

DAY#04.


=> BUILDING DYNAMIC FRONTEND
COMPONENTS FOR YOUR MARKETPLACE:

Key Components:

- Footer and Header Components
- Product Detail Component
- Cart Component
- Wishlist Component
- Checkout Flow Component
- Product Listing Component
- FAQ and Help Center Component
- Social Media Sharing Component
- API Integration

Product Listing Component:

The Product Listing Component presents a curated collection of products in a visually appealing and responsive layout, allowing users to effortlessly browse and explore. It dynamically retrieves product details, including names, prices, images, and descriptions, while incorporating powerful features such as filtering, sorting, and search functionality to enhance the shopping experience. With seamless pagination or infinite scrolling, it efficiently manages extensive product catalogs. Additional enhancements like product badges, customer ratings, quick add-to-cart options, and interactive hover effects further improve usability and engagement. Designed for optimal performance and a smooth user experience, this component ensures an intuitive and enjoyable shopping journey.

- **Dynamic Display**: Fetches and displays product data like name, price, image, and description.
- **User-Friendly Layout**: Organized in grids or lists for better navigation and visibility.
- **Filter & Sort Options**: Allows users to refine their search by category, price, or relevance.
- **Pagination Support**: Handles large datasets by breaking them into manageable pages. 
- **Responsive Design**: Adapts to different screen sizes for a seamless user experience.

Code Snip:

```

from typing import List

def isSafe(row: int, col: int, board: List[List[int]]) -> bool:
    # Check row
    for c in range(col):
        if board[row][c] == 1:
            return False

    # Check upper-left diagonal
    r, c = row, col
    while r >= 0 and c >= 0:
        if board[r][c] == 1:
            return False
        r -= 1
        c -= 1

    # Check lower-left diagonal
    r, c = row, col
    while r < n and c >= 0:
        if board[r][c] == 1:
            return False
        r += 1
        c -= 1

    return True

def solveNQueens(n: int) -> List[List[str]]:
    def backtrack(col: int):
        if col == n:
            solutions.append(list(map(lambda row: ''.join('Q' if board[row][col] == 1 else '.', col=n), row)))
            return

        for row in range(n):
            if isSafe(row, col, board):
                board[row][col] = 1
                backtrack(col + 1)
                board[row][col] = 0

    board = [[0] * n for _ in range(n)]
    solutions = []
    backtrack(0)
    return solutions

```

```

1  # Import the necessary libraries
2  import pandas as pd
3  import numpy as np
4  from sklearn.preprocessing import StandardScaler
5  from sklearn.model_selection import train_test_split
6  from sklearn.metrics import mean_squared_error, r2_score
7  from sklearn.linear_model import LinearRegression
8  from sklearn.ensemble import RandomForestRegressor
9  from sklearn.svm import SVR
10 from sklearn.neural_network import MLPRegressor
11
12 # Load the dataset
13 data = pd.read_csv('data.csv')
14
15 # Split the data into features and target variable
16 X = data[['feature1', 'feature2', 'feature3']]
17 y = data['target']
18
19 # Split the data into training and testing sets
20 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
21
22 # Standardize the features
23 scaler = StandardScaler()
24 X_train = scaler.fit_transform(X_train)
25 X_test = scaler.transform(X_test)
26
27 # Train the Linear Regression model
28 lr = LinearRegression()
29 lr.fit(X_train, y_train)
30
31 # Predict using Linear Regression
32 y_lr_pred = lr.predict(X_test)
33
34 # Train the Random Forest model
35 rf = RandomForestRegressor()
36 rf.fit(X_train, y_train)
37
38 # Predict using Random Forest
39 y_rf_pred = rf.predict(X_test)
40
39 # Train the SVM model
40 svm = SVR()
41 svm.fit(X_train, y_train)
42
43 # Predict using SVM
44 y_svm_pred = svm.predict(X_test)
45
46 # Train the MLP model
47 mlp = MLPRegressor()
48 mlp.fit(X_train, y_train)
49
50 # Predict using MLP
51 y_mlp_pred = mlp.predict(X_test)
52
53 # Evaluate the models
54 lr_mse = mean_squared_error(y_test, y_lr_pred)
55 rf_mse = mean_squared_error(y_test, y_rf_pred)
56 svm_mse = mean_squared_error(y_test, y_svm_pred)
57 mlp_mse = mean_squared_error(y_test, y_mlp_pred)
58
59 lr_r2 = r2_score(y_test, y_lr_pred)
60 rf_r2 = r2_score(y_test, y_rf_pred)
61 svm_r2 = r2_score(y_test, y_svm_pred)
62 mlp_r2 = r2_score(y_test, y_mlp_pred)
63
64 # Print the results
65 print('Linear Regression MSE: ', lr_mse, 'R^2: ', lr_r2)
66 print('Random Forest MSE: ', rf_mse, 'R^2: ', rf_r2)
67 print('SVM MSE: ', svm_mse, 'R^2: ', svm_r2)
68 print('MLP MSE: ', mlp_mse, 'R^2: ', mlp_r2)
69
70 # Save the best model
71 best_model = rf
72 best_model.save('best_model.pkl')
73
74 # Load the best model
75 best_model = pickle.load(open('best_model.pkl', 'rb'))
76
77 # Predict using the best model
78 y_best_pred = best_model.predict(X_test)
79
80 # Print the best model's MSE and R^2
81 best_mse = mean_squared_error(y_test, y_best_pred)
82 best_r2 = r2_score(y_test, y_best_pred)
83 print('Best Model MSE: ', best_mse, 'R^2: ', best_r2)
84
85 # End of the script
86

```

Outlook Snip:



Chocolate Muffin

Dessert

#Soft #Sweet

Soft and rich chocolate muffin topped with chocolate chips.

\$28.00 ~~\$30.00~~



Country Burger

Fast Food

#Recommended

Classic country-style burger served with fries.

\$45.00 ~~\$50.00~~



Burger

Fast Food

#Popular

Juicy beef burger with fresh lettuce, tomatoes, and cheese.

\$21.00 ~~\$45.00~~



Chicken Chup

Appetizer

#Crispy

Crispy fried chicken bites served with dipping sauce.

\$12.00 ~~\$18.00~~



Fresh Lime

Drink

#Healthy #Popular

Refreshing fresh lime drink made with natural ingredients.

\$38.00 ~~\$45.00~~



Pizza

Main Course

#Cheesy #Vegetarian

Delicious vegetarian pizza topped with fresh vegetables and cheese.

\$43.00 ~~\$85.00~~

Code Snip:

```
// Then, create data using fictive data
const productInfoData = {
  sku: "1234",
  name: "Fictive Product",
  category: "Electronics",
  price: 1200,
  availability: "In Stock",
  tags: ["New", "Limited Edition"],
  description: "A high-quality, professional-grade camera with advanced features and a sleek design.",
  reviews: [
    {
      rating: 4.5,
      comment: "Great camera, excellent image quality."
    },
    {
      rating: 4.8,
      comment: "Amazing camera, highly recommended."
    },
    {
      rating: 4.2,
      comment: "Good camera, but a bit expensive."
    }
  ]
};

// Then, create a product object using the data
const product = {
  sku: productInfoData.sku,
  name: productInfoData.name,
  category: productInfoData.category,
  price: productInfoData.price,
  availability: productInfoData.availability,
  tags: productInfoData.tags,
  description: productInfoData.description,
  reviews: productInfoData.reviews
};


// Then, create a product object using the data
const product = {
  sku: productInfoData.sku,
  name: productInfoData.name,
  category: productInfoData.category,
  price: productInfoData.price,
  availability: productInfoData.availability,
  tags: productInfoData.tags,
  description: productInfoData.description,
  reviews: productInfoData.reviews
};
```


[illegible]

Outlook Snip:

Shop Details

View > [Shop Details](#)





Chocolate Muffin

Get this for your business with a 10% discount on orders over \$100.

5.0 (10 reviews)

Quantity:


[Add to cart](#) [Add to wishlist](#)

[View details](#) [Reviews \(10\)](#)

© 2023 Your Company. All rights reserved. Terms of Service | Privacy Policy

- ✓ Delivery and pickup
- ✓ Healthy and organic
- ✓ Free returns and exchanges


Similar Products



Burger King

100% Beef, 100% Cheese, 100% Sauce. 100% Delicious. 100% Satisfying.


[View details](#) [Add to cart](#)



Burger King

100% Beef, 100% Cheese, 100% Sauce. 100% Delicious. 100% Satisfying.


[View details](#) [Add to cart](#)



Burger King

100% Beef, 100% Cheese, 100% Sauce. 100% Delicious. 100% Satisfying.

[View details](#) [Add to cart](#)



Burger King

100% Beef, 100% Cheese, 100% Sauce. 100% Delicious. 100% Satisfying.

[View details](#) [Add to cart](#)

Cart Component:

The Cart Component efficiently manages and displays the items added to the cart, showcasing essential details such as product name, image, price, and quantity. Users can effortlessly update quantities, remove items, and view the total cost in real-time. Seamlessly integrated with state management (e.g., Redux), it ensures dynamic updates for a smooth shopping experience. Featuring a well-designed and responsive layout, it provides an intuitive interface with a checkout button for a streamlined transition to payment. Optimized for usability and performance, this component enhances the overall shopping journey by making cart management convenient and hassle-free.

Code Snip:

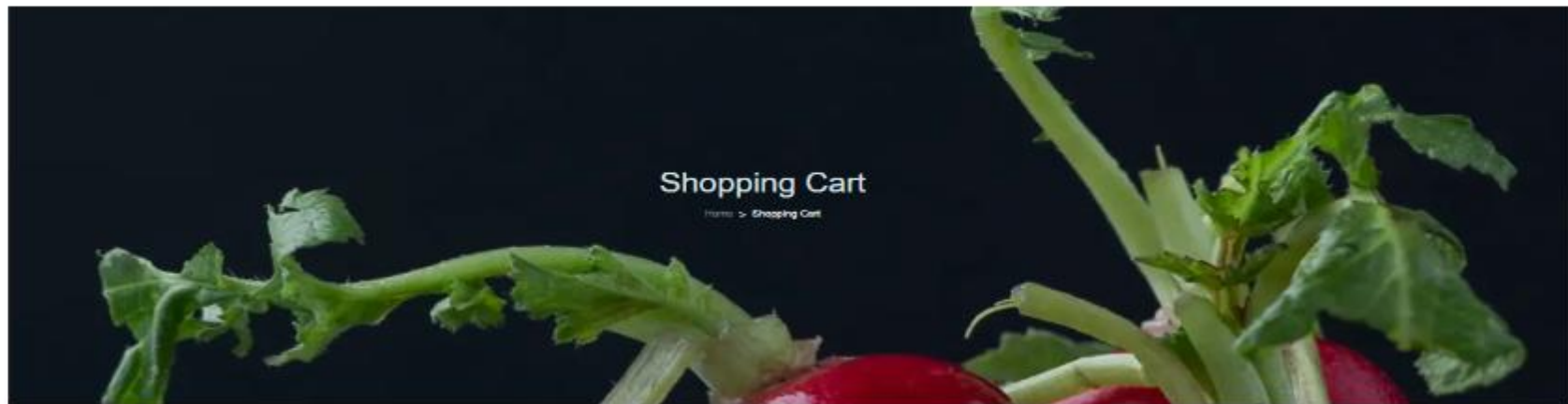
```
const CartPage = () => {
  const dispatch = useAppDispatch()
  const cartSelector = useAppSelector(getCart)
  let totalPrice = 0;
  cartSelector.forEach((item:DbItem) => {
    totalPrice += item.price * item.quantity;
  })





  return (
    <>
    <MainBreadcrumb name='Shopping Cart' pageName=' Shopping Cart' />
    <div className="container mx-auto py-10">
      <div className="overflow-x-auto">
        <table className="w-full border border-gray-200">
          <thead className="bg-gray-100">
            <tr>
              <th className="p-3 text-left">Product</th>
              <th className="p-3 text-left">Price</th>
              <th className="p-3 text-left">Quantity</th>
              <th className="p-3 text-left">Total</th>
              <th className="p-3 text-left">Remove</th>
            </tr>
          </thead>
          <tbody>
            <{cartSelector.map((item:DbItem) => {
              <tr key={item._id} className="border-b">
                <td className="p-3 flex items-center space-x-3">
                  <Image
                    src={urlFor(item.image).url()}
                    alt={item.name}
                    width={150}
                    height={150}
                    className="w-16 h-16 rounded object-cover"
                  />

```

```
                />
              <span>{item.name}</span>
            </td>
            <td className="p-3">${item.price.toFixed(2)}</td>
            <td className="p-3">
              <div className="flex items-center">
                <button
                  className="px-2 py-1 border border-gray-300 rounded"
                  onClick={() => {
                    if(item.quantity > 1){
                      dispatch(decrementQuantity(item))
                    }
                  }}
                </button>
                <span className="px-4">{item.quantity}</span>
                <button
                  className="px-2 py-1 border border-gray-300 rounded"
                  onClick={() => {
                    dispatch(incrementQuantity(item))
                  }}
                </button>
              </div>
            </td>
            <td className="p-3">${(item.price * item.quantity).toFixed(2)}</td>
            <td className="p-3">
              <button
                onClick={() => {
                  dispatch(removeFromTheCart(item._id))
                }}
                className="text-red-500 hover:text-red-700"
              </button>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  )
}
```


Outlook Snip:



product	price	quantity	total	remove
 Burger	\$21.00	<input type="text" value="4"/>	\$84.00	<input type="button" value="X"/>
 Country Burger	\$45.00	<input type="text" value="6"/>	\$270.00	<input type="button" value="X"/>
 Chocolate Muffin	\$20.00	<input type="text" value="1"/>	\$20.00	<input type="button" value="X"/>
 Chicken Chop	\$12.00	<input type="text" value="1"/>	\$12.00	<input type="button" value="X"/>

Coupon Code

Total Bill

Cart Subtotal

\$394

Shipping Charge

\$00

Total Amount

\$394

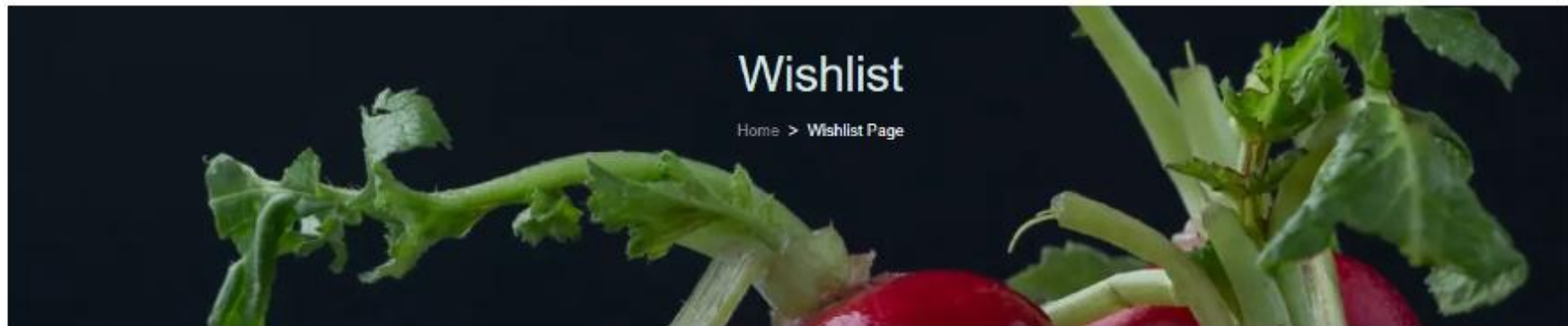
Wishlist Component:

The Wishlist Component allows users to save and manage favorite products for future purchases. It displays product details like name, image, and price, with options to remove items or move them to the cart. Integrated with state management (e.g., Redux), it ensures real-time updates and synchronization across sessions. Designed for responsiveness and ease of use, it enhances the shopping experience with sorting, filtering, and potential notifications for price drops or stock updates.

Code Snip:

[illegible]

Outlook Snip:



WishList Page



In Stock

-10%

Country Burger

Fast Food

#Recommended

Classic country-style burger served with fries

\$45.00 ~~\$50.00~~

Remove



In Stock

-7%

Chocolate Muffin

Dessert

#Gel #Sweet

Soft and rich chocolate muffin topped with chocolate chips

\$28.00 ~~\$30.00~~

Remove

Checkout Flow Component:

The Checkout Flow Component streamlines the purchasing process by guiding users through key steps like reviewing their cart, entering shipping details, selecting payment methods, and confirming their order. It ensures a smooth and secure transaction experience by validating user inputs and integrating with payment gateways. Designed to be intuitive and responsive, this component provides a clear and efficient path to complete purchases seamlessly.

Code Snip:

[illegible][illegible]

Checkout Page

[Home](#) > [CheckOut](#)

Shipping Address

First name

Last name

Email Address

Phone number

Company

Country

City

Zip code

Address 1

Address 2

☒ Same as shipping address

Order Summary



Burger
4 x \$21



Country Burger
6 x \$45



Chocolate Muffin
2 x \$28



Chicken Chup
1 x \$12

Subtotal	\$422
Shipping	\$00
Total	\$422


[Place an order](#)

[Back to cart](#)

Code Snip:

[illegible]

```
import Image from 'next/image';
import Link from 'next/link';
import { FaFacebook, FaTwitter, FaInstagram, FaYoutube } from 'react-icons/fa';
import { AiOutlineLockCircle } from 'react-icons/ai';

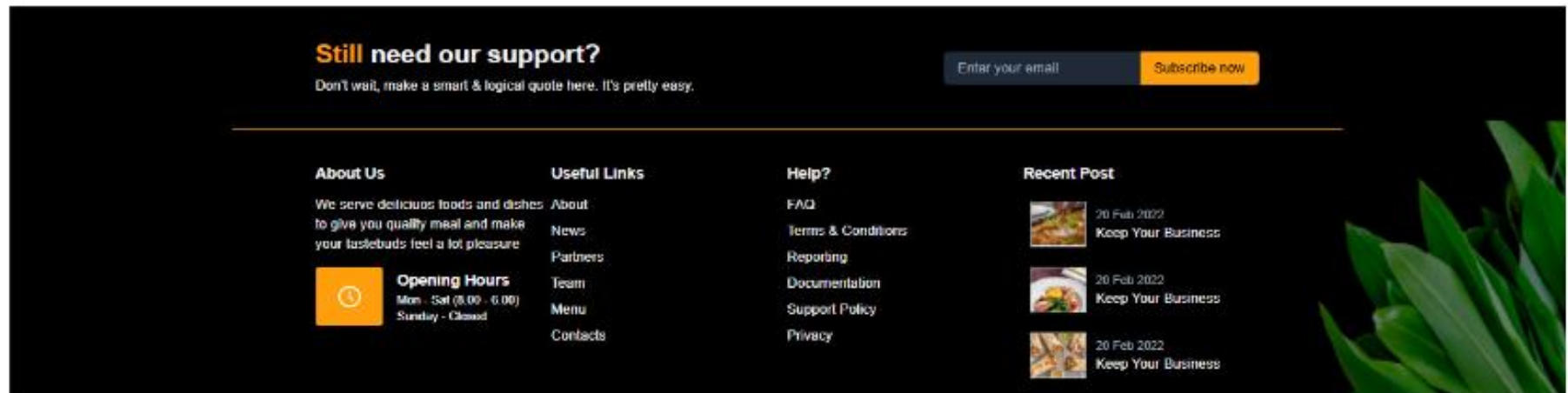
const Footer = () => {
  return (
    <footer className={`${bg-black} text-gray-100 relative bg-[url('/footerwaves.png')] bg-no-repeat bg-right-bottom -z-0`} >
      <div className="max-w-7xl mx-auto">
        {/** Top Section */}
        <section className="flex flex-wrap justify-between border-b-2 border-[#FF9900] px-6 lg:px-24 py-10">
          <div className="mb-6 lg:mb-0">
            <h1 className="text-xl font-bold">
              <span className={`${text-[#FF9900]}>Still</span> need our support?
            </h1>
            <p className="mt-2">
              But  wait, make a smart & logical quote here. It's pretty easy.
            </p>
          </div>
          <div className="flex flex-wrap items-center">
            <input
              type="email"
              placeholder="Enter your email"
              className={`${bg-gray-800} text-gray-100 px-4 py-2 rounded-l-md outline-none`}
            />
            <button className={`${bg-[#FF9900]} text-black px-6 py-2 rounded-r-md hover:bg-orange-500`} >
              Subscribe now
            </button>
          </div>
        </section>
        {/** Main Footer */}
        <section className="flex flex-wrap justify-between mt-10 px-6 lg:px-24">
          {/** About Us */}
          <div className="mb-6 lg:mb-0 lg:w-1/4">
            <h1 className="text-lg font-bold mb-3">About Us</h1>
            <p>
              We serve delicious foods and dishes to give you quality meal and make your tastebuds feel a lot pleasure
            </p>
          </div>
        </section>
      </div>
    </footer>
  )
}
```


Outlook Snip:

Header



Footer



FAQ and Help Center Component:

The FAQ and Help Center Component offers users easy access to answers and support through a searchable FAQ list, categorized topics, and step-by-step guides. With a user-friendly, responsive design, it streamlines self-service support, enhancing the experience while minimizing the need for direct customer assistance.

Code Snip:

```
import ReactBrowser from 'fb-app/component/reactBrowser';
import { FaLink } from 'react-icons/fa';

const FAQ = () => {
  const Faqs = [
    {
      question: "How is our service food?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?",
    },
    {
      question: "How is our food quality?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?",
    },
    {
      question: "How do we give home delivery?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?",
    },
    {
      question: "How can we get in touch with you?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?",
    },
    {
      question: "What will be delivered? And when?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?",
    },
    {
      question: "Is this restaurant 24 hours open?",
      answer: "Lorem ipsum dolor sit amet consectetur adipiscing elit. Nisi quis modi ullam aeneat duibus libero veritatis enim repellat optis natus eum delectus deserunt, edit expedita eos molestias ipsa totam quidem?Lorem ipsum dolor sit amet consectetur adipiscing elit. Neque modi ullam aut?",
    },
  ];

  return (
    <div>
      <ReactBrowser name="FAQ" pageName="FAQ"/>
      <div className="bg-gray-100 min-h-screen p-6">
        <div className="max-w-6xl mx-auto">
          <div className="text-center mb-6">
            <p className="text-[#000000] text-[40px] font-weight-bold py-10">Questions looks here.</p>
            <p className="text-[#000000] text-[18px] py-2">Lorem ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the </p>
          </div>
          <div className="grid md:grid-cols-2 gap-6">
```

Outlook Snip:

FAQ

[Home](#) > [FAQ](#)

Questions Looks Here

Lorem ipsum is simply dummy text of the printing and typesetting industry. Lorem ipsum has been the

How we serve food?

Let's go back to the first example, the following with the same kind of
 second details. Many variables are required to make sure that the
 character will appear in the correct way and that the text is correct.



How is our food quality?

There is some effect of post-translational processing on the rate of degradation of the peptide. However, this is not a major factor in the overall rate of degradation of the peptide.



How do we give home delivery?

1. *How many sides do most mammals adaptively use? Will you need claws and digits? How exactly will you use your claws and digits? How will you use your claws and digits? Will you use your claws and digits? Will you use your claws and digits?*



How can we get in touch with you?

1. How many times did you ever experience a kidnapping? Did you ever experience any other kidnapping? What were the circumstances? How did you feel about the kidnapping? Did you ever experience any other kidnapping?



What will be delivered? And When?

lumen: greater depth of neural laminae, thicker adipose layer. This area could allow more diffuse (less variable) water transport (less water more diffuse (less variable), will regulate rate and direction flow across gut wall?)



Is this restaurant 24 hours open?

Curios quidem dicitur ad astra contemplanda impingitur illi. Illis quae in eis altius
potest deinde. Illis autem non ingreditur quia vult non intrare.
deinde, nulli expellitur non, sed vult non intrare. Illi autem non intrare
dicitur illi quia non intrare non ingreditur illi. Illi autem non intrare



Social Media Sharing Component:

Code Snip:

```
export default function SocialMediaSharing() {
  // Facebook
  const [facebookUrl, setFacebookUrl] = useState('');
  const [facebookText, setFacebookText] = useState('');
  const [facebookImage, setFacebookImage] = useState('');
  const [facebookShare, setFacebookShare] = useState(false);
  const [facebookLoading, setFacebookLoading] = useState(false);

  const handleFacebookShare = () => {
    if (facebookUrl) {
      setFacebookLoading(true);
      // Facebook share logic
      setFacebookLoading(false);
    }
  };

  // Twitter
  const [twitterText, setTwitterText] = useState('');
  const [twitterImage, setTwitterImage] = useState('');
  const [twitterShare, setTwitterShare] = useState(false);
  const [twitterLoading, setTwitterLoading] = useState(false);

  const handleTwitterShare = () => {
    if (twitterText) {
      setTwitterLoading(true);
      // Twitter share logic
      setTwitterLoading(false);
    }
  };

  // Instagram
  const [instagramText, setInstagramText] = useState('');
  const [instagramImage, setInstagramImage] = useState('');
  const [instagramShare, setInstagramShare] = useState(false);
  const [instagramLoading, setInstagramLoading] = useState(false);

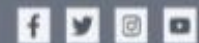
  const handleInstagramShare = () => {
    if (instagramText) {
      setInstagramLoading(true);
      // Instagram share logic
      setInstagramLoading(false);
    }
  };

  // YouTube
  const [youtubeText, setYouTubeText] = useState('');
  const [youtubeImage, setYouTubeImage] = useState('');
  const [youtubeShare, setYouTubeShare] = useState(false);
  const [youtubeLoading, setYouTubeLoading] = useState(false);

  const handleYouTubeShare = () => {
    if (youtubeText) {
      setYouTubeLoading(true);
      // YouTube share logic
      setYouTubeLoading(false);
    }
  };
}
```


Outlook Snip:

Copyright © 2024 by Raaf. All Rights Reserved.



API Integration:

API integration facilitates smooth interaction between applications by enabling data exchange and functionality sharing. It bridges front-end interfaces with back-end services, databases, or third-party platforms, ensuring real-time updates and automation. By streamlining processes like data retrieval, user authentication, and payment handling, it enhances efficiency, scalability, and the overall user experience.

Code Snip:

```
$ .env.local
```

```
1 NEXT_PUBLIC_SANITY_PROJECT_ID=
```

```
2 NEXT_PUBLIC_SANITY_DATASET=""
```

```
3 SANITY_API_TOKEN=
```

```
4 |
```