare attenzione al src adapter-latest.js perché il file anche se inserito nella cartella js\lib con nome adapter.js si è preferito inserire il link github per far si che il file di definizione del applicativo WebRTC sia sempre quello piú aggiornato.

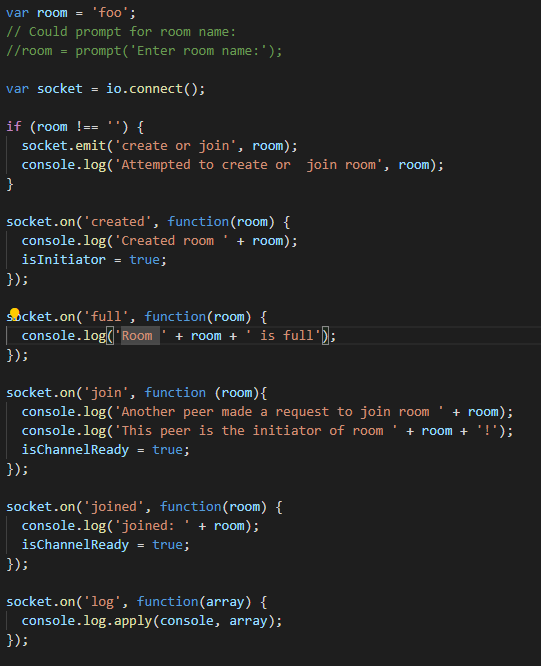
**Main.js**

Il cuore del funzionamento del video streaming è nel file main.js



Vengono definite le variabili che verranno poi utilizzate.

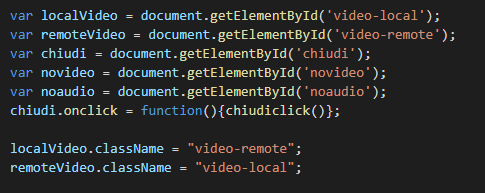
pcConfig contiene un indirizzo di un server turn gratuito quindi non efficientissimo del sito <http://numb.viagenie.ca/> . I dati username e credential sono funzionanti alla stesura del documento sono funzionanti, ma bisognerebbe sostituirli con un turn server piú efficiente. Il server turn si occupa di gestire i flussi di scambio dei media mentre lo stun server fornito da google che si occupa di gestire gli instradamenti dei pacchetti su indirizzi pubblici



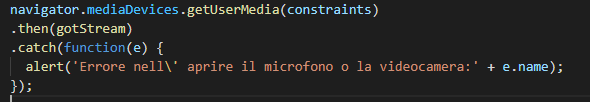
In questa sezione viene creata una stanza che puó ospitare due utenti richiamando le varie funzionalitá della socket



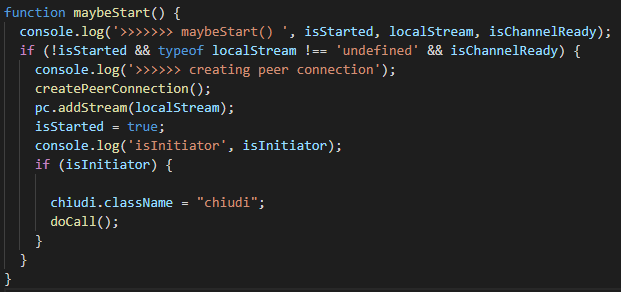
In questa sezione viene avviata l’interconnessione tra i due utenti, ovvero il primo utente che logga si mette in ascolto e il secondo chiede di connettersi al primo



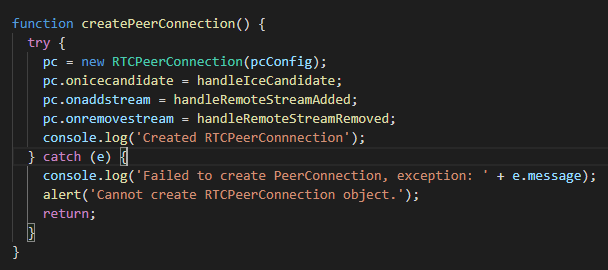
In qua parte di codice viene gestita la connessione tra il front end ed il back end richiamando i vari pulsanti e video tramite il loro id.



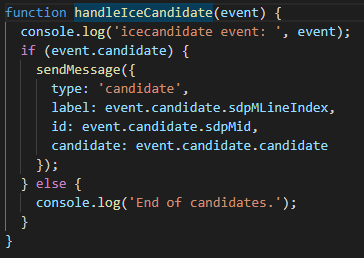
getUserMedia richiede l’accesso alla videocamera ed al microfono e se non ci sono errori inizia lo stream



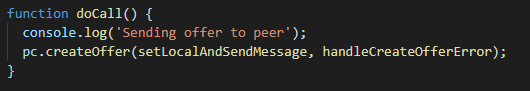
maybeStart è la funzione richiamata per avviare lo scambio del flusso dei dati con connessione peertopeer.



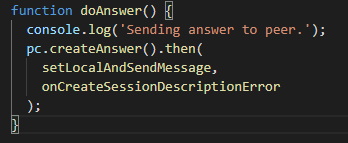
CreatePeerConnection è la funzione che crea e gestisce la connessione PeertoPeer tra i due utenti collegati alla stanza



handleIceCandidate è la funzione che definisce le informazioni del candidato/utente che vuole avviare il video streaming



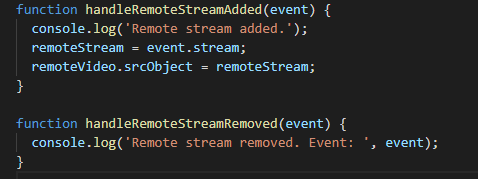
doCall è la funzione che abilita il candidato a ricevere la chiamata



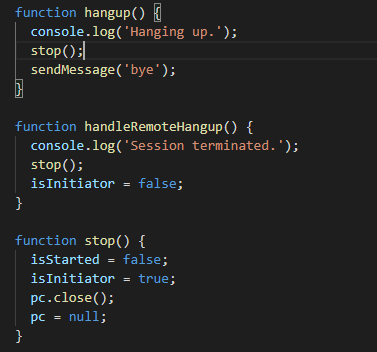
doAnswer è la funzione che fa richiedere l’accesso alla chiamata al richiedente



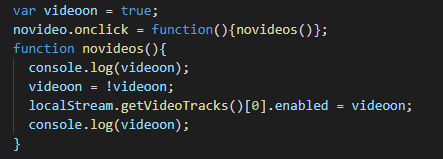
requestTurn è la funzione che avvia la connessione sul server turn inserito nelle costati



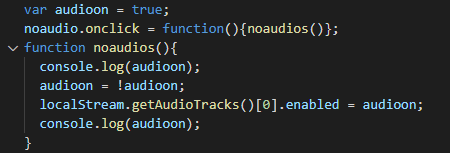
handleRemoteStreamAdded è la funzione che gestisce l’aggiunta a video della connessione della stream remota



Hungup, handleRemoteHangup sono due funzioni che chiudono la chiamata per tutti e due gli utenti e richiamano tutti e due la funzione stop che si occupa di chiudere la connessione



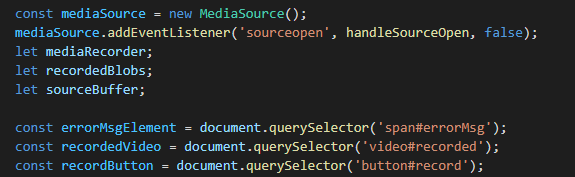
Novideos è la funzione che gestisce l’attivazione e la disattivazione della videocamera durante lo streaming.



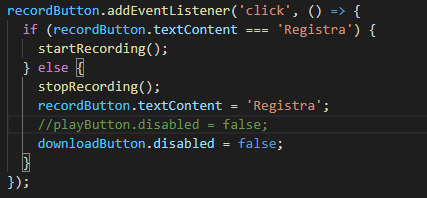
Noaudios è la funzione che gestisce l’attivazione e la disattivazione del microfono durante lo streaming.

**Record.js**

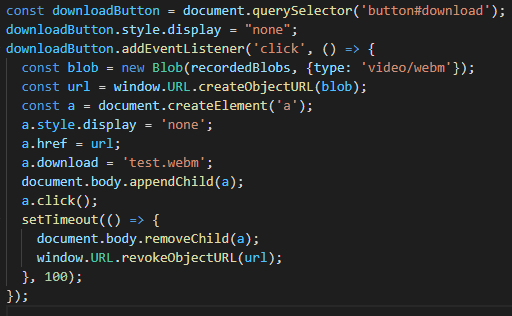
Record.js è il file nel quale è gestita tutta la parte di registrazione della stream remota.



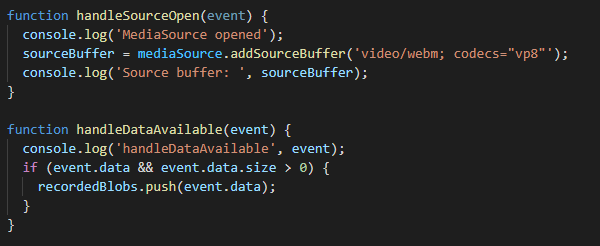
In questa parte di codice vengono settate le costanti ed effettuato il collegamento con il tasto Registra del presente sul front end



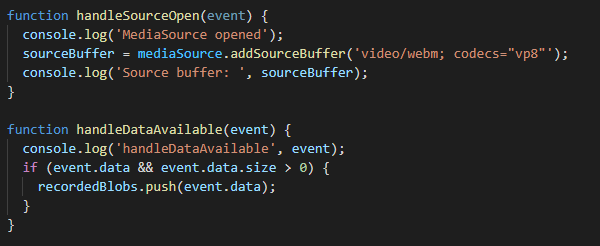
Questa sezione di codice assegna al tasto registra la funzionalitá che permette l’avvio della registrazione e lo stop con poi conseguentemente il download.



Questa sezione di codice assegna al tasto download la funzionalitá che permette il download del video solo dopo che la registrazione è stata conclusa.



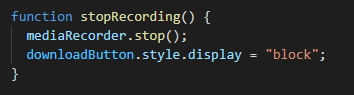
handleSouceOpen è la funzione che crea un buffer di memoria virtuale nel quale viene salvato il momentaneamente il video



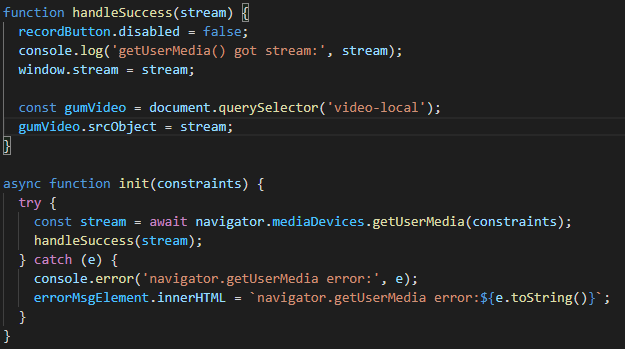
handleDataAvaiable verifica che la registrazione non è nulla o inesistente.



startRecording è la funzione collegata al tasto Registra che si occupa di registrare il video della stream remota fin quando non viene poi chiamata la funzione stopRecording tramite apposito pulsante



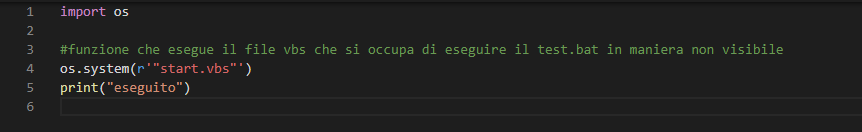
stopRecording è la funzione collegata al pulsante Fine Registrazione che ferma la registrazione e la prepara al download



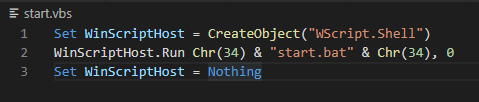
La funzione asincrona init verifica che i media devices siano ancora in funzione e handleSuccess che la connessione tra i due utenti sia ancora aperta.

**Start.py start.vbs start.bat**

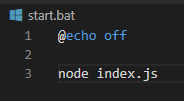
Con i file start.py, start.vbs, e start.bat viene gestito un processo automatico di avvio del server del live streaming.



Il file start.py si occupa di avviare il file start.vbs



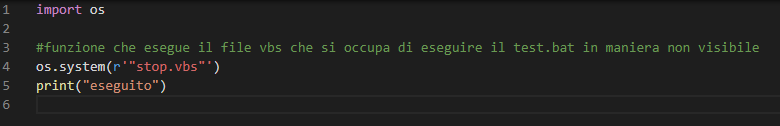
Il file start.vbs si occupa di avviare il file start.bat in un cmd invisibile



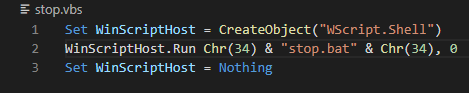
Il file start.bat si occupa di eseguire i comandi di sistema che avviano il server index.js del live streaming

**Stop.py stop.vbs stop.bat**

Con i file stop.py, stop.vbs, e stop.bat viene gestito un processo automatico di stop del server del live streaming avviato in precedenza.



Il file stop.py si occupa di avviare il file stop.vbs



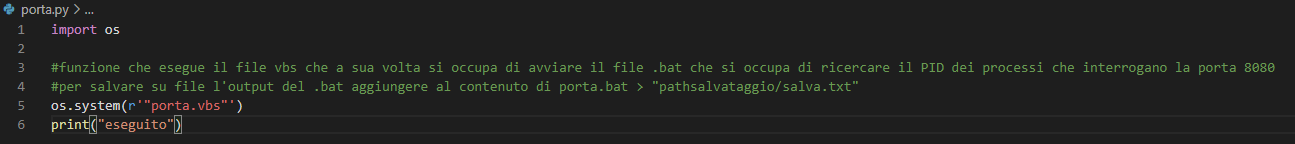
Il file stop.vbs si occupa di avviare il file stop.bat in un cmd invisibile



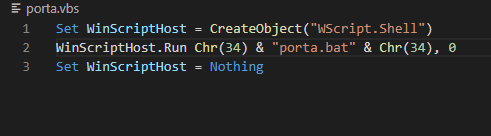
Il file stop.bat si occupa di eseguire i comandi di sistema che fanno una kill di tutti i processi di node js e quindi di index.js ovvero il file che mantiene il funzionamento del live streaming avviato in precedenza.

**Porta.py Porta.vbs Porta.bat**

Con i file posta.py, porta.vbs, porta.bat viene eseguito un processo automatico che individua tutti i processi con i relativi PID che occupano una porta, nel caso del video streaming la 8080



Il file porta.py si occupa di avviare il file porta.vbs



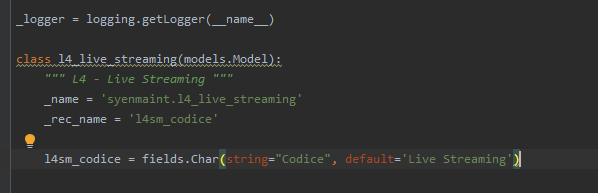
Il file porta.vbs si occupa di avviare il file porta.bat in un cmd invisibile



Il file porta.bat si occupa di eseguire i comandi di sistema che analizzano quali sono i processi che interrogano la porta 8080, in questo caso specifico salvando l’output sul file salva.txt

**L4\_live\_streaming.py**

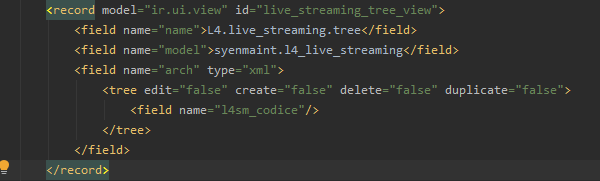
Il file l4\_live\_streaming.py presente nella cartella nel file syenmaint/models/l4\_live\_streaming.py si occupa del back end del video streaming nell’L4



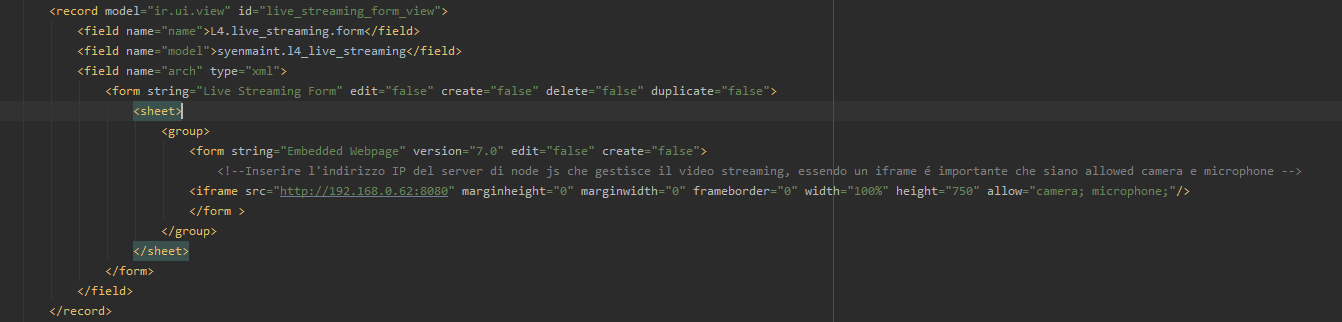
Con questo codice viene definito un modello di assegnazione di risorse sull’L4 al video streaming, assegnando alla variabile l4sm\_codice la stringa di default “Live Streaming”

**L4\_live\_streaming.xml**

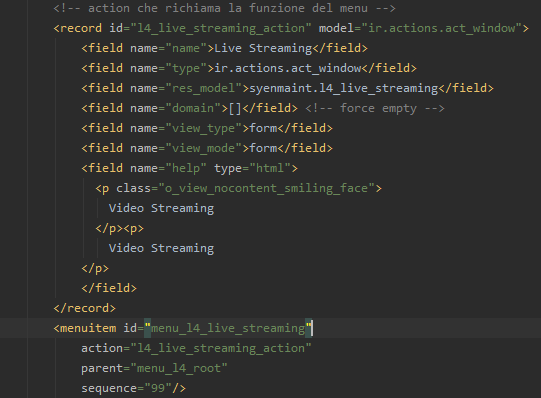
Il file l4\_live\_streaming.xml presente nella cartella syenmaint/views/l4\_live\_streaming.xml si occupa del front end della visualizzazione del video streaming nell’L4



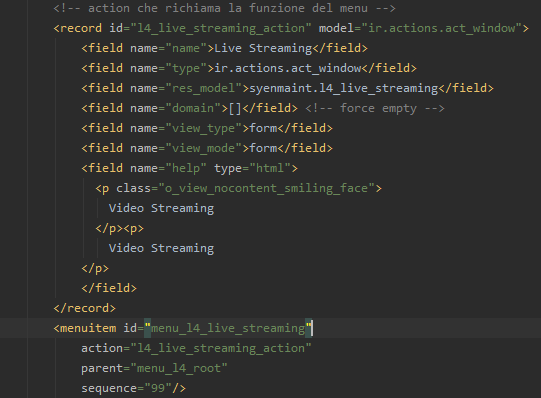
In questa sezione viene creata una tree view con solo il campo l4sm\_codice con impostati a false i valori di edit, create, delete, duplicate.



In questa sezione viene creata la form view in maniera standard con modello di riferimento syenmaint.l4\_live\_streaming. Nella form è presente solo il campo iframe che richiama l’indirizzo pubblico o privato del server di video streaming dove nelle opzioni “allow” viene dato potere di accesso alla videocamera ed al microfono. Anche in questo caso vengono impedite all’utente le possibilitá di edit, creazione, eliminazione e duplicazione in modo da non far visualizzare i pulsanti base dell’ERP di Modifica, creazione, eliminazione e duplicazione.



In questa sezione di codice viene richiamato il tag <menuitem> che permette la visualizzazione della voce Live Streaming all’interno del menu delle sezioni.



In questo codice è presente l’action che viene chiamata alla selezione del pulsante creato dal tag <menuitem> e richiama la visualizzazione, definita nel tag <field name=”view\_mode”>della sola form

**Ir.model.access.csv**

Il file ir.model.access.csv è presente nella cartella syenmaint/security/ir.model.access.csv e si occupa di definire le proprietá di accesso ai singoli modelli

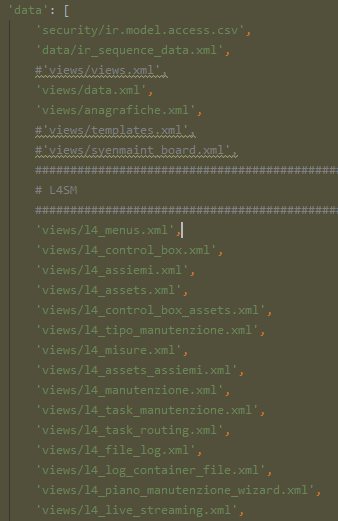
id,name,model\_id:id,group\_id:id,perm\_read,perm\_write,perm\_create,perm\_unlink

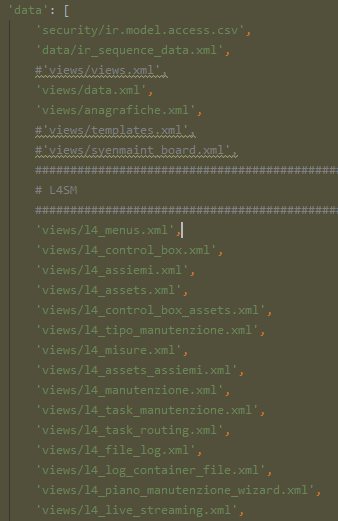


In questa sezione di codice viene assegnato all’utente la sola proprietà di lettura.

**\_\_manifest\_\_.py**

Il file \_\_manifest\_\_.py è presente nella cartella syenmaint/\_\_manifest\_\_.py e si occupa di rendere accessibili i file del front end al sistema.





In questa sezione di codice vengono assegnati alla lista ‘data’ i path di ir.model.access.csv e di l4\_live\_streaming.xml

**\_\_init\_\_.py**

Il file \_\_init\_\_.py è presente nella cartella syenmaint/models/\_\_init\_\_.py e si occupa di rendere accessibili i file del back end al sistema



In questa sezione di codice assegnato al sistema il file l4\_live\_streaming.py

Il risultato finale è:

