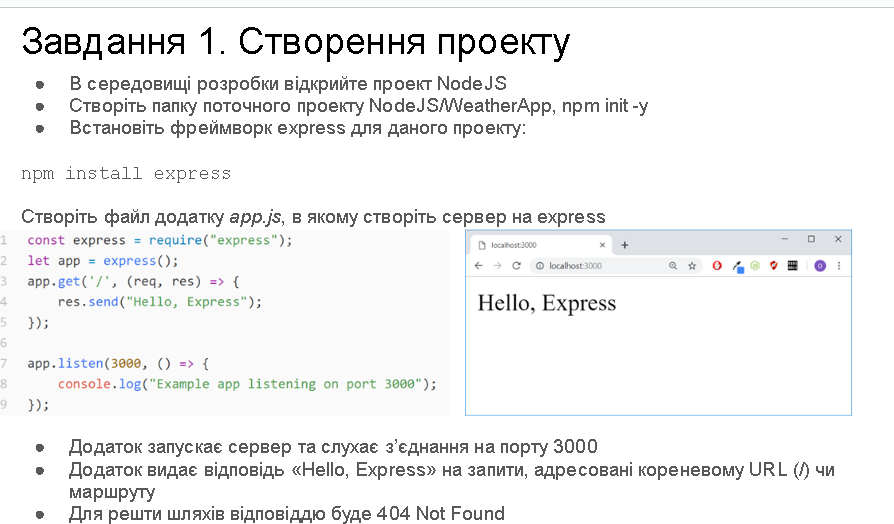
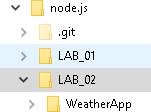
**ЛАБОРАТОРНА РОБОТА № 2**

Express, HBS. Проект “WeatherApp”

**Хід роботи:**



**Створення папки:**



**Прописуємо команду:**



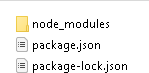
**Результат виконання :**



**Прописуємо команду:**



**Встановленно фреймворк**



**Лістинг програми:**

const express = require('express');

const hbs = require('hbs');

const ip = require('ipinfo');

const { Navigator } = require("node-navigator");

const navigator = new Navigator();

const list = require('./public/cities.json');

hbs.registerPartials(\_\_dirname + '/views/partials');

const app = express();

app.set('view engine', 'hbs');

app.use(express.static(\_\_dirname + '/public'));

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

    res.render('home.hbs');

});

app.get('/weather/:city', async (req, res) => {

    const { city } = req.params;

    const apiKey = '3606ad5fe1f5bba6b232b5b00f2499d9';

    fetch(`https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}`, {

        method: 'GET',

        mode: 'cors'

    })

        .then(response => response.json())

        .then(result => {

            console.log('Отримані дані від сервера OpenWeatherMap:', result); // Виводимо отримані дані на консоль для перевірки

            // Перевіряємо, чи містить отриманий об'єкт result очікувані властивості

            if (result.main && result.main.temp && result.weather && result.weather[0] && result.weather[0].icon && result.main.pressure && result.main.humidity) {

                // Якщо дані коректні, виконуємо рендерінг шаблону weather.hbs з отриманими даними

                res.render('weather.hbs', {

                    list,

                    temp: Math.round(result.main.temp - 273),

                    icon: `https://openweathermap.org/img/wn/${result.weather[0]['icon']}@4x.png`,

                    name: result.name,

                    pressure: result.main.pressure,

                    humidity: result.main.humidity

                });

            } else {

                // Якщо отримані дані не містять необхідні властивості, видаємо помилку 404

                console.error('Отримані дані не містять необхідні властивості');

                res.status(404).render("404.hbs");

            }

        })

        .catch(error => {

            // Обробляємо помилку

            console.error('Error fetching weather data:', error);

            res.status(404).render("404.hbs");

        });

});

app.get('/weather/', (req, res) => {

    ip((err, cLoc) => {

        res.redirect(`/weather/${cLoc.city}`);

    });

});

app.get('/weather-geo/', async (req, res) => {

    navigator.geolocation.getCurrentPosition(async (pos) => {

        const apiKey = '3606ad5fe1f5bba6b232b5b00f2499d9';

        fetch(`https://api.openweathermap.org/data/2.5/onecall?lat=${pos.latitude}&lon=${pos.longitude}&exclude=hourly,daily&appid=${apiKey}`, {

            method: 'GET',

            mode: 'cors'

        })

            .then(response => response.json())

            .then(result => {

                res.render('weather.hbs', {

                    list,

                    temp: Math.round(result.current.temp - 273),

                    icon: `https://openweathermap.org/img/wn/${result.current.weather[0]['icon']}@4x.png`,

                    name: 'ваших поточних координатах',

                    pressure: result.current.pressure,

                    humidity: result.current.humidity

                });

            })

            .catch(error => {

                console.error('Error fetching weather data:', error);

                res.status(404).render("404.hbs");

            });

    });

});

app.use((req, res, next) => {

    res.status(404).render("404.hbs");

});

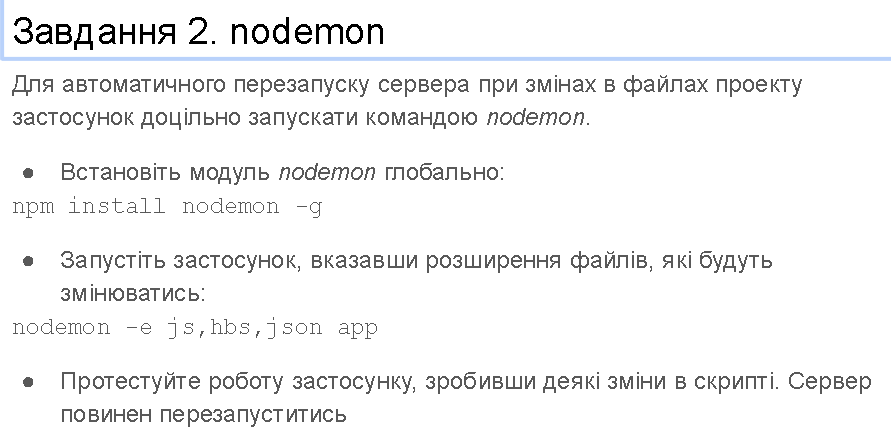
app.listen(3000, () => {

    console.log("Listening on port 3000");

});







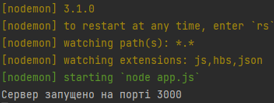
**Встановлення nodemon глобально:**

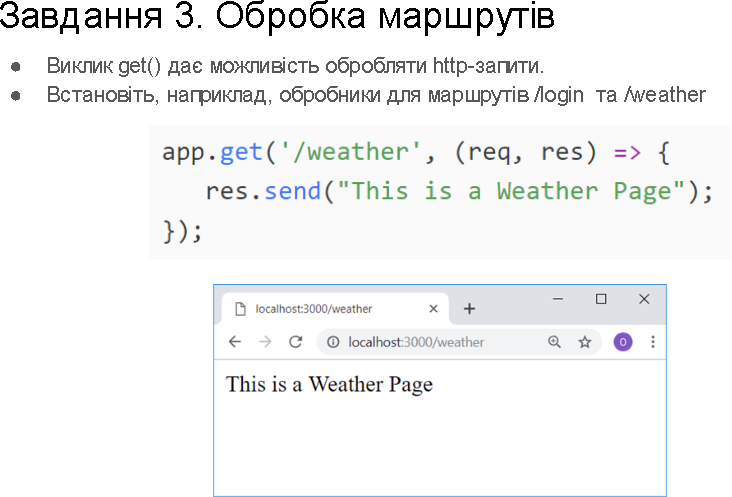


**Вводимо команду:**

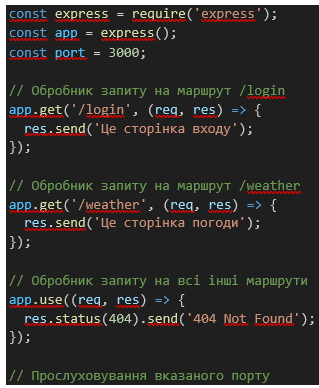


**Результат виконання програми:**

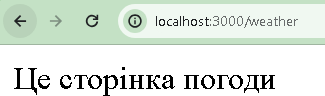
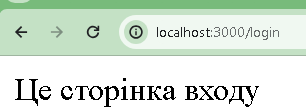


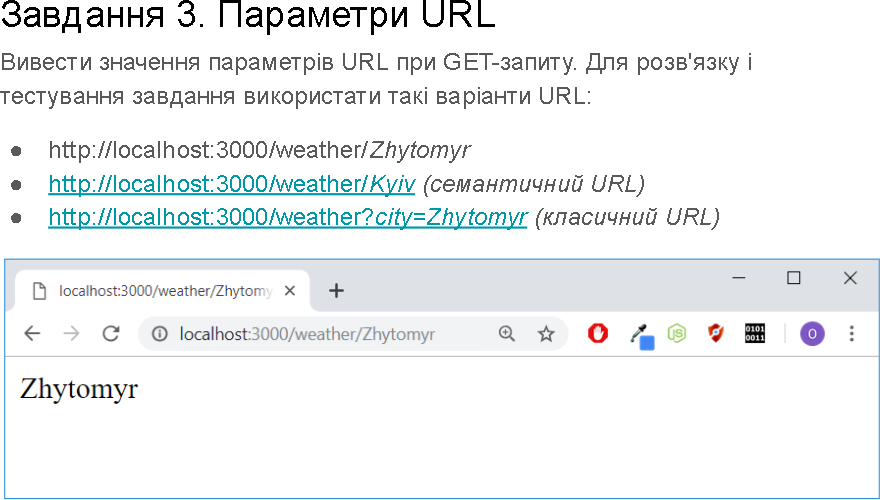


**Лістинг програми:**

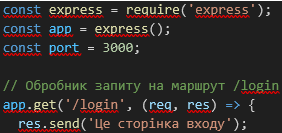


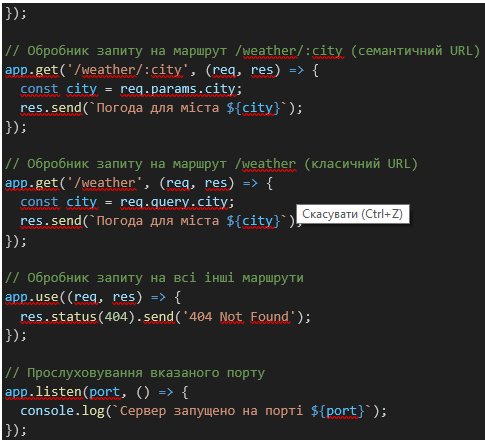




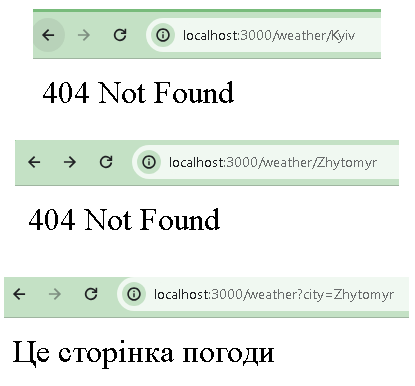


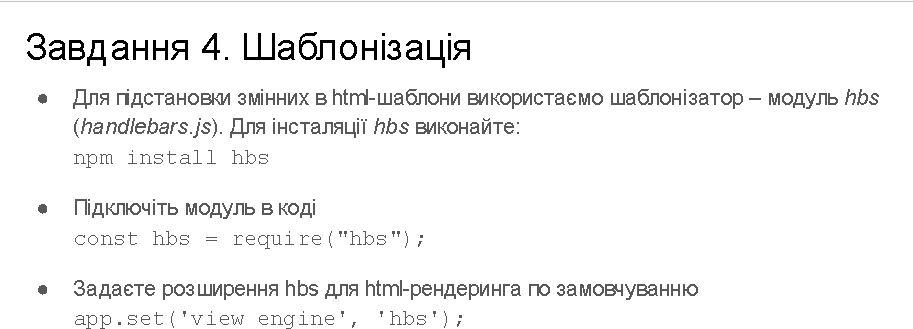
**Лістинг програми:**



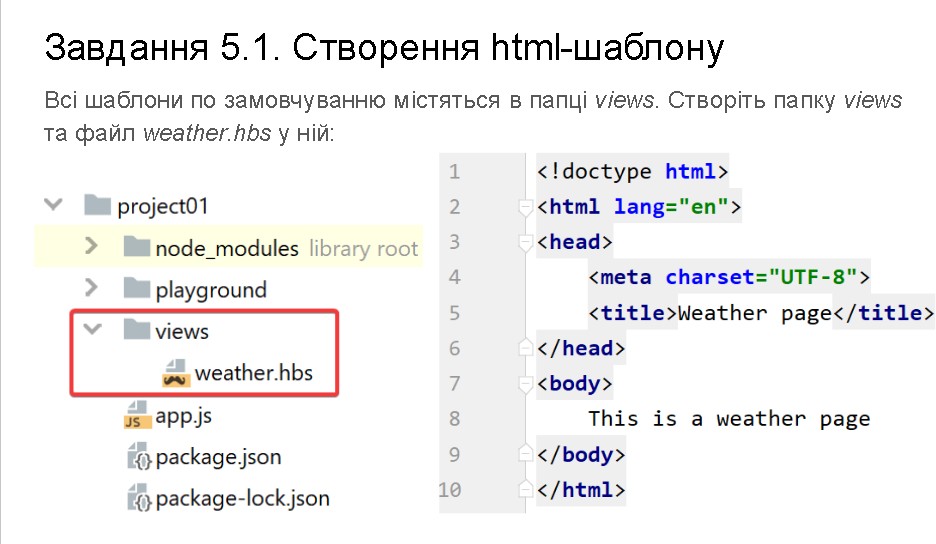


**Результат виконання програми:**











**Лістинг програми:**

{{> header}}

<div id="container" class="row mt-5">

    <div class="row">

         <div id="cities" class="col-6 col-md-4 col-lg-3">

                <div>

                    {{#each list}}

                        <a class="city list-group-item list-group-item-action" href="/weather/{{this}}">{{ this }}</a>

                    {{/each}}

                </div>

                <form method="get" action="/weather-geo/">

                    <button  id="btnCoords" class="btn btn-warning btn-lg">Отримати погаду по координатах</button>

                </form>

         </div>

         <div class="card" style="width: 950px">

            <div class="card-body">

                    <h3 class="card-title" id="currentCity">Погода у  {{ name }}</h3>

                    <img src="{{icon}}" id="icon" alt="">

                    <table class="table table-striped">

                            <tr>

                                    <th>Тиск</th>

                                    <td id ="pressure"> {{pressure}} </td>

                            </tr>

                            <tr>

                                    <th>Вологість</th>

                                    <td id ="humidity"> {{humidity}} </td>

                            </tr>

                            <tr>

                                    <th>Температура</th>

                                    <td id="temperature">{{temp}}</td>

                            </tr>

                    </table>

            </div>

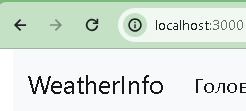
         </div>

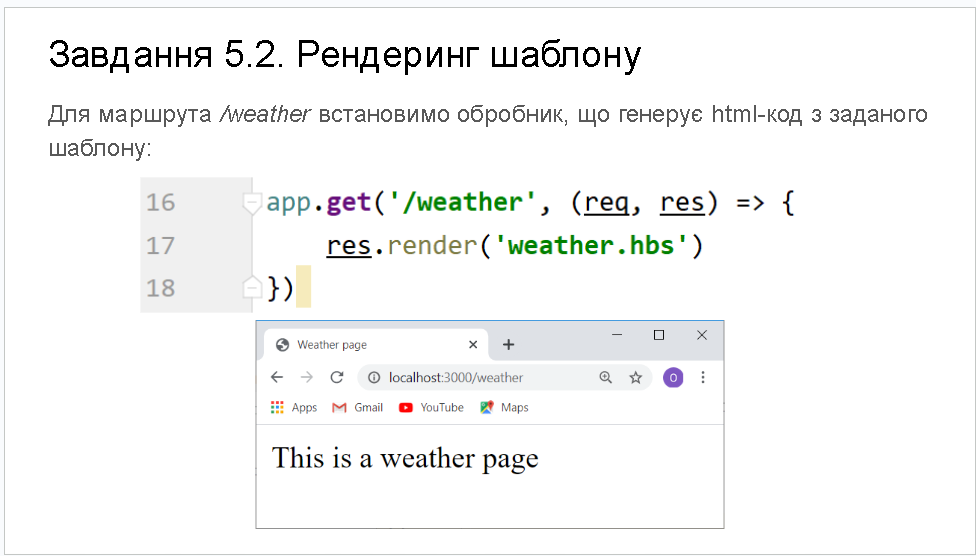
    </div>

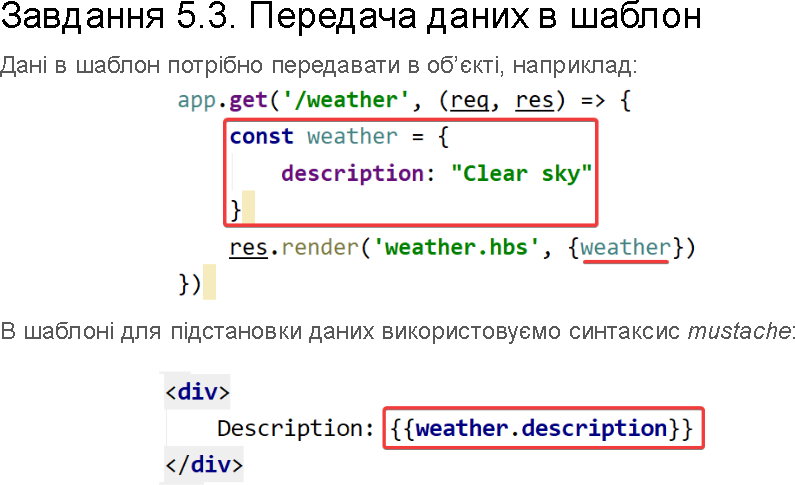
</div>

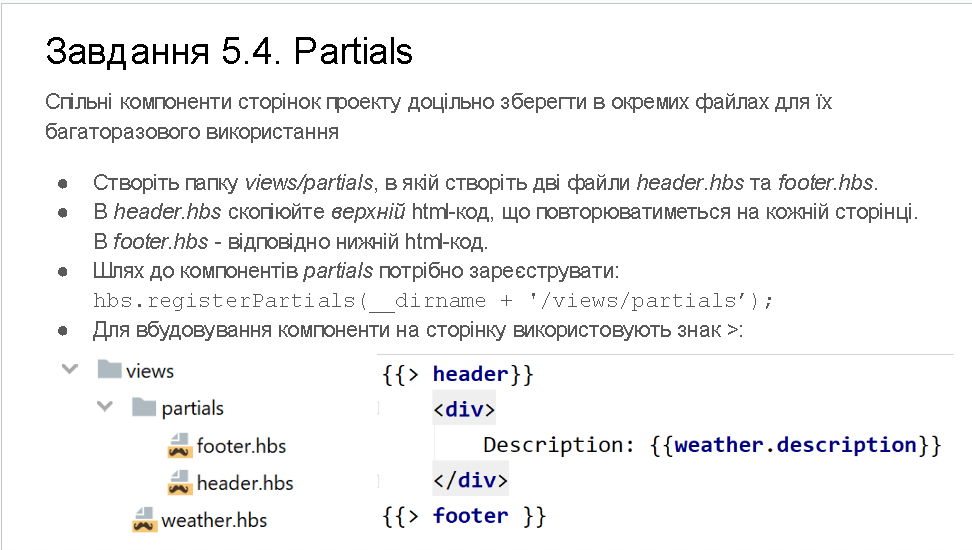
{{> footer}}

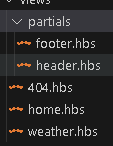
**Результат виконання програми:**





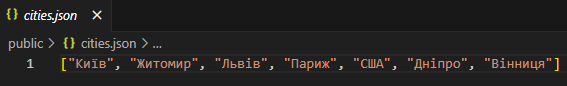








**Json файл:**



**Header.hbs:**

<!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">

    <title>Сторінка погоди</title>

    <link rel="stylesheet" href='/css/styles.css'>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-BmbxuPwQa2lc/FVzBcNJ7UAyJxM6wuqIj61tLrc4wSX0szH/Ev+nYRRuWlolflfl" crossorigin="anonymous">

</head>

<body>

<div class="container-fluid">

    <nav class="navbar navbar-expand-lg navbar-light bg-light">

        <div class="container-fluid">

            <a class="navbar-brand" href="/">WeatherInfo</a>

            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

                <span class="navbar-toggler-icon"></span>

            </button>

            <div class="collapse navbar-collapse" id="navbarSupportedContent">

                <ul class="navbar-nav me-auto mb-2 mb-lg-0">

                    <li class="nav-item">

                        <a class="nav-link active" aria-current="page"

href="/">Головна</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link" href="/weather">Погода</a>

                    </li>

                </ul>

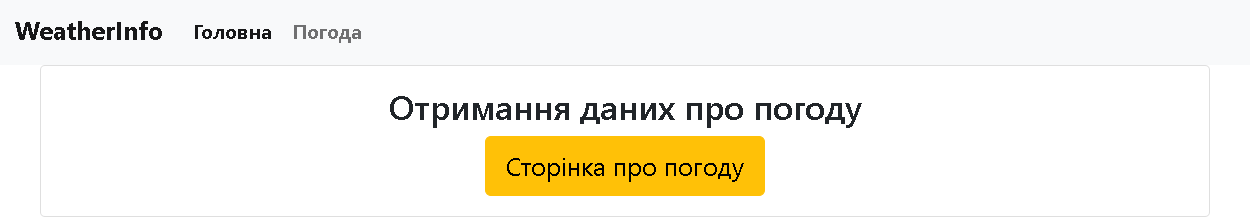
            </div>

        </div>

    </nav>

</div>

**Результат**:



**404.hbs:**

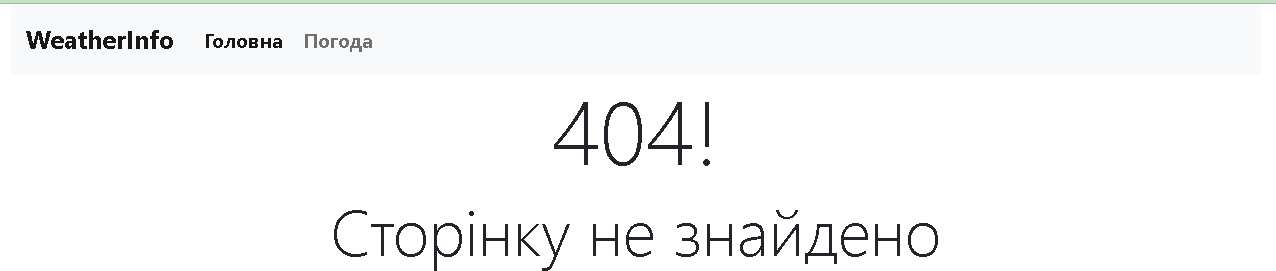
{{> header}}

<h1 class="display-1 text-center">404!</h1>

<h1 class="display-4 text-center">Сторінку не знайдено</h1>

{{> footer}}

**Результат**:



**Home.hbs:**

{{> header}}

<div class="container">

    <div class="card">

        <div class="card-body text-center">

            <h3>Отримання даних про погоду</h3>

            <a href="/weather" class="btn btn-warning btn-lg" role="button">Сторінка про погоду</a>

        </div>

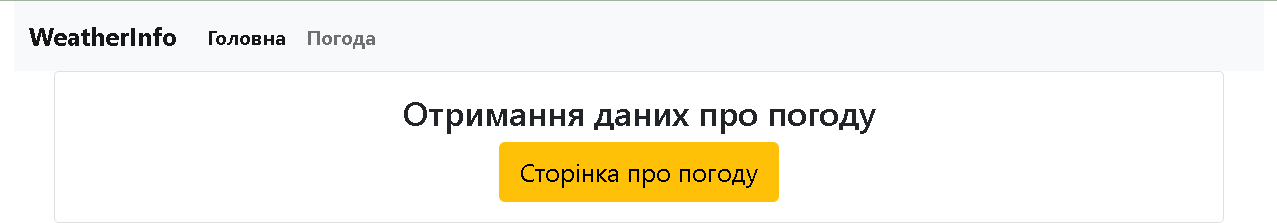
    </div>

</div>

{{> footer}}

**Результат:**

Коли ми перейшли нажавши на кнопку Головна, у нас є вибір, подивитись інфо- рмацію нажавши «сторінка про погоду» або повернутись по верхній панелі назад або обравши іншу кнопку.



**Weather.hbs:**

{{> header}}

<div id="container" class="row mt-5">

    <div class="row">

         <div id="cities" class="col-6 col-md-4 col-lg-3">

                <div>

                    {{#each list}}

                        <a class="city list-group-item list-group-item-action" href="/weather/{{this}}">{{ this }}</a>

                    {{/each}}

                </div>

                <form method="get" action="/weather-geo/">

                    <button  id="btnCoords" class="btn btn-warning btn-lg">Отримати погаду по координатах</button>

                </form>

         </div>

         <div class="card" style="width: 950px">

            <div class="card-body">

                    <h3 class="card-title" id="currentCity">Погода у  {{ name }}</h3>

                    <img src="{{icon}}" id="icon" alt="">

                    <table class="table table-striped">

                            <tr>

                                    <th>Тиск</th>

                                    <td id ="pressure"> {{pressure}} </td>

                            </tr>

                            <tr>

                                    <th>Вологість</th>

                                    <td id ="humidity"> {{humidity}} </td>

                            </tr>

                            <tr>

                                    <th>Температура</th>

                                    <td id="temperature">{{temp}}</td>

                            </tr>

                    </table>

            </div>

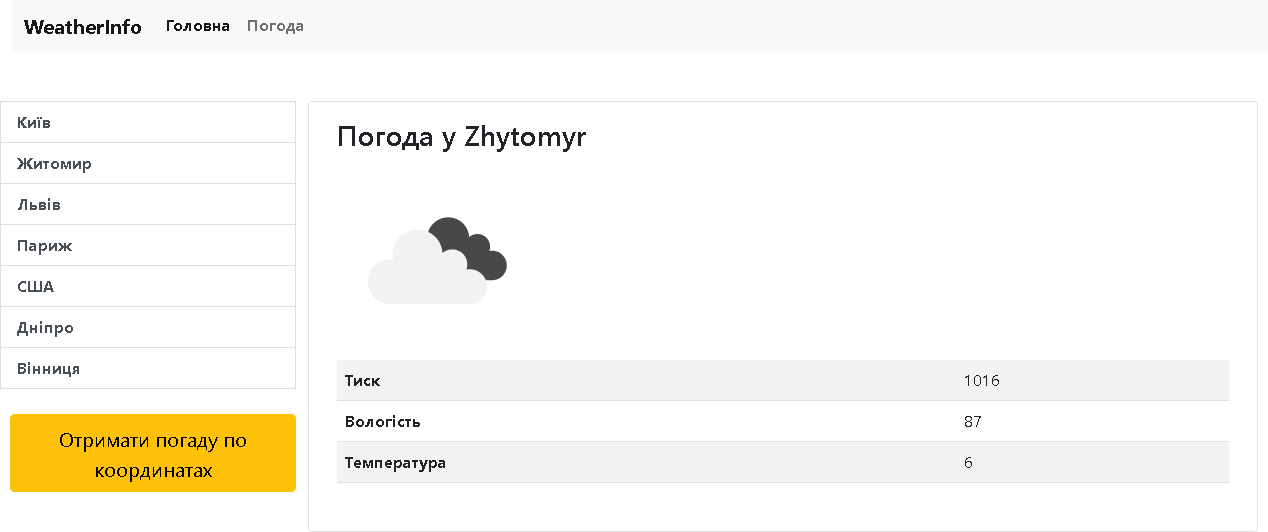
         </div>

    </div>

</div>

{{> footer}}

**Результат:**



Тут ми можемо передивлятись інформацію про погоду зліва по панельці нажимаючи на кожне місто або країну.

Також можна дізнатись яка погода по своїх координатах нажавши на кнопку.

App.js

const express = require('express');

const hbs = require('hbs');

const ip = require('ipinfo');

const { Navigator } = require("node-navigator");

const navigator = new Navigator();

const list = require('./public/cities.json');

hbs.registerPartials(\_\_dirname + '/views/partials');

const app = express();

app.set('view engine', 'hbs');

app.use(express.static(\_\_dirname + '/public'));

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

    res.render('home.hbs');

});

app.get('/weather/:city', async (req, res) => {

    const { city } = req.params;

    const apiKey = '3606ad5fe1f5bba6b232b5b00f2499d9';

    fetch(`https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}`, {

        method: 'GET',

        mode: 'cors'

    })

        .then(response => response.json())

        .then(result => {

            console.log('Отримані дані від сервера OpenWeatherMap:', result); // Виводимо отримані дані на консоль для перевірки

            // Перевіряємо, чи містить отриманий об'єкт result очікувані властивості

            if (result.main && result.main.temp && result.weather && result.weather[0] && result.weather[0].icon && result.main.pressure && result.main.humidity) {

                // Якщо дані коректні, виконуємо рендерінг шаблону weather.hbs з отриманими даними

                res.render('weather.hbs', {

                    list,

                    temp: Math.round(result.main.temp - 273),

                    icon: `https://openweathermap.org/img/wn/${result.weather[0]['icon']}@4x.png`,

                    name: result.name,

                    pressure: result.main.pressure,

                    humidity: result.main.humidity

                });

            } else {

                // Якщо отримані дані не містять необхідні властивості, видаємо помилку 404

                console.error('Отримані дані не містять необхідні властивості');

                res.status(404).render("404.hbs");

            }

        })

        .catch(error => {

            // Обробляємо помилку

            console.error('Error fetching weather data:', error);

            res.status(404).render("404.hbs");

        });

});

app.get('/weather/', (req, res) => {

    ip((err, cLoc) => {

        res.redirect(`/weather/${cLoc.city}`);

    });

});

app.get('/weather-geo/', async (req, res) => {

    navigator.geolocation.getCurrentPosition(async (pos) => {

        const apiKey = '3606ad5fe1f5bba6b232b5b00f2499d9';

        fetch(`https://api.openweathermap.org/data/2.5/onecall?lat=${pos.latitude}&lon=${pos.longitude}&exclude=hourly,daily&appid=${apiKey}`, {

            method: 'GET',

            mode: 'cors'

        })

            .then(response => response.json())

            .then(result => {

                res.render('weather.hbs', {

                    list,

                    temp: Math.round(result.current.temp - 273),

                    icon: `https://openweathermap.org/img/wn/${result.current.weather[0]['icon']}@4x.png`,

                    name: 'ваших поточних координатах',

                    pressure: result.current.pressure,

                    humidity: result.current.humidity

                });

            })

            .catch(error => {

                console.error('Error fetching weather data:', error);

                res.status(404).render("404.hbs");

            });

    });

});

app.use((req, res, next) => {

    res.status(404).render("404.hbs");

});

app.listen(3000, () => {

    console.log("Listening on port 3000");

});

***Висновки:*** в ході виконання лабораторної роботи було створенно проект та опра- цьовано роботу з файлами, Express, HBS. Проект “WeatherApp”