FOOD DELIVERY WEB-APP

TEAM MEMBERS:

ROLL NUMBER:	NAME:
NISHDHARANI P	22ALR059
SANTHIYA E	22ALR086
SASI KUMAR I	22ALR088

INTRODUCTION:

In today's digital age, having a robust online presence is crucial for any restaurant seeking to expand its reach and improve customer service. This report details the development of a specialized Pizza Delivery Website for a specific restaurant, utilizing modern web technologies such as React.js, HTML, and CSS. The website is designed to streamline the ordering process, enhance customer engagement, and drive sales, while prominently featuring the restaurant's unique pizza offerings.

Key Features:

1. **User-Friendly Interface**:
    - **Responsive Design**: Ensure the website is accessible on all devices (desktops, tablets, smartphones).
    - **Intuitive Navigation**: Easy-to-use menu with categories (e.g., pizza types, sides, drinks, desserts).
    - **Search Functionality**: Allow users to quickly find their desired items.
2. **Online Ordering System**:
    - **Menu Display**: High-quality images and descriptions of pizzas and other items.
    - **Add to Cart**: Simple process for adding items to the cart
    - **Order Summary**: Display a detailed summary of the order before checkout.
3. **Contact and Support**:
    - **Contact Information**: Easy access to the restaurant's contact details (phone number, email).
    - **FAQ Section**: Common questions and answers about ordering, delivery, and policies.
4. **Accessibility Features**:
    - **Accessibility Compliance**: Ensure the website is accessible to users with disabilities (e.g., screen reader support, keyboard navigation).
5. **Promotions and Discounts**:
    - **Loyalty Program**: Reward frequent customers with points or discounts.
6. **Reviews and Ratings**:
    - **Customer Reviews**: Allow users to leave reviews and rate their experience.
7. Bell icon:
    - Notifies the user if any item is added to the cart.

8. Mobile-Friendly Interface:

    - Ensures the web application is accessible and fully functional on mobile devices.

REQUIREMENTS:

Front-End Programming:

1. CSS (Cascading Style Sheets):
    - CSS is used to define the visual appearance and layout of web pages. It allows developers to apply styles, such as colors, fonts, spacing, and positioning, to HTML elements, creating a visually appealing and consistent user interface.

2. JavaScript:
    - JavaScript is a powerful programming language that adds interactivity and dynamic behavior to web pages. It enables developers to manipulate and modify HTML and CSS, handle user interactions, perform calculations, validate forms, and make asynchronous requests to servers.

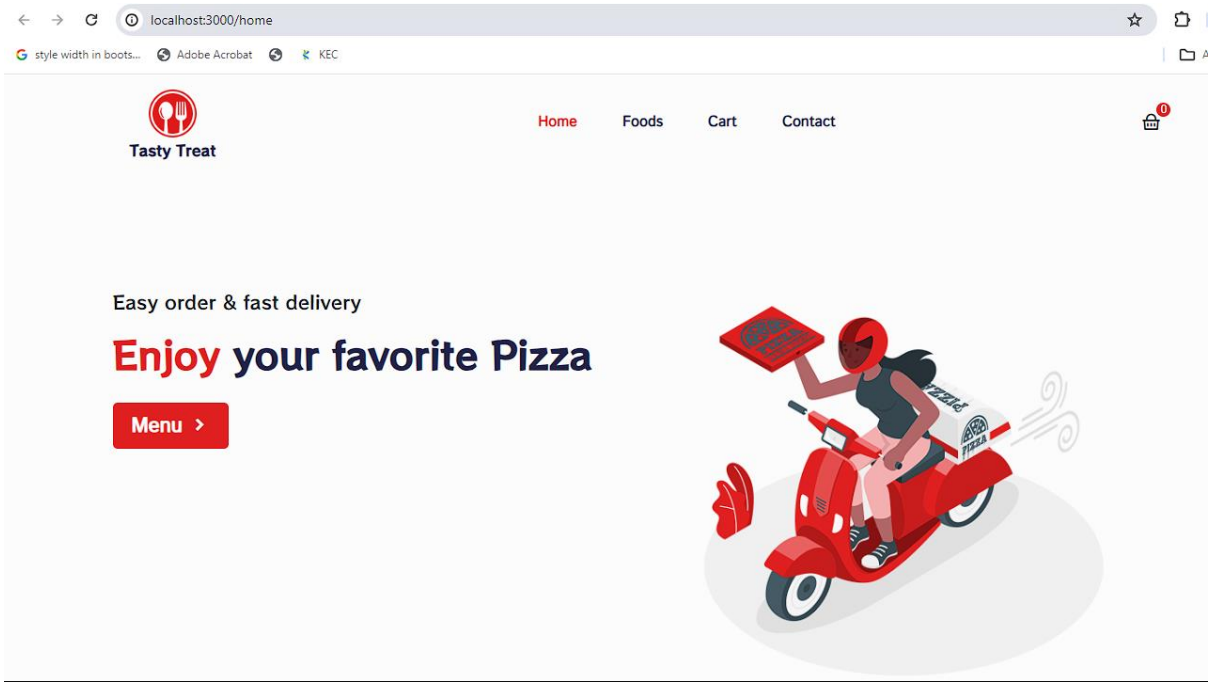
3. React:
    - React is a popular JavaScript library for building user interfaces. It provides a component-based approach to web development, allowing developers to create reusable UI components that update efficiently based on changes in data. React uses a virtual DOM (Document Object Model) to optimize performance and facilitate the building of complex web applications.

Back-End Programming

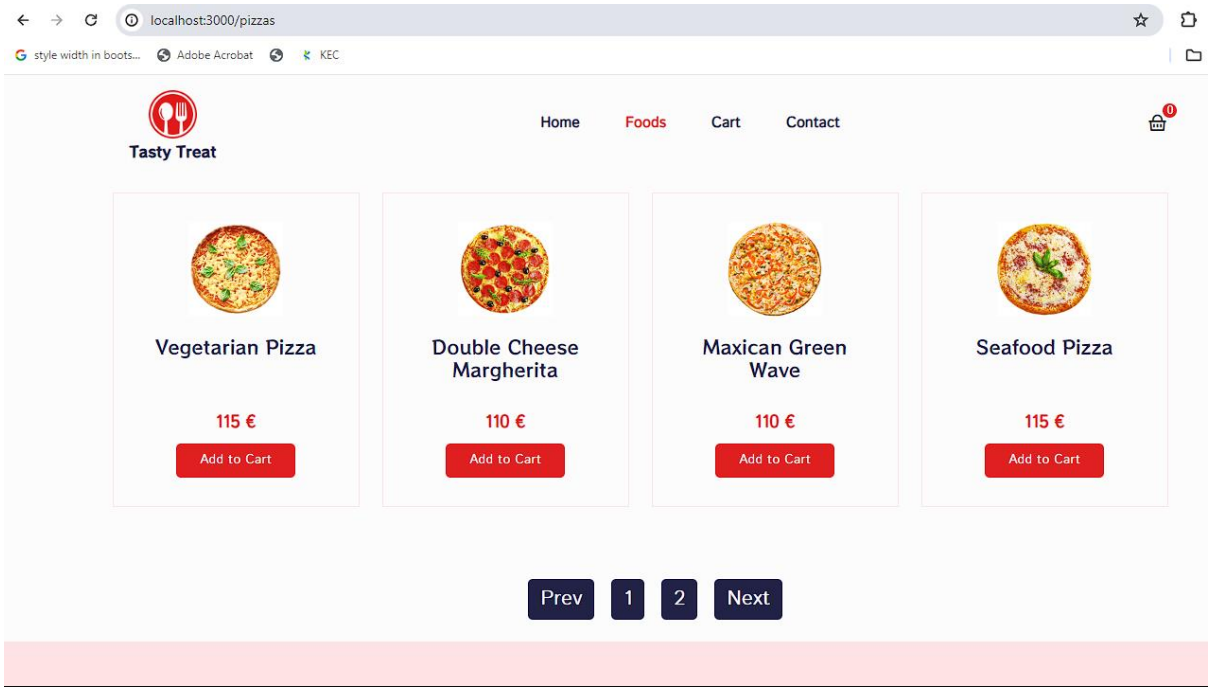
1. Database Management:
    - Backend programming involves working with databases to store and retrieve data. Developers interact with databases using query languages like SQL or NoSQL databases. For this system, MongoDB is used to design and implement database schemas, optimize queries, and ensure data integrity and security.
2. APIs (Application Programming Interfaces):
    - Backend developers build APIs that allow communication between the frontend and backend components of a web application. APIs define the protocols and rules for how different software components can interact and exchange data. Commonly used API standards include RESTful APIs and GraphQL.

UI DESIGN:

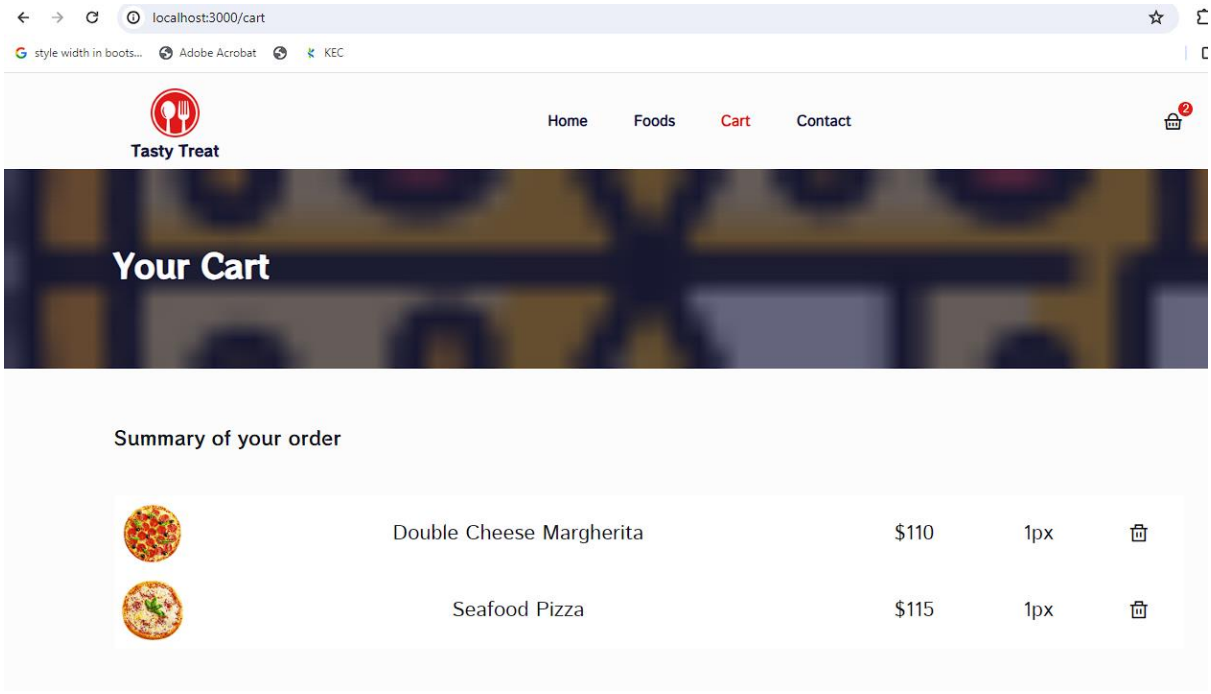
HOME PAGE:

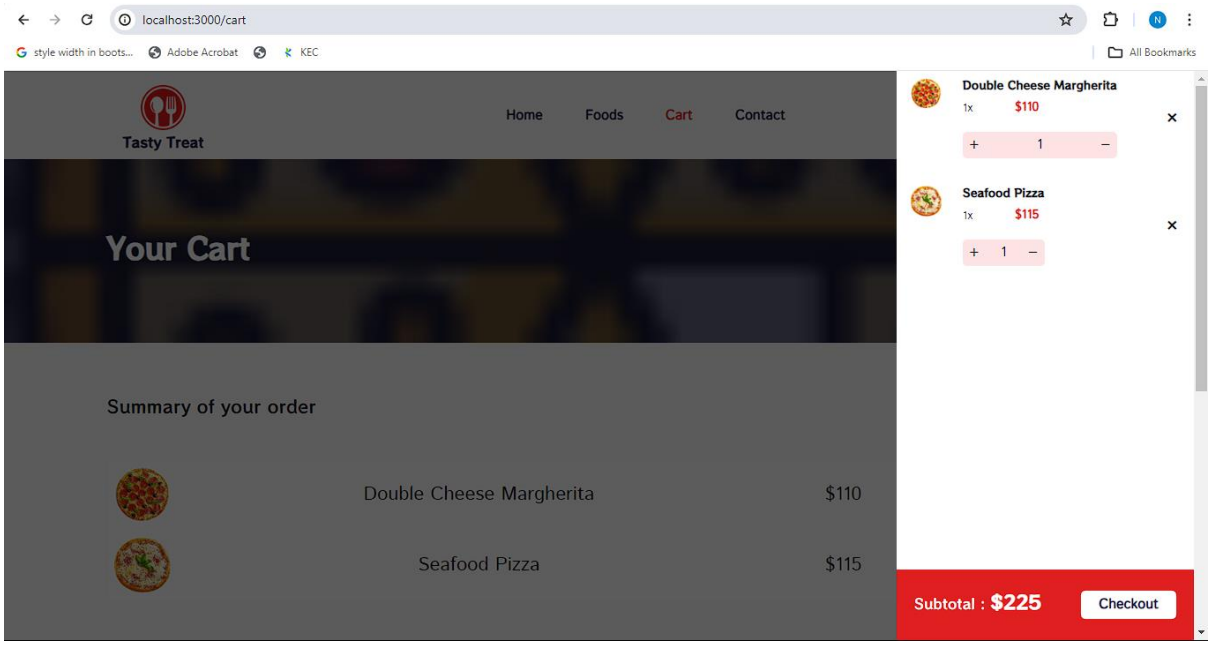


MENU PAGE:

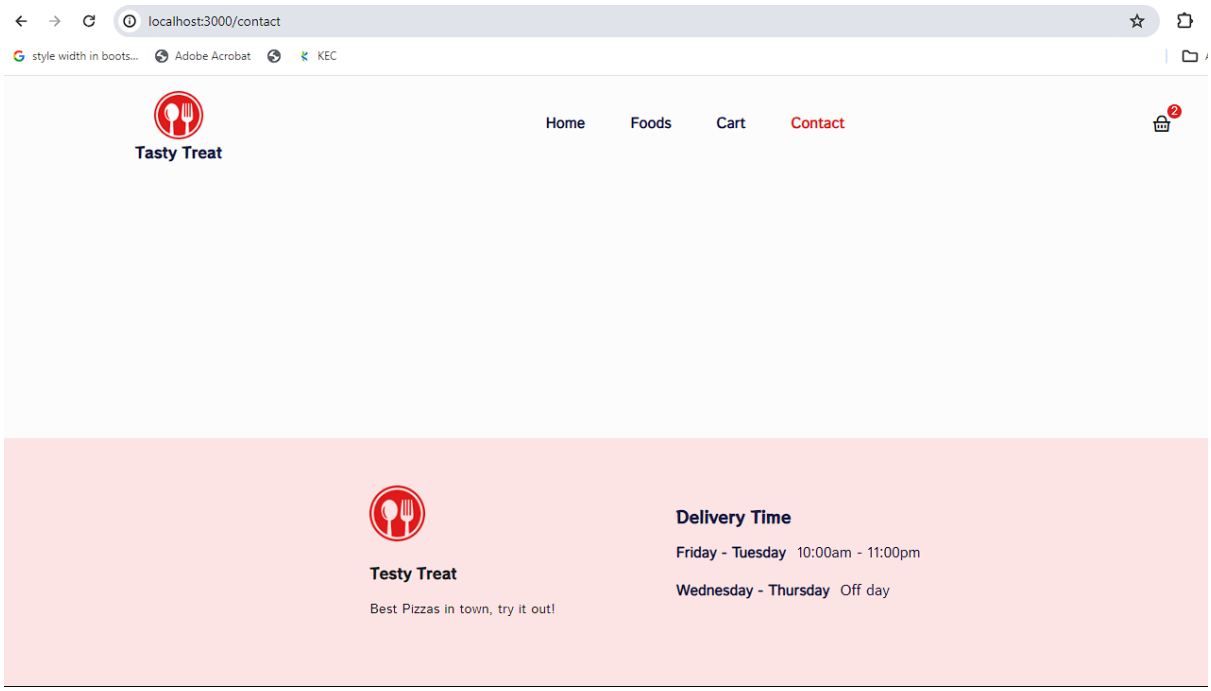


CART:


localhost:3000/cart
style width in boots...
Adobe Acrobat
KEC
Tasty Treat
Home
Foods
Cart
Contact
Your Cart
Summary of your order
Double Cheese Margherita
$110
1px
Seafood Pizza
$115
1px


localhost:3000/cart
style width in boots...
Adobe Acrobat
KEC
All Bookmarks
Tasty Treat
Home
Foods
Cart
Contact
Your Cart
Summary of your order
Double Cheese Margherita
$110
Seafood Pizza
$115
Double Cheese Margherita
1x
$110
+
1
−
Seafood Pizza
1x
$115
+
1
−
Subtotal : $225
Checkout

CONTACT PAGE:



SAMPLE CODING:

Cart.jsx

```
import React from "react";
import CommonSection from "../components/UI/common-section/CommonSection";
import Helmet from "../components/Helmet/Helmet";
import "../styles/cart-page.css";
import { useSelector, useDispatch } from "react-redux";
import { Container, Row, Col } from "reactstrap";
import { cartActions } from "../store/shopping-cart/cartSlice";
import { Link } from "react-router-dom";

const Cart = () => {
  const cartItems = useSelector((state) => state.cart.cartItems);
  const totalAmount = useSelector((state) => state.cart.totalAmount);
  return (
    <Helmet title="Cart">
      <CommonSection title="Your Cart" />
      <section>
        <Container>
          <Row>
            <Col lg="12">
              {cartItems.length === 0 ? (
                <h5 className="text-center">Your cart is empty</h5>
              ) : (
                <>
                  <h5 className="mb-5">Summary of your order</h5>
                  <table className="table table-borderless mb-5 align-middle">
                    <tbody>
```

```
                {cartItems.map((item) => (
                  <Tr item={item} key={item.id} />
                ))}
              </tbody>
            </table>
          </>
        )}

        <div className="mt-4">
          <h6>
            Subtotal: $
            <span className="cart__subtotal">{totalAmount}</span>
          </h6>
          <p>Taxes and shipping will calculate at checkout</p>
          <div className="cart__page-btn">
            <button className="addTOCart__btn me-4">
              <Link to="/pizzas">Continue Shopping</Link>
            </button>
            <button className="addTOCart__btn">
              <Link to="/checkout">Proceed to checkout</Link>
            </button>
          </div>
        </div>
      </Col>
    </Row>
  </Container>
 </section>
</Helmet>
);
```

```
};

const Tr = (props) => {
  const { id, image01, title, price, quantity } = props.item;
  const dispatch = useDispatch();

  const deleteItem = () => {
    dispatch(cartActions.deleteItem(id));
  };
  return (
    <tr>
      <td className="text-center cart__img-box">
        <img src={image01} alt="" />
      </td>
      <td className="text-center">{title}</td>
      <td className="text-center">${price}</td>
      <td className="text-center">{quantity}px</td>
      <td className="text-center cart__item-del">
        <i className="ri-delete-bin-line" onClick={deleteItem}></i>
      </td>
    </tr>
  );
};

export default Cart;
```

Checkout.jsx

```
import "../styles/checkout.css";
```

```
import { AiFillCheckCircle } from "react-icons/ai";

const Checkout = () => {
  return (
    <>
      <div className="checkoutMessage">
        <div className="checkoutTitleContainer">
          <AiFillCheckCircle className="checkoutIcon" />
          <h3>Thank you for your order!</h3>
        </div>
        <span>
          Your order is being processed and will be delivered as fast as
          possible.
        </span>
      </div>
    </>
  );
};
export default Checkout;
```

Home.jsx

```
import React from "react";
import Helmet from "../components/Helmet/Helmet.js";
import { Container, Row, Col } from "reactstrap";

import { Link } from "react-router-dom";

import guyImg from "../assets/images/hero.png";
```

```
import "../styles/hero-section.css";


const Home = () => {
  return (
    <Helmet title="Home">
      <section>
        <Container>
          <Row>
            <Col lg="6" md="6">
              <div className="hero__content">
                <h5 className="mb-3">Easy order & fast delivery</h5>
                <h1 className="mb-4 hero__title">
                  <span>Enjoy</span> your favorite Pizza
                </h1>

                <button className="order__btn d-flex align-items-center justify-content-
between ">
                  <Link to="/pizzas">
                    Menu <i className="ri-arrow-right-s-line"></i>
                  </Link>
                </button>
              </div>
            </Col>

            <Col lg="6" md="6">
              <div className="hero__img">
                <img src={guyImg} alt="delivery-guy" className="w-100" />
              </div>
            </Col>
          </Row>
```

```
        </Container>
      </section>
    </Helmet>
  );
};
export default Home;
```

PizzaDetails.jsx

```
import React, { useState, useEffect } from "react";

import products from "../assets/fake-data/products";
import { useParams } from "react-router-dom";
import Helmet from "../components/Helmet/Helmet";
import CommonSection from "../components/UI/common-section/CommonSection";
import { Container, Row, Col } from "reactstrap";
import ExtraIngredient from '../components/ExtraIngredient/ExtraIngredient.jsx'
import { useDispatch } from "react-redux";
import { cartActions } from "../store/shopping-cart/cartSlice";
import { useSelector } from "react-redux";

import "../styles/product-details.css";
import "../styles/product-card.css";

import ProductCard from "../components/UI/product-card/ProductCard";

const ExtraIngredients = {
  MUSHROOMS: "Mushrooms",
  ONION: "Onion",
  PEPPER: "Pepper",
```

```
  PINAPPLE: "Pinapple",
  TUNA: "Tuna",
  MEAT: "Meat",
  CHEESE: "Cheese",
  HOTSAUCE: "Hot Sauce",
  CORN: "Corn"
}

const PizzaDetails = () => {
  const { id } = useParams();
  const dispatch = useDispatch();
  const [extraIngredients, setExtraIngredients] = useState([]);
  const [isUpdateNotificationDisplayed, setIsUpdateNotificationDisplayed] =
useState(false);
  const product = products.find((product) => product.id === id);
  const cartProducts = useSelector((state) => state.cart.cartItems);
  const [previewImg, setPreviewImg] = useState(product.image01);
  const { title, price, category, desc, image01 } = product;
  const relatedProduct = products.filter((item) => category === item.category);



  useEffect(() => {
    const existingPizza = cartProducts.find(item => item.id === id);
    if(existingPizza) {
      setExtraIngredients(existingPizza.extraIngredients);
    } else {
      setExtraIngredients([]);
    }
  }, [cartProducts, id]);
```

```
const addItem = () => {
  setIsUpdateNotificationDisplayed(true);
    setTimeout(function(){
      setIsUpdateNotificationDisplayed(false);
    },3000)

  dispatch(
    cartActions.addItem({
      id,
      title,
      price,
      image01,
      extraIngredients
    })
    );

  };

  useEffect(() => {
    setPreviewImg(product.image01);
    window.scrollTo(0, 0);
  }, [product]);

  function updateExtraIngredients(ingredient) {
    if(extraIngredients.includes(ingredient)) {
      setExtraIngredients(extraIngredients.filter(item => item !== ingredient));
    } else {
      setExtraIngredients(previousState => [...previousState, ingredient]);
```

```
  }
 }

return (
 <Helmet title="Product-details">
  {isUpdateNotificationDisplayed && (
   <div className="updateCartNotifiation">
    <span>You successfully updated your cart!</span>
   </div>
  )
  }

  <CommonSection title={title} />

  <section>
   <Container>
    <Row>
     <Col lg="2" md="2">
      <div className="product__images">
       <div
        className="img__item mb-3"
        onClick={() => setPreviewImg(product.image01)}
       >
        <img src={product.image01} alt="" className="w-50" />
       </div>
       <div
        className="img__item mb-3"
        onClick={() => setPreviewImg(product.image02)}
       >
```

```
    <img src={product.image02} alt="" className="w-50" />
   </div>

   <div
    className="img__item"
    onClick={() => setPreviewImg(product.image03)}
   >
    <img src={product.image03} alt="" className="w-50" />
   </div>
  </div>
</Col>

<Col lg="4" md="4">
  <div className="product__main-img">
   <img src={previewImg} alt="" className="w-100" />
  </div>
</Col>

<Col lg="6" md="6">
  <div className="single__product-content">
   <h2 className="product__title mb-3">{title}</h2>
   <p className="product__price">
    {" "}
    Price: <span>${price}</span>
   </p>
   <p className="category mb-5">
    Category: <span>{category}</span>
   </p>
```

```
            <button onClick={addItem} className="addTOCART__btn">
              {cartProducts.find(item => item.id === id) ? 'Update Cart' : 'Add to Cart'}
            </button>
          </div>
        </Col>

        <Col lg='12'>
          <div className="extraIngredientsGrid">
            {(Object.values(ExtraIngredients)).map((ingredient) => {
              return (
                <ExtraIngredient
isChecked={extraIngredients.includes(ingredient)}  key={ingredient} onSelect={ingredient
=> updateExtraIngredients(ingredient)} ingredient={ingredient}></ExtraIngredient>
              )
            })}
          </div>
        </Col>

        <Col lg="12">
          <h6 className="description">Description</h6>
          <div className="description__content">
            <p>{desc}</p>
          </div>
        </Col>

        <Col lg="12" className="mb-5 mt-4">
          <h2 className="related__Product-title">You might also like</h2>
        </Col>

        {relatedProduct.map((item) => (
```

```
            <Col lg="3" md="4" sm="6" xs="6" className="mb-4" key={item.id}>
              <ProductCard item={item} />
            </Col>
          ))}
        </Row>
      </Container>
    </section>
  </Helmet>
 );
};

export default PizzaDetails;
```

Future Directions:

- **Mobile App Development**: Expanding the platform to include dedicated mobile apps for iOS and Android to further enhance accessibility and user experience.
- **Advanced Features**: Incorporating features such as loyalty programs, targeted promotions, and AI-driven recommendations to drive customer engagement and retention.
- **Integration with Third-Party Services**: Exploring integrations with third-party delivery services, payment gateways, and customer support tools to broaden the app's functionality and reach.
- **Continuous Improvement**: Regularly updating the app based on user feedback and technological advancements to ensure it remains competitive and aligned with customer expectations.

Conclusion:

The Pizza Delivery Web App project has successfully transformed the way our specific restaurant interacts with its customers and manages its operations. By leveraging modern web technologies, we have created a platform that not only meets the current needs of our customers and staff but also lays a strong foundation for future growth and innovation. As we move forward, we will continue to build on this success, exploring new opportunities to enhance our service offerings and deliver an exceptional dining experience to our customers.