1. What is Express.js ?

Ans: Express.js is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. This is the most popular framework as of now for Node.js.

Express.js is a web application framework for Node.js. It provides various features that make web application development fast and easy which otherwise takes more time using only Node.js.

1. Create Localhost server using express.

Ans:

Const express = require(‘express’);

var app = express();

App.get(‘/’,(req,res) =>={

res.send(‘Hello, World !’);

});

app.listen(3000);

1. Create basic “Hello World” app using node and express

Ans:

Const express = require(‘express’);

var app = express();

App.get(‘/’,(req,res) =>={

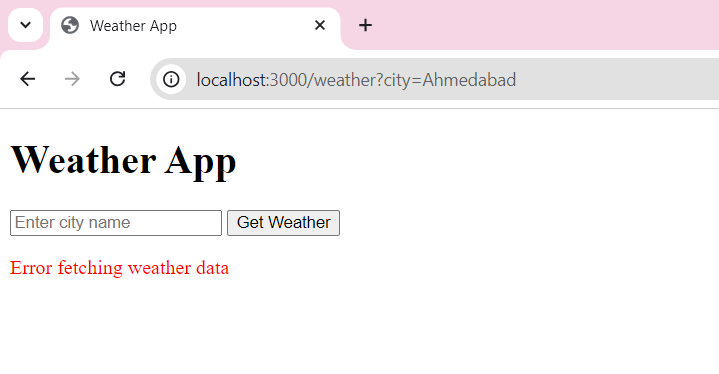
res.send(‘Hello, World !’);

});

app.listen(3000);

1. Create weather App.

Ans:

* Server.js
* const express = require('express');
* const hbs = require('hbs');
* const axios = require('axios');
* const path = require('path');
* const app = express();
* const port = 3000;
* // Set up Handlebars
* app.set('view engine', 'hbs');
* app.set('views', path.join(\_\_dirname, 'views'));
* // Set up static files
* app.use(express.static(path.join(\_\_dirname, 'public')));
* // Route to render the home page
* app.get('/', (req, res) => {
* res.render('index', { title: 'Weather App' });
* });
* // Route to get weather data by city
* app.get('/weather', async (req, res) => {
* const { city } = req.query;
* if (!city) {
* return res.render('index', { title: 'Weather App', error: 'City is required' });
* }
* try {
* const weatherResponse = await axios.get(`https://api.openweathermap.org/data/2.5/weather`, {
* params: {
* q: city,
* appid: '4f0a8998e5d1984a6324c53792d2dba02e66474851cbcb9e4871323e96055011', // Replace with your API key
* units: 'metric'
* }
* });
* if (weatherResponse.data.cod === '404') {
* return res.render('index', { title: 'Weather App', error: 'City not found' });
* }
* const weatherData = weatherResponse.data;
* res.render('weather', {
* title: 'Weather App',
* city: weatherData.name,
* weather: weatherData.weather[0].description,
* temperature: weatherData.main.temp
* });
* } catch (error) {
* console.error('Error fetching weather data:', error);
* res.render('index', { title: 'Weather App', error: 'Error fetching weather data' });
* }
* });
* app.listen(port, () => {
* console.log(`Server is running at http://localhost:${port}`);
* });
* View/weather.hbs
* <!DOCTYPE html>
* <html lang="en">
* <head>
* <meta charset="UTF-8">
* <meta name="viewport" content="width=device-width, initial-scale=1.0">
* <title>{{title}}</title>
* </head>
* <body>
* <h1>{{title}}</h1>
* <form action="/weather" method="get">
* <input type="text" name="city" placeholder="Enter city name" required>
* <button type="submit">Get Weather</button>
* </form>
* <h2>Weather in {{city}}</h2>
* <p>Condition: {{weather}}</p>
* <p>Temperature: {{temperature}}°C</p>
* <a href="/">Back</a>
* </body>
* </html>
* View/index.js
* <!DOCTYPE html>
* <html lang="en">
* <head>
* <meta charset="UTF-8">
* <meta name="viewport" content="width=device-width, initial-scale=1.0">
* <title>{{title}}</title>
* </head>
* <body>
* <h1>{{title}}</h1>
* <form action="/weather" method="get">
* <input type="text" name="city" placeholder="Enter city name" required>
* <button type="submit">Get Weather</button>
* </form>
* {{#if error}}
* <p style="color: red;">{{error}}</p>
* {{/if}}
* </body>
* </html>
* Run cmd -> nodemon server.js
* 

1. Create world clock app.

Ans:

* Server.js

const express = require('express');

const moment = require('moment-timezone');

const app = express();

const port = 3000;

// Serve static files from the 'public' directory

app.use(express.static('public'));

// Mapping of countries to their time zones

const countryTimeZones = {

'United States': [

'America/New\_York', 'America/Chicago', 'America/Denver', 'America/Los\_Angeles'

],

'United Kingdom': ['Europe/London'],

'Japan': ['Asia/Tokyo'],

'Australia': ['Australia/Sydney', 'Australia/Melbourne'],

// Add more countries and their time zones as needed

};

// Route to get time for a specific time zone

app.get('/time/:zone', (req, res) => {

const timeZone = req.params.zone;

const validTimeZones = moment.tz.names();

if (validTimeZones.includes(timeZone)) {

const time = moment().tz(timeZone).format('YYYY-MM-DD HH:mm:ss');

res.json({ time, timeZone });

} else {

res.status(400).json({ error: 'Invalid time zone' });

}

});

// Route to get time zones by country

app.get('/timezones/:country', (req, res) => {

const country = req.params.country;

const timeZones = countryTimeZones[country];

if (timeZones) {

res.json({ timeZones });

} else {

res.status(404).json({ error: 'Country not found' });

}

});

app.listen(port, () => {

console.log(`Server is running at http://localhost:${port}`);

});

* Public/index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>World Clock</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            display: flex;

            flex-direction: column;

            align-items: center;

            justify-content: center;

            height: 100vh;

            margin: 0;

            background-color: #f4f4f4;

        }

        select, button {

            margin: 10px;

            padding: 10px;

            font-size: 16px;

        }

        #time {

            font-size: 24px;

            margin: 20px;

        }

    </style>

</head>

<body>

    <h1>World Clock</h1>

    <select id="country" onchange="loadTimeZones()">

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

        <option value="United States">United States</option>

        <option value="United Kingdom">United Kingdom</option>

        <option value="Japan">Japan</option>

        <option value="Australia">Australia</option>

        <!-- Add more countries as needed -->

    </select>

    <select id="timezone" disabled>

        <option value="">Select Time Zone</option>

    </select>

    <button onclick="getTime()">Get Time</button>

    <div id="time"></div>

    <script>

        function loadTimeZones() {

            const country = document.getElementById('country').value;

            const timezoneSelect = document.getElementById('timezone');

            if (country) {

                fetch(`/timezones/${encodeURIComponent(country)}`)

                    .then(response => {

                        if (!response.ok) {

                            throw new Error('Network response was not ok');

                        }

                        return response.json();

                    })

                    .then(data => {

                        const timeZones = data.timeZones;

                        timezoneSelect.innerHTML = '<option value="">Select Time Zone</option>';

                        timeZones.forEach(zone => {

                            const option = document.createElement('option');

                            option.value = zone;

                            option.textContent = zone;

                            timezoneSelect.appendChild(option);

                        });

                        timezoneSelect.disabled = false;

                    })

                    .catch(error => {

                        console.error('Error loading time zones:', error);

                        timezoneSelect.innerHTML = '<option value="">Select Time Zone</option>';

                        timezoneSelect.disabled = true;

                    });

            } else {

                timezoneSelect.innerHTML = '<option value="">Select Time Zone</option>';

                timezoneSelect.disabled = true;

            }

        }

        function getTime() {

            const timezone = document.getElementById('timezone').value;

            fetch(`/time/${encodeURIComponent(timezone)}`)

                .then(response => {

                    if (!response.ok) {

                        throw new Error('Network response was not ok');

                    }

                    return response.json();

                })

                .then(data => {

                    if (data.error) {

                        document.getElementById('time').innerText = data.error;

                    } else {

                        document.getElementById('time').innerText = `Current time in ${data.timeZone} is ${data.time}`;

                    }

                })

                .catch(error => {

                    console.error('Error fetching time:', error);

                    document.getElementById('time').innerText = 'An error occurred';

                });

        }

    </script>

</body>

</html>

* Cmd -> nodemon server.js

