

**Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки**

**Навчальний проєкт
з дисципліни
«Основи Front-end технологій»**

Виконав:

студент групи ІМ-11
Царик Микола Миколайович
варіант відповідно до списку: 34

Перевірив:

Костянтин Анатолійович Жереб

Київ 2023

Мета: закріпити на практиці всі знання, що були отримані протягом курсу та розробити веб-додаток, який включатиме в себе різні аспекти та прийоми верстки веб-сторінок.

Стек: ReactJs, CSS, HTML

Завдання: розробити Фронт-енд частину багатосторінкового веб-застосунку. Зверстати сайт для суши-ресторану, на якому буде домашня сторінка, сторінка «Меню», сторінка «Про нас» та сторінка «Контакти». Сайт має бути адаптивним і коректно відображатися як на комп'ютері так і на телефоні. Також на сайті має бути адаптивне навігаційне меню (navBar) та Footer.

Хід розробки:

1) Спочатку я вирішив зверстати навігаційне меню так як воно є універсальним для всіх сторінок.

Для створення посилань використовував вбудований компонент `< Link />`

У React компонент **Link** зазвичай використовується для створення гіперпосилань у додатках. Він надає зручний спосіб навігації між різними частинами програми, забезпечуючи перехід без перезавантаження сторінки. Він є частиною бібліотеки React Router, яка надає інструменти для керування маршрутизацією у веб-додатках React.

```
<Link to="/" onClick={showNavbar}>Home</Link>
<Link to="/menu" onClick={showNavbar}>Menu</Link>
<Link to="/about" onClick={showNavbar}>About</Link>
<Link to="/contacts" onClick={showNavbar}>Contacts</Link>
```

Для використання іконок мною була імпортована бібліотека `react-icons/fa`.

У цьому випадку використовуються іконки з колекції Fa (FontAwesome), а саме FaBars та FaTimes (для іконки гамбургер-меню та хрестика).

```
import { FaBars, FaTimes } from "react-icons/fa";
```

```
<FaTimes />
```

Під час адаптації під мобільні пристрої був використаний реакт хук `useRef()`. Використання `ref` дозволяє динамічно змінювати клас елемента при виклику функції `showNavbar`, що в свою чергу змінює його візуальне представлення на сторінці. (Для відкриття та закриття навігаційного меню на екрані смартфона)

```
const navRef = useRef( initialValue: null);  
Nikolay  
const showNavbar = () => {  
  navRef.current.classList.toggle(styles.responsiveNav);  
};
```

Після стилізації компонента можна виділити такі прийоми , які були використані:

- Адаптивний дизайн. Використовуються **медіа-запити**, щоб змінити стилі на маленьких екранах.
- Управління видимістю та прозорістю. **visibility** та **opacity** використовуються для управління відображенням елементів.
- Позиціонування та шари. **position, top, right, left** використовуються для точного позиціонування елементів. **z-index** використовується для керування шарами.

```
.navBtn {  
  padding: 5px;  
  cursor: pointer;  
  background: transparent;  
  border: none;  
  outline: none;  
  color: #eee;  
  visibility: hidden;  
  opacity: 0;  
  font-size: 1.8rem;  
}
```

```
@media only screen and (max-width: 850px) {  
  .links {  
    z-index: 999;  
    position: fixed;  
    top: -100vh;  
    left: 0;
```

2) Розробивши навігаційне меню, я зробив футер, бо він також універсальний для всіх сторінок

Цей компонент також використовував `<Link/>`.

Але тепер ми посилаємо користувача не на певну сторінку в рамках нашого проєкту, а на сторонній ресурс (Instagram/Telegram/YouTube), тому ми використовуємо атрибут `target="_blank"` щоб відкривати посилання у новій вкладці браузера.

Для кожного з компонентів проєкту я використовував техніку використання локальних CSS-стилів.

Локальні стилі-модулі (CSS Modules) є методологією організації стилів у React-додатках, надаючи локалізацію стилів для конкретних компонентів. Це дозволяє уникнути конфліктів імен стилів між різними компонентами, що є особливо корисним у великих проєктах.

```
import styles from './Footer.module.css';
```

```
<div className={styles.container}>  
  <div className={styles.links}>
```

```
.container {...}  
.socialMedia svg {...}  
.links a{...}  
.links {...}
```

3) Далі я написав базову структуру сторінок, та додав **Router** для переміщення між сторінками

Структура маршрутизації: Використовується компонент **BrowserRouter** із бібліотеки **react-router-dom** для обгортання всієї програми та включення маршрутизації.

Компонент **Routes** використовується визначення маршрутів за допомогою компонента **Route**. Кожен маршрут має свій шлях (**path**) та компонент, який відображатиметься (**element**).

Маршрути та компоненти: Визначено маршрути для чотирьох сторінок: Home, Menu, About, Contacts.

Кожен маршрут пов'язаний з відповідним компонентом: **<Home/>**, **<Menu/>**, **<About/>**, **<Contacts/>**.

Імпорт компонентів: Імпортовано компоненти **Navbar** та **Footer** із відповідних файлів.

```
import Navbar from "../components/0thers/Navbar";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import Home from "../pages/Home";
```

```
<Router>
  <Navbar />
  <Routes>
    <Route path="/" exact element={<Home />} />
    <Route path="/menu" exact element={<Menu />} />
    <Route path="/about" exact element={<About />} />
    <Route path="/contacts" exact element={<Contacts />} />
  </Routes>
  <Footer />
</Router>
```

4) Далі я почав розробляти сторінки і першою біла домашня сторінка

Background Image (Фонове зображення):

У стилі компонента (HomePage.module.css) використав властивість `backgroundImage` для визначення фонового зображення.

У коді компонента це властивість задається вбудованим стилем `style={{ backgroundImage: url(/images/homeBackground.jpg) }}`.

`url(/images/homeBackground.jpg)` вказує на шлях до зображення, яке буде використовуватися як фонове.

Зауважу, що зображення у проєкті знаходяться у папці **public/images**. Папка `public` у проєктах React використовується для статичних ресурсів, таких як зображення, та забезпечує простий доступ до них.

```
<div
  className={styles.container}
  style={{ backgroundImage: `url(/images/homeBackground.jpg)` }}
>
```

Дивлячись на стилі можна помітити такі особливості:

Шрифти: Здійснено імпорт шрифтів з Google Fonts, таких як 'Nova Square' та 'Varela Round'. Шрифт 'Nova Square' використовується для основного тексту в контейнері.

```
@import url('https://fonts.googleapis.com/css2?family=Nova+Square&family=Varela+Round&display=swa
```

Фонове зображення: Контейнер має фонове зображення, яке розташоване зверху, не повторюється і розтягується, щоб покрити весь контейнер.

```
background-position: top;
background-repeat: no-repeat;
background-size: cover ;
```

Адаптивний дизайн: Використовуються медіа-запити (`@media`) для адаптації стилів до різних розмірів екрану.

```
@media only screen and (max-width: 1450px) {
```

```
@media only screen and (max-width: 750px) {
```

Анімація кнопки при наведенні: для кнопки визначено анімацію при наведенні (`hover`), яка включає зміну кольору фону та затримку для плавного ефекту.

```
.content button:hover {
```

5) Після цього я взявся за розробку меню ресторану

Було створено 3 компоненти: `<MenuPage/>`, `<MenuList/>` `<MenuItem/>`

На сторінці меню я відображаю лист зі всіх доступних продуктів.

Для цього у `<MenuItem/>` я повертаю розмітку:

```
return (  
  <div className={styles.itemCard}>  
    <img src={"/" + image} alt={title} className={styles.itemImage} />  
    <div className={styles.itemDetails}>  
      <h2 className={styles.itemTitle}>{title}</h2>  
      <p className={styles.itemDescription}>{description}</p>  
      <div className={styles.itemInfo}>  
        <p className={styles.itemValue}>  
          <strong>{weight}</strong>  
        </p>  
        <p className={styles.itemValue}>  
          <strong>${price.toFixed( fractionDigits: 2)}</strong>  
        </p>  
      </div>  
    </div>  
  </div>  
)
```

В якості значень для полів я використовую значення , які приходять з пропсів які я передаю з `<MenuList/>`.

```
const MenuItem = (props) => {  
  const { title, description, weight, price, image } = props;
```

`<MenuList/>` в свою чергу повертає список зі всіх доступних елементів.

Значення він бере зі створеного масиву, в якому зберігаються такі значення продукту як: **id, title, description, weight, price, image**. Цей масив він тапжржж отримує через пропс, який йому передають в `<MenuPage/>`

```
const MenuList = (props) => {  
  return (  
    <div className={styles.list}>  
      {props.list.map((item) => {...})}  
    </div>  
  );  
};
```

В `<MenuPage/>` ми викликаємо функцію `getAllProducts`, яка створена в окремому файлі разом з масивом даних.

```
const MenuPage = () => {
  const sushiList = getAllProducts();
  return (
    <div className={styles.container}>
      <MenuList list={sushiList}></MenuList>
    </div>
  );
};
```

```
export function getAllProducts() {
  return sushiList;
}
```

```
const sushiList = [
  {
    id: 1,
    title: "Philadelphia Roll",
    description: "Salmon, avocado, cucumber, cream cheese, nori, rice",
    weight: "200g",
    price: 12.99,
    image: "images/sushi1.png",
  },
  {id: 2...},
  {id: 3...},
  {id: 4...},
  {id: 5...},
  {id: 6...},
  {id: 7...},
  {id: 8...},
  {id: 9...},
];
```


6) Закінчивши з меню, я перейшов до сторінки «Про нас»

Ця сторінка була створена за схожим принципом і також використовувала структуру: сторінка => список => елемент.

Цього разу в якості пропсів у Item ми передаємо посилання на зображення, заголовок, контент (текст блоку), а також номер стилю блоку (якщо номер = 1, то зображення буде розташовано праворуч, якщо 0 то ліворуч)

```
<AboutItem
  blockLayoutNumber={item.blockLayoutNumber}
  imgSrc={item.imgSrc}
  title={item.title}
  content={item.content}
/>
```

Різні стилі були написані як для першого типу блоків так і для другого, але також були стилі які є спільними для обох типів блоків, тому стилі виглядають таким чином:

```
.block{...}
.block h1{...}
.block p{...}
.block img{...}
.additionalImage{...}

.block1 h1{...}
.block1 p{...}
.block1 img{...}
.block1 .additionalImage {...}

.block2 h1{...}
.block2 p{...}
.block2 img{...}
.block2 .additionalImage{...}
```

Було використано 2 важливі властивості css:

order: Визначає порядок відображення елементів у flex контейнері.

Застосовується для керування порядком блоків .block1 та .block2 залежно від контексту.

z-index: Визначає шар, на якому відображається елемент в просторі.

Використовується для визначення, яке зображення чи блок буде вище в ієрархії на сторінці.

7) Останньою я розробив сторінку «Контакти»

Я створив окремі компоненти для різних типів питань щоб їх можна було використовувати в програмі багато разів додаючи мінімальну кількість нового коду. Мною були створені такі компоненти як: `<ContactPage/>`, `<Form/>`, `<FormMessage/>`, `<FormTextArea/>`.

```
const ContactPage = () => {
  return (
    <div className={styles.contacts}>
      <div className={styles.leftSide}>
        <h1>Contacts Us</h1>
        <Form />
      </div>
      <div
        className={styles.rightSide}
        style={{ backgroundImage: `url(/images/contacts1.jpg)` }}
      ></div>
    </div>
  );
};
```

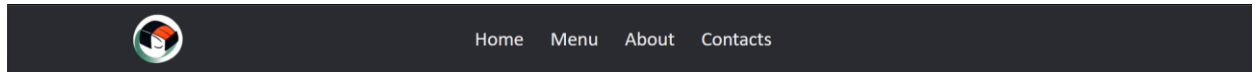
```
const Form = () => {
  return (
    <div className={styles.container}>
      <form method="POST" id="contactsForm" className={styles.form}>
        <FormQuestion name="name" type="text" title="Full name" />
        <FormQuestion name="email" type="email" title="Email" />
        <FormQuestion name="message" type="text" title="Message" />
        <FormTextArea
          name="message"
          rows="6"
          placeholder="Enter your message"
          required={true}
          className={styles.textArea}
        />
        <Link to="/">
          <button type="submit" className={styles.button}>
            Send message
          </button>
        </Link>
      </form>
    </div>
  );
};
```

В якості пропсів я передавав текст, ім'я та тип, які потім використовувалися в тегах `<label>` та `<input>`

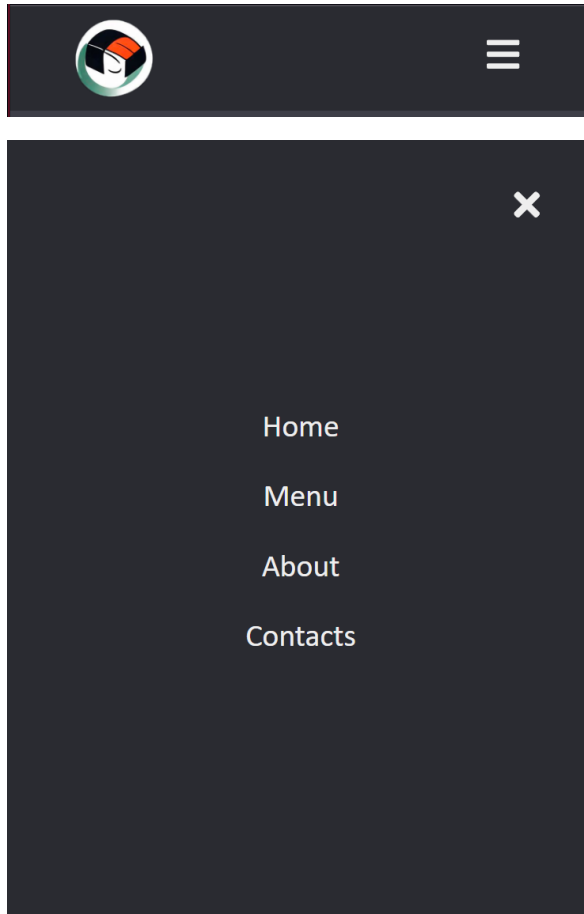
```
const FormQuestion = ({ name, type }) => {
  return (
    <div>
      <label htmlFor={name}></label>
      <input name={name} placeholder={`Enter ${name}`} type={type} />
    </div>
  );
};
```

Скріншоти готового сайту:

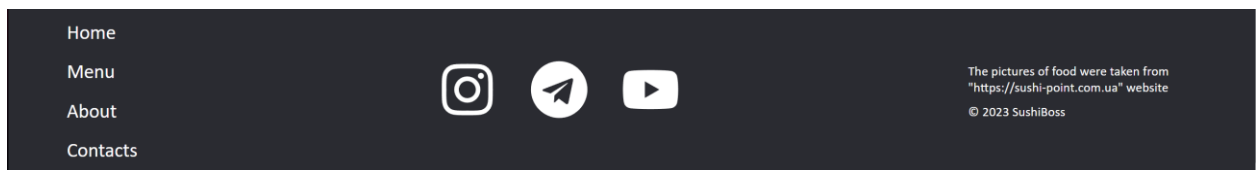
Навігаційне меню (Великий екран)



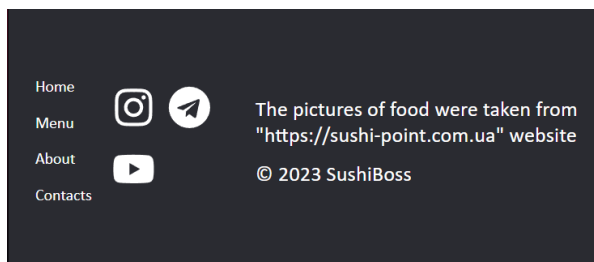
Навігаційне меню (Екран телефону)



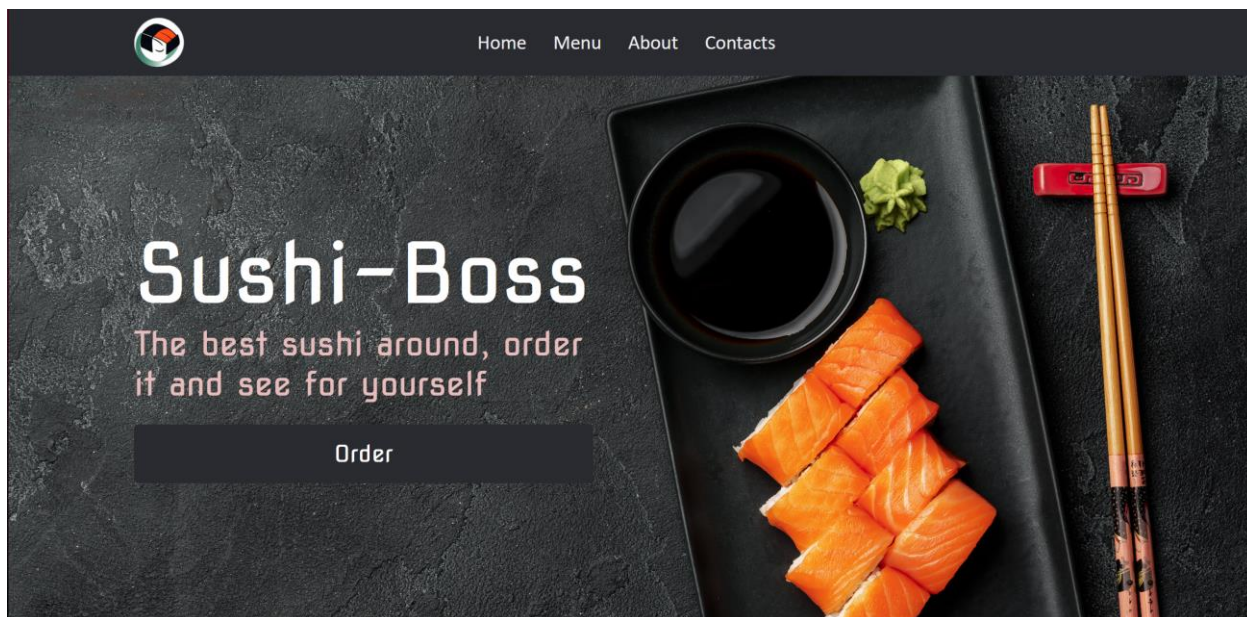
Футер (Великий екран)



Футер (Екран телефону)



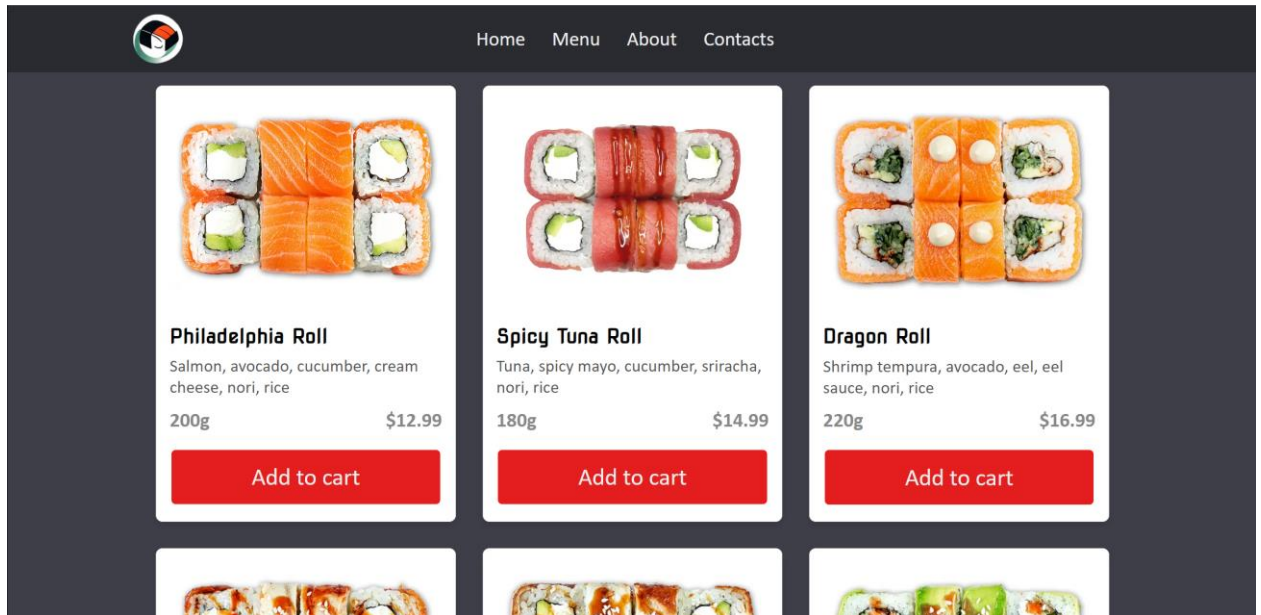
Домашня сторінка (Великий екран)



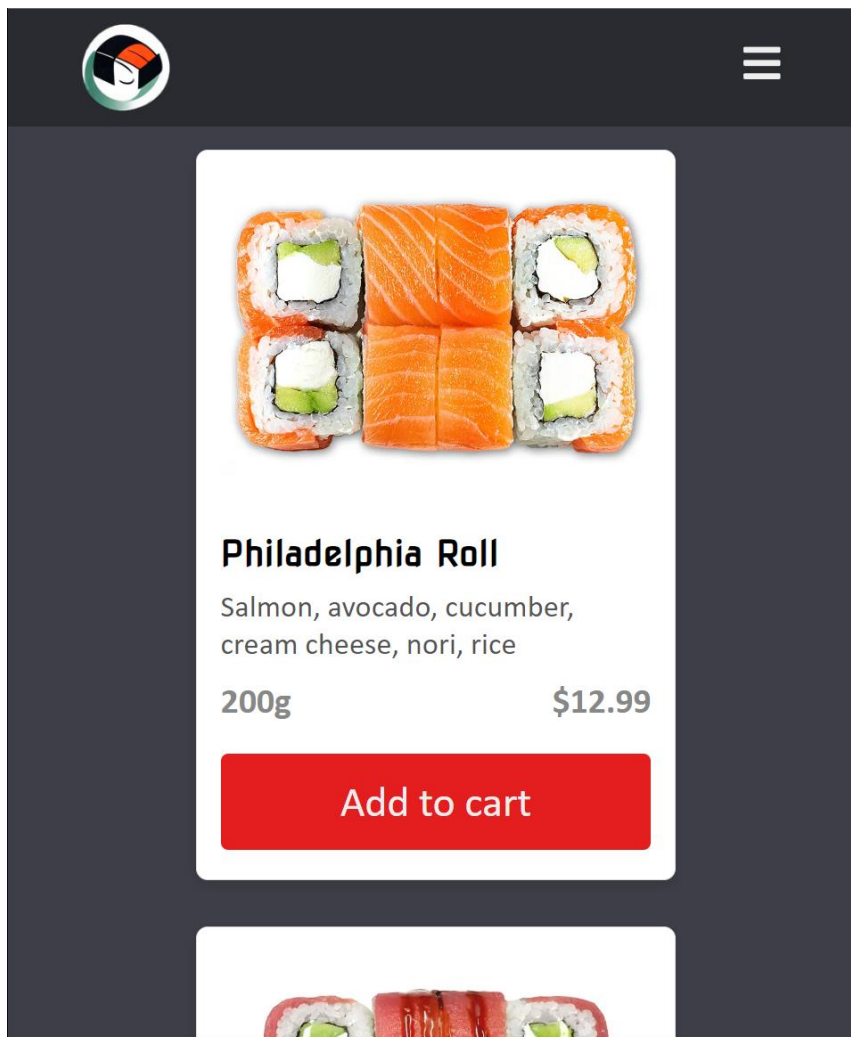
Домашня сторінка (Екран телефону)




Сторінка «Меню» (Великий екран)



Сторінка «Меню» (Екран телефону)




Сторінка «Про нас» (Великий екран)




HomeMenuAboutContacts

Welcome

Welcome to SushiBoss - a culinary haven where artistry meets tradition, and every bite tells a story. Nestled in the heart of Kyiv, our sushi restaurant takes you on an exquisite journey through the delicate flavors and meticulous craftsmanship of Japanese cuisine.






The art of cook

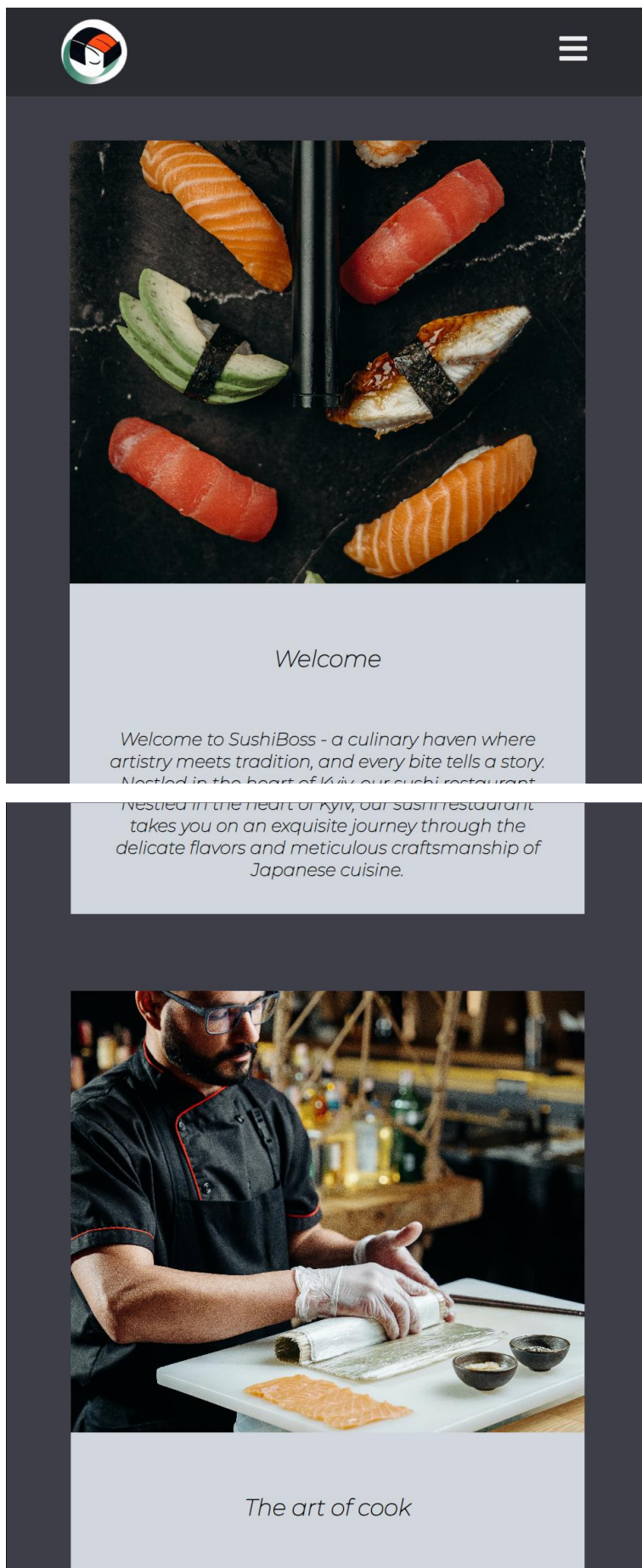
Our chefs, masters of their craft, skillfully blend the finest ingredients to create a symphony of tastes that dance on your palate. From the pristine waters of the Pacific to the vibrant markets of Tokyo, each element is carefully sourced to ensure the highest quality.

Elevating Japanese elegance

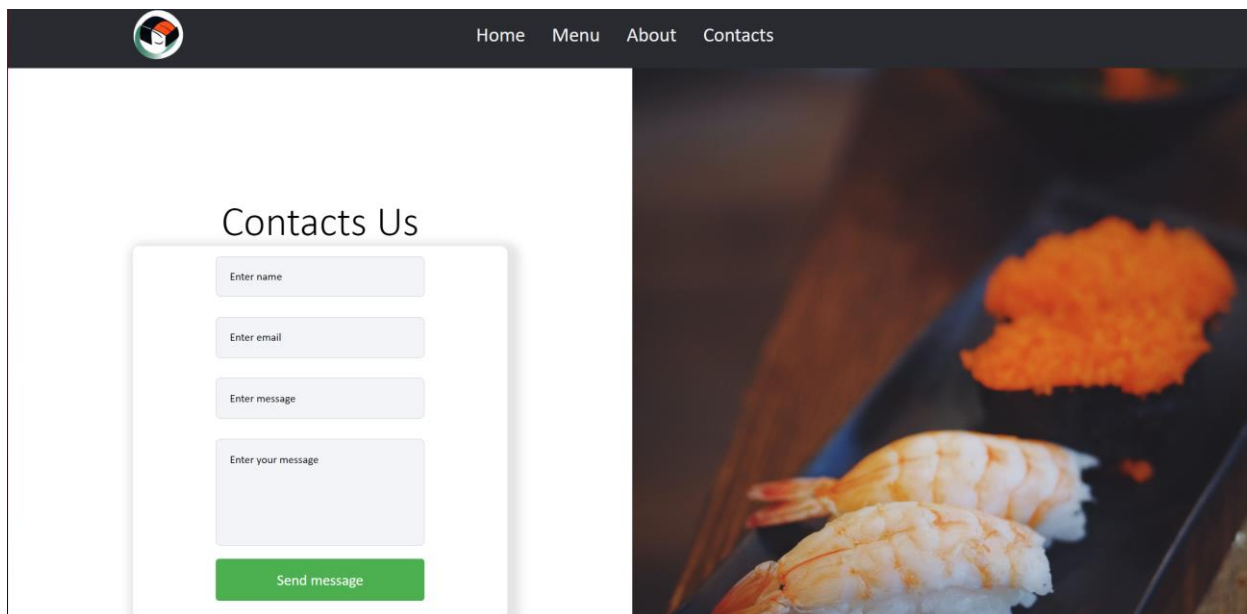
The tranquil surroundings provide the perfect backdrop for savoring our extensive menu, which boasts a diverse selection of sushi and sashimi, expertly crafted rolls, and innovative fusion creations that push the boundaries of traditional Japanese fare.



Сторінка «Про нас» (Екран телефону)



Сторінка «Контакти» (Великий екран)



The desktop version of the 'Contacts Us' page features a dark header with a logo on the left and navigation links 'Home', 'Menu', 'About', and 'Contacts' on the right. The main content area is split into two columns. The left column contains a white contact form titled 'Contacts Us' with four input fields: 'Enter name', 'Enter email', 'Enter message', and 'Enter your message'. A green 'Send message' button is at the bottom of the form. The right column displays a large, high-quality photograph of two pieces of shrimp sushi on a dark plate, with a pile of orange tobiko (flying fish roe) in the background.

Contacts Us

Enter name

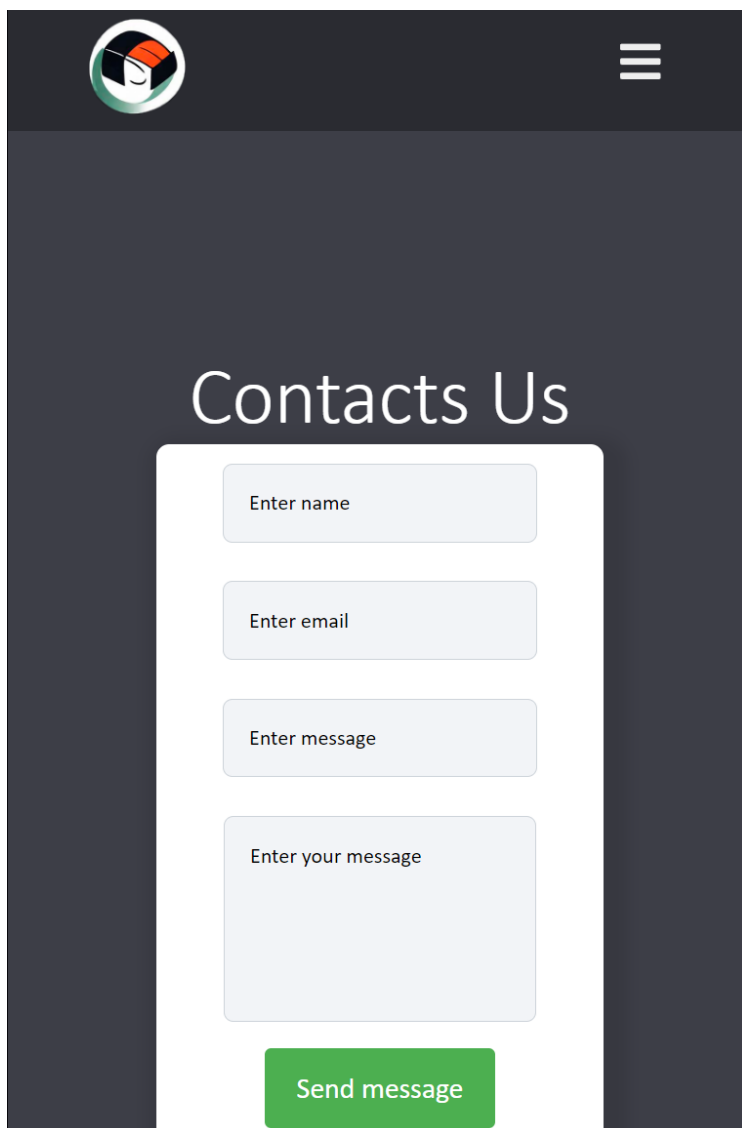
Enter email

Enter message

Enter your message

Send message

Сторінка «Контакти» (Екран телефону)



The mobile version of the 'Contacts Us' page has a dark header with a logo on the left and a hamburger menu icon on the right. The background is a solid dark grey. The contact form is centered and consists of a white container with the title 'Contacts Us' in large white text. Below the title are four light grey input fields: 'Enter name', 'Enter email', 'Enter message', and 'Enter your message'. A green 'Send message' button is positioned at the bottom of the form.

Contacts Us

Enter name

Enter email

Enter message

Enter your message

Send message

Висновок:

Під час розробки проєкту я використовував набуті знання з ReactJs, CSS та HTML для створення багатосторінкового веб-застосунку для суши-ресторану. Весь процес включав в себе розробку домашньої сторінки, сторінки "Меню", сторінки "Про нас" та сторінки "Контакти".

Однією з ключових задач було забезпечити адаптивність сайту, щоб він коректно відображався як на комп'ютері, так і на мобільних пристроях. Особливу увагу приділив реалізації адаптивного навігаційного меню (navBar) та Footer, щоб забезпечити зручну навігацію для користувачів на різних пристроях.

Я вивчив багато нових технік, таких як імпорт зображень у HTML файли, створення масиву макетних даних імітуючих базу даних, нові прийоми адаптивної верстки такі як object-fit та media запити, робота з іконками, використання локальних css стилів, структуризація React застосунку і багато іншого.

Під час розробки виникли виклики такі як налаштування структури файлів, робота з картинками що накладаються одне на одного, адаптація всіх сторінок під мобільні пристрої, але завдяки вмінню працювати з React та використанню сучасних практик верстки, мені вдалося подолати їх та зверстати веб-сайт для суши-ресторану. Цей проєкт дозволив мені закріпити та застосувати свої знання на практиці та отримати цінний досвід у розробці фронт-енд частини веб-застосунків.