

# NodeJS

Express, HBS. Проект “WeatherApp”

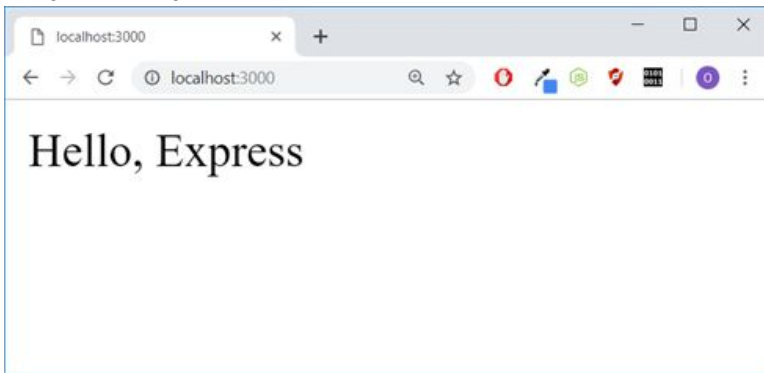
# Завдання 1. Створення проекту

- В середовищі розробки відкрийте проект NodeJS
- Створіть папку поточного проекту NodeJS/WeatherApp, `npm init -y`
- Встановіть фреймворк `express` для даного проекту:

```
npm install express
```

Створіть файл додатку `app.js`, в якому створіть сервер на `express`

```
1  const express = require("express");
2  let app = express();
3  app.get('/', (req, res) => {
4    res.send("Hello, Express");
5  });
6
7  app.listen(3000, () => {
8    console.log("Example app listening on port 3000");
9  });
```



- Додаток запускає сервер та слухає з'єднання на порту 3000
- Додаток видає відповідь «Hello, Express» на запити, адресовані кореневому URL (/) чи маршруту
- Для решти шляхів відповіддю буде 404 Not Found

## Завдання 2. nodemon

Для автоматичного перезапуску сервера при змінах в файлах проекту застосунок доцільно запускати командою *nodemon*.

- Встановіть модуль *nodemon* глобально:

```
npm install nodemon -g
```

- Запустіть застосунок, вказавши розширення файлів, які будуть змінюватись:

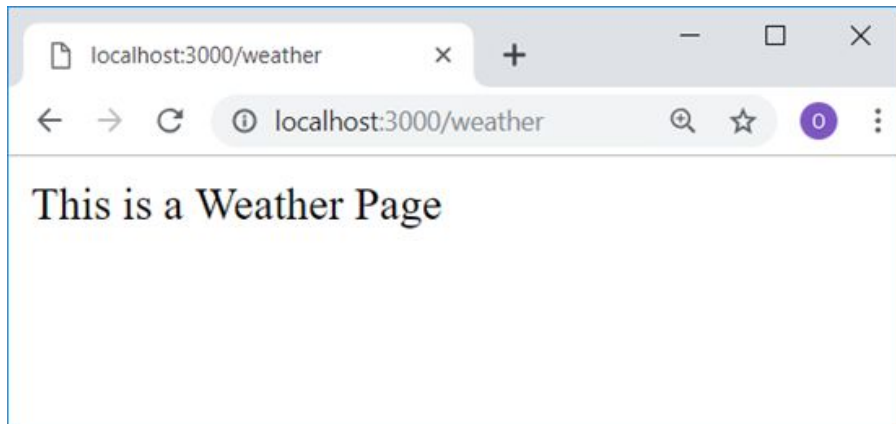
```
nodemon -e js,hbs,json app
```

- Протестуйте роботу застосунку, зробивши деякі зміни в скрипті. Сервер повинен перезапуститись

## Завдання 3. Обробка маршрутів

- Виклик `get()` дає можливість обробляти `http`-запити.
- Встановить, наприклад, обробники для маршрутів `/login` та `/weather`

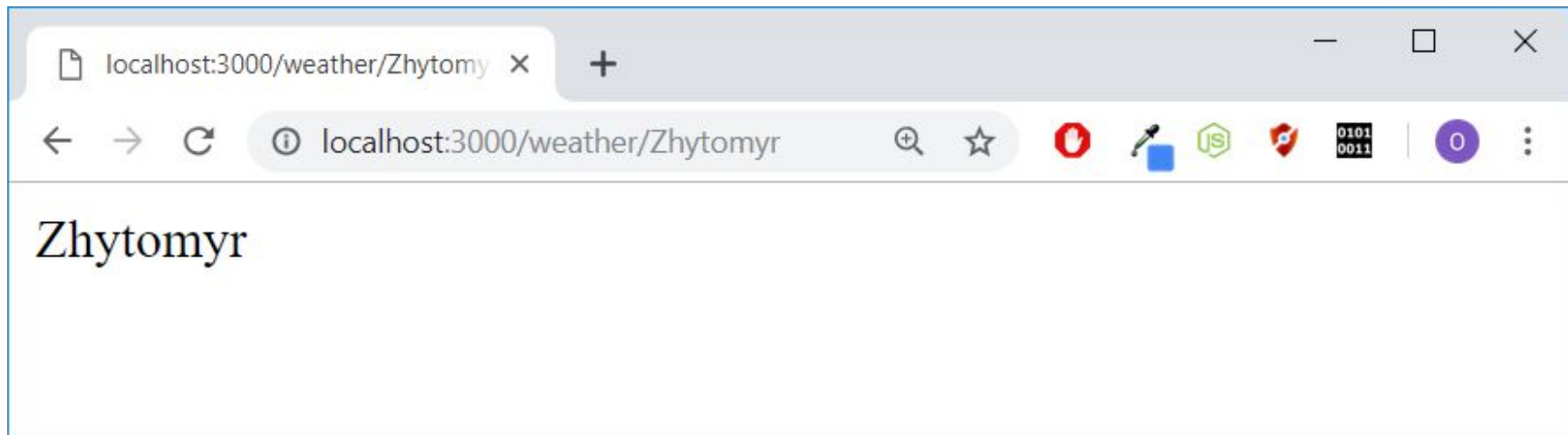
```
app.get('/weather', (req, res) => {  
    res.send("This is a Weather Page");  
});
```



## Завдання 3. Параметри URL

Вивести значення параметрів URL при GET-запиту. Для розв'язку і тестування завдання використати такі варіанти URL:

- `http://localhost:3000/weather/Zhytomyr`
- <http://localhost:3000/weather/Kyiv> (семантичний URL)
- <http://localhost:3000/weather?city=Zhytomyr> (класичний URL)



## Завдання 4. Шаблонізація

- Для підстановки змінних в html-шаблони використаємо шаблонізатор – модуль *hbs* (*handlebars.js*). Для інсталяції *hbs* виконайте:

```
npm install hbs
```

- Підключіть модуль в коді

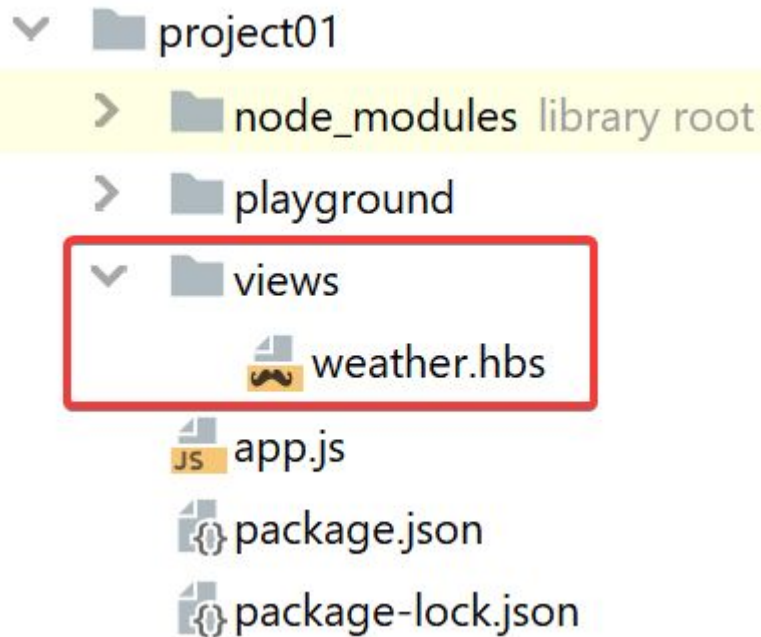
```
const hbs = require("hbs");
```

- Задаєте розширення *hbs* для html-рендеринга по замовчуванню

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

# Завдання 5.1. Створення html-шаблону

Всі шаблони по замовчуванню містяться в папці *views*. Створіть папку *views* та файл *weather.hbs* у ній:

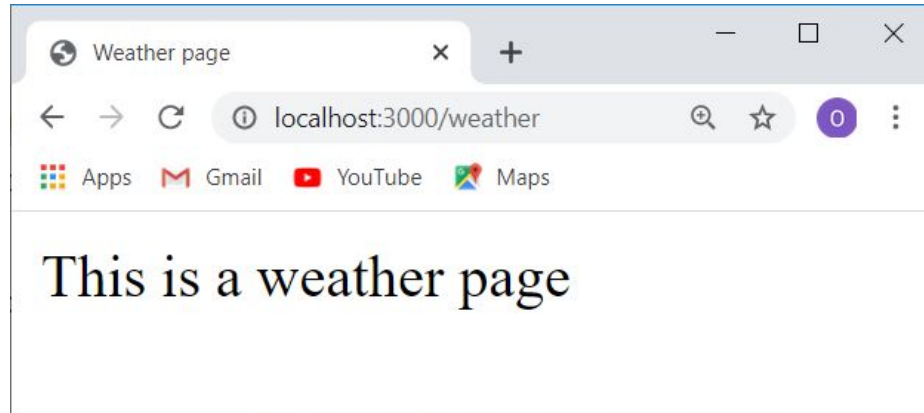


```
1  <!doctype html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Weather page</title>
6  </head>
7  <body>
8      This is a weather page
9  </body>
10 </html>
```

## Завдання 5.2. Рендеринг шаблону

Для маршрута `/weather` встановимо обробник, що генерує html-код з заданого шаблону:

```
16 app.get('/weather', (req, res) => {  
17   res.render('weather.hbs')  
18 })
```





## Завдання 5.3. Передача даних в шаблон

Дані в шаблон потрібно передавати в об'єкті, наприклад:

```
app.get('/weather', (req, res) => {  
  const weather = {  
    description: "Clear sky"  
  }  
  res.render('weather.hbs', {weather})  
})
```

В шаблоні для підстановки даних використовуємо синтаксис *mustache*:

```
<div>  
  Description: {{weather.description}}  
</div>
```

## Завдання 5.4. Partials

Спільні компоненти сторінок проекту доцільно зберегти в окремих файлах для їх багаторазового використання

- Створіть папку *views/partials*, в якій створіть дві файли *header.hbs* та *footer.hbs*.
- В *header.hbs* скопіюйте *верхній* html-код, що повторюватиметься на кожній сторінці. В *footer.hbs* - відповідно нижній html-код.

- Шлях до компонентів *partials* потрібно зареєструвати:


```
hbs.registerPartials(__dirname + '/views/partials');
```

- Для вбудовування компоненти на сторінку використовують знак `>`:

▼ views

▼ partials

 footer.hbs

 header.hbs

 weather.hbs

```
{{> header}}
```

```
| <div>
```

```
    Description: {{weather.description}}
```

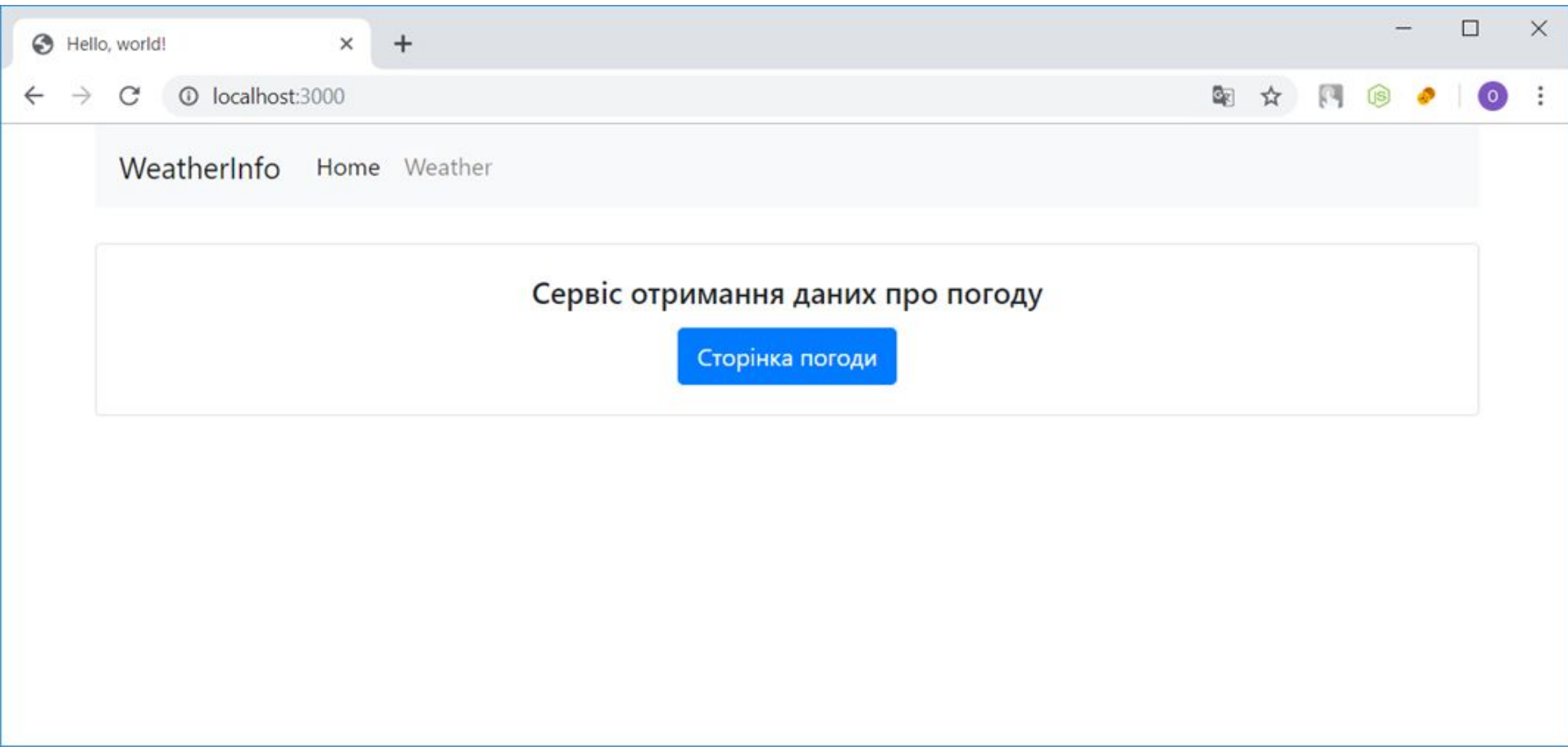
```
| </div>
```

```
{{> footer }}
```

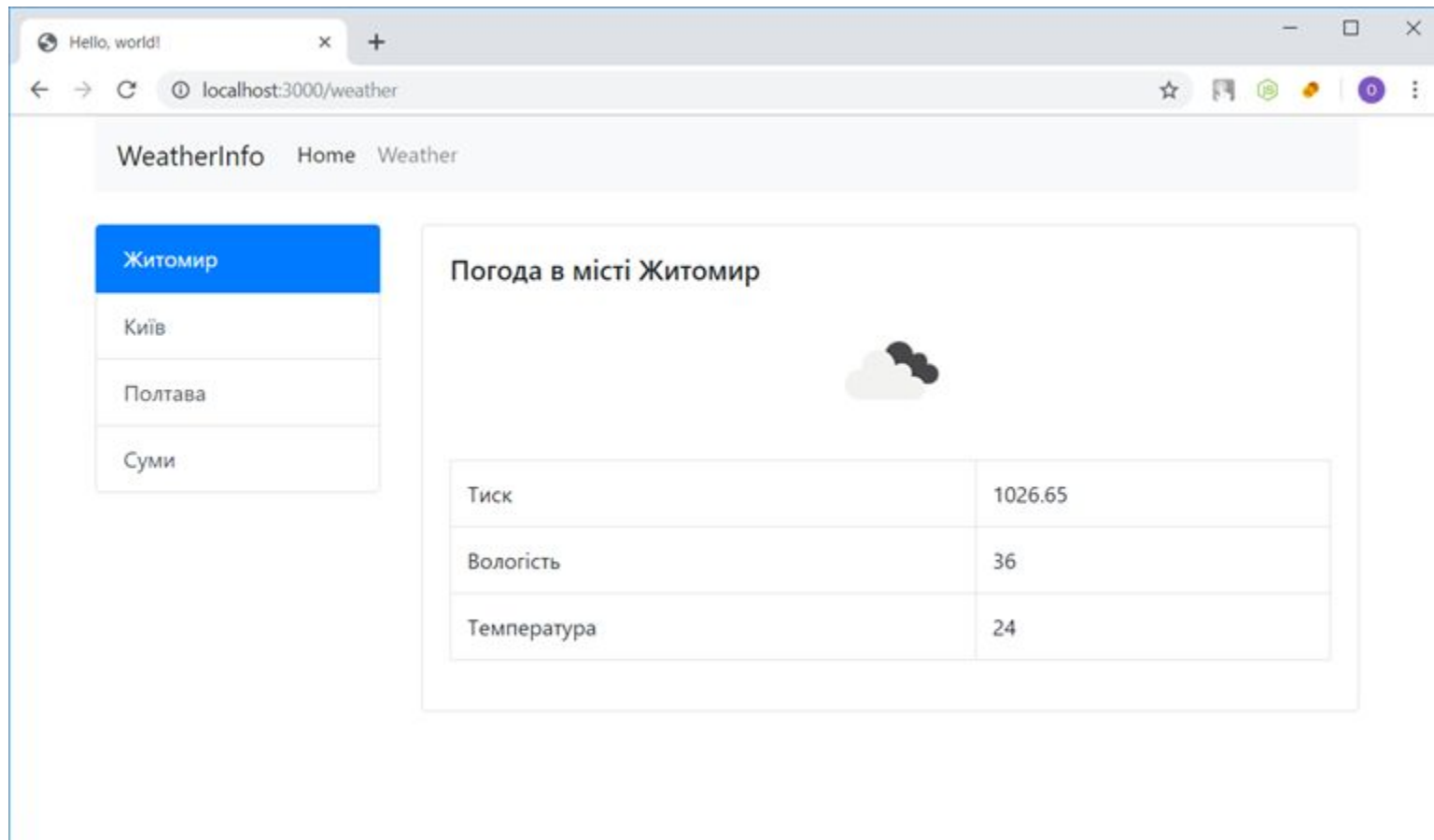
## Завдання 6. Розробити веб-застосунок для отримання даних про погоду

- Дані про міста повинні отримуватись із *json*-файлу та передаватись в *html*-шаблон у вигляді масиву
- В шаблоні потрібно сформувати меню посилань з назвами міст
- Формат рядка запиту для отримання даних про погоду:  
*/weather/{city}*, де *city* - назва вибраного міста
- Дані про погоду можна отримати відправкою запиту на *OpenWeatherMap*
- Advanced. Отримати дані про погоду в місцезнаходженні користувача за таким URI:  
*/weather/*

# Головна сторінка



# Сторінки з погодою



# Матеріали до роботи

Іструкція для виконання <https://youtu.be/mj3Md9OOMxY>