<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

body {font-family: Arial, Helvetica, sans-serif;}

/\* Full-width input fields \*/

input[type=text], input[type=password] {

width: 100%;

padding: 12px 20px;

margin: 8px 0;

display: inline-block;

border: 1px solid #ccc;

box-sizing: border-box;

}

/\* Set a style for all buttons \*/

button {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

width: 100%;

}

button:hover {

opacity: 0.8;

}

/\* Extra styles for the cancel button \*/

.cancelbtn {

width: auto;

padding: 10px 18px;

background-color: #f44336;

}

/\* Center the image and position the close button \*/

.imgcontainer {

text-align: center;

margin: 24px 0 12px 0;

position: relative;

}

img.avatar {

width: 40%;

border-radius: 50%;

}

.container {

padding: 16px;

}

span.psw {

float: right;

padding-top: 16px;

}

/\* The Modal (background) \*/

.modal {

display: none; /\* Hidden by default \*/

position: fixed; /\* Stay in place \*/

z-index: 1; /\* Sit on top \*/

left: 0;

top: 0;

width: 100%; /\* Full width \*/

height: 100%; /\* Full height \*/

overflow: auto; /\* Enable scroll if needed \*/

background-color: rgb(0,0,0); /\* Fallback color \*/

background-color: rgba(0,0,0,0.4); /\* Black w/ opacity \*/

padding-top: 60px;

}

/\* Modal Content/Box \*/

.modal-content {

background-color: #fefefe;

margin: 5% auto 15% auto; /\* 5% from the top, 15% from the bottom and centered \*/

border: 1px solid #888;

width: 80%; /\* Could be more or less, depending on screen size \*/

}

/\* The Close Button (x) \*/

.close {

position: absolute;

right: 25px;

top: 0;

color: #000;

font-size: 35px;

font-weight: bold;

}

.close:hover,

.close:focus {

color: red;

cursor: pointer;

}

/\* Add Zoom Animation \*/

.animate {

-webkit-animation: animatezoom 0.6s;

animation: animatezoom 0.6s

}

@-webkit-keyframes animatezoom {

from {-webkit-transform: scale(0)}

to {-webkit-transform: scale(1)}

}

@keyframes animatezoom {

from {transform: scale(0)}

to {transform: scale(1)}

}

/\* Change styles for span and cancel button on extra small screens \*/

@media screen and (max-width: 300px) {

span.psw {

display: block;

float: none;

}

.cancelbtn {

width: 100%;

}

}

</style>

</head>

<body>

<h2>REAL TIME COMMUNICATION SYSTEM POWERED BY AI FOR SPECIALLY ABLED</h2>

<button onclick="document.getElementById('id01').style.display='block'" style="width:auto;">Login</button>

<div id="id01" class="modal">

<form class="modal-content animate" action="/action\_page.php" method="post">

<div class="imgcontainer">

<span onclick="document.getElementById('id01').style.display='none'" class="close" title="Close Modal">&times;</span>

<img src="https://static.vecteezy.com/system/resources/thumbnails/007/407/996/small/user-icon-person-icon-client-symbol-login-head-sign-icon-design-vector.jpg">

</div>

<div class="container">

<label for="uname"><b>Username</b></label>

<input type="text" placeholder="Enter Username" name="uname" required>

<label for="psw"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="psw" required>

<button type="submit">Login</button>

<label>

<input type="checkbox" checked="checked" name="remember"> Remember me

</label>

</div>

<div class="container" style="background-color:#f1f1f1">

<button type="button" onclick="document.getElementById('id01').style.display='none'" class="cancelbtn">Cancel</button>

<span class="psw">Forgot <a href="#">password?</a></span>

</div>

</form>

</div>

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<link rel="stylesheet" href="style.css">

<title>Document</title>

</head>

<body>

<div class="display-cover">

<video autoplay></video>

<canvas class="d-none"></canvas>

<div class="video-options">

<select name="" id="" class="custom-select">

<option value="">Select camera</option>

</select>

</div>

<img class="screenshot-image d-none" alt="">

<div class="controls">

<button class="btn btn-danger play" title="Play"><i data-feather="play-circle"></i></button>

<button class="btn btn-info pause d-none" title="Pause"><i data-feather="pause"></i></button>

<button class="btn btn-outline-success screenshot d-none" title="ScreenShot"><i data-feather="image"></i></button>

</div>

</div>

<script src="https://unpkg.com/feather-icons"></script>

<script src="script.js"></script>

</body>

<html><head>

</head><body>

<video src="" ></video>

<br />

<button id='flipCamera'>Flip</button>

</body>

<script>

var front = false;

var video = document.querySelector('video');

document.getElementById('flipCamera').onclick = function() { front = !front; };

var constraints = { video: { facingMode: (front? "user" : "environment"), width: 640, height: 480 } };

navigator.mediaDevices.getUserMedia(constraints)

.then(function(mediaStream) {

video.srcObject = mediaStream;

video.onloadedmetadata = function(e) {

video.play();

};

})

.catch(function(err) { console.log(err.name + ": " + err.message); })

</script></html>

</html>

<style>

.screenshot-image {

width: 150px;

height: 90px;

border-radius: 4px;

border: 2px solid whitesmoke;

box-shadow: 0 1px 2px 0 rgba(0, 0, 0, 0.1);

position: absolute;

bottom: 5px;

left: 10px;

background: white;

}

.display-cover {

display: flex;

justify-content: center;

align-items: center;

width: 70%;

margin: 5% auto;

position: relative;

}

video {

width: 100%;

background: rgba(0, 0, 0, 0.2);

}

.video-options {

position: absolute;

left: 20px;

top: 30px;

}

.controls {

position: absolute;

right: 20px;

top: 20px;

display: flex;

}

.controls > button {

width: 45px;

height: 45px;

text-align: center;

border-radius: 100%;

margin: 0 6px;

background: transparent;

}

.controls > button:hover svg {

color: white !important;

}

@media (min-width: 300px) and (max-width: 400px) {

.controls {

flex-direction: column;

}

.controls button {

margin: 5px 0 !important;

}

}

.controls > button > svg {

height: 20px;

width: 18px;

text-align: center;

margin: 0 auto;

padding: 0;

}

.controls button:nth-child(1) {

border: 2px solid #D2002E;

}

.controls button:nth-child(1) svg {

color: #D2002E;

}

.controls button:nth-child(2) {

border: 2px solid #008496;

}

.controls button:nth-child(2) svg {

color: #008496;

}

.controls button:nth-child(3) {

border: 2px solid #00B541;

}

.controls button:nth-child(3) svg {

color: #00B541;

}

.controls > button {

width: 45px;

height: 45px;

text-align: center;

border-radius: 100%;

margin: 0 6px;

background: transparent;

}

.controls > button:hover svg {

color: white;

}

</style>

<script>

// Get the modal

var modal = document.getElementById('id01');

// When the user clicks anywhere outside of the modal, close it

window.onclick = function(event) {

if (event.target == modal) {

modal.style.display = "none";

}

}

feather.replace();

const controls = document.querySelector('.controls');

const cameraOptions = document.querySelector('.video-options>select');

const video = document.querySelector('video');

const canvas = document.querySelector('canvas');

const screenshotImage = document.querySelector('img');

const buttons = [...controls.querySelectorAll('button')];

let streamStarted = false;

const [play, pause, screenshot] = buttons;

const constraints = {

video: {

width: {

min: 1280,

ideal: 1920,

max: 2560,

},

height: {

min: 720,

ideal: 1080,

max: 1440

},

}

};

</script>

<script>

const getCameraSelection = async () => {

const devices = await navigator.mediaDevices.enumerateDevices();

const videoDevices = devices.filter(device => device.kind === 'videoinput');

const options = videoDevices.map(videoDevice => {

return `<option value="${videoDevice.deviceId}">${videoDevice.label}</option>`;

});

cameraOptions.innerHTML = options.join('');

};

</script>

<script>

play.onclick = () => {

if (streamStarted) {

video.play();

play.classList.add('d-none');

pause.classList.remove('d-none');

return;

}

if ('mediaDevices' in navigator && navigator.mediaDevices.getUserMedia) {

const updatedConstraints = {

...constraints,

deviceId: {

exact: cameraOptions.value

}

};

startStream(updatedConstraints);

}

};

const startStream = async (constraints) => {

const stream = await navigator.mediaDevices.getUserMedia(constraints);

handleStream(stream);

};

const handleStream = (stream) => {

video.srcObject = stream;

play.classList.add('d-none');

pause.classList.remove('d-none');

screenshot.classList.remove('d-none');

streamStarted = true;

};

getCameraSelection();

cameraOptions.onchange = () => {

const updatedConstraints = {

...constraints,

deviceId: {

exact: cameraOptions.value

}

};

startStream(updatedConstraints);

};

const pauseStream = () => {

video.pause();

play.classList.remove('d-none');

pause.classList.add('d-none');

};

const doScreenshot = () => {

canvas.width = video.videoWidth;

canvas.height = video.videoHeight;

canvas.getContext('2d').drawImage(video, 0, 0);

screenshotImage.src = canvas.toDataURL('image/webp');

screenshotImage.classList.remove('d-none');

};

pause.onclick = pauseStream;

screenshot.onclick = doScreenshot;

</script>

</body>

</html>