Telemedicine Connection-Reliability Checklist

Run this before launch. A clinical call must survive a bad network — detection, recovery, degradation, fallback. Engineering guidance, not legal advice — confirm specifics with counsel.

1 · DETECT (know you are in trouble)

- [ ] Watch connectionState: connected/completed = healthy; disconnected = wait; failed = act now
- [ ] Two-speed response: short timer on disconnected, immediate repair on failed
- [ ] Show the patient an honest 'Reconnecting…' indicator the moment a path drops
- [ ] Reconnection logs stay inside the compliance boundary — they can contain PHI
- [ ] Availability of ePHI is a Security Rule goal (45 CFR 164.306(a)(1)) — reliability is a control

2 · RECOVER (climb the cheapest rung first)

- [ ] ICE restart (restartIce()) wired to failed — recovers in ~1-2 s, media preserved
- [ ] Reconnect signaling FIRST, then restartIce() — a dead WebSocket sinks the restart
- [ ] Confirm your SFU / video API supports ICE restart for multi-party calls
- [ ] Escalate only if needed: rebuild RTCPeerConnection > re-join room > wait & resume
- [ ] Don't tear down the whole call on every blip — that throws away the DTLS keys

3 · DEGRADE & FALL BACK (bend before you break)

- [ ] Protect audio first: shed video quality to keep the conversation clear
- [ ] Ladder: lower resolution/frame rate > drop layers (SFU) > audio-only > phone
- [ ] Set degradationPreference per clinical context (motion vs detail)
- [ ] Build a PSTN dial-in fallback — audio-only telehealth is reimbursable (CMS thru 2027)
- [ ] Design the phone path as first-class care, not an apology screen

4 · RESUME & TEST (prove it on a hostile network)

- [ ] Keep session state server-side so a reconnect restores the room intact
- [ ] Signaling reconnect uses exponential backoff and re-auth; rejoin is idempotent
- [ ] Test the Wi-Fi-to-cellular handover on a real phone, walking out of range
- [ ] Test with simulated packet loss and bandwidth caps — verify the ladder triggers
- [ ] Test total drop (airplane mode) > restore > confirm clean automatic rejoin

THE RELIABILITY MATH

Say you run 1,000 consults/day and 4% hit a network event = 40 calls at risk. ICE restart silently recovers ~80% = 32 saved; the rebuild and re-join rungs catch most of the rest. The difference between 'drops the call' and 'freezes for two seconds' is the difference between a support ticket and a non-event — multiplied by every consult you will ever run. Reliability is the layer you cannot see in a demo and cannot skip in production.