

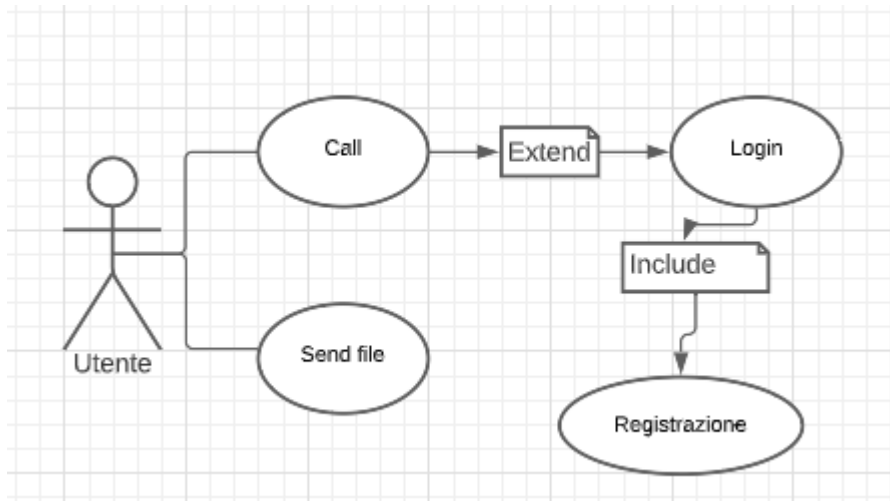
Relazione Tecnologie Internet

P2P VideoChat

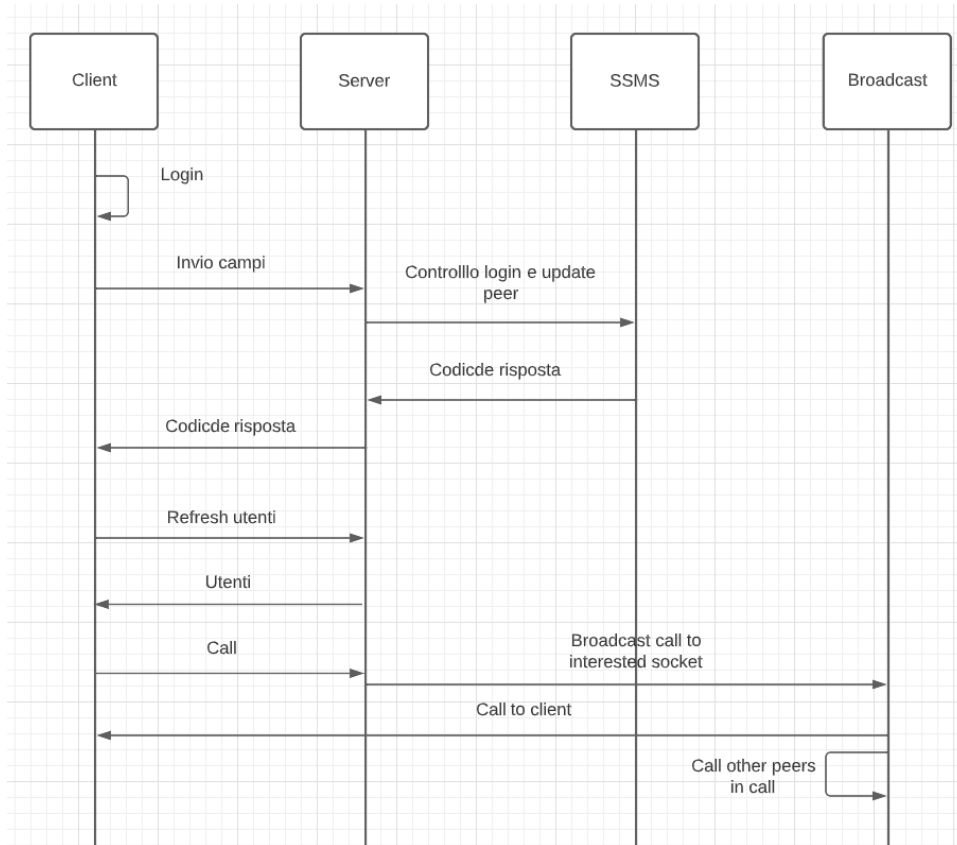
Per lo sviluppo del progetto sono state utilizzate le seguenti tecnologie:

- Node.js : ambiente di runtime per eseguire file js indipendentemente da un browser, utile per lo sviluppo di applicazioni web che integrano javascript
- Express.js : framework di Node.js per la creazione di server
- Socket.io : libreria per le comunicazioni C/S in real time
- PeerJS (WebRTC) : tecnologia per semplificare comunicazioni WebRTC tra C/S
- SQLServer : ambiente per lo sviluppo del db per l'autenticazione
- Tedious : libreria per la comunicazione con il DB SQL

Flusso Utilizzo Utente



Comunicazioni chiamate



Installazione ed esecuzione:

Per eseguire il programma occorre installare Node.js e le librerie di PerJs, Express, Tedious

Dopo aver clonato la cartella di github, per eseguire il programma bisogna:

Aprire un istanza di sql server e creare il DB con tutte le sue funzionalità, eseguendo il file DBALL.tsq. Successivamente bisognerà modificare inserendo i propri dati di accesso a SQLSERVER nella pagina /public/web_server.js dove avviene la connessione al db, bisognerà modificare i campi "userName" e "password" e "database"

```

authentication: {
  options: {
    userName: "", //insert here your sql username
    password: "" //insert here your sql password
  },
  type: "default"
},
server: "localhost",
options: {
  database: "", //insert here your sql db name
  encrypt: true,
  trustServerCertificate: true //importante
}

```

Successivamente per eseguire l'applicazione tramite IDE di Node.js o usando Visual Studio Code, aprire il terminale nella cartella del progetto ed eseguire prima il comando "node /server/peerserver.js" per creare un'istanza di un server PeerJS e successivamente "node /public/web_server.js" per eseguire il server della nostra applicazione.

Una volta eseguiti i file aprire un browser di nostra preferenza e navigando all'URL localhost:8080 dove il nostro server è stato eseguito e troveremo la nostra pagina index.html


Name

Password

Choose a peer:

Put a file: Nessun file selezionato

Our Video



Successivamente si potrà creare un profilo ed accedervi per comparire nella lista degli utenti online, cliccando su refresh si aggiornerà la lista degli utenti online e selezionando quello desiderato e cliccando "Call Selected" si chiamerà l'utente, si avvierà così la chiamata tra i 2 utenti che visualizzeranno il proprio video e quello dell'altro. Selezionando un altro utente dalla lista e chiamandolo, questo verrà unito alla chiamata e il suo video verrà aggiunto alle pagine di tutti gli altri utenti.

Non ho capito se i servizi RESTFUL fossero obbligatori o meno visto che in questo caso l'utilizzo di socket risulta molto più efficiente, ne spiego qualche motivazione :

1. Ci basiamo su indirizzi IP e Porte per la comunicazione tra i vari peer questo richiede le socket
2. Le socket sono un protocollo STATEFUL a differenza dei servizi RESTFUL
3. Necessitiamo di un'applicazione in Real Time che consiglia l'uso di socket
4. Abbiamo poche richieste di risorse se non nulle
5. Con l'utilizzo di WebSocket la comunicazione rimane più leggera e più stabile per grossi utilizzi