INTEGRATING SHIPENGINE IN A NEXT.JS PROJECT

Introduction

ShipEngine is a powerful API for shipping and tracking logistics. This document explains how to integrate ShipEngine into a Next.js project using React and Tailwind CSS to fetch shipping rates, create labels, and track shipments.

Prerequisites

- A Next.js project set up
- ShipEngine API key
- Axios for API requests
- React hooks for state management
- Tailwind CSS for styling

Setting Up the Shipping Component

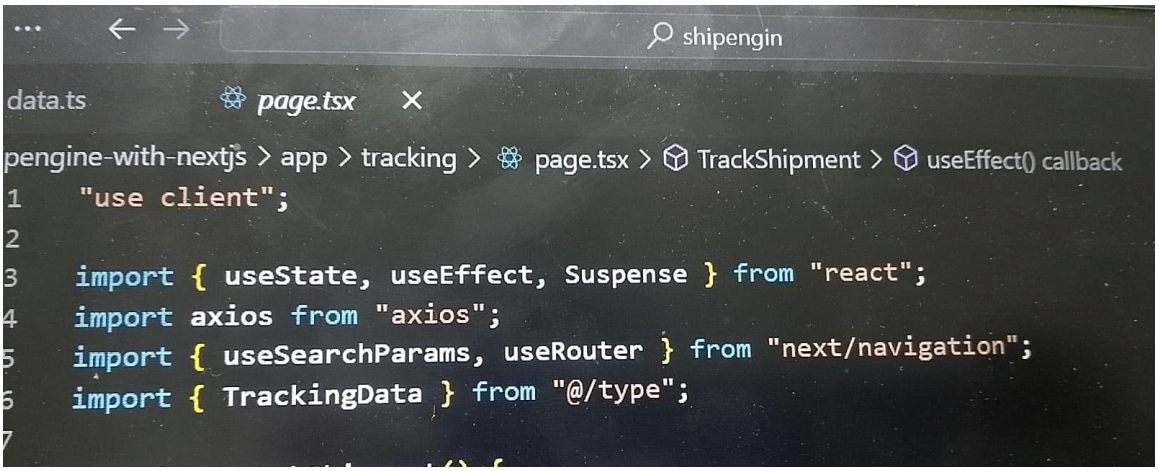
Importing Dependencies

```
"use client";

import React, { useState } from "react";

import axios from "axios";

import { Address, Rate, trackingObjType, Package } from "@/type";
```

import { cartProductsWhichCanBeShipped } from "@/data";

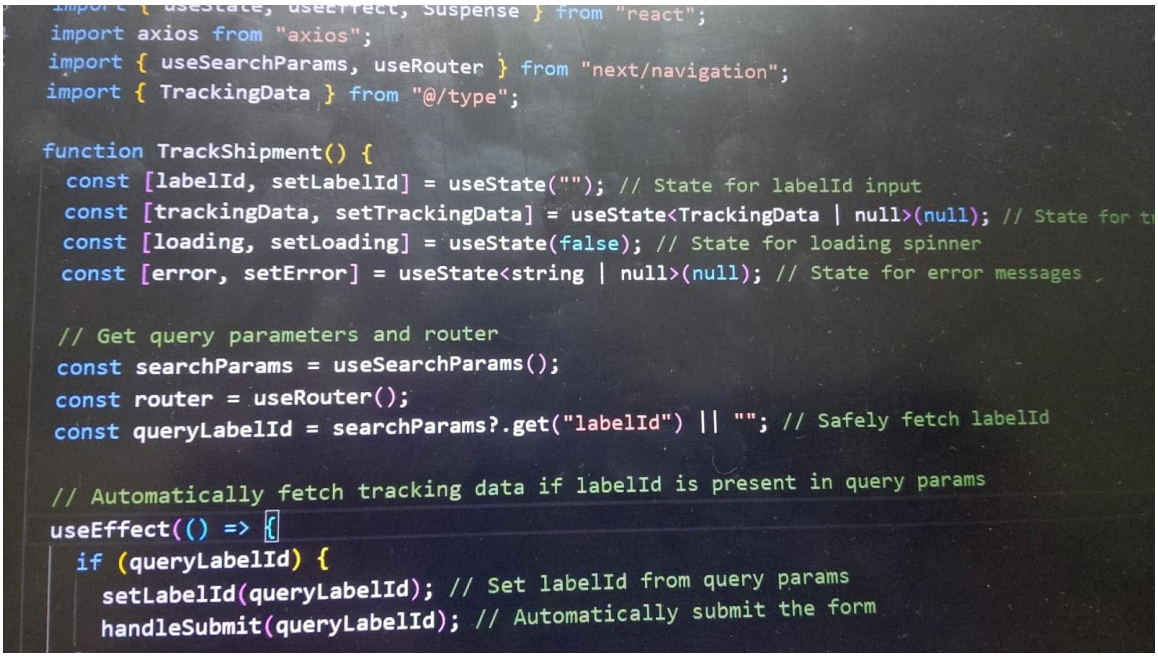
import Link from "next/link";

here is the picture



Defining State Variables

```
const [shipeToAddress, setshipeToAddress] = useState<Address>({...});

const [rates, setRates] = useState<Rate[]>([]);

const [rateId, setrateId] = useState<string | null>(null);

const [labelPdf, setLabelPdf] = useState<string | null>(null);

const [trackingObj, setTrackingObj] = useState<trackingObjType |
null>(null);

const [loading, setLoading] = useState(false);

const [errors, setErrors] = useState<string[]>([]);
```



```
import { useState, useEffect, Suspense } from "react";
import axios from "axios";
import { useSearchParams, useRouter } from "next/navigation";
import { TrackingData } from "@/type";

function TrackShipment() {
  const [labelId, setLabelId] = useState(""); // State for labelId input
  const [trackingData, setTrackingData] = useState<TrackingData | null>(null); // State for t
  const [loading, setLoading] = useState(false); // State for loading spinner
  const [error, setError] = useState<string | null>(null); // State for error messages

  // Get query parameters and router
  const searchParams = useSearchParams();
  const router = useRouter();
  const queryLabelId = searchParams?.get("labelId") || ""; // Safely fetch labelId

  // Automatically fetch tracking data if labelId is present in query params
  useEffect(() => {
    if (queryLabelId) {
      setLabelId(queryLabelId); // Set labelId from query params
      handleSubmit(queryLabelId); // Automatically submit the form
```

Fetching Shipping Rates

```
const handleSubmit = async (e: React.FormEvent) => {
    e.preventDefault();
    setLoading(true);
    setErrors([]);
    setRates([]);

    try {
        const response = await axios.post("/api/shipengine/get-rates", {
            shipeToAddress,
            packages: cartProductsWhichCanBeShipped.map((product) => ({
                weight: product.weight,
```

```
            dimensions: product.dimensions,
        })),
      });
      setRates(response.data.shipmentDetails.rateResponse.rates);
    } catch (error) {
      setErrors(["An error occurred while fetching rates."]);
    } finally {
      setLoading(false);
    }
};
```

Creating a Shipping Label

```
const handleCreateLabel = async () => {
    if (!rateId) {
      alert("Please select a rate to create a label.");
    }
    setLoading(true);
    setErrors([]);

    try {
```

```
      const response = await axios.post("/api/shipengine/label", { rateId });
      const labelData = response.data;
      setLabelPdf(labelData.labelDownload.href);
      setTrackingObj({
        trackingNumber: labelData.trackingNumber,
        labelId: labelData.labelId,
        carrierCode: labelData.carrierCode,
      });
    } catch (error) {
      setErrors(["An error occurred while creating the label."]);
    } finally {
      setLoading(false);
    }
};
```

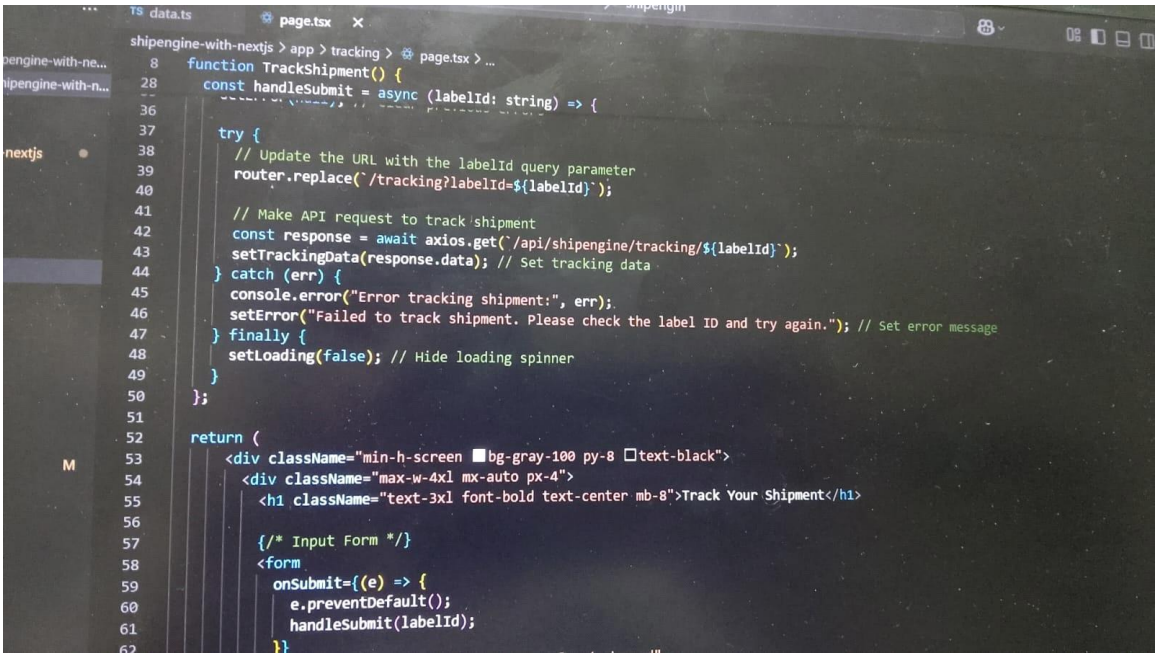
Tracking Shipments Page

```
"use client";
import { useState, useEffect, Suspense } from "react";
import axios from "axios";
import { useSearchParams, useRouter } from "next/navigation";
```

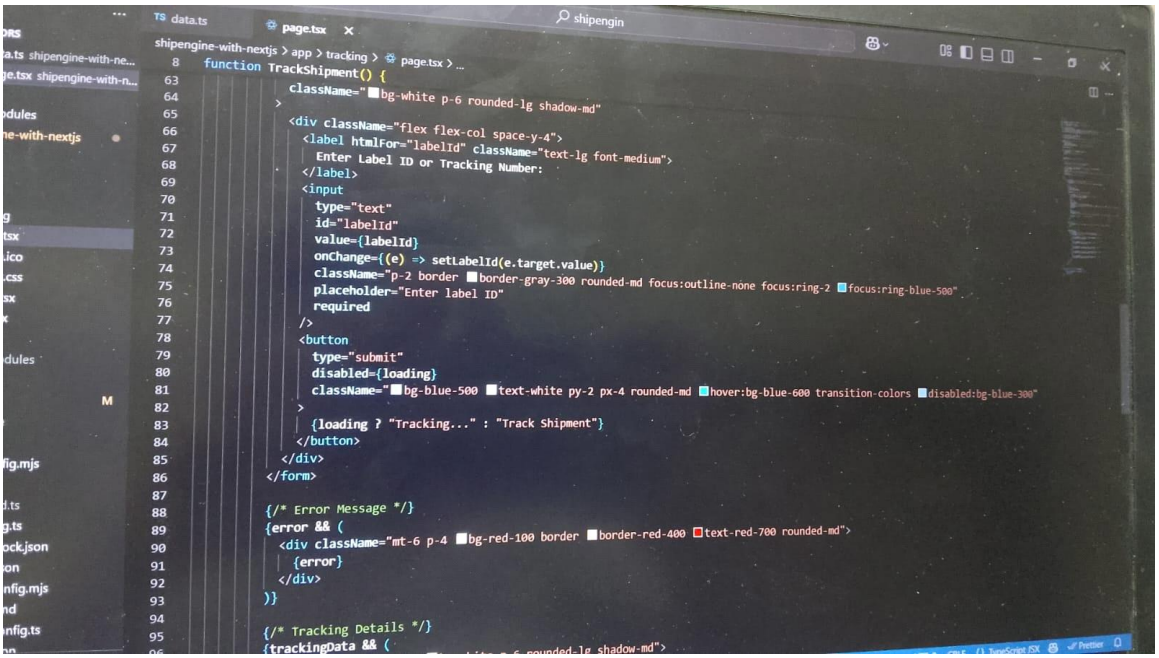
```
import { TrackingData } from "@/type";


function TrackShipment() {
   const [labelId, setLabelId] = useState("");
   const [trackingData, setTrackingData] = useState<TrackingData |
null>(null);
   const [loading, setLoading] = useState(false);
   const [error, setError] = useState<string | null>(null);
   const searchParams = useSearchParams();
   const router = useRouter();
   const queryLabelId = searchParams?.get("labelId") || "";


   useEffect(() => {
      if (queryLabelId) {
         setLabelId(queryLabelId);
         handleSubmit(queryLabelId);
      }
   }, [queryLabelId]);
```

```
  const handleSubmit = async (labelId: string) => {
    if (!labelId) {
      setError("Label ID is required.");
      return;
    }
    setLoading(true);
    setError(null);
    try {
      router.replace(`/tracking?labelId=${labelId}`);
      const response = await
axios.get(`/api/shipengine/tracking/${labelId}`);
      setTrackingData(response.data);
    } catch (err) {
      setError("Failed to track shipment. Please check the label ID and try
again.");
    } finally {
      setLoading(false);
    }
  };
```

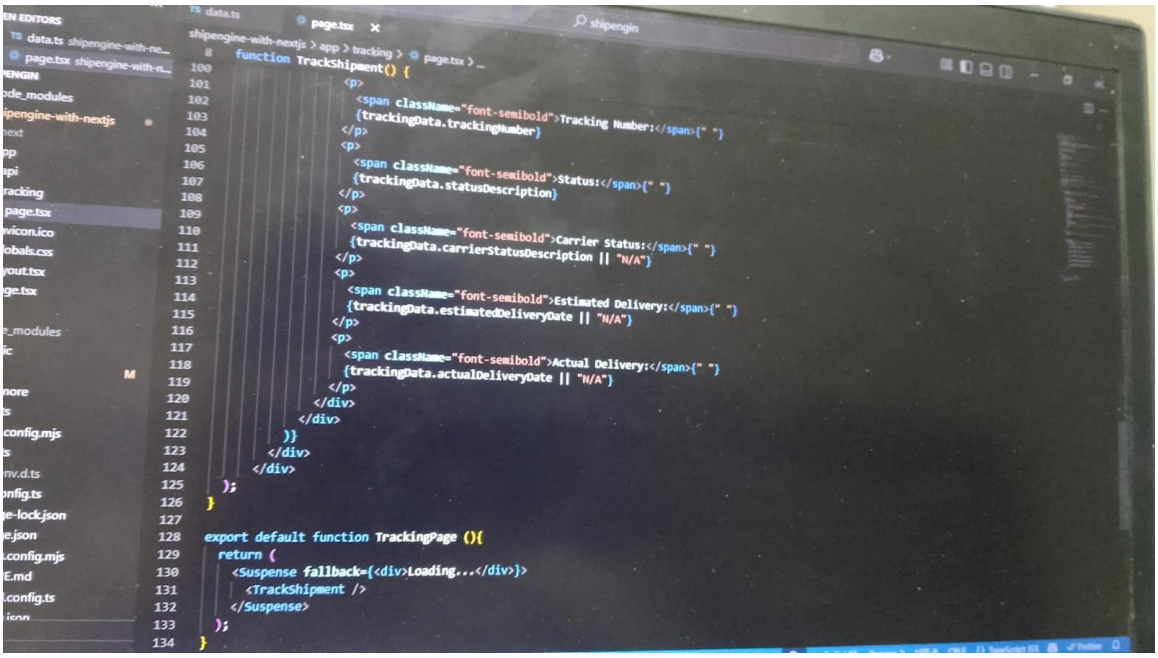
```
  return (
    <Suspense fallback={<div>Loading...</div>}>
      <div>
        <h1>Track Your Shipment</h1>
        <input type="text" value={labelId} onChange={(e) =>
setLabelId(e.target.value)} placeholder="Enter label ID" />
        <button onClick={() => handleSubmit(labelId)}>Track</button>
        {error && <p>{error}</p>}
        {trackingData && <p>Tracking Number:
{trackingData.trackingNumber}</p>}
      </div>
    </Suspense>
  );
}

export default TrackShipment;
```



```
shipengine-with-nextjs > app > tracking > page.tsx > ...
8   function TrackShipment() {
28    const handleSubmit = async (labelId: string) => {
36
37      try {
38        // Update the URL with the labelId query parameter
39        router.replace(`/tracking?labelId=${labelId}`);
40
41        // Make API request to track shipment
42        const response = await axios.get(`/api/shipengine/tracking/${labelId}`);
43        setTrackingData(response.data); // Set tracking data
44      } catch (err) {
45        console.error("Error tracking shipment:", err);
46        setError("Failed to track shipment. Please check the label ID and try again."); // Set error message
47      } finally {
48        setLoading(false); // Hide loading spinner
49      }
50    };
51
52    return (
53      <div className="min-h-screen bg-gray-100 py-8 text-black">
54        <div className="max-w-4xl mx-auto px-4">
55          <h1 className="text-3xl font-bold text-center mb-8">Track Your Shipment</h1>
56
57          {/* Input Form */}
58          <form
59            onSubmit={(e) => {
60              e.preventDefault();
61              handleSubmit(labelId);
62            }}
```



```
shipengine-with-nextjs > app > tracking > page.tsx > ...
8   function TrackShipment() {
63              className="bg-white p-6 rounded-lg shadow-md"
64            >
65              <div className="flex flex-col space-y-4">
66                <label htmlFor="labelId" className="text-lg font-medium">
67                  Enter Label ID or Tracking Number:
68                </label>
69                <input
70                  type="text"
71                  id="labelId"
72                  value={labelId}
73                  onChange={(e) => setLabelId(e.target.value)}
74                  className="p-2 border border-gray-300 rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500"
75                  placeholder="Enter label ID"
76                  required
77                />
78                <button
79                  type="submit"
80                  disabled={loading}
81                  className="bg-blue-500 text-white py-2 px-4 rounded-md hover:bg-blue-600 transition-colors disabled:bg-blue-300"
82                >
83                  {loading ? "Tracking..." : "Track Shipment"}
84                </button>
85              </div>
86            </form>
87
88            {/* Error Message */}
89            {error && (
90              <div className="mt-6 p-4 bg-red-100 border border-red-400 text-red-700 rounded-md">
91                {error}
92              </div>
93            )}
94
95            {/* Tracking Details */}
            {trackingData && (
```



```
function TrackShipment() {
                    <p>
                      <span className="font-semibold">Tracking Number:</span>{" "}
                      {trackingData.trackingNumber}
                    </p>
                    <p>
                      <span className="font-semibold">Status:</span>{" "}
                      {trackingData.statusDescription}
                    </p>
                    <p>
                      <span className="font-semibold">Carrier Status:</span>{" "}
                      {trackingData.carrierStatusDescription || "N/A"}
                    </p>
                    <p>
                      <span className="font-semibold">Estimated Delivery:</span>{" "}
                      {trackingData.estimatedDeliveryDate || "N/A"}
                    </p>
                    <p>
                      <span className="font-semibold">Actual Delivery:</span>{" "}
                      {trackingData.actualDeliveryDate || "N/A"}
                    </p>
                  </div>
                </div>
              )}
            </div>
          </div>
  );
}

export default function TrackingPage (){
  return (
    <Suspense fallback={<div>Loading...</div>}>
      <TrackShipment />
    </Suspense>
  );
}
```

Conclusion

This integration allows fetching shipping rates, generating shipping labels, and tracking shipments efficiently using ShipEngine in a Next.js project. You can further enhance it by implementing user authentication and advanced error handling.