Appendix F

 SAMPLE PROGRAM CODE

import React, { useEffect, useState, useContext, useRef } from 'react';

import { useHistory } from 'react-router-dom';

import Peer from 'peerjs';

import axios from 'axios';

import {Grid, Snackbar,Fab,FormControl,InputLabel,Select,MenuItem,Dialog,

DialogTitle,DialogContent,DialogActions,Button,Card,Avatar,Typography,

} from '@mui/material';

import { io } from 'socket.io-client';

import { Editor } from '../components/CodeEditor';

import TimerCountdown from '../components/TimerCountdown';

import { Rnd } from 'react-rnd';

import { UserContext } from '../contexts/UserContext';

import ReviewPost from '../components/ReviewPost';

import { Widget, addResponseMessage } from 'react-chat-theme';

import styled from 'styled-components';

import { ReviewContext } from '../contexts/ReviewContext';

import MicIcon from '@mui/icons-material/Mic';

import MicOffIcon from '@mui/icons-material/MicOff';

import VideocamIcon from '@mui/icons-material/Videocam';

import VideocamOffSharpIcon from '@mui/icons-material/VideocamOffSharp';

import PlayArrowIcon from '@mui/icons-material/PlayArrow';

import WebAssetIcon from '@mui/icons-material/WebAsset';

import CallEndIcon from '@mui/icons-material/CallEnd';

const Session = ({ match }) => {

  const { user, urlAPI } = useContext(UserContext);

  const history = useHistory();

  const { reviewer, setReviewer } = useContext(ReviewContext);

  //Teacher

  const [teacher, setTeacher] = useState();

  const [teacherID, setTeacherID] = useState();

  //Video Cam MySelf

  const myFace = useRef(null);

  const peerFace = useRef(null);

  const peerInstance = useRef(null);

  const myStream = useRef();

  const socket = io(urlAPI);

  const [isValid, setIsValid] = useState(false);

  const [fullname, setFullname] = useState(

    `${user.firstname} ${user.lastname}`

  );

  const [description, setDescription] = useState();

  const [language, setLanguage] = useState('C#');

  const [state, setState] = useState({ open: false,vertical: 'top', horizontal: 'center', });

  const [expireDate, setExpireDate] = useState();

  const [startingDate, setStartingDate] = useState();

  //Button Video Chat

  const [isMuted, setIsMuted] = useState(true);

  const [isCameraOn, setIsCameraOn] = useState(true);

  //End Button Video Chat

  const { vertical, horizontal, open } = state;

  const handleOpen = (newState) => {

    setState({ open: true, ...newState });

  };

  const handleClose = () => {

    setState({ ...state, open: false });

  };

  const handleChangeLanguage = (e) => {

    setLanguage(e.target.value);

  };

  //-- -- Start ----End Session Button

  const [endSessionBtn, setEndSessionBtn] = useState(false);

  const handleEndSessionOpen = async (value) => {

    await getTeacher();

    setEndSessionBtn(value);

  };

  const handleEndSessionClose = () => {

    setEndSessionBtn(false);

  };

  const [endSessionTeacherBtn, setEndSessionTeacherBtn] = useState(false);

  const handleEndSessionTeacherOpen = () => {

    setEndSessionTeacherBtn(true);

  };

  const handleEndSessionTeacherClose = () => {

    setEndSessionTeacherBtn(false);

  };

  // ------End ----

  // Double Click Video fullscreen

  function openFullscreen() {

    if (myFace.current.requestFullscreen) {

      myFace.current.requestFullscreen();

    } else if (myFace.current.webkitRequestFullscreen) {

      /\* Safari \*/

      myFace.current.webkitRequestFullscreen();

    } else if (myFace.current.msRequestFullscreen) {

      /\* IE11 \*/

      myFace.current.msRequestFullscreen();

    }

  }

  // Mute Microphone

  const muteMic = () => {

    myStream.current

      .getAudioTracks()

      .forEach((track) => (track.enabled = !track.enabled));

    setIsMuted(!isMuted);

  };

  // Turn of Camera

  const turnOffCam = () => {

    myStream.current

      .getVideoTracks()

      .forEach((track) => (track.enabled = !track.enabled));

    setIsCameraOn(!isCameraOn);

    myFace.current.hidden = !myFace.current.hidden;

    socket.emit( 'hidePeerFace', match.params.id, user.id, isCameraOn ? true : false);};