**Web RTC Peer Connection**

const localvideo = document.getElementById('localvideo')

const remotevideo = document.getElementById('remotevideo')

const textoffer = document.getElementById('text-offer')

const textanswer = document.getElementById('text-answer')

const textsend = document.getElementById('text-send')

const textmessage = document.getElementById('text-message')

let peer;

let remoteStream;

let localStream;

let dataChannel;

let me;

let friend;

init = async () => {

localStream = await navigator.mediaDevices.getUserMedia({video:true,audio:true})

localvideo.srcObject = localStream

await testGet()

}

createPeerConnection = async () => {

const config = {iceServers:[{urls:'stun:stun.l.google.com:19302'}]}

peer = new RTCPeerConnection(config)

remoteStream = new MediaStream()

remotevideo.srcObject = remoteStream

localStream.getTracks().forEach(track => peer.addTrack(track,localStream))

peer.ontrack = e => e.streams[0].getTracks().forEach(track => remoteStream.addTrack(track))

}

createOffer = async () => {

await createPeerConnection()

dataChannel = await peer.createDataChannel("channel")

dataChannel.onopen = () => console.log("open channel")

dataChannel.onmessage = msg => textmessage.value += "[Teman] : " + msg.data + "\n"

const offer = await peer.createOffer()

await peer.setLocalDescription(offer)

textoffer.value = JSON.stringify(peer.localDescription)

peer.onicecandidate = e => {

if(e.candidate) textoffer.value = JSON.stringify(peer.localDescription)

}

}

createAnswer = async () => {

await createPeerConnection()

peer.ondatachannel = e => {

dataChannel = e.channel

dataChannel.onopen = () => console.log("open channel")

dataChannel.onmessage = msg => textmessage.value += "[Teman] : " + msg.data + "\n"

showMessage("Berhasil Menghubungkan","display-none")

}

const offer = JSON.parse(textoffer.value)

peer.setRemoteDescription(offer)

const answer = await peer.createAnswer()

await peer.setLocalDescription(answer)

textanswer.value = JSON.stringify(peer.localDescription)

peer.onicecandidate = e => {

if(e.candidate) textanswer.value = JSON.stringify(peer.localDescription)

}

}

addAnswer = async () => {

const answer = JSON.parse(textanswer.value)

await peer.setRemoteDescription(answer)

}

init()

document.getElementById('btn-offer').onclick = async () => await createOffer()

document.getElementById('btn-answer').onclick = async () => await createAnswer()

document.getElementById('add-answer').onclick = async () => await addAnswer()

document.getElementById('btn-send').onclick = () => {

const message = textsend.value

textsend.value = ""

textmessage.value += "[Aku] : " + message + "\n"

dataChannel.send(message)

}

document.querySelector('.flip').addEventListener('click', () => {

if(localvideo.classList.contains('localvideo')){

localvideo.className = "remotevideo"

remotevideo.className = "localvideo"

}else{

localvideo.className = "localvideo"

remotevideo.className = "remotevideo"

}

})

localvideo.onclick = () => localvideo.muted = !localvideo.muted

remotevideo.onclick = () => remotevideo.muted = !remotevideo.muted

async function testPost(storeData){

const data = await DB.postData(storeData)

console.log(data)

}

async function testGet(){

me = Date.now()+""+Math.floor(Math.random()\*100000)

const data = await DB.getData()

if(data.length == 1){

showMessage("Menunggu Teman Terhubung..","")

await createOffer()

setTimeout(async () => {

const dataObject = {

timestamp: "offer",

username: me,

sdp: textoffer.value

}

await testPost(dataObject)

const checkFriendConnected = setInterval( async () => {

const data = await DB.getData()

if(data.length == 3){

textanswer.value = (data[2][2])

showMessage("Berhasil Menghubungkan","")

await DB.deleteData()

await addAnswer()

showMessage("Berhasil Menghubungkan","display-none")

clearInterval(checkFriendConnected)

}else console.log("waiting...")

}, 3000);

checkFriendConnected()

}, 3000);

}else if(data.length == 2){

showMessage("Menghubungkan Dengan Teman..","")

textoffer.value = (data[1][2]) // get offer

await createAnswer()

setTimeout( async () => {

const dataObject = {

timestamp: "answer",

username: me,

sdp: textanswer.value

}

await testPost(dataObject)

}, 3000);

}else {}

}

function showMessage(message, classname){

const elMessage = document.querySelector('.waiting-message')

elMessage.innerHTML = message

if(classname!="") elMessage.classList.add(classname)

}