How to Use Your Phone as a PC Webcam

A Complete Setup Guide

Ronak Hingonia

February 21, 2025

Contents

1 Introduction

This guide demonstrates how to transform a smartphone into a webcam for a PC using WebRTC. The phone captures its camera feed via getUserMedia and streams it in real time, while a lightweight Node.js signaling server handles the exchange of connection details. On the PC side, a corresponding interface receives and displays the video stream, enabling a seamless, hardware-free webcam solution

2 Prerequisites

- **Hardware Requirements**
  - Smartphone (Android/iOS)
  - PC or laptop
  - USB cable (optional for wired connection)
  - Stable Wi-Fi connection
- **Software Requirements**
  - A browser installed on the phone (Safari, Chrome, etc.)
  - Node.js installed on PC (16 or higher)
  - Python installed on PC (3.x.x or higher)
  - OBS Studio installed on PC
  - Operating system requirements

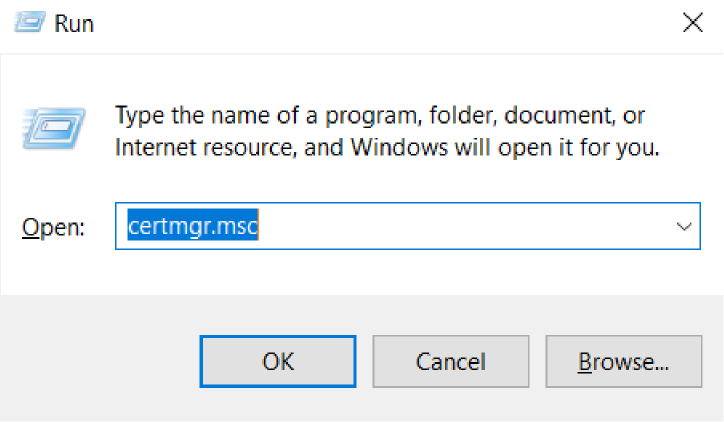
3 Setup Process

3.1 Creating and Installing Root CA

1. After cloning the project locally on your PC, open a terminal in it.
2. Run the create_ca.py file via command

```
$ python create_ca.py
```

3. `rootCA.crt` and `rootCA.key` files should appear in the directory
4. **Installing the CA in PC**
   - Open Run (Win + R)
   - Type `certmgr.msc` → Hit Enter.

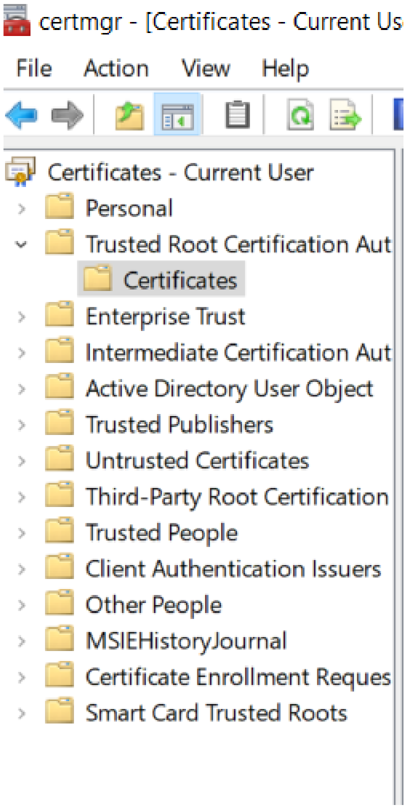


Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: certmgr.msc

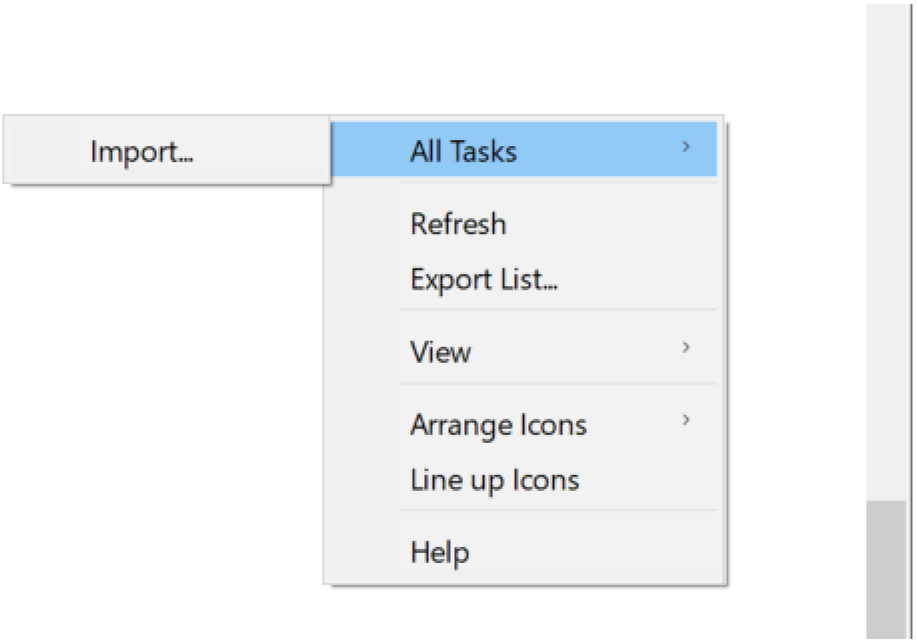
-

- Navigate to Trusted Root Certification Authorities → Certificates.

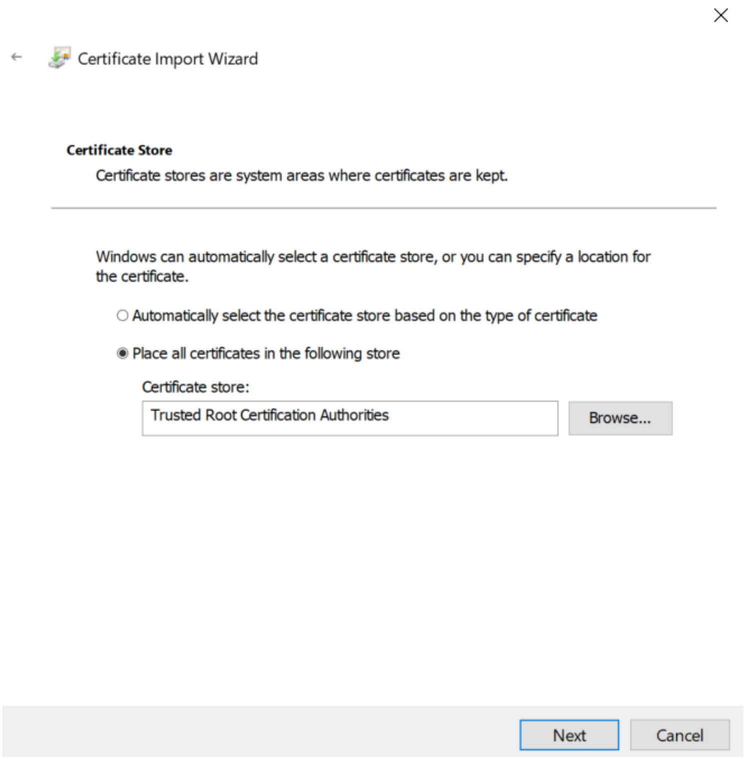

certmgr - [Certificates - Current Us
File
Action
View
Help
Certificates - Current User
Personal
Trusted Root Certification Aut
Certificates
Enterprise Trust
Intermediate Certification Aut
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification
Trusted People
Client Authentication Issuers
Other People
MSIEHistoryJournal
Certificate Enrollment Reques
Smart Card Trusted Roots

-

- Right-click → All Tasks → Import.



-
- Hit Next
- Browse the file → Hit Next
- Stay on Place all Certificates in the following store with the following store pre-written.



Certificate Import Wizard

Certificate Store

Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

○ Automatically select the certificate store based on the type of certificate

● Place all certificates in the following store

Certificate store: Trusted Root Certification Authorities

-
- Next → Complete the installation and agree to it’s conditions.

Why have we done this?

Web browsers and devices only trust websites that have certificates issued by recognized authorities (like Let's Encrypt). Since we are creating our own self-signed certificates, browsers and devices don't trust them by default.

To fix this, we create our own Certificate Authority (CA). This CA acts like a personal verification system that tells devices, *"Hey, any certificate signed by me is safe!"*

Why this helps?

Once the Root CA certificate is installed on a device, any future certificates signed with this CA will be automatically trusted. This means you won't have to install new certificates every time you generate one.

This setup allows a secure connection between your phone and PC without manual certificate installation each time :)

-However, if you change your mobile device or PC device (or both), you'll have to install the Root CA certificate again. This is because the new device doesn't recognize the previous Root CA, and you need to set it up again for trusted connections.

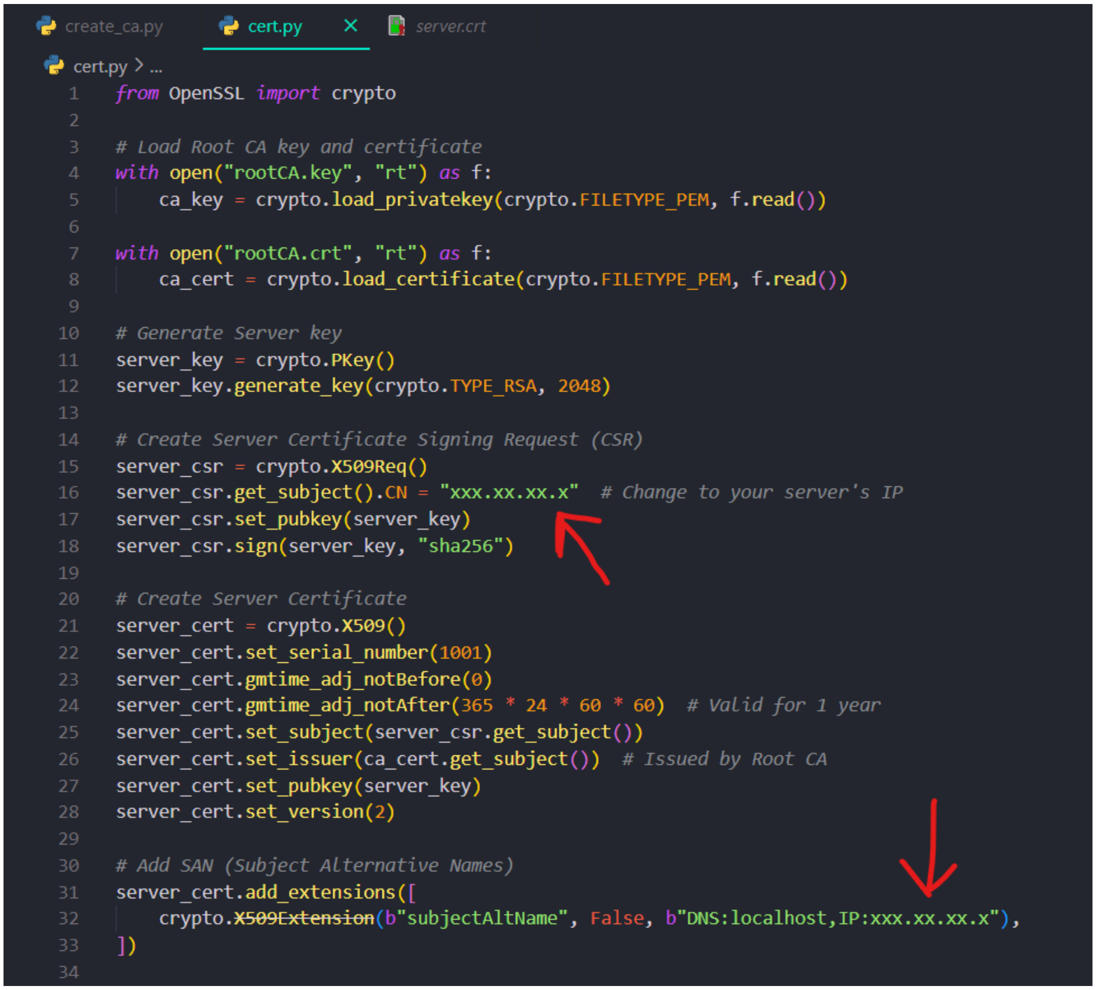
5. **Install the same certificate on your mobile device.** *You can use mail to send the file.*

3.2 Certificate Generation and Configuration

1. Connect your mobile device to your computer using a USB cable, ensuring a stable connection is established.
2. Enable USB debugging on your Android device:
   - Go to Settings → About Phone
   - Tap "Build Number" seven times to enable Developer Options
   - Navigate to Settings → Developer Options
   - Enable "USB Debugging"
3. If iOS, simply turn on personal hotspot including USB connection i.e. USB threatening
4. Obtain the IP address for the USB connection:
   - For Windows: Open Command Prompt and execute `ipconfig`
   - For macOS/Linux: Open Terminal and execute `ifconfig`

- Locate and copy the IPv4 Address listed under ”Ethernet Adapter” or ”USB Connection”

5. Configure the certificate generation script as follows:



```python
from OpenSSL import crypto

# Load Root CA key and certificate
with open("rootCA.key", "rt") as f:
    ca_key = crypto.load_privatekey(crypto.FILETYPE_PEM, f.read())

with open("rootCA.crt", "rt") as f:
    ca_cert = crypto.load_certificate(crypto.FILETYPE_PEM, f.read())

# Generate Server key
server_key = crypto.PKey()
server_key.generate_key(crypto.TYPE_RSA, 2048)

# Create Server Certificate Signing Request (CSR)
server_csr = crypto.X509Req()
server_csr.get_subject().CN = "xxx.xx.xx.x"  # Change to your server's IP
server_csr.set_pubkey(server_key)
server_csr.sign(server_key, "sha256")

# Create Server Certificate
server_cert = crypto.X509()
server_cert.set_serial_number(1001)
server_cert.gmtime_adj_notBefore(0)
server_cert.gmtime_adj_notAfter(365 * 24 * 60 * 60)  # Valid for 1 year
server_cert.set_subject(server_csr.get_subject())
server_cert.set_issuer(ca_cert.get_subject())  # Issued by Root CA
server_cert.set_pubkey(server_key)
server_cert.set_version(2)

# Add SAN (Subject Alternative Names)
server_cert.add_extensions([
    crypto.X509Extension(b"subjectAltName", False, b"DNS:localhost,IP:xxx.xx.xx.x"),
])

```

6. Verify certificate generation:

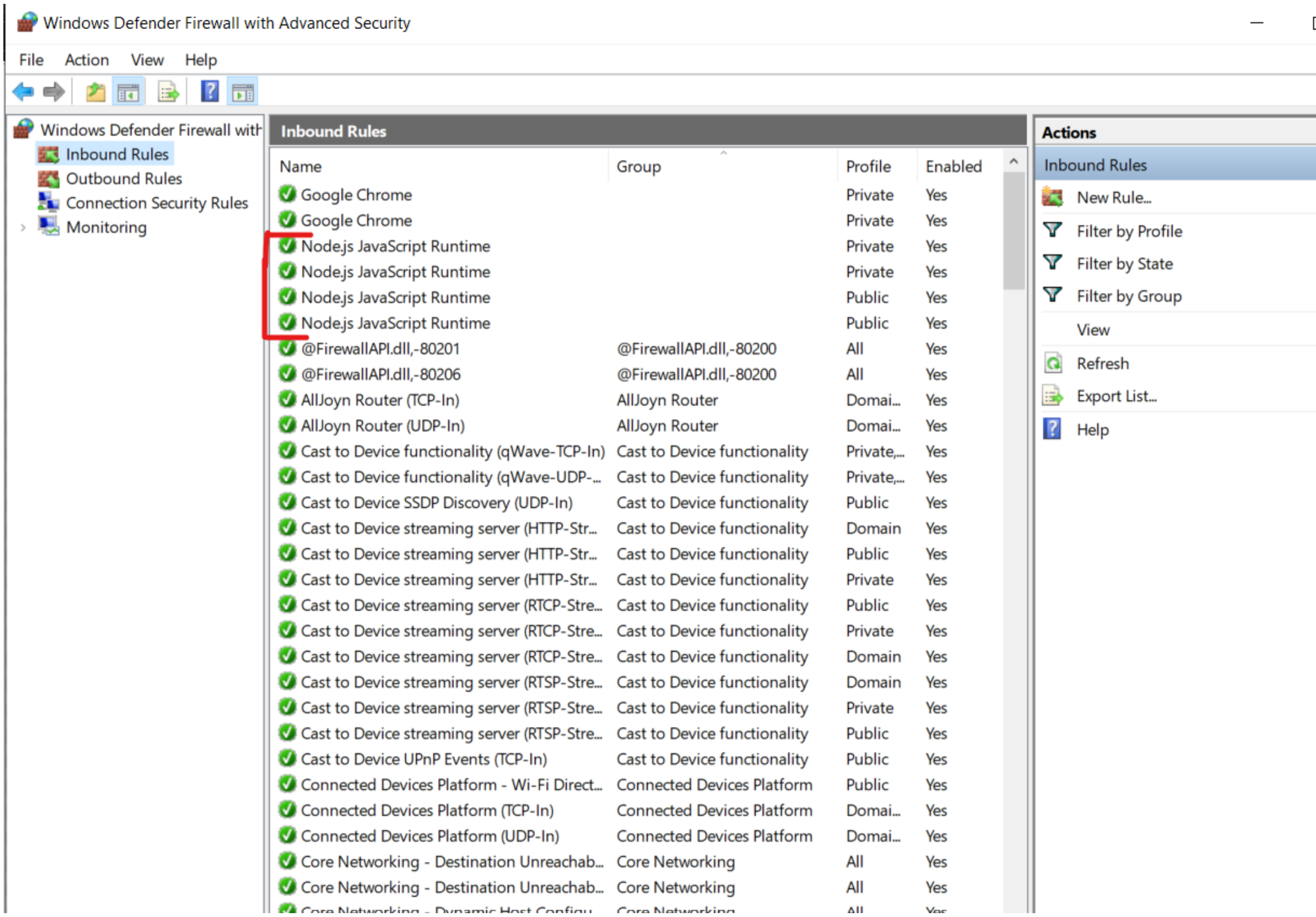
- Run the script using

```
$ python cert.py
```

- Check the output directory for the generated certificate files
- Ensure both files were created successfully
  - Server.crt (Security Certificate)
  - Server.key (Keynote Presentation)

3.3 Firewall Configuration

1. Head to Advanced settings in Windows Defender Firewall
2. Go to Inbound Rules
3. Make sure that both public & private profiles are allowed.



Name	Group	Profile	Enabled
Google Chrome		Private	Yes
Google Chrome		Private	Yes
Node.js JavaScript Runtime		Private	Yes
Node.js JavaScript Runtime		Private	Yes
Node.js JavaScript Runtime		Public	Yes
Node.js JavaScript Runtime		Public	Yes
@FirewallAPI.dll,-80201	@FirewallAPI.dll,-80200	All	Yes
@FirewallAPI.dll,-80206	@FirewallAPI.dll,-80200	All	Yes
AllJoyn Router (TCP-In)	AllJoyn Router	Domai...	Yes
AllJoyn Router (UDP-In)	AllJoyn Router	Domai...	Yes
Cast to Device functionality (qWave-TCP-In)	Cast to Device functionality	Private,...	Yes
Cast to Device functionality (qWave-UDP-...	Cast to Device functionality	Private,...	Yes
Cast to Device SSDP Discovery (UDP-In)	Cast to Device functionality	Public	Yes
Cast to Device streaming server (HTTP-Str...	Cast to Device functionality	Domain	Yes
Cast to Device streaming server (HTTP-Str...	Cast to Device functionality	Public	Yes
Cast to Device streaming server (HTTP-Str...	Cast to Device functionality	Private	Yes
Cast to Device streaming server (RTCP-Stre...	Cast to Device functionality	Public	Yes
Cast to Device streaming server (RTCP-Stre...	Cast to Device functionality	Private	Yes
Cast to Device streaming server (RTCP-Stre...	Cast to Device functionality	Domain	Yes
Cast to Device streaming server (RTSP-Stre...	Cast to Device functionality	Domain	Yes
Cast to Device streaming server (RTSP-Stre...	Cast to Device functionality	Private	Yes
Cast to Device streaming server (RTSP-Stre...	Cast to Device functionality	Public	Yes
Cast to Device UPnP Events (TCP-In)	Cast to Device functionality	Public	Yes
Connected Devices Platform - Wi-Fi Direct...	Connected Devices Platform	Public	Yes
Connected Devices Platform (TCP-In)	Connected Devices Platform	Domai...	Yes
Connected Devices Platform (UDP-In)	Connected Devices Platform	Domai...	Yes
Core Networking - Destination Unreachab...	Core Networking	All	Yes
Core Networking - Destination Unreachab...	Core Networking	All	Yes
Core Networking - Dynamic Host Configu...	Core Networking	All	Yes

4. If not, go to properties by right-clicking on the rule.
5. In General tab, in 'Action' section, - Allow the connection.

3.4 PC Setup

1. Head to the root directory and Run Server.js file

```
$ node server.js
```

2. Now the server is running, open a browser and visit https://localhost/pc.html -*Initially, it would show a small black screen*

3.5 Mobile Setup

1. Similarly open a browser and visit
   https://172.xx.xx.x/phone.html - *Replace it with the actual IPv4 Address*
2. Hit allow to grant camera access
3. And that's it!!

3.6 OBS Setup

1. Click on + (add) in the 'Source' tab.
2. Choose **Window Capture**
3. Select **Create new** (*name it anything you like*), and make sure to check - **Make source visible**
4. Choose the Browser window where the stream is live.
5. **Tip:** Make the stream a separate window to avoid interruption
6. Move and expand the red box to adjust the stream in frame.
7. And that's it, just hit **Start Virtual Camera** from the controls panel on the bottom right.

Note:

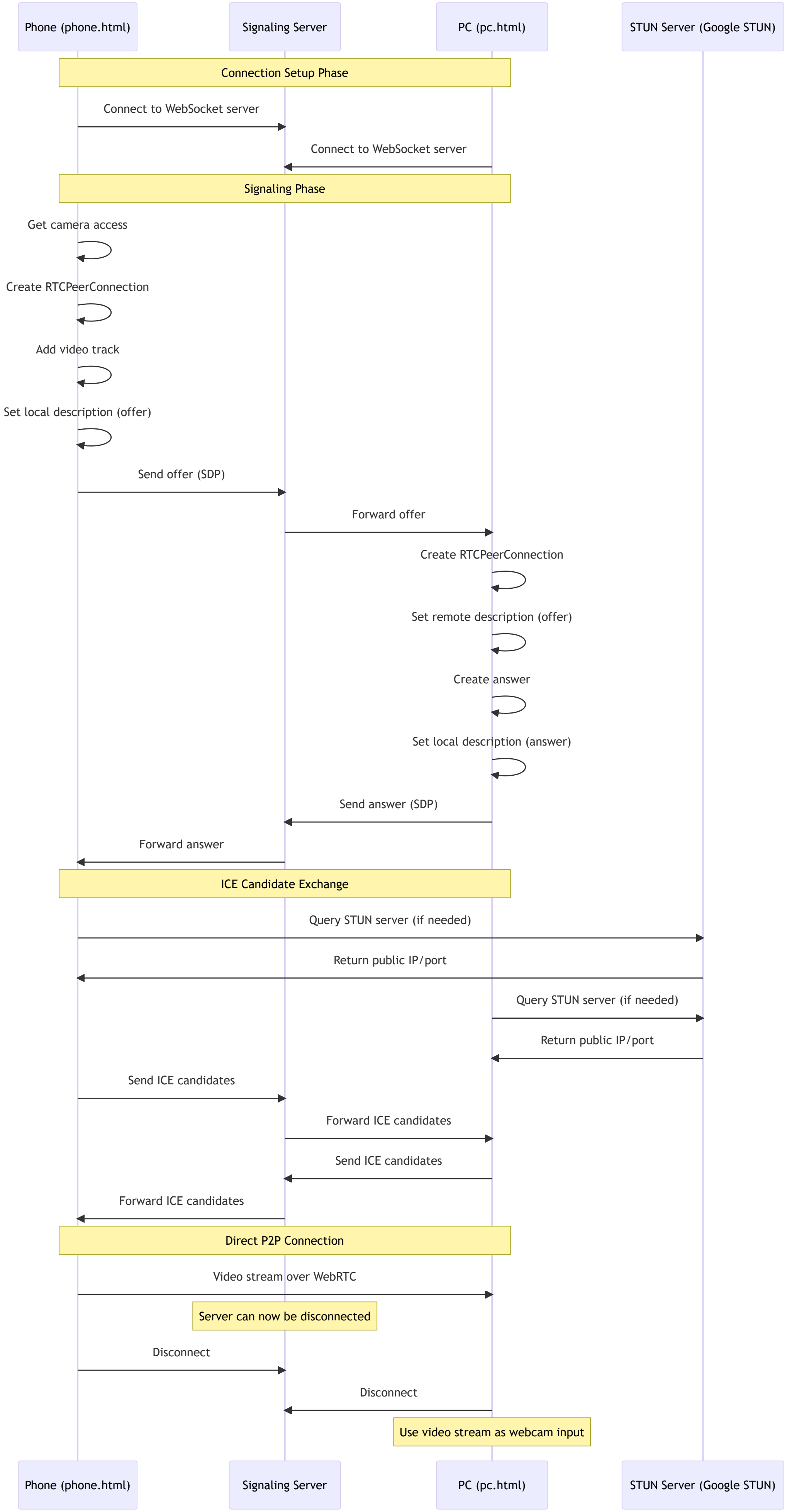
Do not minimize the browser window displaying the live stream from your phone's camera. Minimizing it will cause the OBS to freeze on black screen.

Tip: Use Alt+Tab to switch between windows

4 Flowchart

And there you have it - I've mapped out the complete flow of how your phone transforms into a PC webcam. The flowchart breaks down the whole process, from the initial server handshake to the final video streaming connection. You can see how the phone and PC find each other through the signaling server, set up their direct connection, and then handle the video feed independently.Now that you've used the system, you'll recognize each of these steps from your experience.

I'd love to hear your experience with the project :)


Phone (phone.html)
Signaling Server
PC (pc.html)
STUN Server (Google STUN)
Connection Setup Phase
Connect to WebSocket server
Connect to WebSocket server
Signaling Phase
Get camera access
Create RTCPeerConnection
Add video track
Set local description (offer)
Send offer (SDP)
Forward offer
Create RTCPeerConnection
Set remote description (offer)
Create answer
Set local description (answer)
Send answer (SDP)
Forward answer
ICE Candidate Exchange
Query STUN server (if needed)
Return public IP/port
Query STUN server (if needed)
Return public IP/port
Send ICE candidates
Forward ICE candidates
Send ICE candidates
Forward ICE candidates
Direct P2P Connection
Video stream over WebRTC
Server can now be disconnected
Disconnect
Disconnect
Use video stream as webcam input
Phone (phone.html)
Signaling Server
PC (pc.html)
STUN Server (Google STUN)