

TOR VERGATA
UNIVERSITÀ DEGLI STUDI DI ROMA

Studio e sperimentazione della tecnologia WebRTC

Relatore: Pietro Frasca

Candidato: Orlando Parente

Anno Accademico 2020/2021

Content

- Introduzione alla tecnologia webRTC
- Introduzione all'applicativo “Local Peer RTC”
- Videochiamata con “Local Peer RTC”
- Fase di Signaling: Server PHP
- API webRTC in “Local Peer RTC”
- Conclusioni

Introduzione alla tecnologia webRTC (1)

- È un progetto open source
- Fornisce supporto per lo sviluppo di applicativi browser con capacità di *Real Time Comunication (RTC)*
- È uno standard



Introduzione alla tecnologia webRTC (2)

- A e B vogliono comunicare
- A genera la *SDP Offer* che invia (nella *fase di signaling*) a B
- B genera la *SDP Answer* che invia (nella *fase di signaling*) ad A
- A genera una connessione il più efficiente possibile dalle informazioni collezionate

Introduzione applicativo “Local Peer RTC”

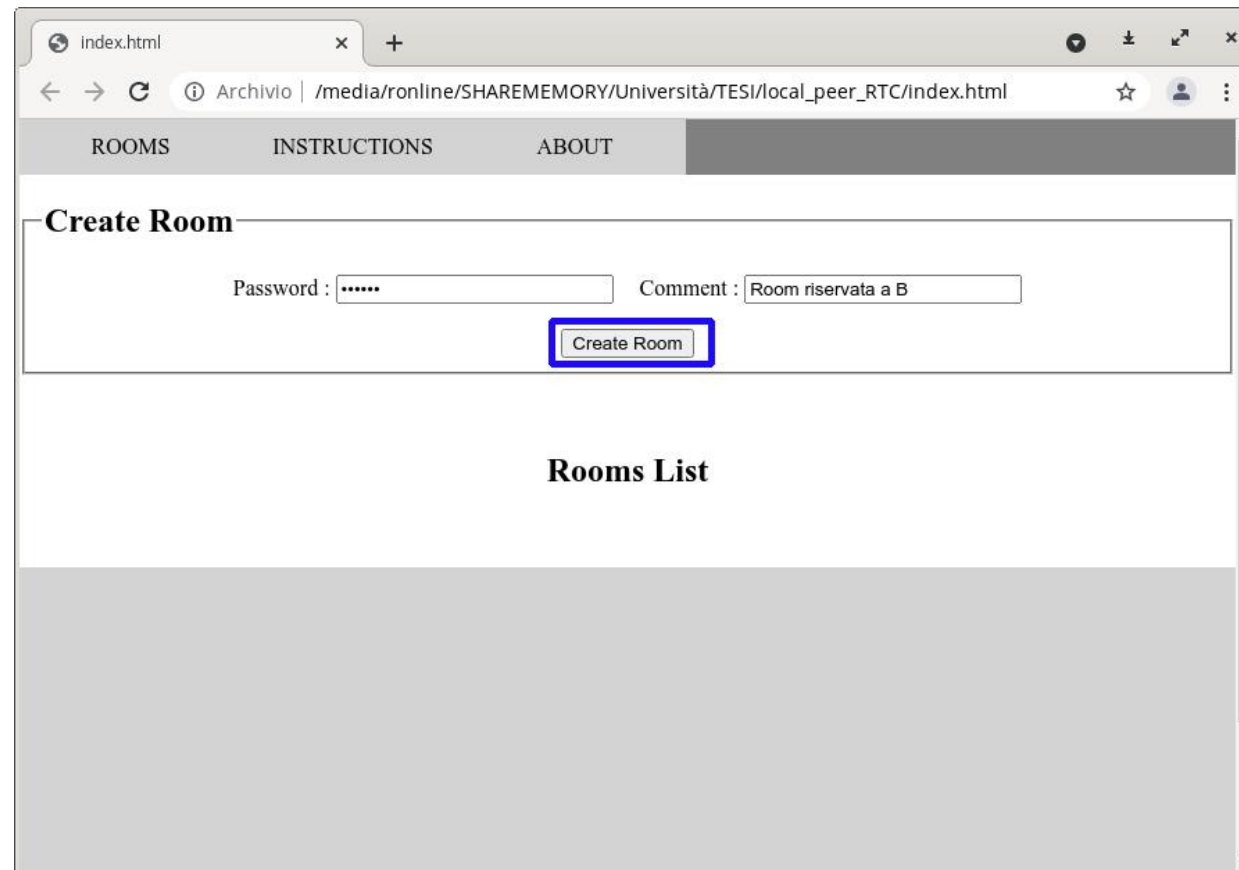
- Sviluppato con le API standard webRTC
- Permette di effettuare una videochiamata tra *due* host
- Permette lo scambio di messaggi

Videochiamata con “Local Peer RTC” (1)

- A vuole effettuare una videochiamata con B

Videochiamata con “Local Peer RTC” (2)

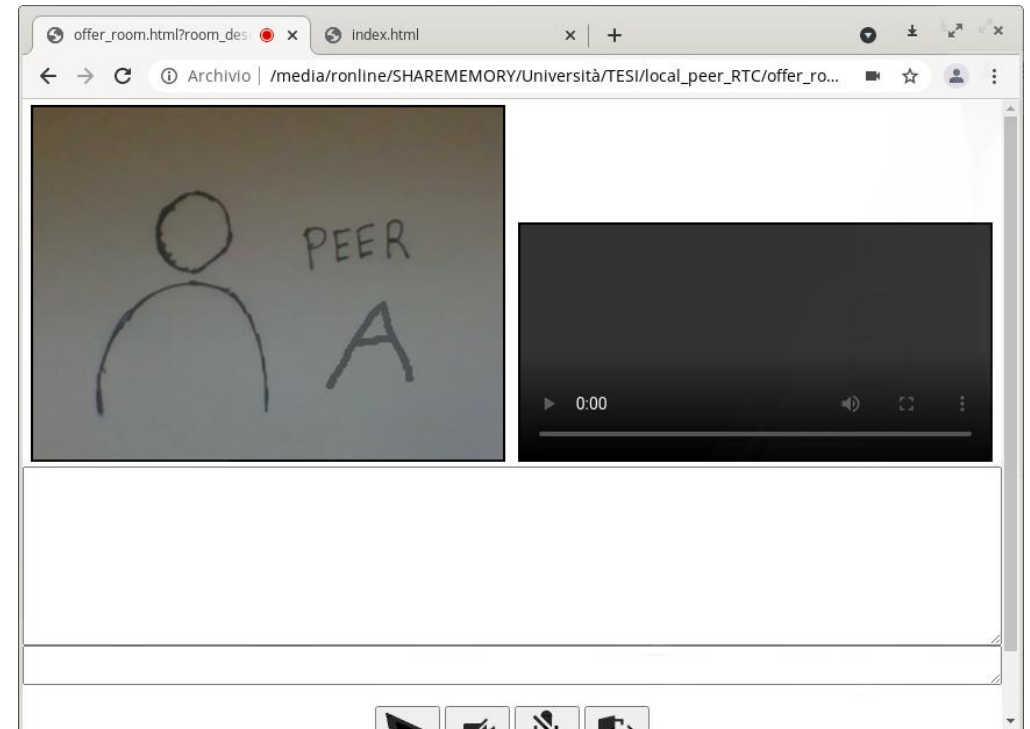
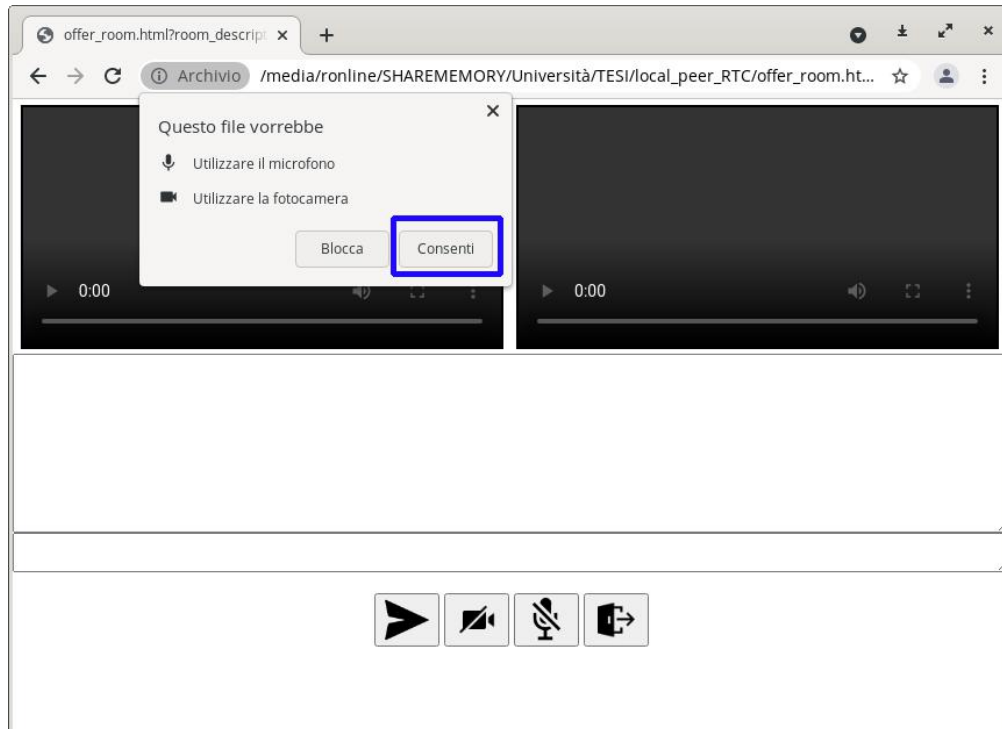
- PASSO 1: A genera una “room”



The screenshot shows a web browser window with the address bar displaying the URL: `/media/ronline/SHAREMEMORY/Università/TESI/local_peer_RTC/index.html`. The page has a navigation bar with three tabs: **ROOMS**, **INSTRUCTIONS**, and **ABOUT**. The **ROOMS** tab is currently selected. Below the navigation bar, there is a section titled **Create Room**. This section contains two input fields: a **Password** field with masked characters (dots) and a **Comment** field with the text "Room riservata a B". Below these fields is a **Create Room** button, which is highlighted with a blue rectangular border. Below the **Create Room** section, there is a section titled **Rooms List**, which is currently empty and has a light gray background.

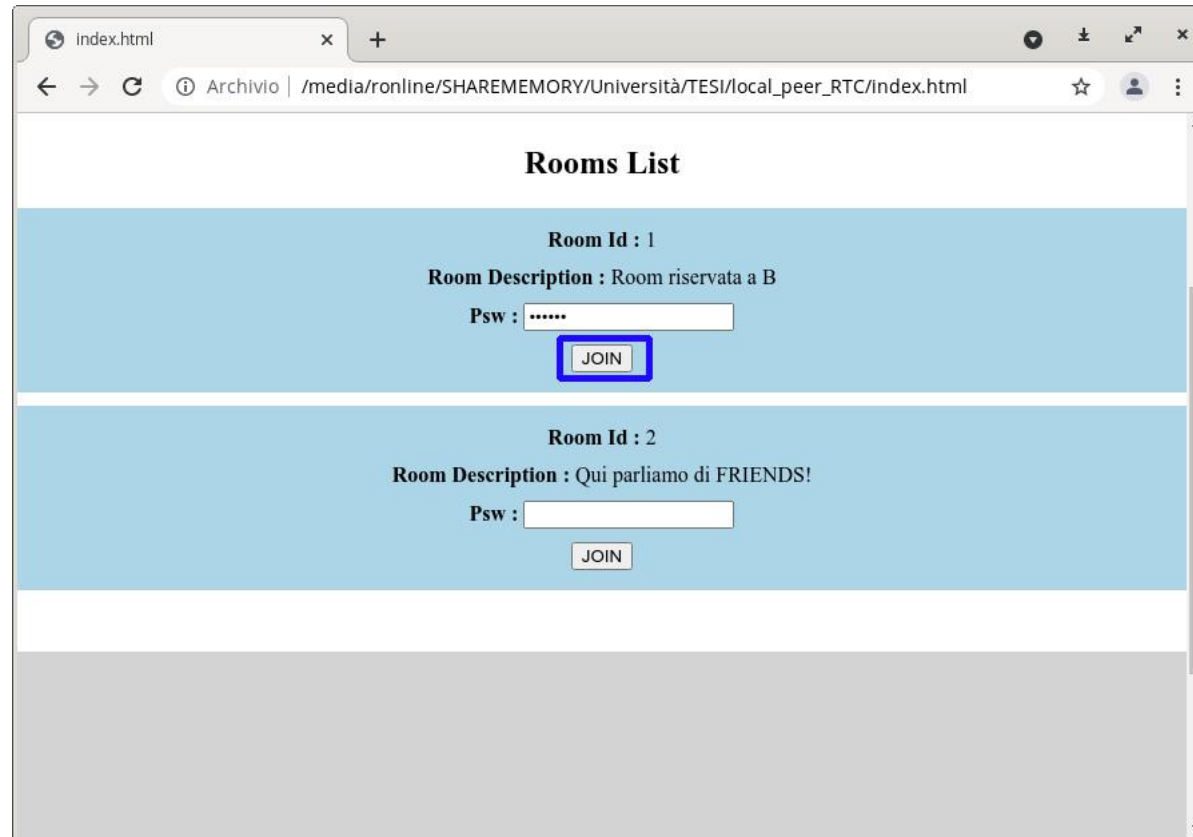
Videochiamata con “Local Peer RTC”

- PASSO 2: *getUserMedia()* richiede il permesso per catturare lo stream della telecamera e del microfono.
- Se i permessi vengono negati, viene lanciato il **PermissionDeniedError** che l'applicativo gestisce reindirizzando l'utente alla pagina principale
- Se vengono concessi, la “room” viene effettivamente creata



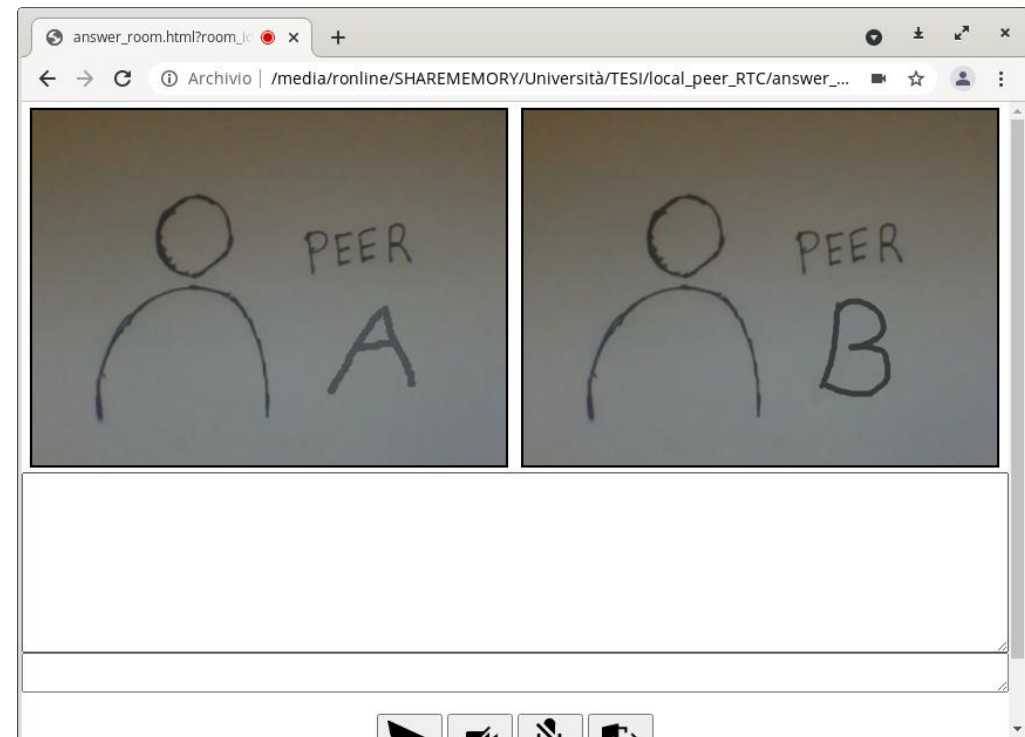
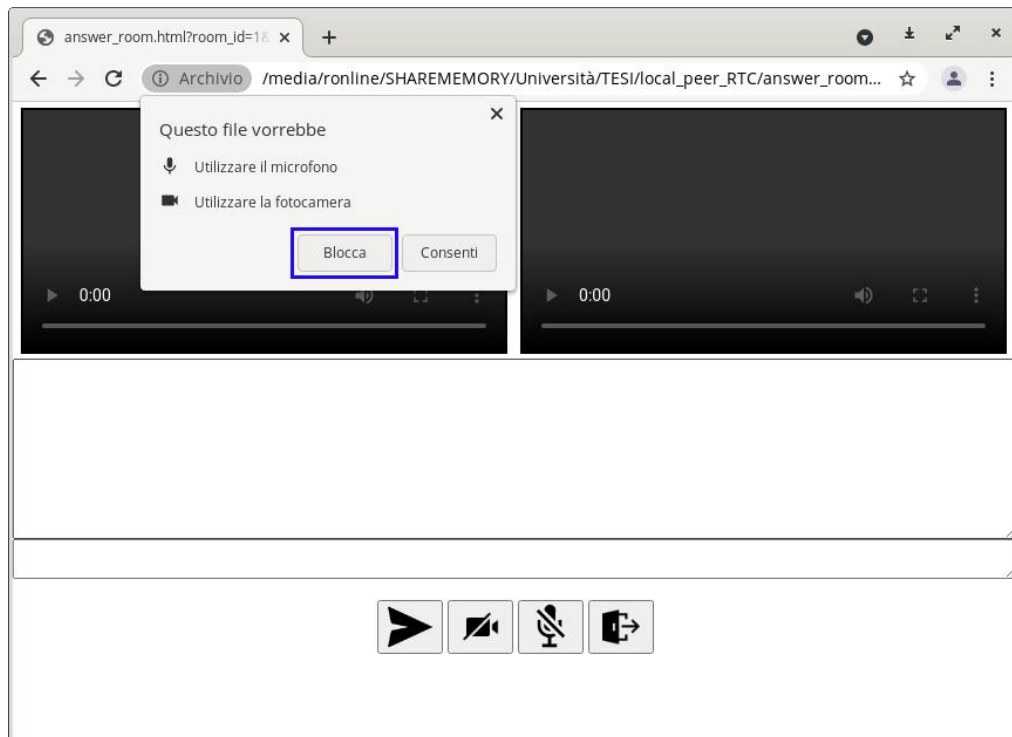
Videochiamata con “Local Peer RTC” (3)

- PASSO 3: B cerca, fra le “room” disponibili quella creata da A e cliccla sul bottone “join” corrispondente



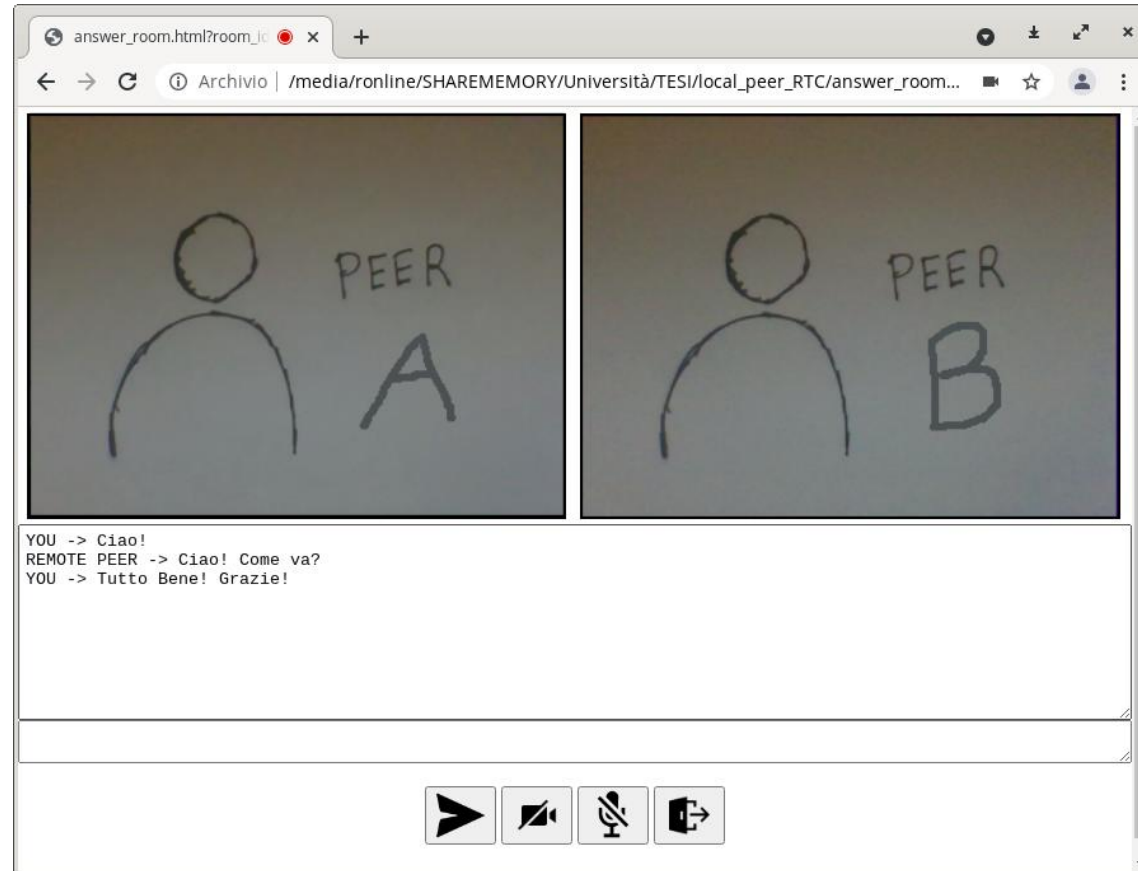
Videochiamata con “Local Peer RTC” (4)

- PASSO 4: Anche B deve concedere a *getUserMedia()* il permesso di accedere allo stream dei dispositivi.



Videochiamata con “Local Peer RTC” (5)

- A e B sono in videochiamata, possono scambiarsi messaggi mutare video e/o audio oppure uscire dalla “room” con il tasto “quit”



Fase di Signaling: Server PHP (1)

- La fase di Signaling è la fase in cui i due host devono scambiarsi la *SDP Offer* e la *SDP Answer*
- Non viene specificato (né consigliato) un modo per implementare questa fase: il “come” è lasciato decidere allo sviluppatore
- Nell'applicativo “Local Peer RTC” si è deciso di utilizzare un server PHP che salva la *SDP Offer* e la *SDP Answer* in un oggetto json, in modo che qualsiasi host possa accedervi.

API webRTC in “Local Peer RTC”

- *RTCPeerConnection* per instaurare e gestire la connessione
- *MediaStream (getUserMedia)* per catturare lo stream di microfono e telecamera
- *DataChannel* per l’invio e la ricezione dei messaggi

Conclusioni

- API standard e semplici da usare
- Interoperabilità (diversi browser possono effettuare *una real time* communication tra di loro)
- Connessione più efficiente possibile