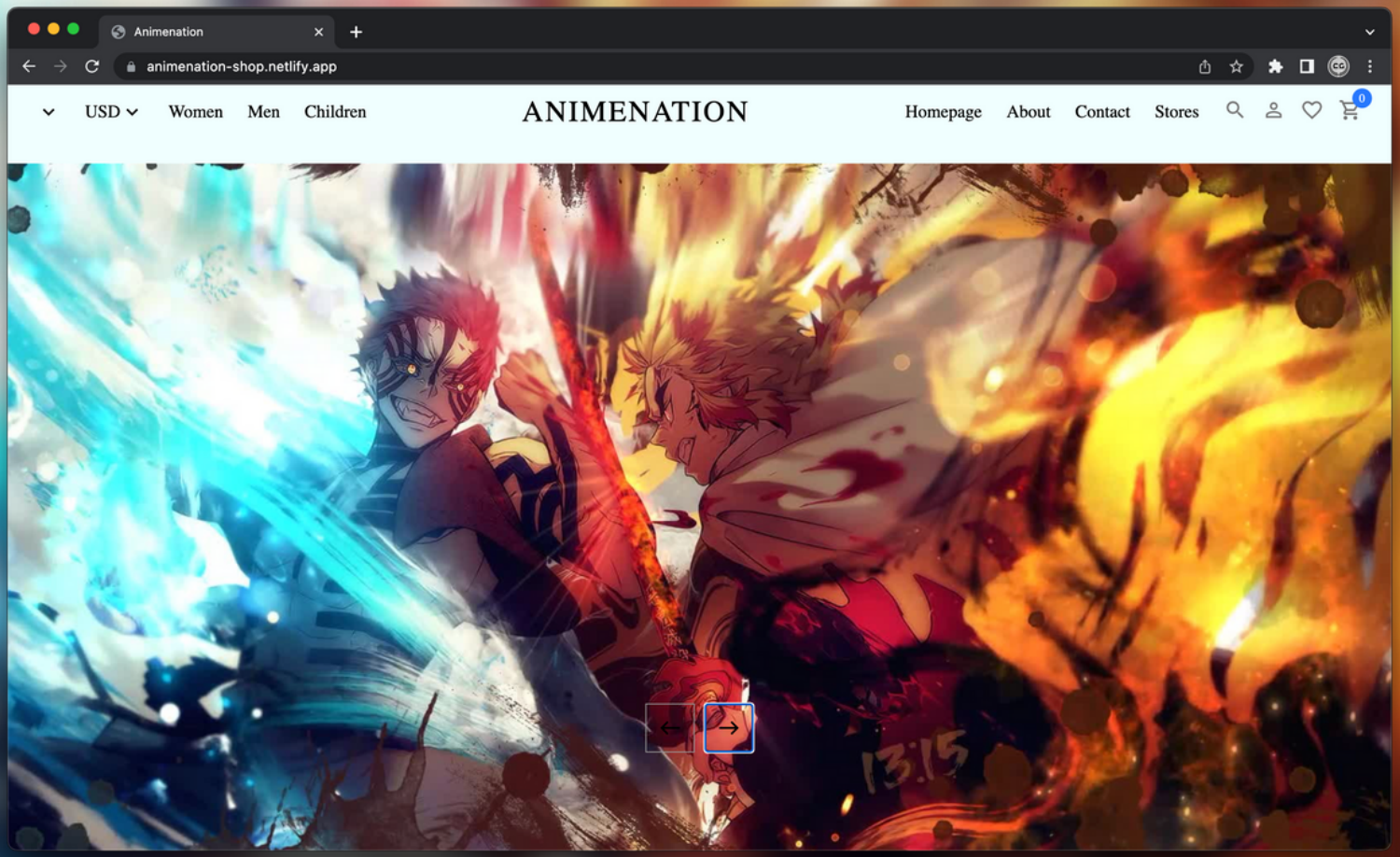
Animenation

CASE STUDY



Overview:

Animenation is a React website application that is featured as an anime apparel online E-commerce site. This website is only intended to showcase some of my web developing skills and is not to be used as services or actual sales. Animenation Shop features a Home page, that leads into its featured and trending products. Each product has its own individual page that users can cycle and filter through their desired category. Users can add these items to a cart, and proceed to a checkout page.

Goals:

- Build an E-commerce website for other companies to see.
- Build a REST API using Strapi.io to host the products and contents of the online store.
- Integrates Stripe`s API, an online payment processing and credit card processing platform API, to allow users to purchase product.
- Expand and grow my web developing skills by building an E-commerce site and use new APIs, that I hadn't built or used before.

Role & Tools Used:

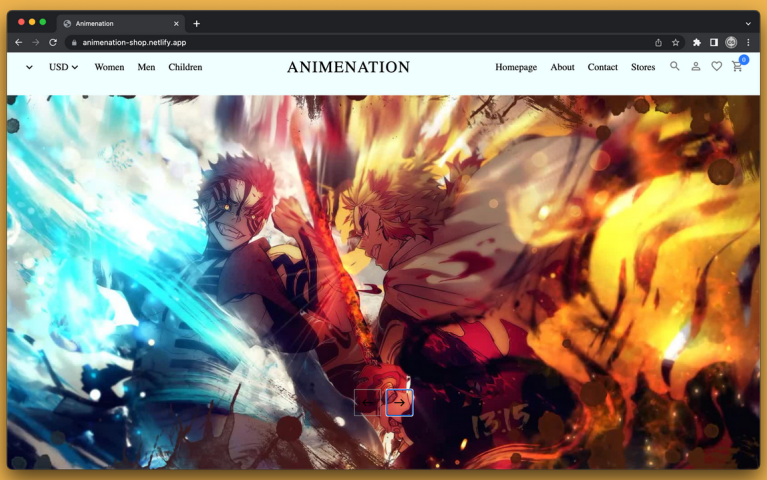
- Full-Stack Web Developer
- Strapi.io API (Backend)
- Stripe API
- React (Frontend)
- Vite (Development Environment)

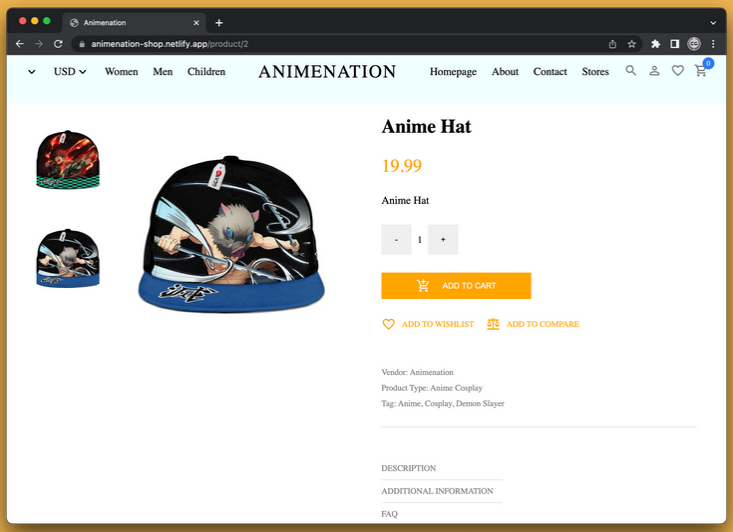
APPROACH

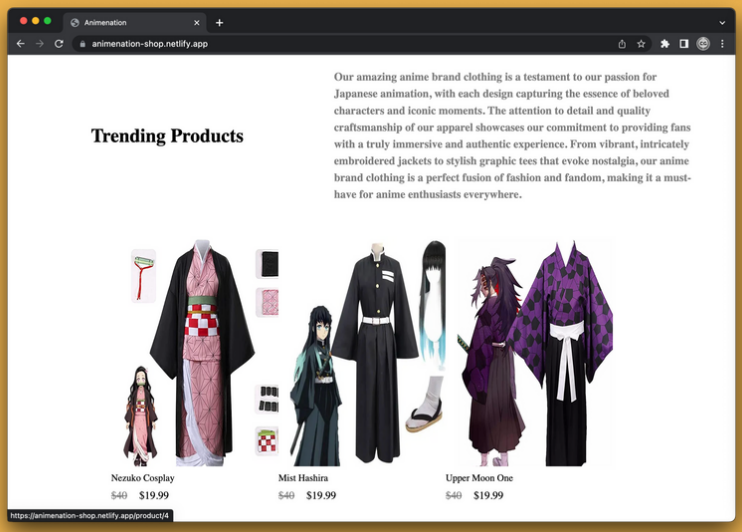
Frontend:

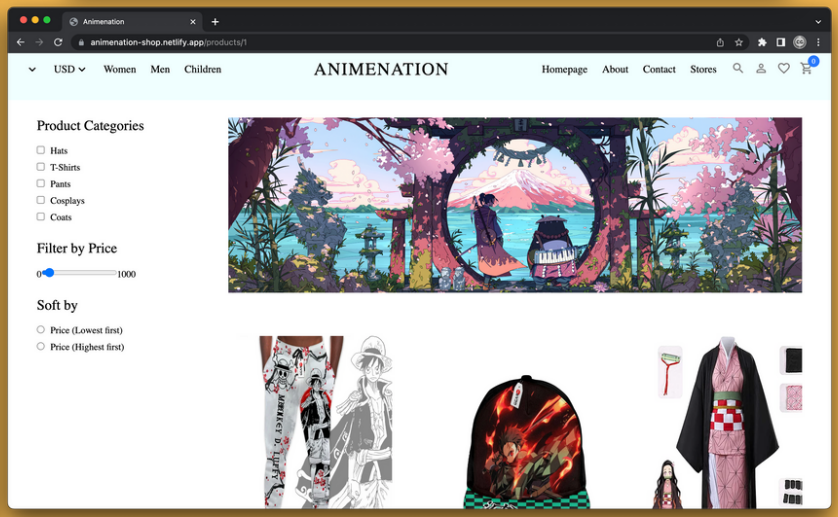
Using React, I utilized its components to quickly build the UI of the website that users would be able to browse through. I used React for my front end because its components are extremely useful when building the UI of applications. Components are independent, reusable bit of code which divides the UI into smaller pieces, making it very easy to make changes and adjustments to the UI without having to continuously update and change the entire file. React is fast, scalable, reusability features, and can handle high performance environments without compromising in other areas. Upon entering, users are greeted with a Homepage that consists of the featured products and trending products of their favorite anime. Each of the products can be accessed and filtered through to find a specific category of clothing. A shopping cart is available for users to add their desired products for checkout later on.

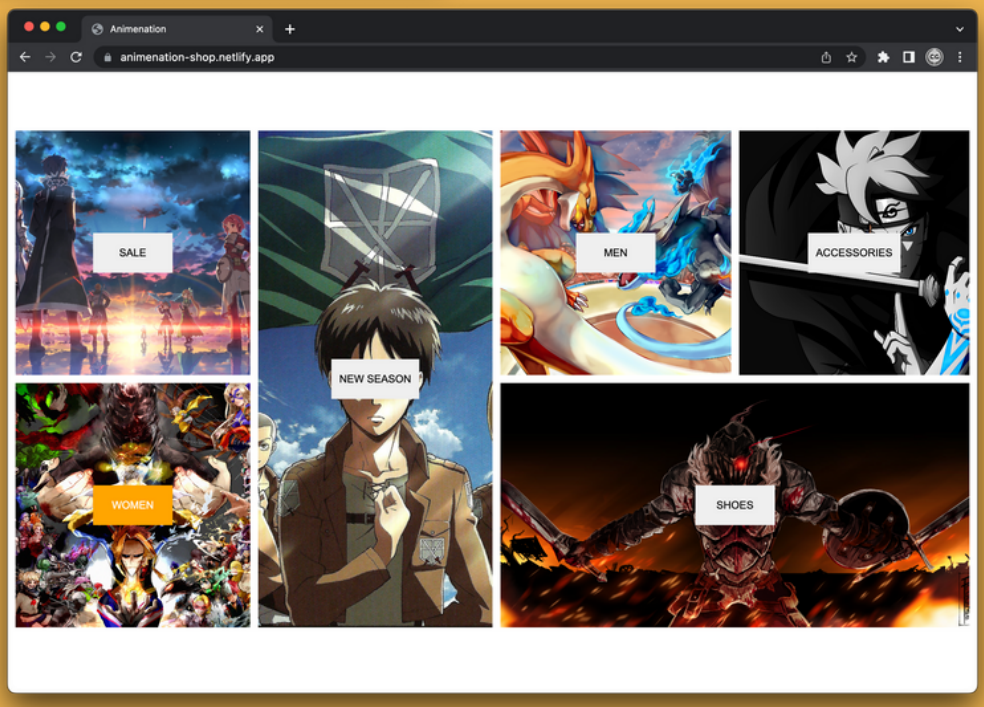
Frontend


Animation
animenation-shop.netlify.app
USD
Women
Men
Children
ANIMENATION
Homepage
About
Contact
Stores


Animation
animenation-shop.netlify.app/product/2
USD
Women
Men
Children
ANIMENATION
Homepage
About
Contact
Stores
Anime Hat
19.99
Anime Hat
ADD TO CART
ADD TO WISHLIST
ADD TO COMPARE
Vendor: Animenation
Product Type: Anime Cosplay
Tag: Anime, Cosplay, Demon Slayer
DESCRIPTION
ADDITIONAL INFORMATION
FAQ


Animation
animenation-shop.netlify.app
Trending Products
Our amazing anime brand clothing is a testament to our passion for Japanese animation, with each design capturing the essence of beloved characters and iconic moments. The attention to detail and quality craftsmanship of our apparel showcases our commitment to providing fans with a truly immersive and authentic experience. From vibrant, intricately embroidered jackets to stylish graphic tees that evoke nostalgia, our anime brand clothing is a perfect fusion of fashion and fandom, making it a must-have for anime enthusiasts everywhere.
Nezuko Cosplay
$40 $19.99
Mist Hashira
$40 $19.99
Upper Moon One
$40 $19.99
https://animenation-shop.netlify.app/product/4


Animation
animenation-shop.netlify.app/products/1
USD
Women
Men
Children
ANIMENATION
Homepage
About
Contact
Stores
Product Categories
Hats
T-Shirts
Pants
Cosplays
Coats
Filter by Price
0
1000
Soft by
Price (Lowest first)
Price (Highest first)


Animation
animenation-shop.netlify.app
SALE
NEW SEASON
MEN
ACCESSORIES
WOMEN
SHOES

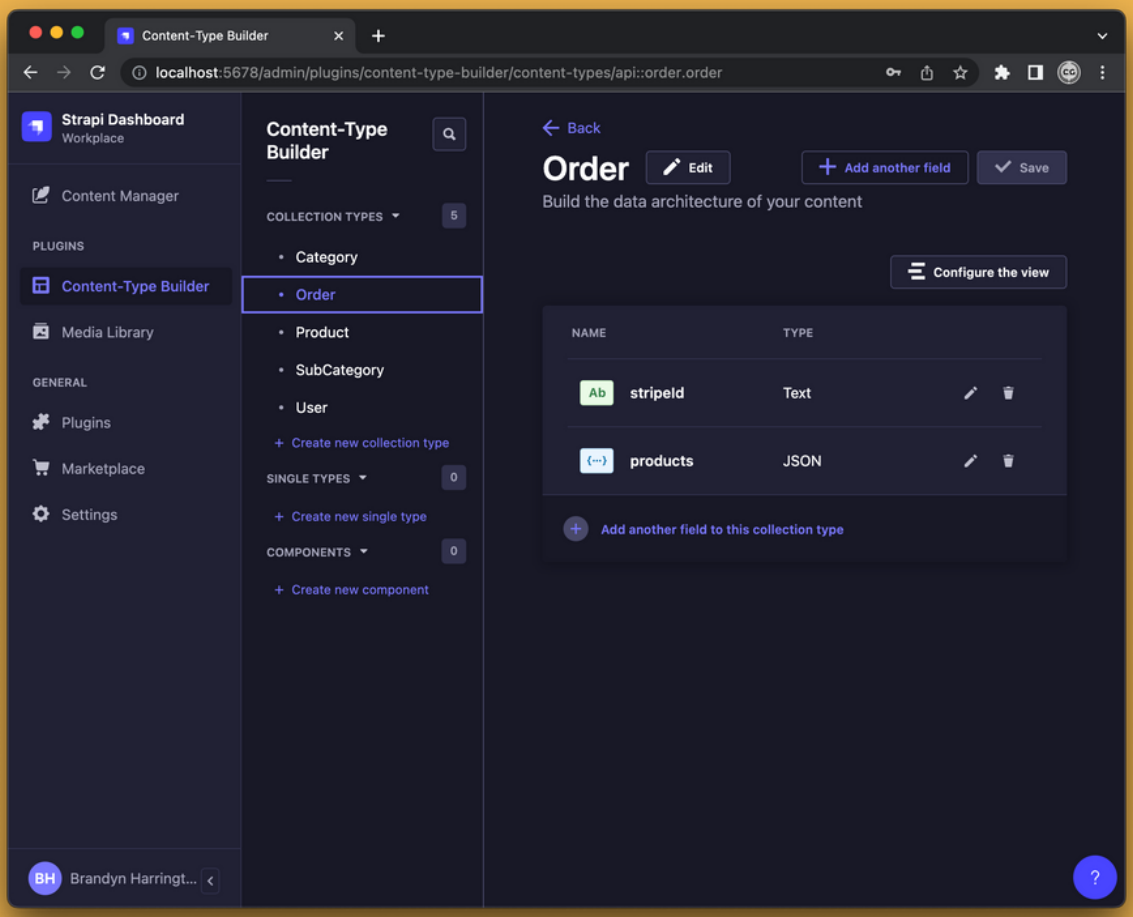
Backend:

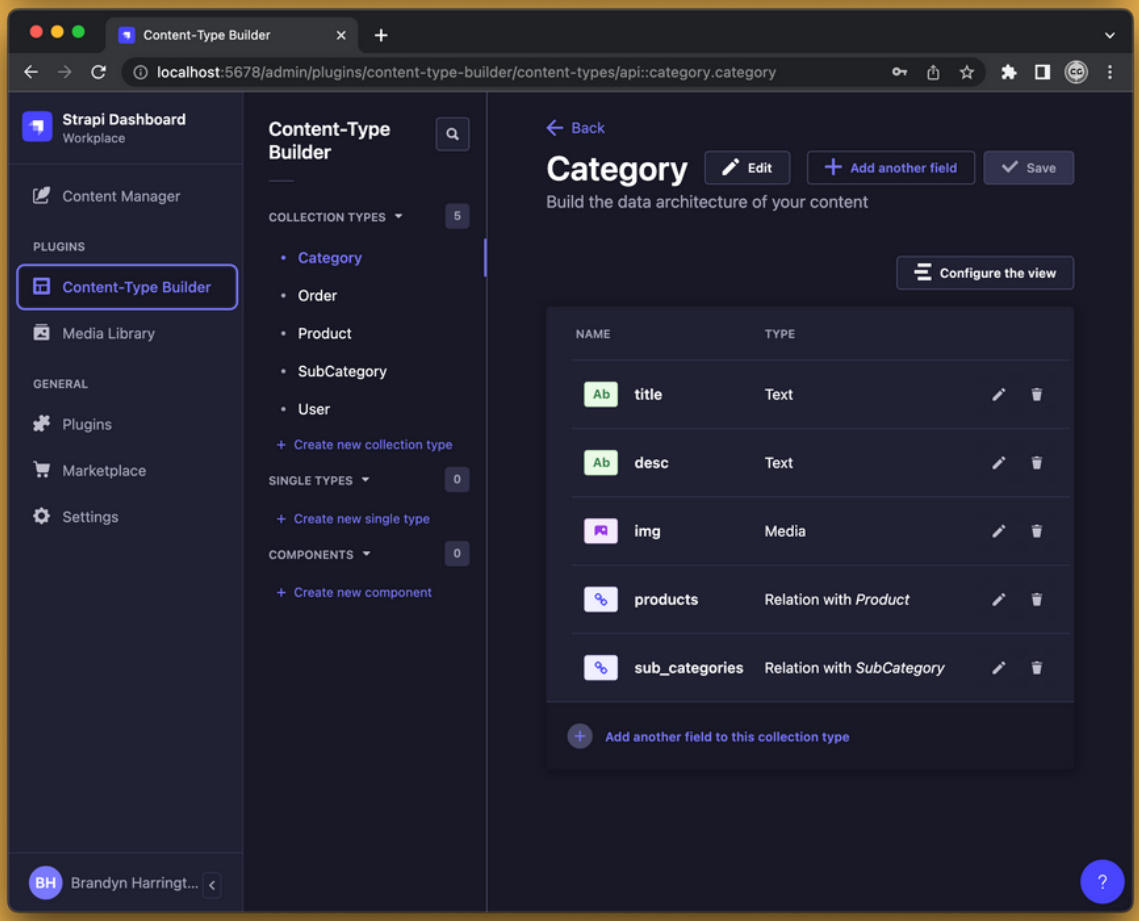
After I had designed the frontend of the E-commerce website, it was time to utilize **Strapi.io API** and **Stripe API**. I decided to use Strapi as my REST API to hold the data for my products. This includes the products title, description, image, secondary image, category, subcategory, price, and type. I also decided to go with Strapi due to its modular architecture, allowing you to personalize it and add your own features, very simply. As a result, the system is extremely adaptable to any project. Strapi is also easy to scale and can handle a high volume of traffic in a real E-commerce environment. Strapi has great compatibility with **Stripe API**, an online payment processing and credit card processing platform API.

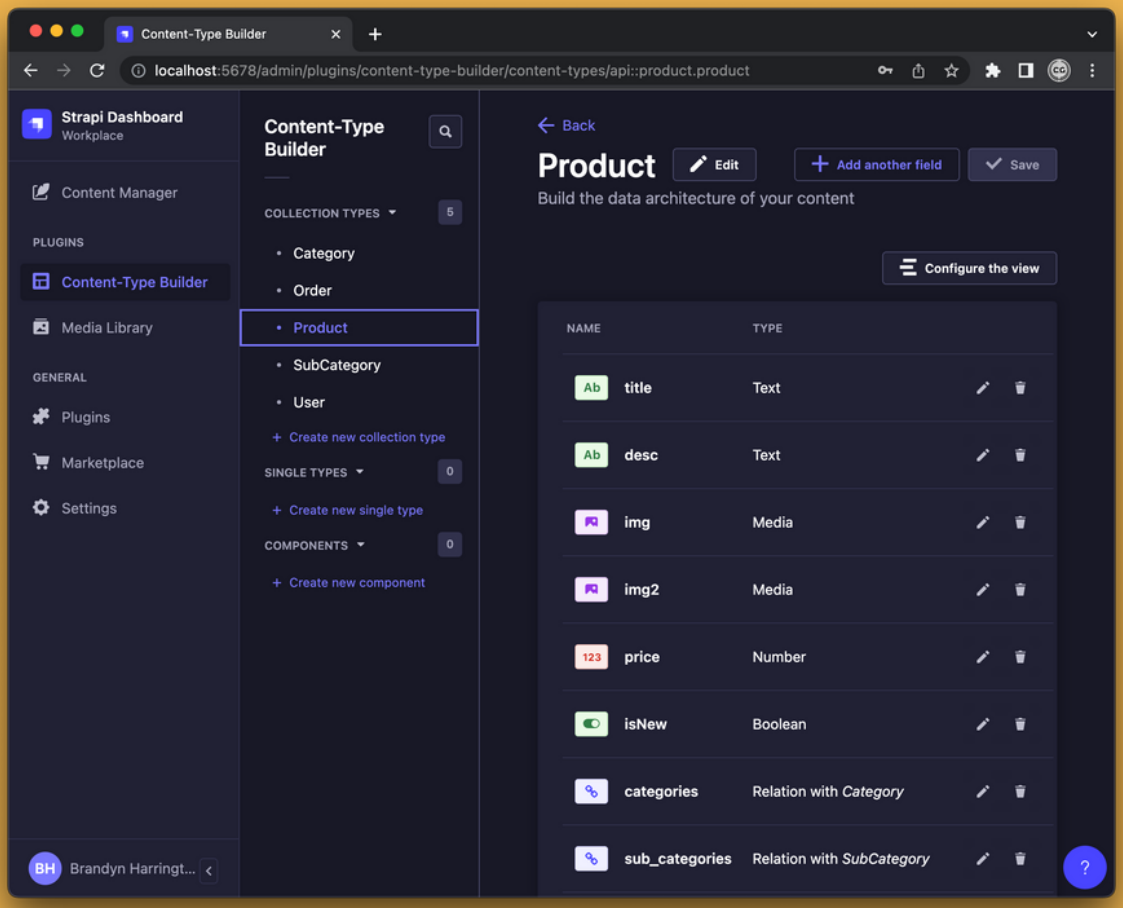
With **Strapi.io API**, I was able to easily enter the data for all of my products. This API made it simple to establish relationships across each product, category, and subcategory.

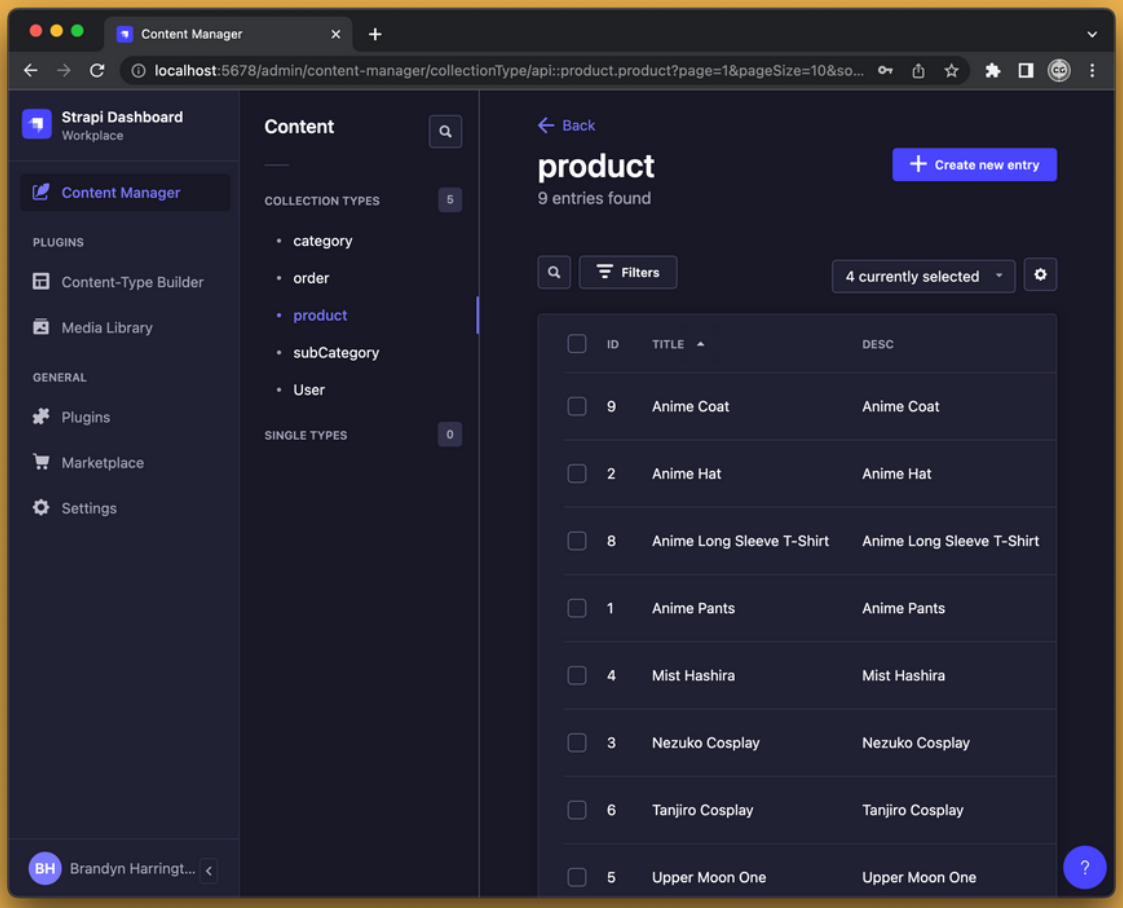
Stripe API allowed my to easily connected to my client-side and server-side application, allowing users of the site to process payments for products.

The following images include some of Strapi.io API's dashboard, showing how the products and easily assigned types and connected through different relationships.


Content-Type Builder
localhost:5678/admin/plugins/content-type-builder/content-types/api::order.order
Strapi Dashboard
Workplace
Content Manager
PLUGINS
Content-Type Builder
Media Library
GENERAL
Plugins
Marketplace
Settings
BH
Brandyn Harringt...
Content-Type Builder
COLLECTION TYPES 5
Category
Order
Product
SubCategory
User
+ Create new collection type
SINGLE TYPES 0
+ Create new single type
COMPONENTS 0
+ Create new component
Back
Order
Edit
Add another field
Save
Build the data architecture of your content
Configure the view
NAME
TYPE
Ab
stripeId
Text
{...}
products
JSON
Add another field to this collection type
?


Content-Type Builder
localhost:5678/admin/plugins/content-type-builder/content-types/api::category.category
Strapi Dashboard
Workplace
Content Manager
PLUGINS
Content-Type Builder
Media Library
GENERAL
Plugins
Marketplace
Settings
BH
Brandyn Harringt...
Content-Type Builder
COLLECTION TYPES 5
Category
Order
Product
SubCategory
User
+ Create new collection type
SINGLE TYPES 0
+ Create new single type
COMPONENTS 0
+ Create new component
Back
Category
Edit
Add another field
Save
Build the data architecture of your content
Configure the view
NAME
TYPE
Ab
title
Text
Ab
desc
Text
img
Media
products
Relation with Product
sub_categories
Relation with SubCategory
Add another field to this collection type
?


Content-Type Builder
localhost:5678/admin/plugins/content-type-builder/content-types/api::product.product
Strapi Dashboard
Workplace
Content Manager
PLUGINS
Content-Type Builder
Media Library
GENERAL
Plugins
Marketplace
Settings
Brandyn Harringt...
Content-Type Builder
COLLECTION TYPES 5
Category
Order
Product
SubCategory
User
+ Create new collection type
SINGLE TYPES 0
+ Create new single type
COMPONENTS 0
+ Create new component
Back
Product
Edit
Add another field
Save
Build the data architecture of your content
Configure the view
NAME
TYPE
title Text
desc Text
img Media
img2 Media
price Number
isNew Boolean
categories Relation with Category
sub_categories Relation with SubCategory


Content Manager
localhost:5678/admin/content-manager/collectionType/api::product.product?page=1&pageSize=10&so...
Strapi Dashboard
Workplace
Content Manager
PLUGINS
Content-Type Builder
Media Library
GENERAL
Plugins
Marketplace
Settings
Brandyn Harringt...
Content
COLLECTION TYPES 5
category
order
product
subCategory
User
SINGLE TYPES 0
Back
product
9 entries found
Create new entry
Filters
4 currently selected
ID
TITLE
DESC
9 Anime Coat Anime Coat
2 Anime Hat Anime Hat
8 Anime Long Sleeve T-Shirt Anime Long Sleeve T-Shirt
1 Anime Pants Anime Pants
4 Mist Hashira Mist Hashira
3 Nezuko Cosplay Nezuko Cosplay
6 Tanjiro Cosplay Tanjiro Cosplay
5 Upper Moon One Upper Moon One

CONCLUSION

Difficulties & Challenges:

One of my main issues and complications I came across was trying to connect my REST API database's product information with my client-side web application. I continued to come across HTML Request errors because my API key had locked permissions. After some digging through Strapi.io API documention, along with others users questions, I managed to solve the problem by editing the "Roles & Permissions" section in Strapi's dashboard. This section allows the Administrator of the website to make adjustments to what users can do when they have gained access to the backends API key. I only needed to give permission to public users to view and find the products on the page.

Final Take-Away:

I learned a lot working through this project, from building an E-commerce website, converting product information into RESTful API data to be used in a backend environment, to using external APIs in a production level environment like processing payments. This project helped me grow in my capacity as a full-stack web developer, and I can use what I learned in future projects and possibly employments.