Filename::index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Stopwatch & Clock App</title>

<link rel="stylesheet" href="main.css" />

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&display=swap" rel="stylesheet">

</head>

<body>

<div class="container">

<h1>Stopwatch & Clock</h1>

<div class="clock-wrapper" aria-label="Current time and date">

<div class="clock" id="clock">

<div class="time">--:--:-- <span class="ampm">--</span></div>

<div class="date">---</div>

</div>

<canvas id="analogClock" width="100" height="100" aria-label="Analog clock"></canvas>

</div>

<div class="stopwatch-time" id="stopwatch" aria-live="polite">00:00:00.000</div>

<div class="buttons">

<button id="startBtn" aria-label="Start stopwatch">Start</button>

<button id="stopBtn" aria-label="Stop stopwatch" disabled>Stop</button>

<button id="resetBtn" aria-label="Reset stopwatch" disabled>Reset</button>

</div>

</div>

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

</body>

</html>

Filename::main.css

\* {

box-sizing: border-box;

}

body {

margin: 0;

padding: 20px;

font-family: 'Inter', -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto,

Oxygen, Ubuntu, Cantarell, "Open Sans", "Helvetica Neue", sans-serif;

background: linear-gradient(135deg, #081526, #15395f);

color: #e0e6f0;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

min-height: 100vh;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

user-select: none;

}

.container {

background: #192a56;

border-radius: 20px;

padding: 40px 50px;

box-shadow: 0 8px 24px rgba(15, 40, 75, 0.5);

max-width: 420px;

width: 100%;

text-align: center;

}

h1 {

margin: 0 0 20px;

font-weight: 700;

font-size: 2.25rem;

letter-spacing: 1.5px;

color: #f5f7fa;

text-shadow: 0 1px 3px rgba(0,0,0,0.6);

}

.clock-wrapper {

display: flex;

justify-content: space-between; /\* Adjusted to space between \*/

align-items: center;

margin-bottom: 40px;

}

.clock {

font-weight: 600;

color: #82cfff;

text-shadow: 0 1px 5px rgba(0, 126, 255, 0.7);

user-select: none;

min-width: 120px; /\* Adjusted width \*/

text-align: center;

}

.clock .time {

font-size: 2rem; /\* Decreased font size \*/

letter-spacing: 2px; /\* Adjusted letter spacing \*/

line-height: 1.2;

}

.clock .ampm {

font-size: 1rem; /\* Decreased font size \*/

font-weight: 600;

color: #4dacff;

vertical-align: super;

letter-spacing: 1px;

}

.clock .date {

font-size: 0.8rem; /\* Decreased font size \*/

margin-top: 4px; /\* Adjusted margin \*/

color: #aad4ff;

letter-spacing: 1px;

}

#analogClock {

width: 100px; /\* Decreased width \*/

height: 100px; /\* Decreased height \*/

background: #0e1f44;

border-radius: 50%;

box-shadow:

inset 0 0 12px rgba(0, 102, 255, 0.6),

0 0 12px rgba(0, 102, 255, 0.9);

}

.stopwatch-time {

font-size: 4rem;

font-weight: 800;

letter-spacing: 4px;

margin-bottom: 30px;

color: #ffa502;

text-shadow: 0 1px 4px rgba(255, 165, 0, 0.8);

transition: color 0.2s ease;

font-variant-numeric: tabular-nums;

}

.buttons {

display: flex;

justify-content: center;

gap: 18px;

}

button {

background: linear-gradient(135deg, #0066ff, #0050cc);

border: none;

color: #f5f7fa;

padding: 14px 24px;

font-size: 1.1rem;

font-weight: 600;

border-radius: 12px;

cursor: pointer;

box-shadow: 0 4px 14px rgba(0, 102, 255, 0.5);

transition: background 0.3s ease, box-shadow 0.3s ease;

flex: 1;

user-select: none;

}

button:hover:not(:disabled) {

background: linear-gradient(135deg, #0050cc, #003d99);

box-shadow: 0 6px 20px rgba(0, 80, 204, 0.8);

}

button:disabled {

background: #384a6a;

cursor: not-allowed;

box-shadow: none;

color: #7a8ca6;

}

@media (max-width: 440px) {

.container {

padding: 30px 30px;

}

h1 {

font-size: 1.8rem;

}

.clock .time {

font-size: 1.75rem; /\* Adjusted font size \*/

letter-spacing: 2px;

line-height: 1.1;

}

.clock .ampm {

font-size: 0.9rem; /\* Adjusted font size \*/

vertical-align: top;

margin-left: 0;

letter-spacing: 0.5px;

}

.clock .date {

font-size: 0.7rem; /\* Adjusted font size \*/

}

.stopwatch-time {

font-size: 3rem;

letter-spacing: 3px;

margin-bottom: 24px;

}

button {

padding: 12px 18px;

font-size: 1rem;

}

}

Filename::main.js

function updateClock() {

const clockElement = document.getElementById('clock');

const timeDiv = clockElement.querySelector('.time');

const ampmSpan = timeDiv.querySelector('.ampm');

const dateDiv = clockElement.querySelector('.date');

const now = new Date();

let hours = now.getHours();

const minutes = now.getMinutes();

const seconds = now.getSeconds();

const ampm = hours >= 12 ? 'PM' : 'AM';

hours = hours % 12 || 12;

const hStr = hours.toString().padStart(2, '0');

const mStr = minutes.toString().padStart(2, '0');

const sStr = seconds.toString().padStart(2, '0');

const options = { weekday: 'short', year: 'numeric', month: 'short', day: '2-digit' };

const dateStr = now.toLocaleDateString(undefined, options);

timeDiv.innerHTML = `${hStr}:${mStr}:${sStr} <span class="ampm">${ampm}</span>`;

dateDiv.textContent = dateStr;

}

setInterval(updateClock, 1000);

updateClock();

const canvas = document.getElementById('analogClock');

const ctx = canvas.getContext('2d');

const scale = window.devicePixelRatio || 1;

canvas.width \*= scale;

canvas.height \*= scale;

ctx.scale(scale, scale);

const radius = canvas.height / (2 \* scale);

ctx.translate(radius, radius);

const clockRadius = radius \* 0.90;

function drawFace() {

ctx.beginPath();

ctx.arc(0, 0, clockRadius, 0, 2 \* Math.PI);

ctx.fillStyle = '#0e1f44';

ctx.fill();

ctx.strokeStyle = '#1b2e70';

ctx.lineWidth = radius \* 0.07;

ctx.shadowColor = 'rgba(0, 102, 255, 0.6)';

ctx.shadowBlur = 10;

ctx.stroke();

ctx.shadowBlur = 0;

ctx.beginPath();

ctx.arc(0, 0, radius \* 0.05, 0, 2 \* Math.PI);

ctx.fillStyle = '#3977ff';

ctx.fill();

}

function drawNumbers() {

ctx.font = radius \* 0.15 + "px 'Inter', sans-serif";

ctx.fillStyle = '#82cfff';

ctx.textBaseline = "middle";

ctx.textAlign = "center";

for (let num = 1; num <= 12; num++) {

const ang = num \* Math.PI / 6;

ctx.rotate(ang);

ctx.translate(0, -clockRadius \* 0.82);

ctx.rotate(-ang);

ctx.fillText(num.toString(), 0, 0);

ctx.rotate(ang);

ctx.translate(0, clockRadius \* 0.82);

ctx.rotate(-ang);

}

}

function drawHand(pos, length, width, color) {

ctx.beginPath();

ctx.lineWidth = width;

ctx.lineCap = "round";

ctx.strokeStyle = color;

ctx.moveTo(0, 0);

ctx.rotate(pos);

ctx.lineTo(0, -length);

ctx.stroke();

ctx.rotate(-pos);

}

function drawTime() {

const now = new Date();

let hour = now.getHours() % 12;

const minute = now.getMinutes();

const second = now.getSeconds();

const hourPos = (hour + minute / 60 + second / 3600) \* Math.PI / 6;

const minutePos = (minute + second / 60) \* Math.PI / 30;

const secondPos = second \* Math.PI / 30;

drawHand(hourPos, clockRadius \* 0.5, radius \* 0.07, '#ffa502');

drawHand(minutePos, clockRadius \* 0.75, radius \* 0.05, '#4db8ff');

drawHand(secondPos, clockRadius \* 0.85, radius \* 0.015, '#ff6b6b');

}

function drawClock() {

ctx.clearRect(-radius, -radius, canvas.width, canvas.height);

drawFace();

drawNumbers();

drawTime();

}

setInterval(drawClock, 1000);

drawClock();

let startTime = 0;

let elapsedTime = 0;

let stopwatchInterval;

const stopwatchEl = document.getElementById('stopwatch');

const startBtn = document.getElementById('startBtn');

const stopBtn = document.getElementById('stopBtn');

const resetBtn = document.getElementById('resetBtn');

function timeToString(time) {

const diffInHrs = time / 3600000;

const hours = Math.floor(diffInHrs);

const diffInMin = (diffInHrs - hours) \* 60;

const minutes = Math.floor(diffInMin);

const diffInSec = (diffInMin - minutes) \* 60;

const seconds = Math.floor(diffInSec);

const diffInMs = (diffInSec - seconds) \* 1000;

const milliseconds = Math.floor(diffInMs);

return (

String(hours).padStart(2, '0') + ':' +

String(minutes).padStart(2, '0') + ':' +

String(seconds).padStart(2, '0') + '.' +

String(milliseconds).padStart(3, '0')

);

}

function startStopwatch() {

startTime = Date.now() - elapsedTime;

stopwatchInterval = setInterval(() => {

elapsedTime = Date.now() - startTime;

stopwatchEl.textContent = timeToString(elapsedTime);

stopwatchEl.style.color = '#ffda79';

}, 10);

startBtn.disabled = true;

stopBtn.disabled = false;

resetBtn.disabled = false;

}

function stopStopwatch() {

clearInterval(stopwatchInterval);

stopwatchEl.style.color = '#ffa502';

startBtn.disabled = false;

stopBtn.disabled = true;

resetBtn.disabled = false;

}

function resetStopwatch() {

clearInterval(stopwatchInterval);

elapsedTime = 0;

stopwatchEl.textContent = '00:00:00.000';

stopwatchEl.style.color = '#ffa502';

startBtn.disabled = false;

stopBtn.disabled = true;

resetBtn.disabled = true;

}

startBtn.addEventListener('click', startStopwatch);

stopBtn.addEventListener('click', stopStopwatch);

resetBtn.addEventListener('click', resetStopwatch);