Stripe Payment Integration Setup

react ts and node take payment with stripe project

Creating a project that takes payments using React, TypeScript (TS), and Node.js with Stripe involves several steps. Below is a high-level overview of the process:

Prerequisites

- **Node.js** installed.
- **React** environment set up.
- A **Stripe** account.

Project Structure

Here's a simple structure for the project:

```lua

/my-stripe-project
|-- /client (React app)
|   |-- /src
|   |-- /components
|   |-- App.tsx
|   |-- index.tsx
|-- /server (Node.js app)
|   |-- /src
|   |-- /routes
|   |-- index.ts
|-- package.json
```

Step 1: Initialize the Project

1. **Create the project directory:**

   ```bash
   mkdir my-stripe-project
   cd my-stripe-project
   ```

2. **Set up the server:**

```bash
mkdir server
cd server
npm init -y
npm install express stripe body-parser cors dotenv
```

3. **Set up the client:**

```bash
npx create-react-app client --template typescript
cd client
npm install @stripe/stripe-js @stripe/react-stripe-js axios
```

Step 2: Backend - Node.js/Express with Stripe

1. **Set up the server (`index.ts`):**
   - Create a `server/src/index.ts` file.
   - Install TypeScript and other dev dependencies:

```bash
npm install typescript ts-node @types/node @types/express @types/cors --save-dev
```

   - Create a `tsconfig.json` file:

```json
{
  "compilerOptions": {
    "target": "ES6",
    "module": "commonjs",
    "strict": true,
    "esModuleInterop": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true
  }
}
```

   - Implement the server code:

typescript

```typescript
import express from 'express';
import Stripe from 'stripe';
import bodyParser from 'body-parser';
import cors from 'cors';
import dotenv from 'dotenv';

dotenv.config();

const app = express();
const stripe = new Stripe(process.env.STRIPE_SECRET_KEY!, {
  apiVersion: '2022-11-15',
});

app.use(cors());
app.use(bodyParser.json());

app.post('/create-payment-intent', async (req, res) => {
  const { amount } = req.body;

  try {
    const paymentIntent = await stripe.paymentIntents.create({
      amount,
      currency: 'usd',
    });

    res.status(200).send({
      clientSecret: paymentIntent.client_secret,
    });
  } catch (error) {
    res.status(500).send({ error: error.message });
  }
});

app.listen(5000, () => console.log('Server running on port 5000'));
```

- Run the server:

bash

```bash
npx ts-node src/index.ts
```

2. **Environment variables (`.env`):**
   - In the `server` directory, create a `.env` file and add your Stripe secret key:

```env
STRIPE_SECRET_KEY=sk_test_...
```

Step 3: Frontend - React with Stripe

1. **Setup the payment form in `App.tsx` :**
   - Import the necessary libraries:

```typescript
import React, { useState } from 'react';
import { loadStripe } from '@stripe/stripe-js';
import { Elements, CardElement, useStripe, useElements } from
'@stripe/react-stripe-js';
import axios from 'axios';

const stripePromise = loadStripe('pk_test_...');

const CheckoutForm: React.FC = () => {
  const stripe = useStripe();
  const elements = useElements();
  const [error, setError] = useState(null);
  const [loading, setLoading] = useState(false);

  const handleSubmit = async (event: React.FormEvent) => {
    event.preventDefault();
    setLoading(true);

    if (!stripe || !elements) {
      return;
    }

    const cardElement = elements.getElement(CardElement);

    try {
      const { data: { clientSecret } } = await
axios.post('http://localhost:5000/create-payment-intent', {
        amount: 1000, // Amount in cents
```

```
      });

      const paymentResult = await stripe.confirmCardPayment(clientSecret, {
        payment_method: {
          card: cardElement!,
        },
      });

      if (paymentResult.error) {
        setError(paymentResult.error.message);
      } else {
        if (paymentResult.paymentIntent?.status === 'succeeded') {
          alert('Payment successful!');
        }
      }
    } catch (error) {
      setError(error.message);
    }

    setLoading(false);
  };

  return (
    <form onSubmit={handleSubmit}>
      <CardElement />
      <button type="submit" disabled={!stripe || loading}>
        Pay
      </button>
      {error && <div>{error}</div>}
    </form>
  );
};

const App: React.FC = () => {
  return (
    <Elements stripe={stripePromise}>
      <CheckoutForm />
    </Elements>
  );
};

export default App;
```

2. **Run the React application:**

```bash
npm start
```

Step 4: Testing and Deployment

- **Test** your application locally using test card numbers provided by Stripe.
- **Deploy** the server and client to a hosting service like Heroku, Vercel, or Netlify.

This guide should help you set up a simple payment system using React, TypeScript, Node.js, and Stripe. If you have any specific issues or questions, feel free to ask!