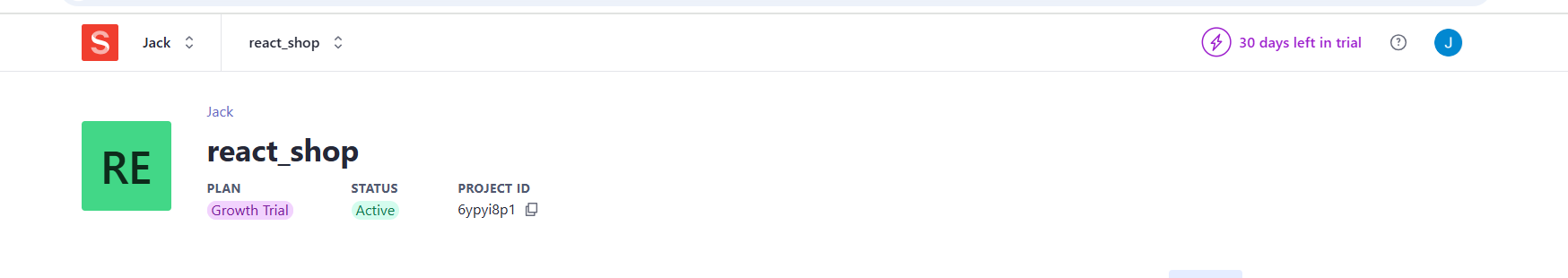
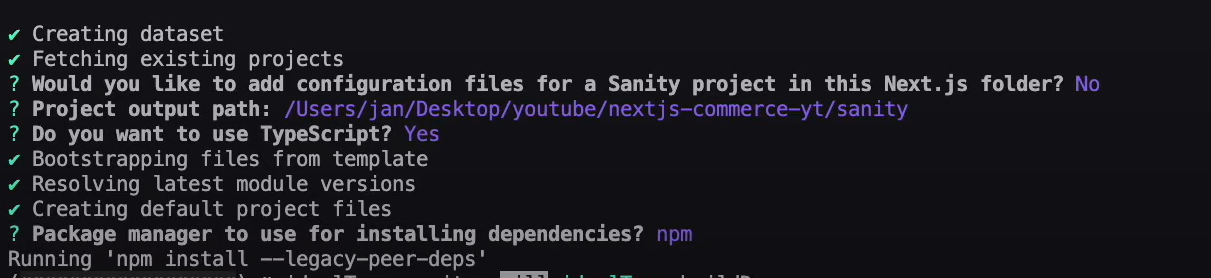
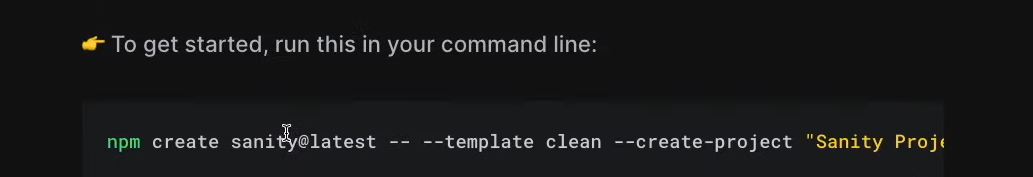
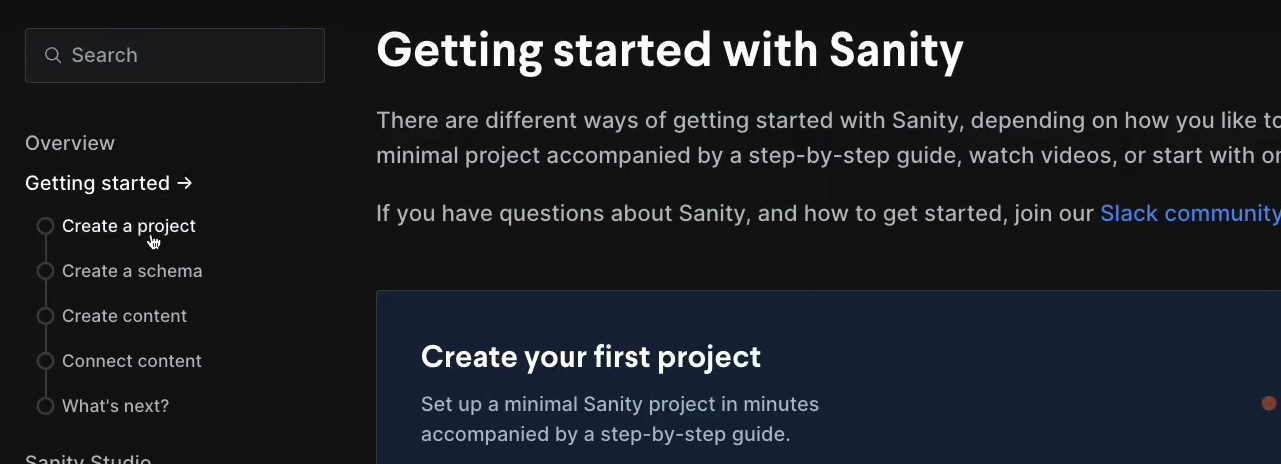
React專案連接sanity

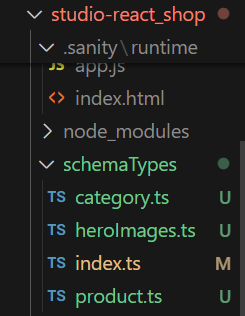
建立sanity專案



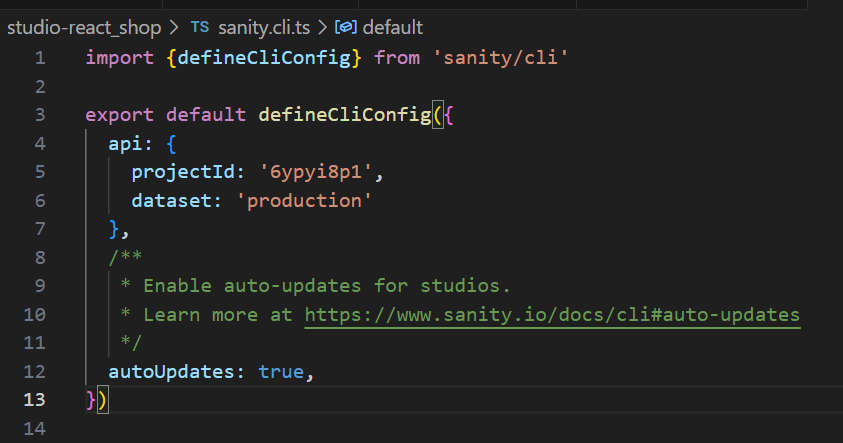
在終端機輸入指令安裝sanity的目錄

在sanity目錄建立category、product、heroImages.ts檔案

Index.ts主要為引入輸出檔案的



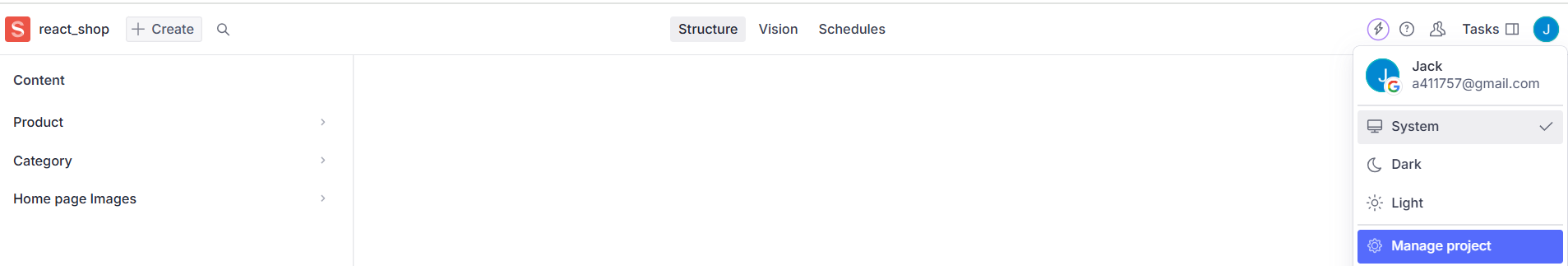
查看sanity內的sanity.cli.ts的projectId是否符合專案的id



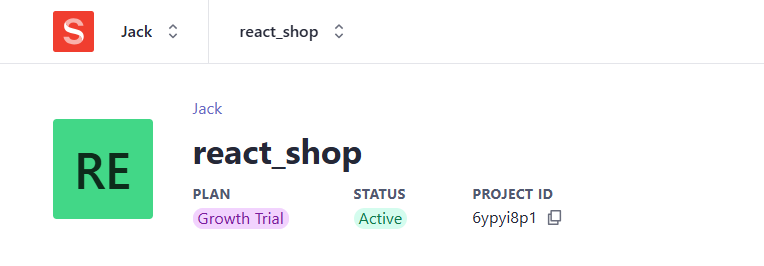
進入sanity目錄開啟專案



選manage可以查看專案id



專案id



目前sanity 專案schema內的product.ts

export default {

    name: 'product',

    type: 'document',

    title: 'Product',

    fields: [

      {

        name: 'name',

        type: 'string',

        title: 'Name of Product',

      },

      {

        name: 'images',

        type: 'array',

        title: 'Product Images',

        of: [{type: 'image'}],

      },

      {

        name: 'description',

        type: 'text',

        title: 'Description of product',

      },

      {

        name: 'slug',

        type: 'slug',

        title: 'Product Slug',

        options: {

          source: 'name',

        },

      },

      {

        name: 'price',

        title: 'Price',

        type: 'number',

      },

      {

        name: 'category',

        title: 'Product Category',

        type: 'reference',

        to: [

          {

            type: 'category',

          },

    ],

  }

]

}

Category.ts

export default {

    name: 'category',

    type: 'document',

    title: 'Category',

    fields: [

      {

        name: 'name',

        type: 'string',

        title: 'Category Name',

      },

      {

        name: 'description',

        type: 'text',

        title: 'Category Description',

      },

      {

        name: 'slug',

        type: 'slug',

        title: 'Category Slug',

        options: {

          source: 'name',

          maxLength: 96,

        },

      },

    ],

  };

heroImages.ts

export default {

    name: 'heroImage',

    type: 'document',

    title: 'Home page Images',

    fields: [

      {

        name: 'image1',

        type: 'image',

        title: 'First Image',

      },

      {

        name: 'image2',

        type: 'image',

        title: 'Second Image',

      },

    ],

  }

第一開始主要輸出首頁的Images，所以先建立Hero的元件輸出sanity的圖片。

先在sanity目錄新增sanity.ts檔案，建立url變數可以在Hero元件使用輸出圖片。



import { createClient } from '@sanity/client';

import imageUrlBuilder from '@sanity/image-url';

export const client = createClient({

  projectId: "6ypyi8p1",

  dataset: "production",

  apiVersion: "2022-03-25",

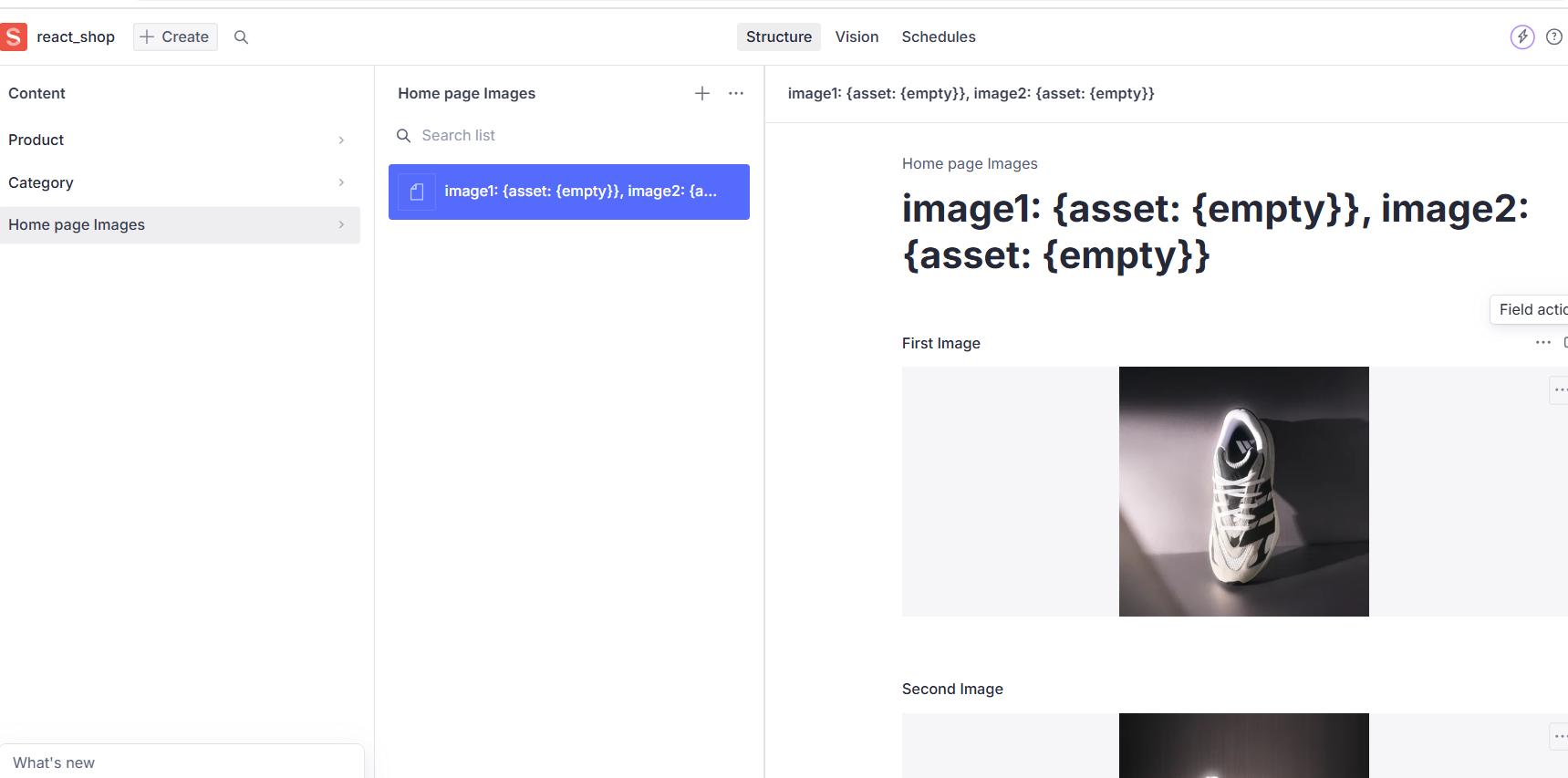
  useCdn: true,

});

const builder = imageUrlBuilder(client);

export const urlFor = (source) => builder.image(source);

在sanity的console引入兩個圖片



Hero元件引入

import  { useEffect, useState } from 'react';

import { client, urlFor } from '../../studio-react\_shop/sanity'; // 保留對 Sanity 的連結

const Hero = () => {

  const [data, setData] = useState(null);

  // 獲取數據

  useEffect(() => {

    async function getData() {

      const query = "\*[\_type == 'heroImage'][0]";

      const result = await client.fetch(query);

      setData(result);

    }

    getData();

  }, []);

  // 當數據尚未加載時，顯示載入狀態

  if (!data) {

    return <div>Loading...</div>;

  }

  return (

    <section className="mx-auto max-w-2xl px-4 sm:pb-6 lg:max-w-7xl lg:px-8">

      <div className="mb-8 flex flex-wrap justify-between md:mb-16">

        {/\* 文字部分 \*/}

        <div className="mb-6 flex w-full flex-col justify-center sm:mb-12 lg:mb-0 lg:w-1/3 lg:pb-24 lg:pt-48">

          <h1 className="mb-4 text-4xl font-bold text-black sm:text-5xl md:mb-8 md:text-6xl">

            Top Fashion for a top price!

          </h1>

          <p className="max-w-md leading-relaxed text-gray-500 xl:text-lg">

            We sell only the most exclusive and high-quality products for you.

            We are the best, so come and shop with us.

          </p>

        </div>

        {/\* 圖片部分 \*/}

        <div className="mb-12 flex w-full md:mb-16 lg:w-2/3">

          <div className="relative left-12 top-12 z-10 -ml-12 overflow-hidden rounded-lg bg-gray-100 shadow-lg md:left-16 md:top-16 lg:ml-0">

            <img

              src={urlFor(data.image1).url()}

              alt="Great Photo"

              className="h-full w-full object-cover object-center"

              width={400}

              height={500}

            />

          </div>

          <div className="overflow-hidden rounded-lg bg-gray-100 shadow-lg">

            <img

              src={urlFor(data.image2).url()}

              alt="Great Photo"

              className="h-full w-full object-cover object-center"

              width={500}

              height={500}

            />

          </div>

        </div>

      </div>

      {/\* 三個 Link 部分 \*/}

    </section>

  );

};

export default Hero;

安裝react-router-dom建立路由，先查看Navbar有幾個分頁建立元件。

用Navbar元件和Footer元件包住SetRoutes元件，這樣在SetRoutes內的所有頁面都會輸出Navbar和Footer元件。



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

import { SetRoutes } from "./SetRoutes";

import { Navbar } from "./components/Navbar";

import Footer from "./components/Footer";

const App = () => {

  return (

    <BrowserRouter>

      <Navbar /> {/\* Navbar在所有頁面中都會顯示 \*/}

      <SetRoutes />

      <Footer />  {/\* Footer在所有頁面中都會顯示 \*/}

    </BrowserRouter>

  );

};

export default App;

建立SetRoutes元件



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

import Home from "./components/Home";

export const SetRoutes = () => {

  return useRoutes([

    { path: "/", element: <Home /> },

  ]);

};

Navbar元件引入react-router-dom的Link元件



import { motion } from "framer-motion";

import { useState } from "react";

import { FiMenu, FiArrowRight } from "react-icons/fi";

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

export const Navbar = () => {

  return (

    <div className="bg-gray-50">

      <FlipNav />

      <div className="h-72" />

    </div>

  );

};

const FlipNav = () => {

  const [isOpen, setIsOpen] = useState(false);

  return (

    <nav className="bg-white p-4 border-b-[1px] border-gray-200 flex items-center justify-between relative">

      <NavLeft setIsOpen={setIsOpen} />

      <NavRight />

      <NavMenu isOpen={isOpen} />

    </nav>

  );

};

const Logo = () => {

  return (

    <svg

      width="50"

      height="39"

      viewBox="0 0 50 39"

      fill="none"

      xmlns="http://www.w3.org/2000/svg"

      className="fill-gray-800"

    >

      <path

        d="M16.4992 2H37.5808L22.0816 24.9729H1L16.4992 2Z"

        stopColor="#000000"

      ></path>

      <path

        d="M17.4224 27.102L11.4192 36H33.5008L49 13.0271H32.7024L23.2064 27.102H17.4224Z"

        stopColor="#000000"

      ></path>

    </svg>

  );

};

const NavLeft = ({ setIsOpen }) => {

  return (

    <div className="flex items-center gap-6">

      <motion.button

        whileHover={{ scale: 1.05 }}

        whileTap={{ scale: 0.95 }}

        className="block lg:hidden text-gray-950 text-2xl"

        onClick={() => setIsOpen((pv) => !pv)}

      >

        <FiMenu />

      </motion.button>

      <Logo />

      <NavLink to="/" text="Home" /> {/\* 使用 Link 來導航到首頁 \*/}

      <NavLink to="/solutions" text="Solutions" />

      <NavLink to="/community" text="Community" />

      <NavLink to="/pricing" text="Pricing" />

      <NavLink to="/company" text="Company" />

    </div>

  );

};

const NavLink = ({ to, text }) => {

  return (

    <Link

      to={to} // 使用 Link 元件的 `to` 屬性來導航

      className="hidden lg:block h-[30px] overflow-hidden font-medium"

    >

      <motion.div whileHover={{ y: -30 }}>

        <span className="flex items-center h-[30px] text-gray-500">{text}</span>

        <span className="flex items-center h-[30px] text-indigo-600">

          {text}

        </span>

      </motion.div>

    </Link>

  );

};

const NavRight = () => {

  return (

    <div className="flex items-center gap-4">

      <motion.button

        whileHover={{ scale: 1.05 }}

        whileTap={{ scale: 0.95 }}

        className="px-4 py-2 bg-gradient-to-r from-violet-600 to-indigo-600 bg-clip-text text-transparent font-medium rounded-md whitespace-nowrap"

      >

        Sign in

      </motion.button>

      <motion.button

        whileHover={{ scale: 1.05 }}

        whileTap={{ scale: 0.95 }}

        className="px-4 py-2 bg-gradient-to-r from-violet-600 to-indigo-600 text-white font-medium rounded-md whitespace-nowrap"

      >

        Sign up

      </motion.button>

    </div>

  );

};

const NavMenu = ({ isOpen }) => {

  return (

    <motion.div

      variants={menuVariants}

      initial="closed"

      animate={isOpen ? "open" : "closed"}

      className="absolute p-4 bg-white shadow-lg left-0 right-0 top-full origin-top flex flex-col gap-4"

    >

      <MenuLink to="/solutions" text="Solutions" />

      <MenuLink to="/community" text="Community" />

      <MenuLink to="/pricing" text="Pricing" />

      <MenuLink to="/company" text="Company" />

    </motion.div>

  );

};

const MenuLink = ({ to, text }) => {

  return (

    <motion.div

      variants={menuLinkVariants}

      className="h-[30px] overflow-hidden font-medium text-lg flex items-start gap-2"

    >

      <Link to={to}>

        <motion.span variants={menuLinkArrowVariants}>

          <FiArrowRight className="h-[30px] text-gray-950" />

        </motion.span>

        <motion.div whileHover={{ y: -30 }}>

          <span className="flex items-center h-[30px] text-gray-500">{text}</span>

          <span className="flex items-center h-[30px] text-indigo-600">{text}</span>

        </motion.div>

      </Link>

    </motion.div>

  );

};

export default Navbar;

const menuVariants = {

  open: {

    scaleY: 1,

    transition: {

      when: "beforeChildren",

      staggerChildren: 0.1,

    },

  },

  closed: {

    scaleY: 0,

    transition: {

      when: "afterChildren",

      staggerChildren: 0.1,

    },

  },

};

const menuLinkVariants = {

  open: {

    y: 0,

    opacity: 1,

  },

  closed: {

    y: -10,

    opacity: 0,

  },

};

const menuLinkArrowVariants = {

  open: {

    x: 0,

  },

  closed: {

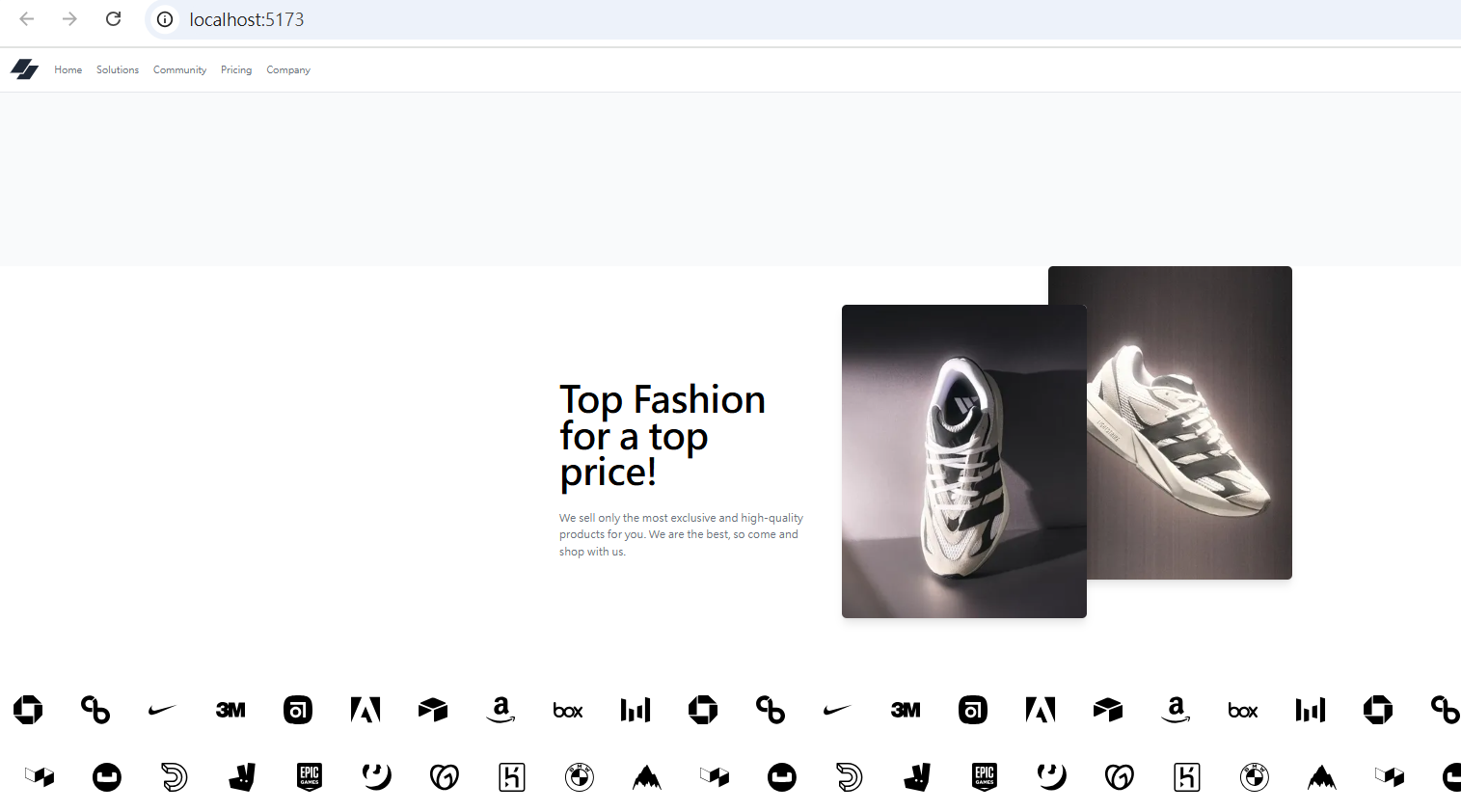
    x: -4,

  },

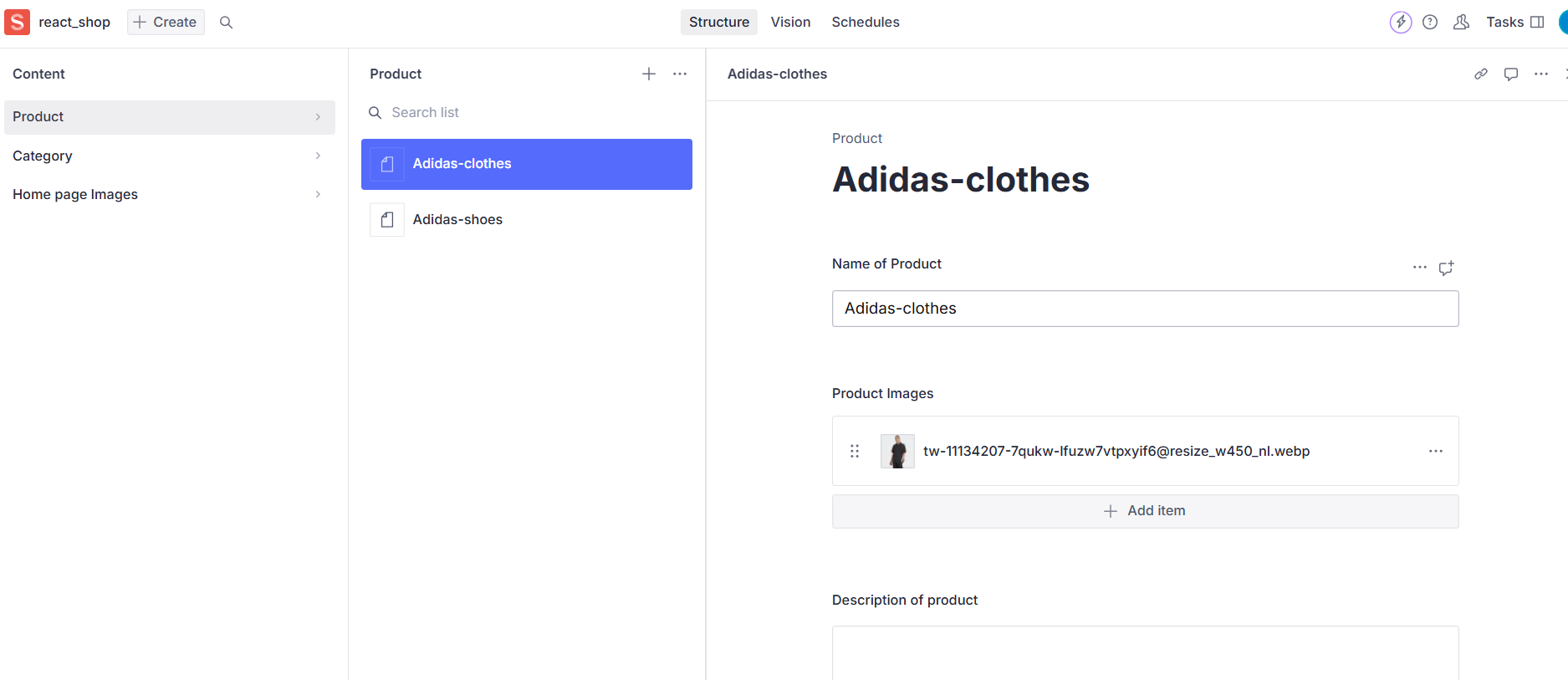
};

輸出:

點擊Home或者其他Link都會出現Navbar和Footer



在首頁輸出sanity的prodcut.ts，先在sanity的console新增prodcut的產品圖片和資訊，這裡需要注意產品名稱用英文。



建立HomeProducts元件



import { useState, useEffect } from "react";

import { Link } from "react-router-dom"; // React Router 的 Link

import { client } from "../../studio-react\_shop/sanity"; // 確保這是你 sanity 客戶端的正確路徑

const getData = async () => {

  const query = `\*[\_type == "product"][0...4] | order(\_createdAt desc) {

        \_id,

        price,

        name,

        "slug": slug.current,

        "categoryName": category->name,

        "imageUrl": images[0].asset->url

    }`;

  const data = await client.fetch(query); // 從 Sanity 擷取資料

  return data;

};

const HomeProduct = () => {

  const [data, setData] = useState([]); // 使用 useState 來儲存資料

  useEffect(() => {

    const fetchData = async () => {

      const result = await getData(); // 擷取資料

      setData(result); // 將資料設置到 state

    };

    fetchData(); // 在組件掛載時執行資料擷取

  }, []); // 空的依賴陣列表示只在組件首次掛載時執行

  return (

    <div className="bg-white">

      <div className="mx-auto max-w-2xl px-4 py-16 sm:px-6 sm:py-24 lg:max-w-7xl lg:px-8">

        <div className="flex justify-between items-center">

          <h2 className="text-2xl font-bold tracking-tight text-gray-900">

            最新產品

          </h2>

        </div>

        <div className="mt-6 grid grid-cols-1 gap-x-6 gap-y-10 sm:grid-cols-2 lg:grid-cols-4 xl:gap-x-8">

          {data.map((product) => (

            <div key={product.\_id} className="group relative">

              <div className="aspect-square w-full overflow-hidden rounded-md bg-gray-200 group-hover:opacity-75 lg:h-80">

                <img

                  src={product.imageUrl}

                  alt="Product image"

                  className="w-full h-full object-cover object-center lg:h-full lg:w-full"

                  width={300}

                  height={300}

                />

              </div>

              <div className="mt-4 flex justify-between">

                <div>

                  <h3 className="text-sm text-gray-700">

                    <Link to={`/product/${product.slug}`}>

                      {product.name}

                    </Link>

                  </h3>

                  <p className="mt-1 text-sm text-gray-500">

                    {product.categoryName}

                  </p>

                </div>

                <p className="text-sm font-medium text-gray-900">

                  ${product.price}

                </p>

              </div>

            </div>

          ))}

        </div>

      </div>

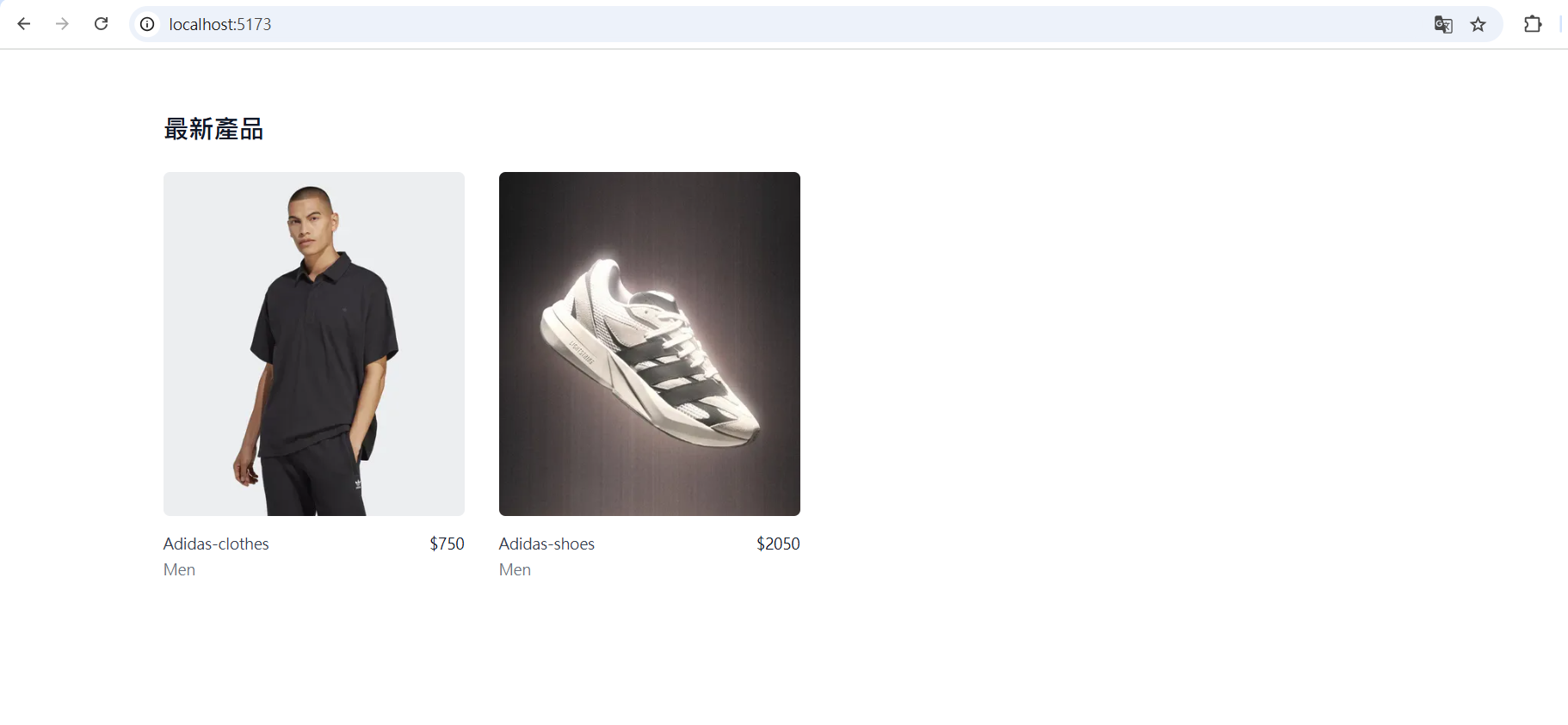
    </div>

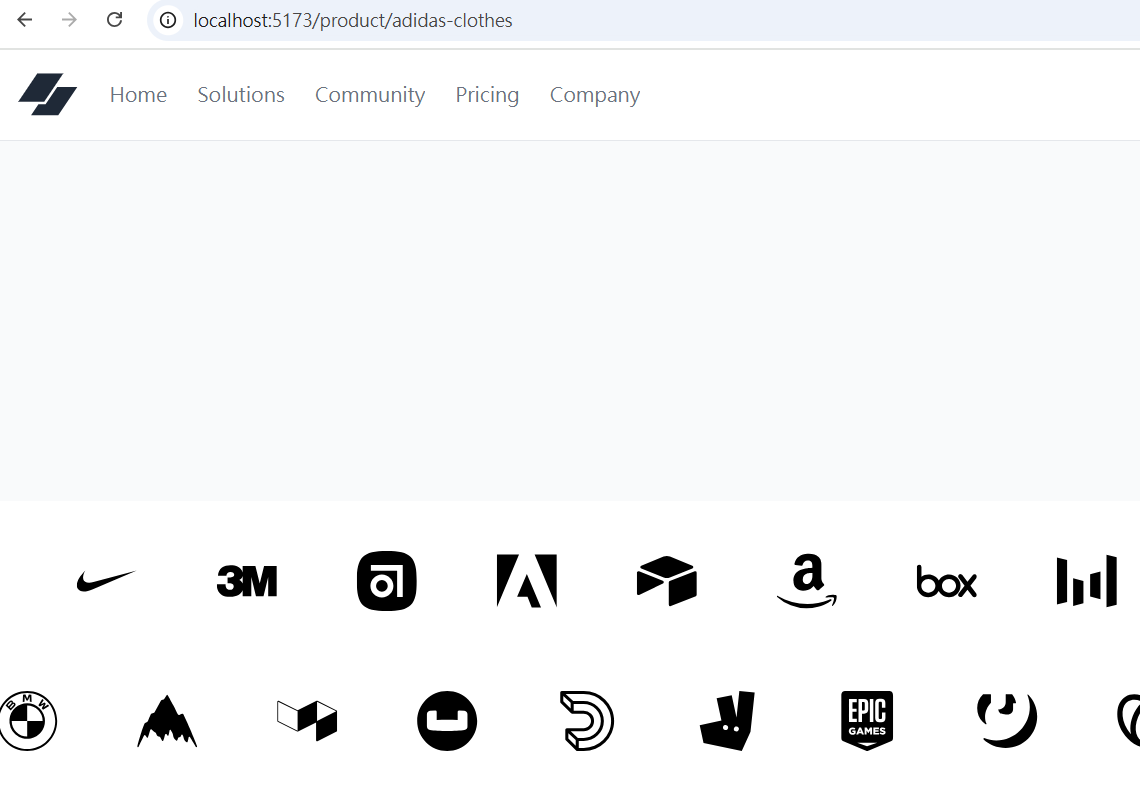
  );

};

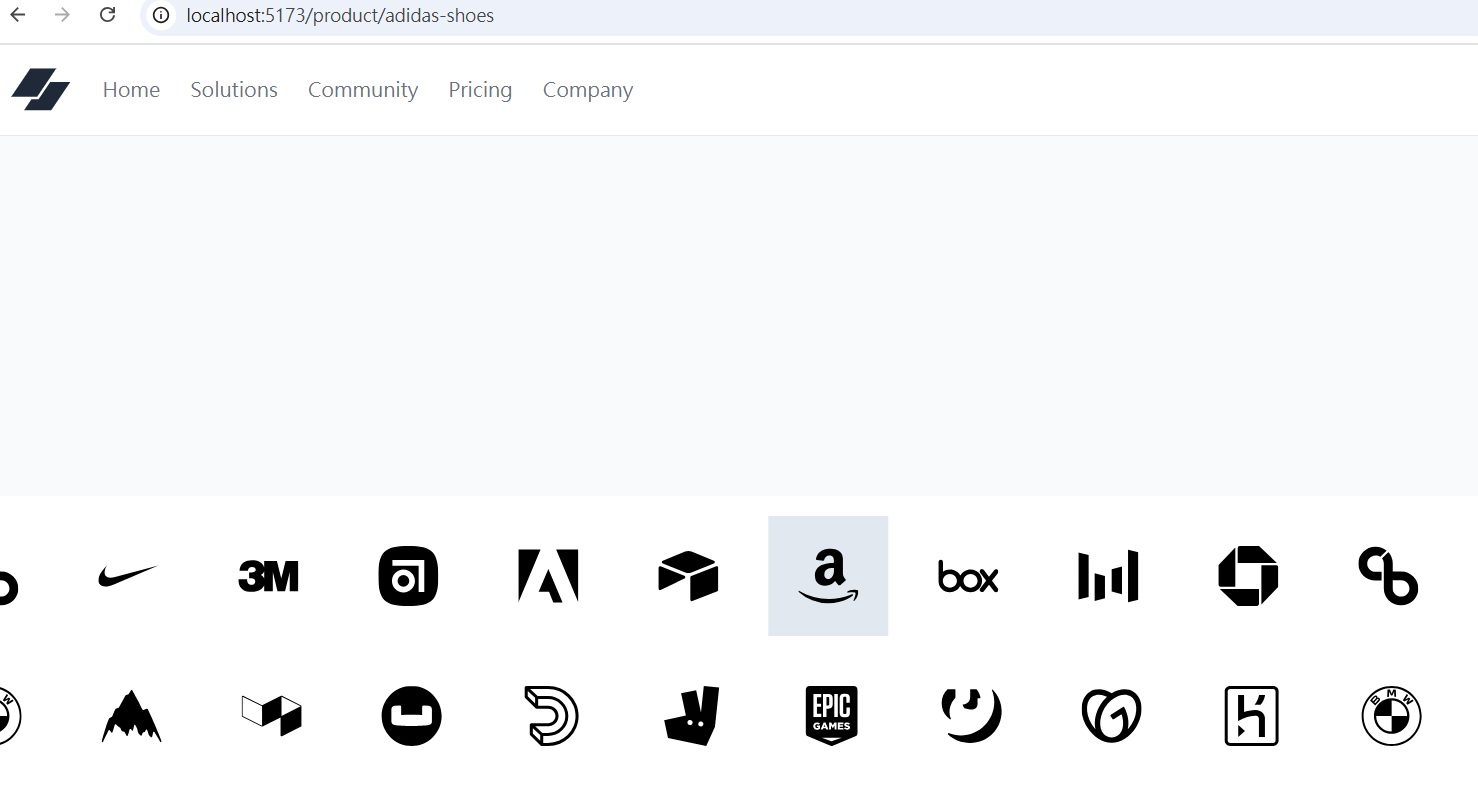
export default HomeProduct;

成功輸出:

當點擊產品名稱的時候會導引到跟產品名字一樣的路由，例如點Adidas-clothes



點Adidas-shoes



接下來建立ProductDetail元件當使用者點擊產品的Link的時候可以輸出，sanity的product.ts屬性。

建立ProductDetail元件



import { useEffect, useState } from "react";

import { useParams } from "react-router-dom"; // 用來抓取 URL 參數

import { client } from "../../studio-react\_shop/sanity"; // 確保這是你 sanity 客戶端的正確路徑

const ProductDetail = () => {

  const { slug } = useParams(); // 從 URL 中取得 slug 參數

  const [product, setProduct] = useState(null); // 儲存取得的產品資料

  const [loading, setLoading] = useState(true); // 處理載入狀態

  useEffect(() => {

    const getProductDetail = async () => {

      const query = `\*[\_type == "product" && slug.current == $slug][0] {

        \_id,

        name,

        price,

        description,

        "slug": slug.current,

        "categoryName": category->name,

        "imageUrl": images[0].asset->url

      }`;

      const data = await client.fetch(query, { slug }); // 根據 slug 查詢產品資料

      setProduct(data); // 設置產品資料

      setLoading(false); // 停止載入

    };

    getProductDetail(); // 呼叫函數來取得產品資料

  }, [slug]); // 當 slug 改變時重新抓取資料

  if (loading) return <div>Loading...</div>; // 如果資料還在載入中，顯示 Loading

  return (

    <div className="product-detail">

      <div className="product-image">

        <img src={product.imageUrl} alt={product.name} className="w-full h-auto" />

      </div>

      <div className="product-info">

        <h2 className="text-3xl font-bold">{product.name}</h2>

        <p className="text-xl text-gray-700">{product.categoryName}</p>

        <p className="text-lg text-gray-900">${product.price}</p>

        <p className="text-md text-gray-600 mt-4">{product.description}</p>

      </div>

    </div>

  );

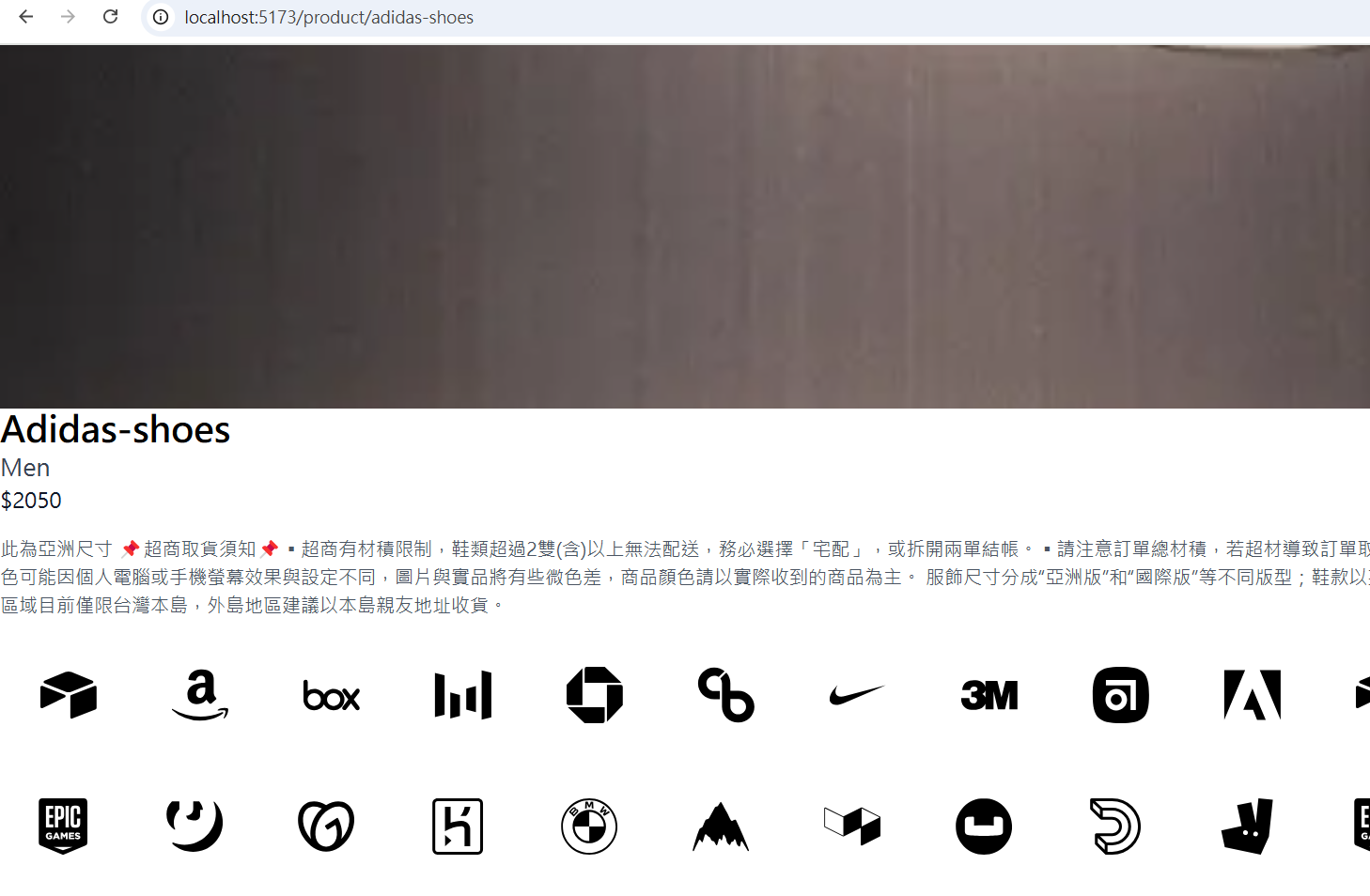
};

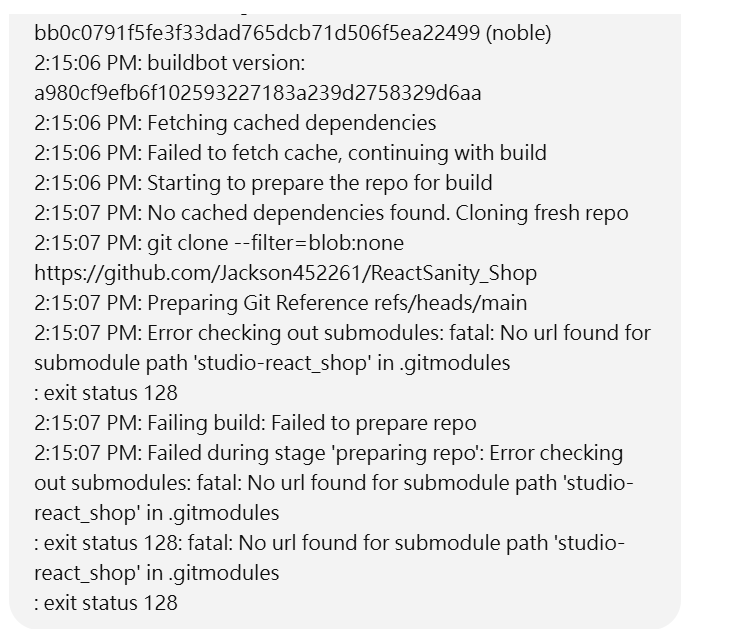
export default ProductDetail;

SetRoutes元件新增導引到每個產品slug特定的ProductDetail元件。



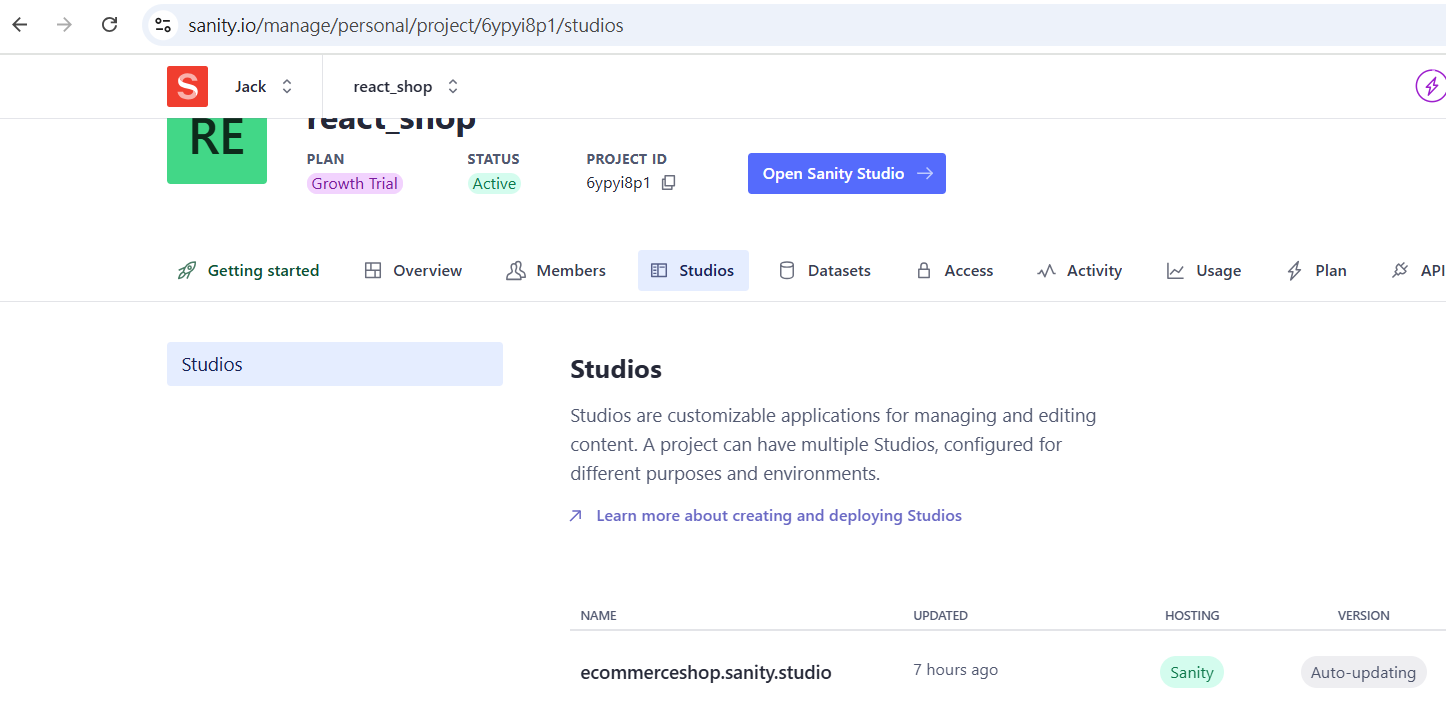
成功輸出:



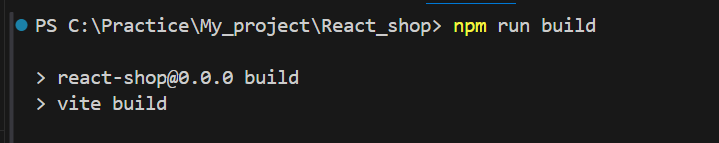
把目前專案上傳到Github再推送到Netlify的時候遇到問題: 

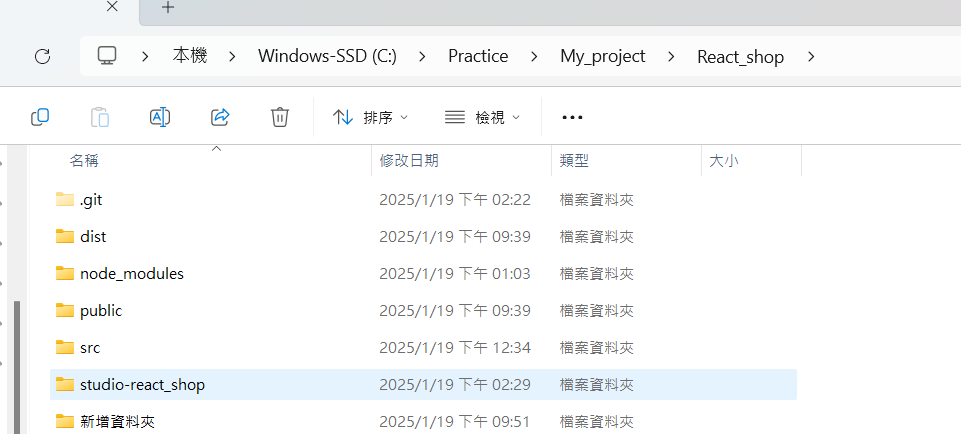
目前解決方法:

先進入sanity的目錄輸入指令sanity deploy得到Url網址



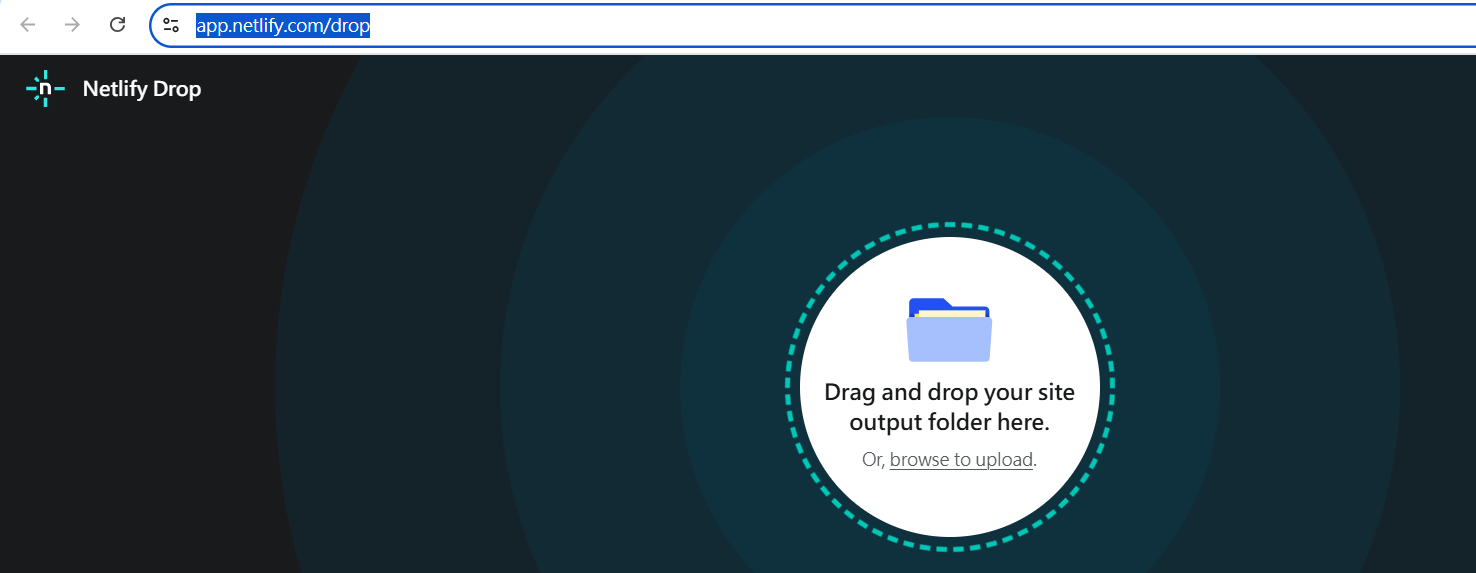
前端目錄輸入指令建立dist資料夾



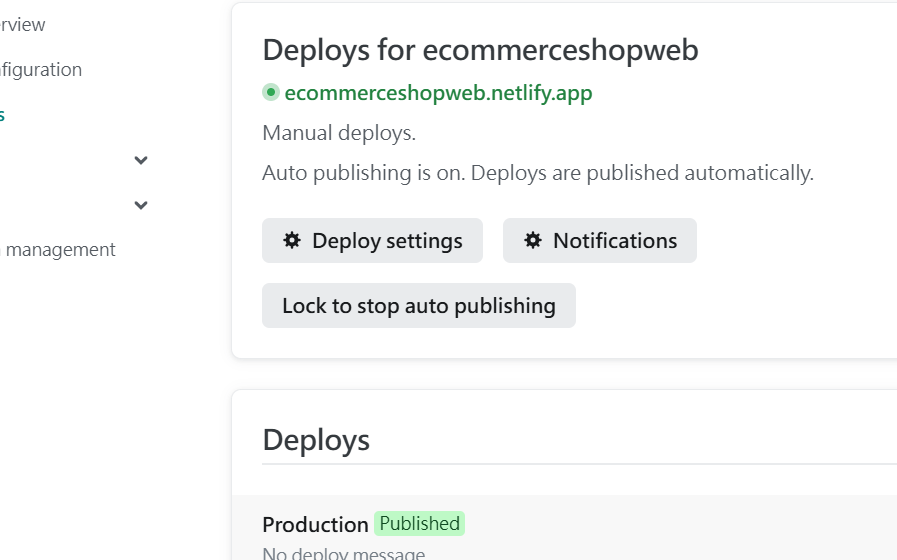


<https://app.netlify.com/drop>

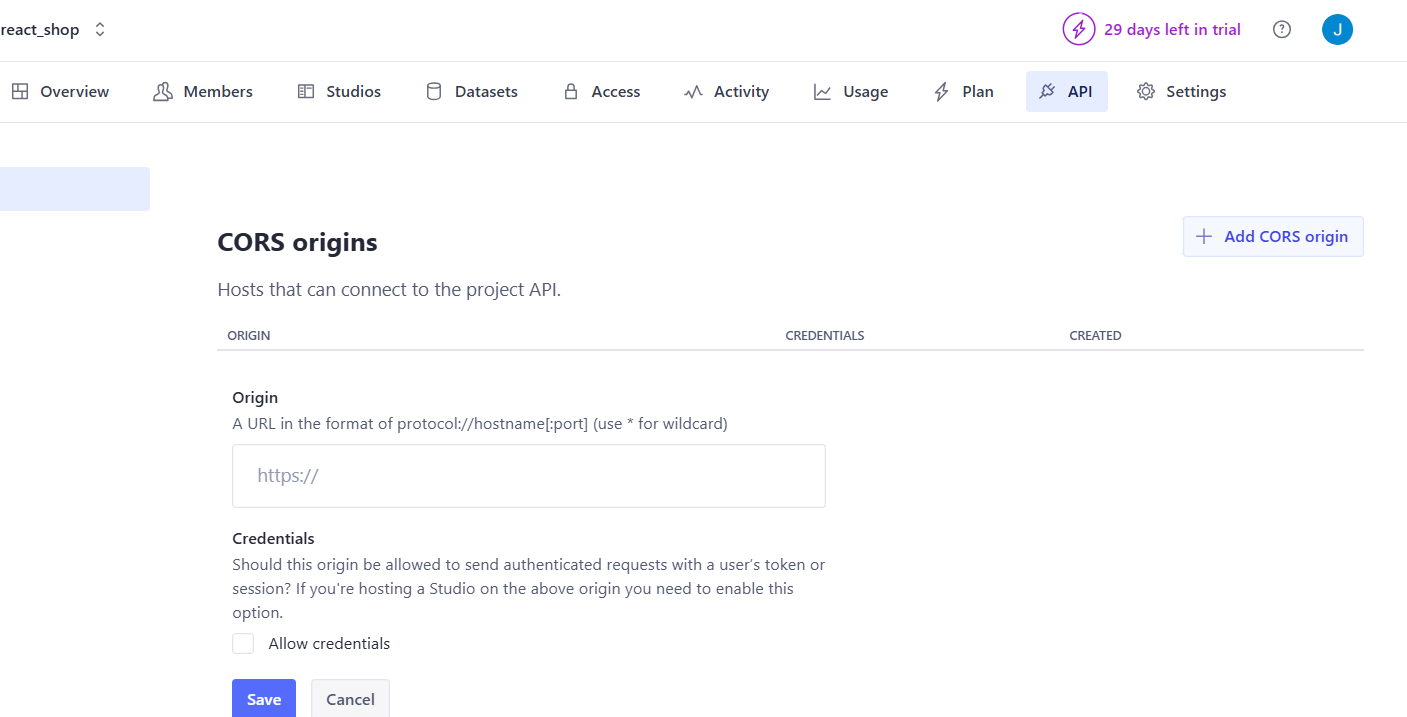
把dist資料夾移動到netlify drop



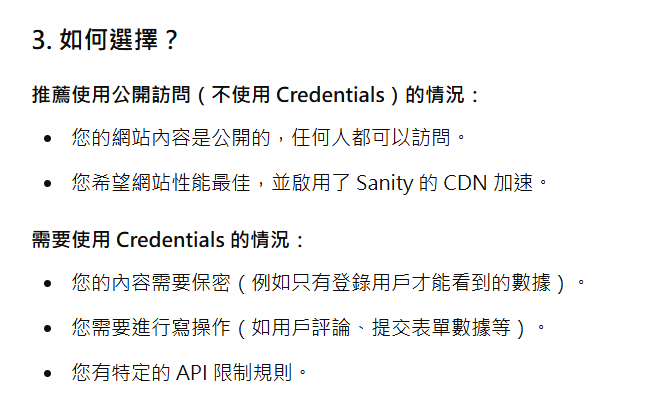
成功上傳

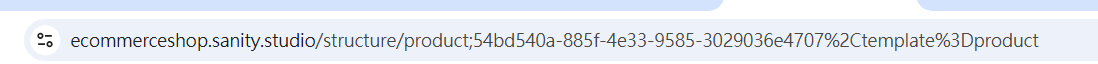


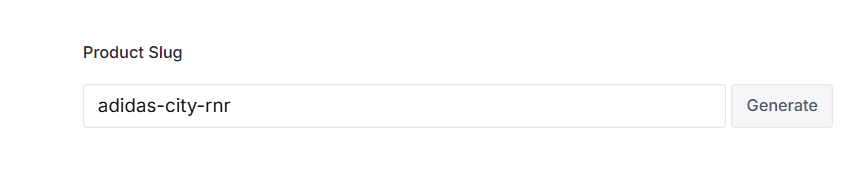
得到Netlify網址之後進入到sanity的API設置允許CROS跨域設定



不需要勾選Allow credentials



測試假如在sanity已經deploy的url新增一個產品，這次新增的產品名稱改用中文來測試會不會有錯誤: 



成功輸出:



Sanity的product slug必須是英文要不然會有錯誤

