**Веб-програмування**

## **Лабораторна робота №3**

«Рефакторинг Bootstrap-лендінгу з використанням SCSS/SASS»

*Виконав Заяць Ілля, група ІПЗ-41*

**Мета:**

Освоїти основні концепції та прийоми роботи з препроцесором SCSS/SASS для оптимізації процесу написання стилів та підвищення ефективності розробки веб-сторінок.

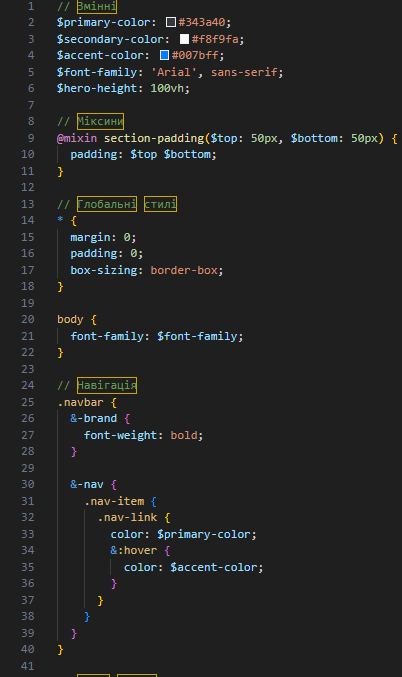
Хід роботи

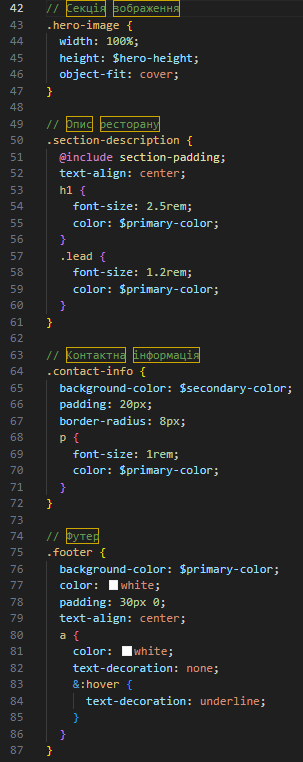
1.)- Виконати рефакторинг коду розробленого в попередній лабораторній роботі лендінгу, який буде використовувати основні можливості SCSS.

2.)- Розширити функціональність лендінгу, реалізувавши 2 додаткових завдання на вибір з використанням просунутих можливостей SCSS.

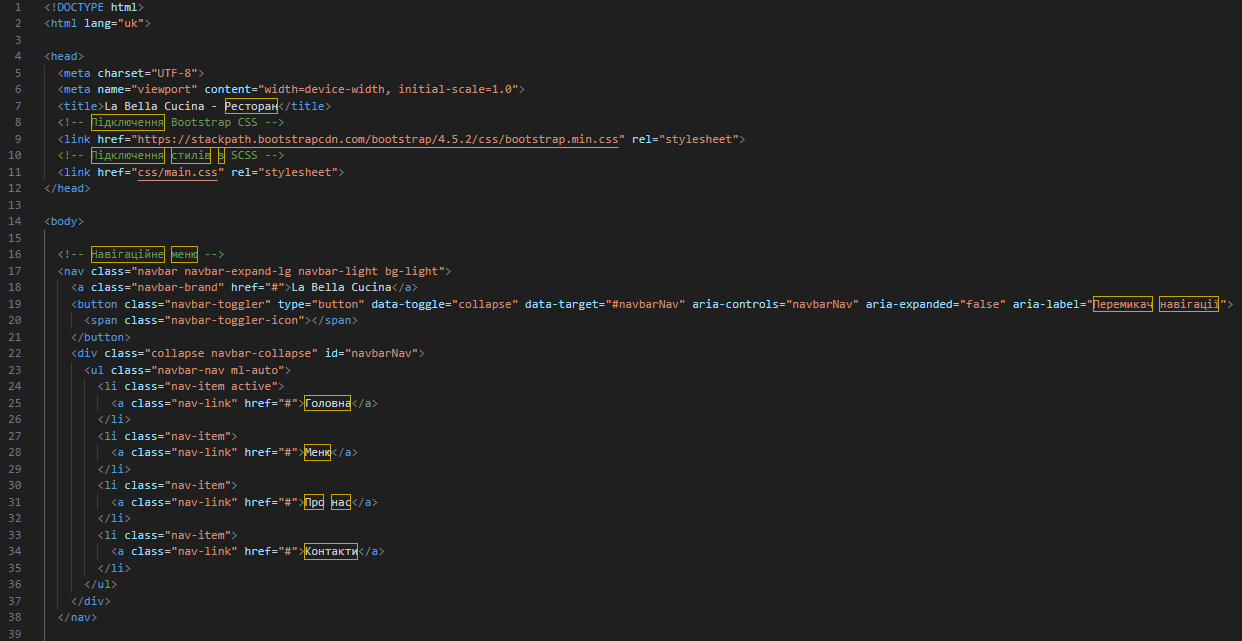
## *Завдання 1*

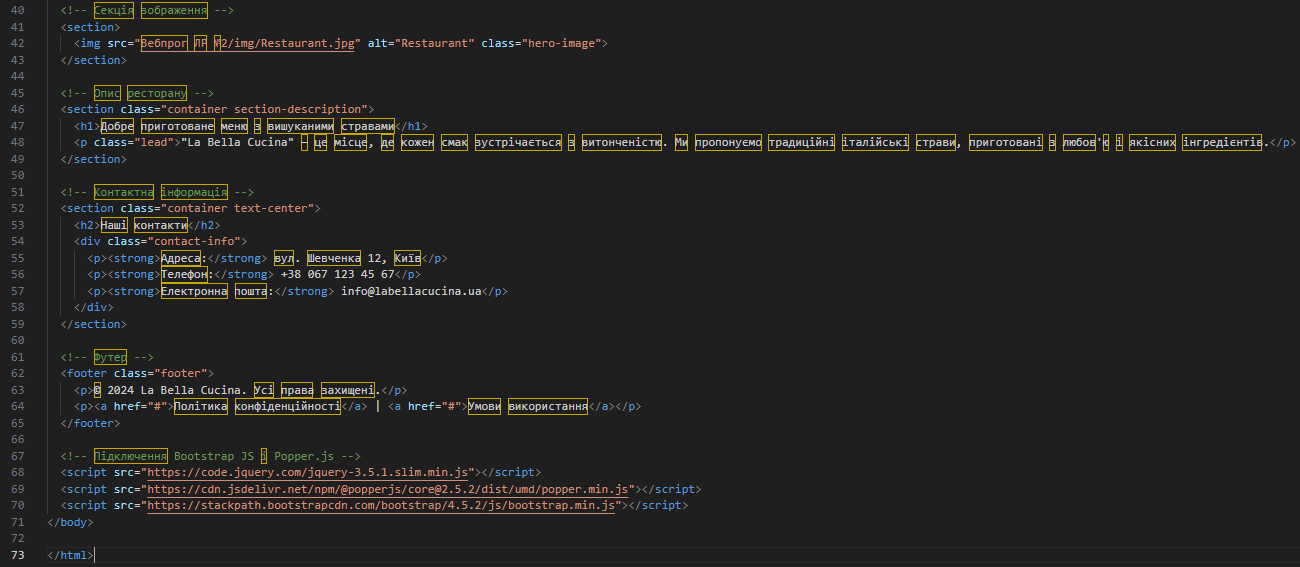
1.)- Потрібно виконати рефакторинг коду розробленого в попередній лабораторній роботі лендінгу, який буде використовувати основні можливості SCSS. Тут я коротко поясню структуру оновленого лендінгу. Змінні SCSS: Визначають основні кольори, шрифти та інші параметри для використання в стилях. Міксини: Створено міксин section-padding, щоб полегшити додавання відступів до різних секцій. Вкладеність: У SCSS ми використовуємо вкладеність для зручності й чистоти коду (наприклад, .navbar .nav-item .nav-link). Генерація CSS: Після компіляції SCSS в CSS, файл main.css буде містити згенеровані стилі, які підключаються до HTML. Зображення все ще не хоче під’єднуватись і відображатись, тому я вирішив не марнувати на нього час.





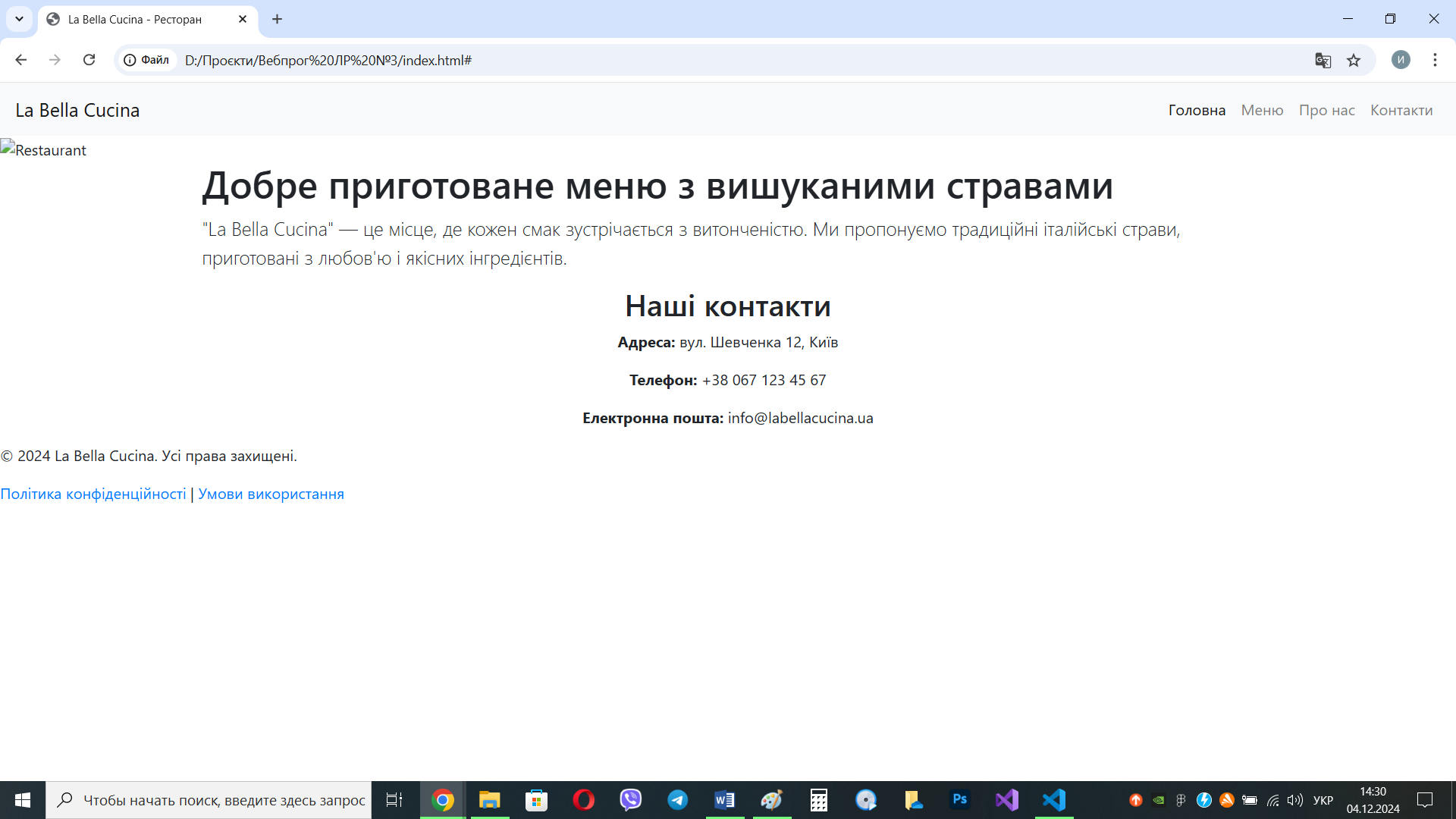
Файл styles.scss





Основний HTML-файл index.html

2.)- Результат виведення



## *Завдання 2:*

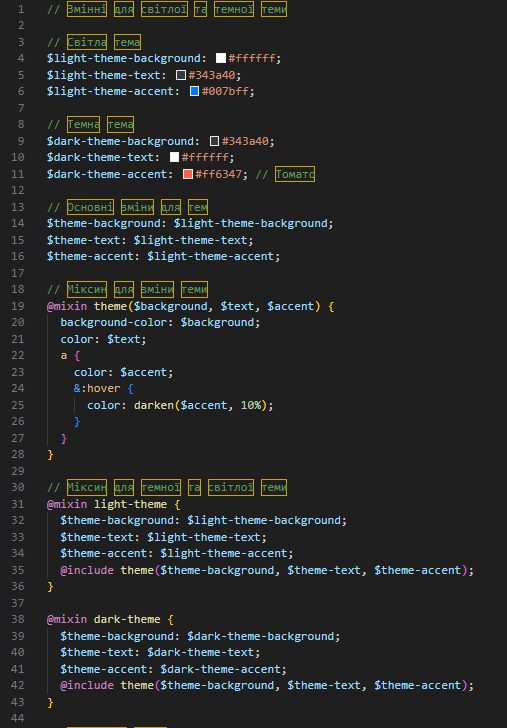
1.)- Я вибрав наступні додаткові завдання:

1. Створити систему змінних для світлої та темної теми. Реалізувати міксин для генерації стилів теми. Додати JavaScript для перемикання між темами.

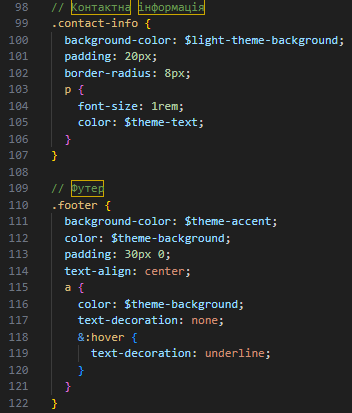
2. Створити файл \_functions.scss та реалізувати кілька корисних функцій SCSS (наприклад, для конвертації px в rem).

Я вирішив не розписувати про них окремо, а відразу про всі змінні і їх значення.

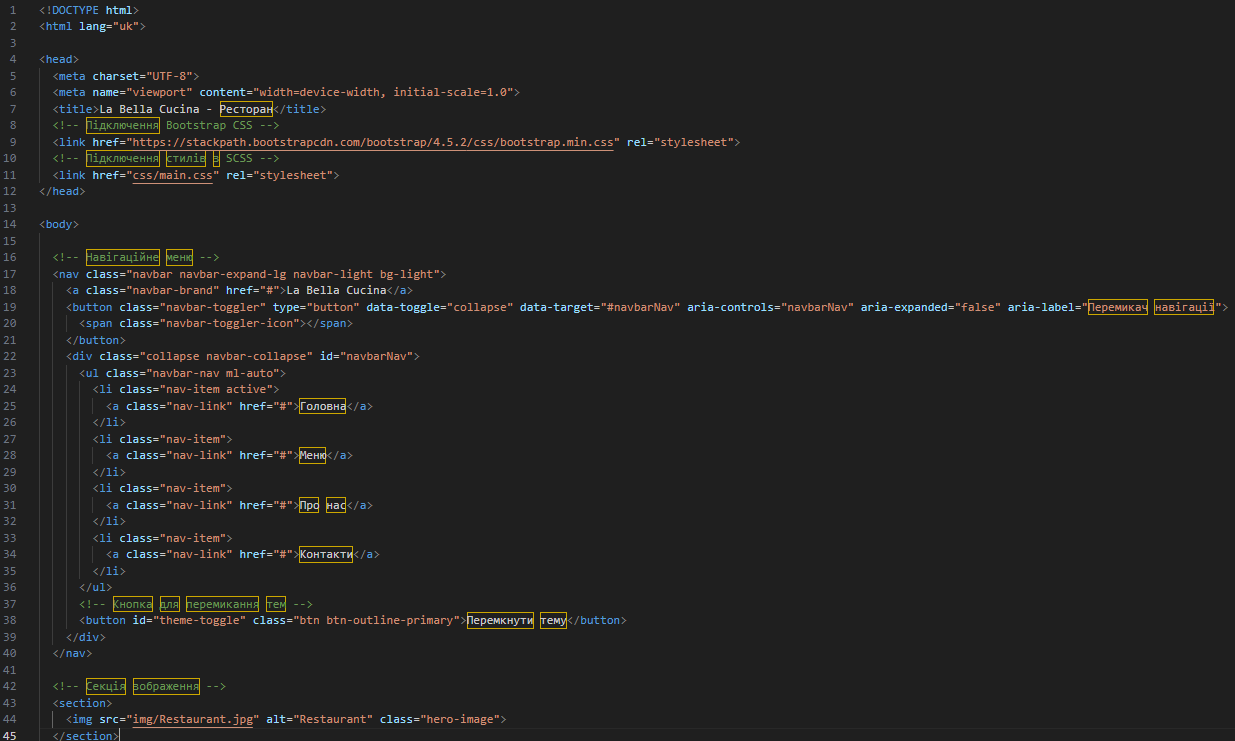
1. Спочатку визначимо змінні для кольорів у світлій та темній темах в SCSS. Створимо окремі змінні для кожної теми, а потім через міксин згенеруємо стилі теми.
2. Додамо міксини для теми та створимо JS для перемикання теми
3. Створимо файл \_functions.scss для корисних функцій. У ньому створимо функцію px-to-rem, за допомогою якої тепер, замість того, щоб використовувати пікселі безпосередньо в стилях, можна застосовувати цю функцію для перерахунку пікселів у rem.

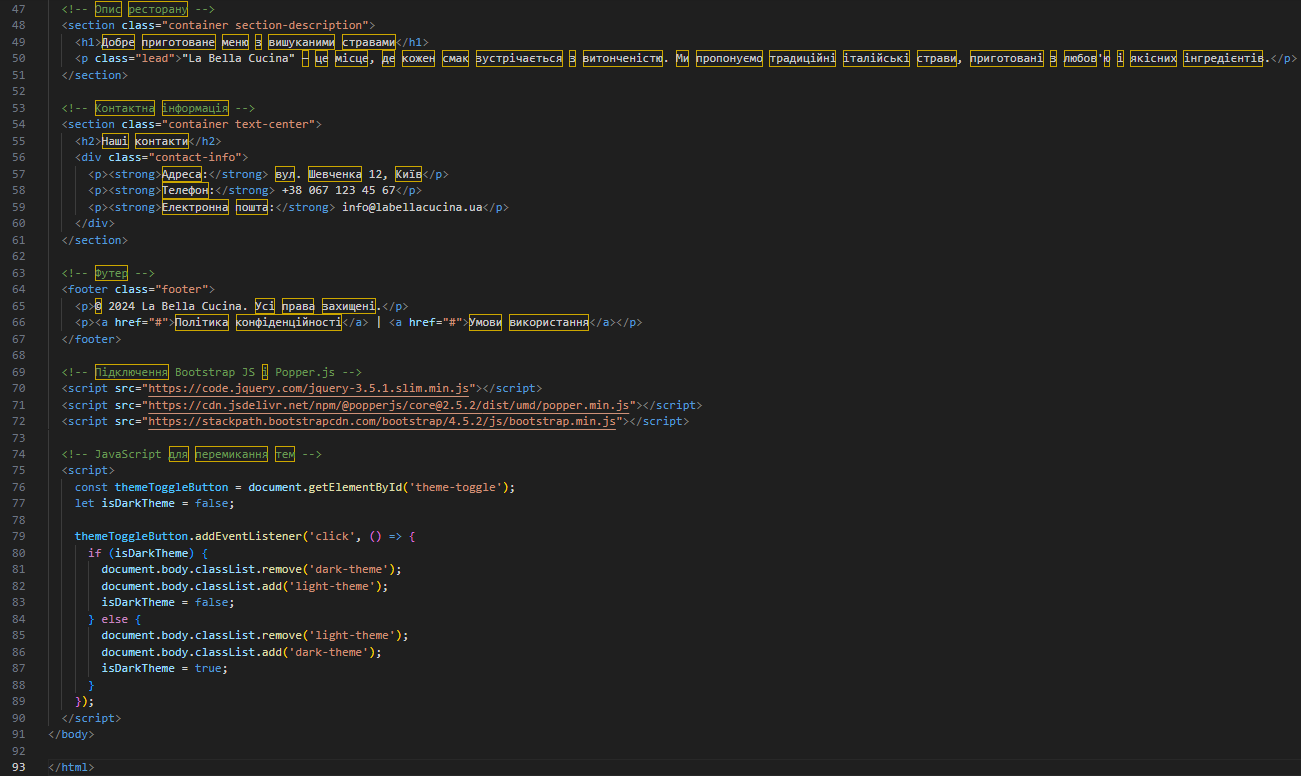




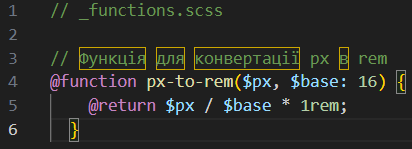


Оновлений файл styles.scss



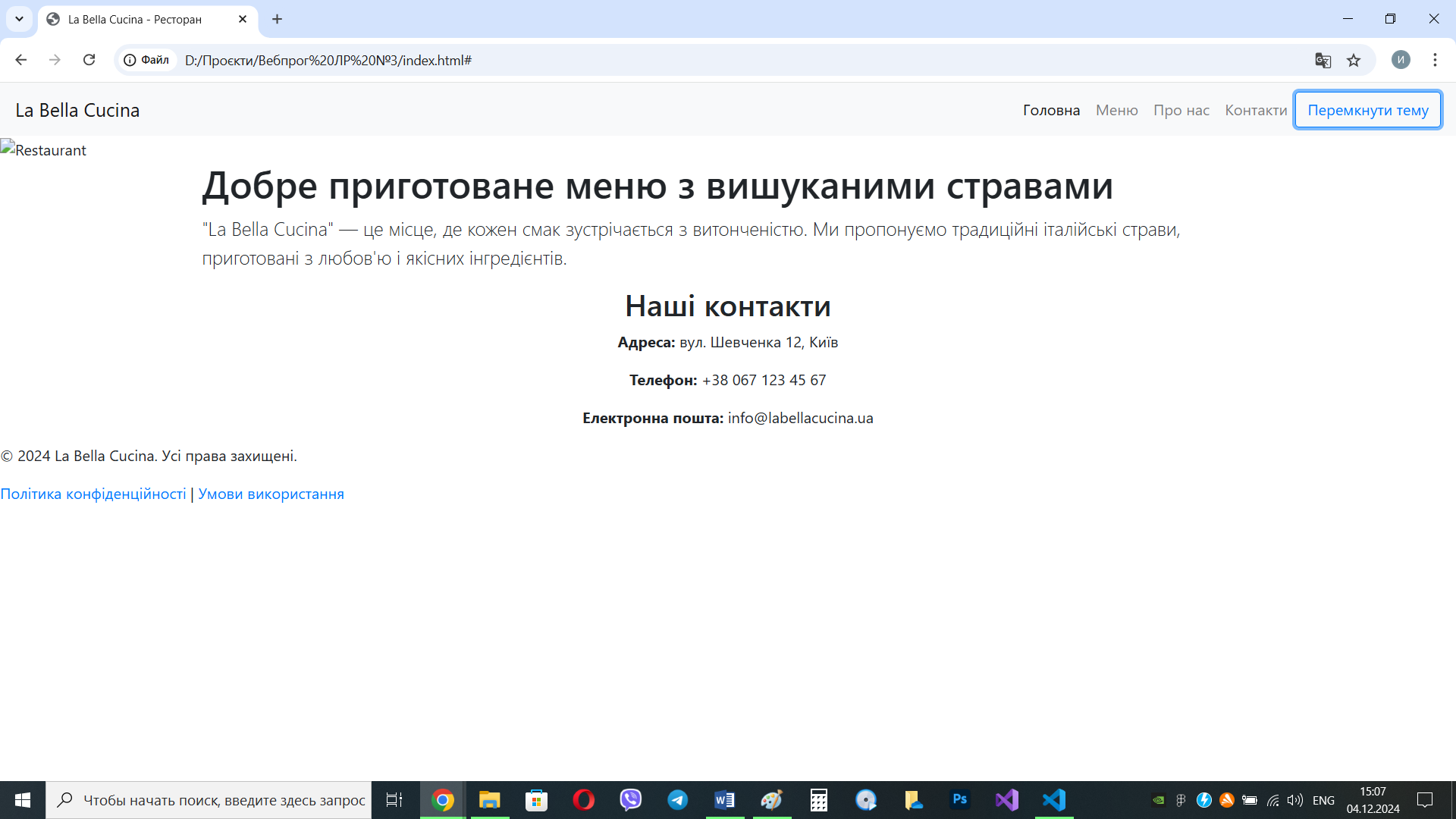


Оновлений HTML-файл index.html



Файл \_functions.scss

2.)- Результат виведення:



**Висновки:** код програми був оновлений з використанням SCSS. Після цього я доповнив код, щоб реалізувати систему світлої та темної теми, та функцію конвертації px в rem. Зробив я наступне: я реалізував змінні для світлої та темної теми в SCSS; створив міксин для генерації стилів теми і додав можливість перемикання теми через JavaScript; додав файл \_functions.scss для конвертації пікселів у rem за допомогою функції px-to-rem.