Headless Ecommerce:

Headless Ecommerce is basically use of Backend api of some provider like shopify and frontend is made by developers themselves. JAMSTACK is best for Ecommerce now a days as it is fast and SEO friendly

Project-01 (Gatsby-stripe-checkout)

This is the most easiest method for connecting payment gateways. Stripe provide everything like payment gateway, checkout page etc. No custom api creation required or no server is involved. Client side is directly connecting to stripe. Means checkout page will be of stripe

*Creating Project:*

gatsby new gatsby-ecommerce-1-stripe-checkout <https://github.com/gatsbyjs/gatsby-starter-minimal>

*Starting project:*

netlify dev

or,

gatsby develop

*Tutorial to be followed:*

[Gatsby E-commerce Tutorial | Gatsby (gatsbyjs.com)](https://www.gatsbyjs.com/tutorial/ecommerce-tutorial/)

Clean index.js file like below:

import \* as React from "react"

// markup

const IndexPage = () => {

  return (

    <div>

      Hello world

    </div>

  )

}

export default IndexPage

*Dependencies*

Npm I @stripe/stripe-js

Npm I dotenv

npm install gatsby-source-stripe

npm i --save stripe (stripe for server side)

*Get your Api or published key:*

Developers -> api (link: [API keys – personal – Stripe [Test]](https://dashboard.stripe.com/test/apikeys) )

pk\_test\_51JhERSGkYc9g1BvdAdto0rO94IZQ6BAqLaShXV1pie6mHqnY0dTObrEPGgBsZRffjvKBvB2SBviBv1FIPQ1Mlpit00Yw4DxR4C

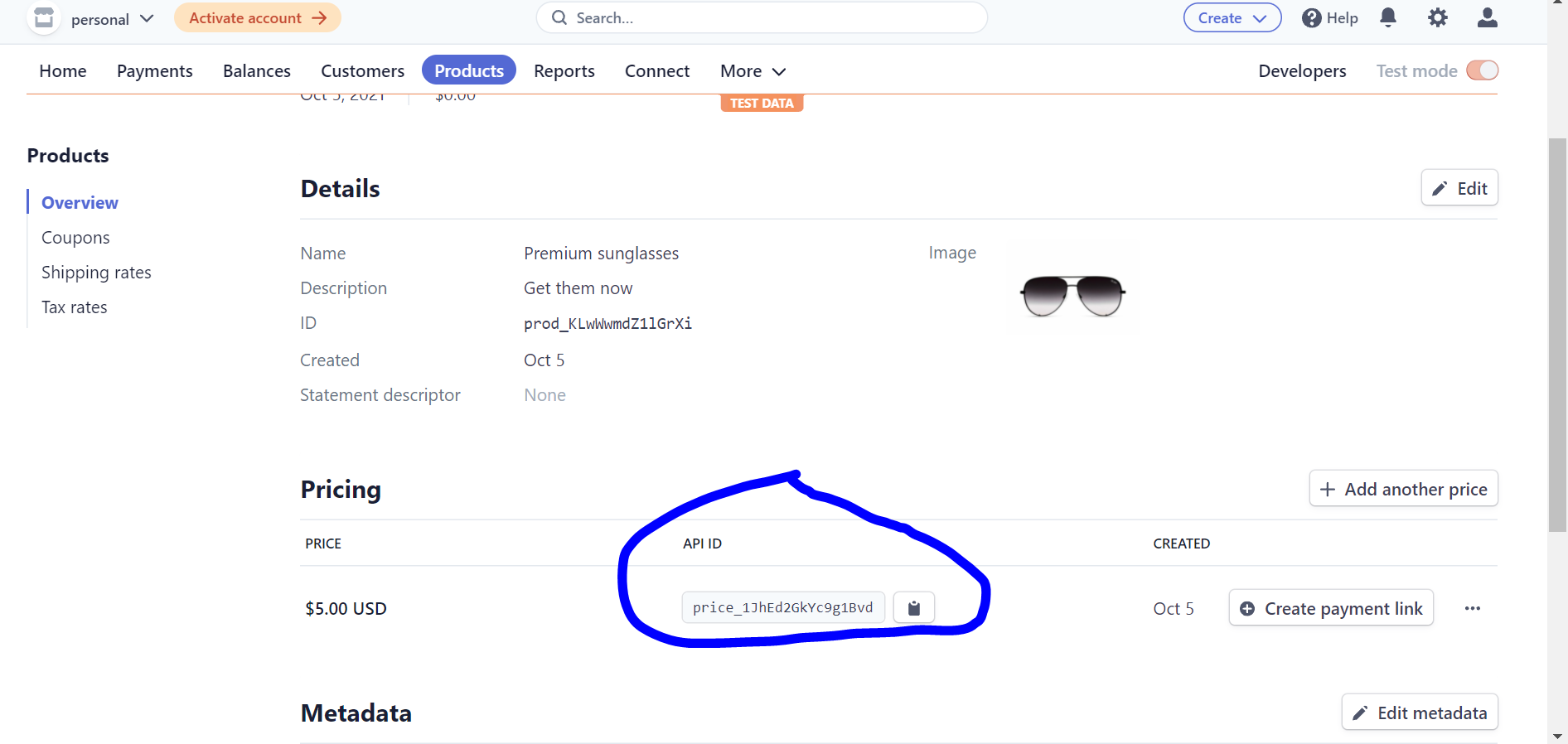
*Important thing:*

1. Create account on stripe
2. Go to below link and as we are integrating client side to stripe, so enable or click on client integration. After that first need to create a product

[Checkout and Payment Links – New Business – Stripe [Test]](https://dashboard.stripe.com/settings/checkout)

1. Go to this link to create your first product. Create 2 3 test products

[Checkout and Payment Links – New Business – Stripe [Test]](https://dashboard.stripe.com/settings/checkout)



After creating product, the highlighted api id is very much important

*First Part:*

Simple stripe checkout page containing all products there

Main file is index.js (other are payment-success and payment-error)

To get stripe checkout page or basic payment gateway code:

import \* as React from "react"

import { loadStripe } from "@stripe/stripe-js"

const stripePromise = loadStripe("pk\_test\_51JhERSGkYc9g1BvdAdto0rO94IZQ6BAqLaShXV1pie6mHqnY0dTObrEPGgBsZRffjvKBvB2SBviBv1FIPQ1Mlpit00Yw4DxR4C")

// markup

const IndexPage = () => {

  const redirectToCheckout = async () => {

    const stripe = await stripePromise

    const result = await stripe.redirectToCheckout({

      mode: "payment",

      lineItems: [

        { price: "price\_1JhEhQGkYc9g1Bvdhk8kZJ8z", quantity: 3 },

        { price: "price\_1JhEd2GkYc9g1Bvd737V5xW2", quantity: 6 }

    ], //array of objects

      successUrl: `http://localhost:8888/payment-success/`,

      cancelUrl: `http://localhost:8888/payment-error`,

    })

  }

  return (

    <div>

      <div>

      Hello world

      </div>

      <button onClick={redirectToCheckout}>Checkout</button>

    </div>

  )

}

export default IndexPage

Updated version of above with location.origin to avoid hardcoded url, it will by default take whatever url is using

import \* as React from "react"

import { loadStripe } from "@stripe/stripe-js"

const stripePromise = loadStripe("pk\_test\_51JhERSGkYc9g1BvdAdto0rO94IZQ6BAqLaShXV1pie6mHqnY0dTObrEPGgBsZRffjvKBvB2SBviBv1FIPQ1Mlpit00Yw4DxR4C")

// markup

const IndexPage = ({location}) => {

  const redirectToCheckout = async () => {

    const stripe = await stripePromise

    const result = await stripe.redirectToCheckout({

      mode: "payment",

      lineItems: [

        { price: "price\_1JhEhQGkYc9g1Bvdhk8kZJ8z", quantity: 3 },

        { price: "price\_1JhEd2GkYc9g1Bvd737V5xW2", quantity: 6 }

    ], //array of objects

      successUrl: `${location.origin}/payment-success/`,

      cancelUrl: `${location.origin}/payment-error`,

    })

  }

  return (

    <div>

      <div>

      Hello world

      </div>

      <button onClick={redirectToCheckout}>Checkout</button>

    </div>

  )

}

export default IndexPage

Also create payment-succeful.js and payment-error.js so to redirect if payment succeful or failed

Under lineitems, priceid is basically api id from product page as discussed earlier and passed public key to loadstripe

*How to Update Price:*

Go to product, then click on add price and add new price their and old price also their, similarly we can also choose some different currency. After that it will again give the new product Id which we can pass under LineItems

*SECOND PART:*

Previously we directly have checkout button and on clicking that we get products there Now we need to fetch our all products from stripe on some page so user can choose to buy which product he wants, so we need to fetch data from stripe

Main file is productList.js (other are payment-success and payment-error)

Now install dependency -> npm I Gatsby-source-stripe

Now add plugin in Gatsby-config.js

Gatsby-config.js

require("dotenv").config({

    path: `.env.${process.env.NODE\_ENV}`,

  })

module.exports = {

    siteMetadata: {

        siteUrl: `https://www.yourdomain.tld`,

    },

    plugins: [

        {

          resolve: `gatsby-source-stripe`,

          options: {

            objects: ["Price"],

            secretKey: process.env.STRIPE\_SECRET\_KEY,

            downloadFiles: false,

          },

        },

      ],

}

productList.js

As we want to see all products infront of us and checkout for that product. For that create productList.js file which can be route on localhost:8800/productList

import \* as React from "react"

import { loadStripe } from "@stripe/stripe-js"

import { useStaticQuery } from "gatsby"

import { graphql } from 'gatsby'

const stripePromise = loadStripe("pk\_test\_51JhERSGkYc9g1BvdAdto0rO94IZQ6BAqLaShXV1pie6mHqnY0dTObrEPGgBsZRffjvKBvB2SBviBv1FIPQ1Mlpit00Yw4DxR4C")

// markup

const ProductList = ({location}) => {

    const data = useStaticQuery(

        graphql`

            query ProductPrices {

                prices : allStripePrice {

                    edges {

                        node {

                            id

                            active

                            currency

                            unit\_amount

                            product {

                                id

                                name

                                images

                            }

                        }

                    }

                }

            }

        `

    );

console.log('Stripe data', data)

const redirectToCheckout = async (id) => {

    const stripe = await stripePromise

    const result = await stripe.redirectToCheckout({

      mode: "payment",

      lineItems: [

        { price: id, quantity: 1 },

    ], //array of objects

      successUrl: `${location.origin}/payment-success/`,

      cancelUrl: `${location.origin}/payment-error`,

    })

  }

  return (

    <div style={{margin: "0px", backgroundColor: "lightgray", width:"100wh", height:"100vh", display:"flex", flexDirection:"column", alignItems:"center"}}>

      <div>

        <h1 style={{display:"flex", alignItems:"center", justifyContent:"center"}}>Product List</h1>

        <div style={{display:"flex", alignItems:"center"}}>

            {data.prices.edges.map((edge)=>(

                //console.log(edge.node.product.name)

                <div style={{backgroundColor: "White", borderRadius: "5px", padding:"10px", margin:"10px", displayL:"flex", flexDirection:"column", alignItems:"center", justifyContent:"center"}} key={edge.node.product.id}>

                    <img style={{display:"flex", alignItems:"center", margin:"auto"}}width="100px" height="100px" src={edge.node.product.images[0]} alt=""/>

                    <h2>Product Name: <span style={{fontSize: "20px"}}>{edge.node.product.name}</span></h2>

                    <h3>Product Price: <span style={{fontSize: "16px"}}>{edge.node.unit\_amount}</span></h3>

                    <button style={{cursor:"pointer", display:"flex", alignItems:"center", justifyContent:"center", margin:"auto", backgroundColor: "teal", border: "none", borderRadius:"5px", padding: "7px 10px", color: "white", fontSize:"16px", fontWeight: "500"}} onClick={()=>redirectToCheckout(edge.node.id)}>Checkout</button>

                </div>

            ))}

        </div>

      </div>

    </div>

  )

}

export default ProductList

In this we are fetching data from stripe and then simply mapping on it to extract the data, file is simple just has inline styling to it. This is our main file now, not index.js is our main file

*Third Part:*

In this, we put our redirectToCheckout on server which is functions in Gatsby

Create netlify.toml:

[build]

command = "yarn build" or “gatsby build <for this build as not deploying>”

functions = “functions”

publish = "public" or “build <for react>”

Create functions folder:

Inside that -> netlify functions:create checkout (And select hello world, this will give template checkout.js file)

Main files are checkout.js (server) and checkout-session.js (client)

Dependency:

npm i --save stripe (installing stripe for server side)

checkout.js

// Docs on event and context https://www.netlify.com/docs/functions/#the-handler-method

const stripe = require('stripe')(process.env.STRIPE\_SECRET\_KEY);

const handler = async (event) => {

  const session = await stripe.checkout.sessions.create({

    payment\_method\_types: ['card'],

    line\_items: [{

      price\_data: {

        currency: 'usd',

        product\_data: {

          name: 'T-shirt',

          images: ["https://ae01.alicdn.com/kf/HTB1eB\_4XtjvK1RjSspiq6AEqXXaU/Fengdong-fashion-school-backpack-for-girls-school-bags-female-bagpack-children-backpacks-kids-cute-USB-bag.jpg"]

        },

        unit\_amount: 2000,

      },

      quantity: 1,

    }],

    mode: "payment",

    successUrl: `http://localhost:8888/payment-success/`,

    cancelUrl: `http://localhost:8888/payment-error`,

  });

  try {

    return {

      statusCode: 200,

      body: JSON.stringify({ id: session.id }),

      // // more keys you can return:

      // headers: { "headerName": "headerValue", ... },

      // isBase64Encoded: true,

    }

  } catch (error) {

    return { statusCode: 500, body: error.toString() }

  }

}

module.exports = { handler }

checkout-session.js (client):

getting response from the server which is checkout file under functions

import React from "react"

import { loadStripe } from "@stripe/stripe-js"

const stripePromise = loadStripe("pk\_test\_51JhERSGkYc9g1BvdAdto0rO94IZQ6BAqLaShXV1pie6mHqnY0dTObrEPGgBsZRffjvKBvB2SBviBv1FIPQ1Mlpit00Yw4DxR4C")

export default function CheckoutSession({location}) {

    const redirectToCheckout = async ()=> {

        const stripe = await stripePromise;

        const response = await fetch('/.netlify/functions/checkout');

        const data = await response.json();

        const result = await stripe.redirectToCheckout({

            sessionId: data.id

        })

    }

  return (

      <div>

        <div>Hello Checkout Session</div>

        <button onClick={redirectToCheckout}>Checkout</button>

      </div>

    )

}