**Web Technology**

**Naresh Kumar**

**22IT3025**

**Lab-8**

**using react**

**T1:**

import React, { useState } from 'react';

const CurrencyConverter = () => {

const [amount, setAmount] = useState('');

const [fromCurrency, setFromCurrency] = useState('USD');

const [toCurrency, setToCurrency] = useState('EUR');

const [convertedAmount, setConvertedAmount] = useState(null);

const exchangeRates = {

USD: {

EUR: 0.85,

INR: 75.32,

GBP: 0.73,

},

EUR: {

USD: 1.18,

INR: 88.68,

GBP: 0.86,

},

INR: {

USD: 0.013,

EUR: 0.011,

GBP: 0.0097,

},

GBP: {

USD: 1.38,

EUR: 1.16,

INR: 103.36,

},

};

const handleAmountChange = (e) => {

setAmount(e.target.value);

};

const handleFromCurrencyChange = (e) => {

setFromCurrency(e.target.value);

};

const handleToCurrencyChange = (e) => {

setToCurrency(e.target.value);

};

const convertCurrency = () => {

const exchangeRate = exchangeRates[fromCurrency][toCurrency];

const converted = amount \* exchangeRate;

setConvertedAmount(converted.toFixed(2));

};

return (

<div>

<h1>Currency Converter</h1>

<div>

<label>

Amount:

<input type="number" value={amount} onChange={handleAmountChange} />

</label>

</div>

<div>

<label>

From Currency:

<select value={fromCurrency} onChange={handleFromCurrencyChange}>

{Object.keys(exchangeRates).map((currency) => (

<option key={currency} value={currency}>

{currency}

</option>

))}

</select>

</label>

</div>

<div>

<label>

To Currency:

<select value={toCurrency} onChange={handleToCurrencyChange}>

{Object.keys(exchangeRates[fromCurrency]).map((currency) => (

<option key={currency} value={currency}>

{currency}

</option>

))}

</select>

</label>

</div>

<button onClick={convertCurrency}>Convert</button>

{convertedAmount && (

<p>

Converted Amount: {convertedAmount} {toCurrency}

</p>

)}

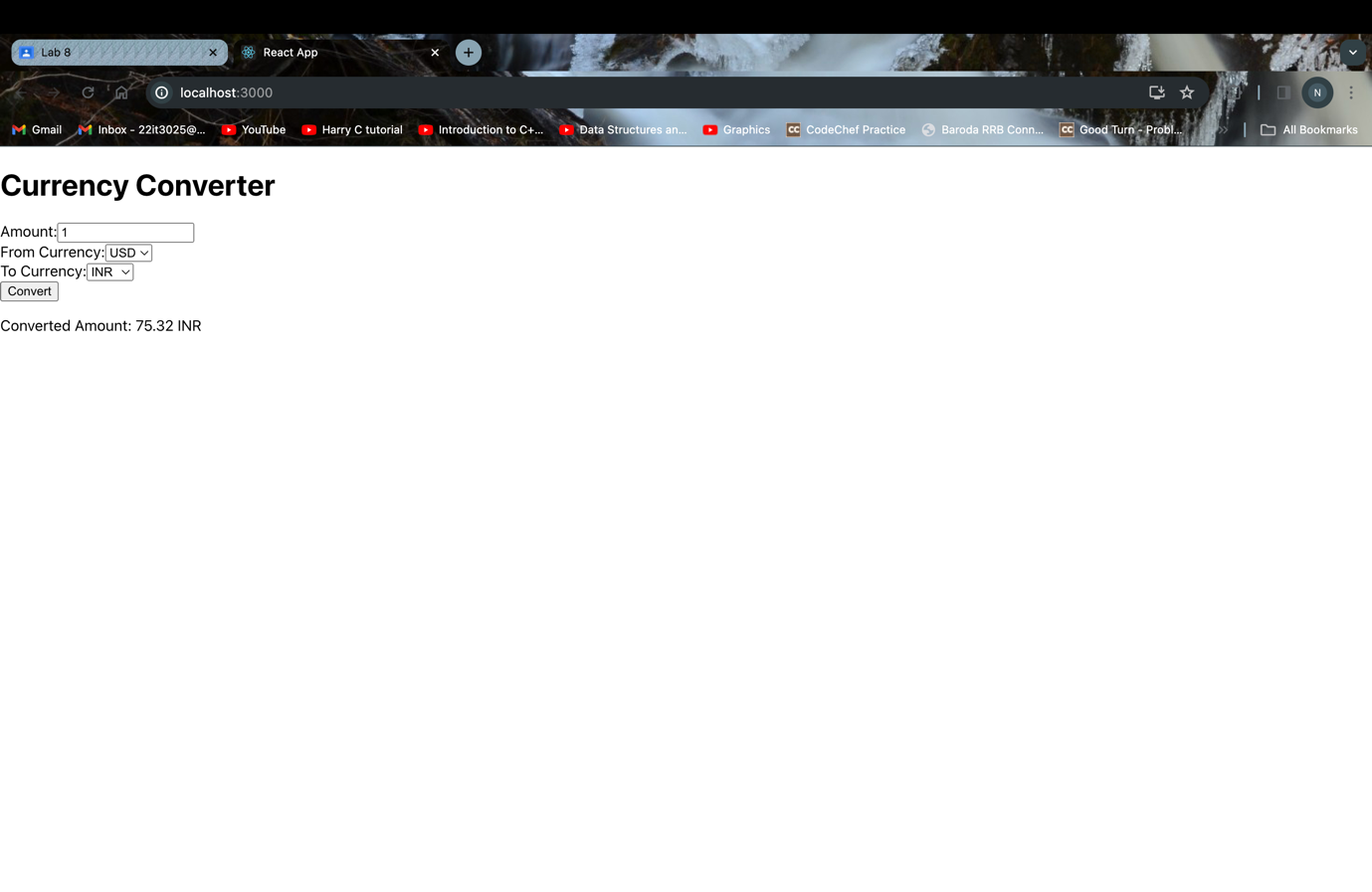
</div>

);

};

export default CurrencyConverter;

OUTPUT



T2:

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

function Stopwatch() {

const [time, setTime] = useState(0);

const [isRunning, setIsRunning] = useState(false);

useEffect(() => {

let timer;

if (isRunning) {

timer = setInterval(() => {

setTime(prevTime => prevTime + 1);

}, 1000);

} else {

clearInterval(timer);

}

return () => clearInterval(timer);

}, [isRunning]);

const startStopwatch = () => {

setIsRunning(true);

};

const pauseStopwatch = () => {

setIsRunning(false);

};

const resetStopwatch = () => {

setTime(0);

setIsRunning(false);

};

const formatTime = () => {

const hours = Math.floor(time / 3600);

const minutes = Math.floor((time % 3600) / 60);

const seconds = time % 60;

const formattedHours = String(hours).padStart(2, '0');

const formattedMinutes = String(minutes).padStart(2, '0');

const formattedSeconds = String(seconds).padStart(2, '0');

return `${formattedHours}:${formattedMinutes}:${formattedSeconds}`;

};

return (

<div className="stopwatch">

<h1>Stopwatch</h1>

<div className="time">{formatTime()}</div>

<div className="controls">

{!isRunning ? (

<button onClick={startStopwatch}>Start</button>

) : (

<button onClick={pauseStopwatch}>Pause</button>

)}

<button onClick={resetStopwatch}>Reset</button>

</div>

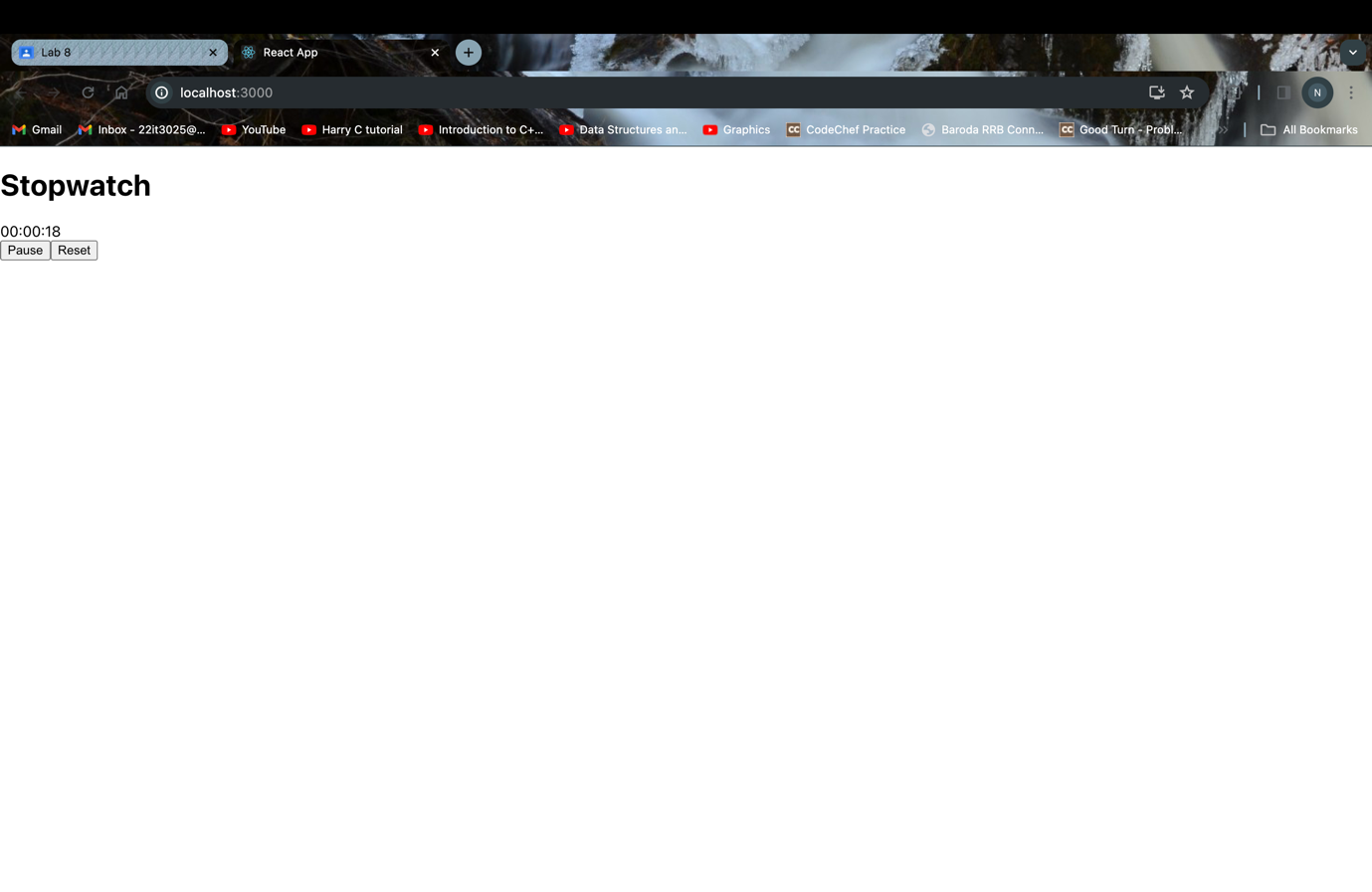
</div>

);

}

export default Stopwatch;

OUTPUT



Using Vue

T1:

<template>

<div>

<h1>Currency Converter</h1>

<div>

<label>

Amount:

<input type="number" v-model="amount" />

</label>

</div>

<div>

<label>

From Currency:

<select v-model="fromCurrency">

<option value="USD">USD</option>

<option value="EUR">EUR</option>

<option value="GBP">GBP</option>

<option value="INR">INR</option>

</select>

</label>

</div>

<div>

<label>

To Currency:

<select v-model="toCurrency">

<option value="EUR">EUR</option>

<option value="USD">USD</option>

<option value="GBP">GBP</option>

<option value="INR">INR</option>

</select>

</label>

</div>

<button @click="convertCurrency">Convert</button>

<p v-if="convertedAmount !== null">

Converted Amount: {{ convertedAmount }} {{ toCurrency }}

</p>

</div>

</template>

<script>

export default {

data() {

return {

amount: '',

fromCurrency: 'USD',

toCurrency: 'EUR',

convertedAmount: null,

exchangeRates: {

USD: {

EUR: 0.85,

GBP: 0.73,

INR: 75.32,

},

EUR: {

USD: 1.18,

GBP: 0.86,

INR: 88.68,

},

GBP: {

USD: 1.38,

EUR: 1.16,

INR: 103.36,

},

INR: {

USD: 0.013,

EUR: 0.011,

GBP: 0.0097,

},

},

};

},

methods: {

convertCurrency() {

const exchangeRate = this.exchangeRates[this.fromCurrency][this.toCurrency];

const converted = this.amount \* exchangeRate;

this.convertedAmount = converted.toFixed(2);

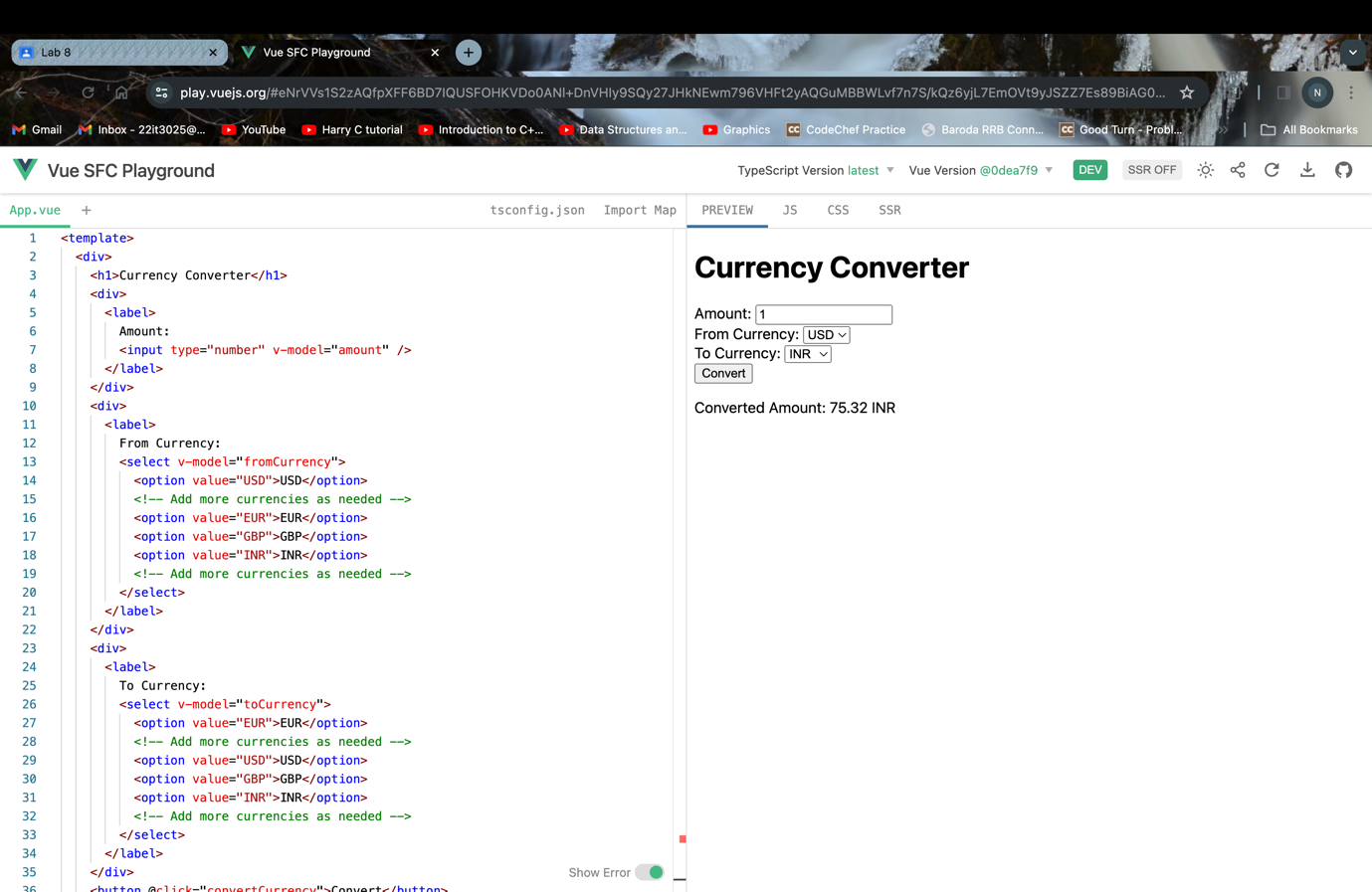
},

},

};

</script>

OUTPUT



T2:

<template>

<div class="stopwatch">

<h1>Stopwatch</h1>

<div class="timer">{{ formatTime }}</div>

<div class="buttons">

<button @click="startStopwatch" v-if="!isRunning">Start</button>

<button @click="pauseStopwatch" v-if="isRunning">Pause</button>

<button @click="resetStopwatch">Reset</button>

</div>

</div>

</template>

<script>

export default {

data() {

return {

isRunning: false,

startTime: null,

currentTime: 0,

timerInterval: null,

};

},

computed: {

formatTime() {

const minutes = Math.floor(this.currentTime / 60);

const seconds = this.currentTime % 60;

return `${String(minutes).padStart(2, '0')}:${String(seconds).padStart(2, '0')}`;

},

},

methods: {

startStopwatch() {

this.isRunning = true;

this.startTime = Date.now() - this.currentTime \* 1000;

this.timerInterval = setInterval(() => {

this.currentTime = Math.floor((Date.now() - this.startTime) / 1000);

}, 1000);

},

pauseStopwatch() {

this.isRunning = false;

clearInterval(this.timerInterval);

},

resetStopwatch() {

this.isRunning = false;

clearInterval(this.timerInterval);

this.currentTime = 0;

},

},

destroyed() {

clearInterval(this.timerInterval); // Cleanup interval when component is destroyed

},

};

</script>

OUTPUT

