Capstone - Metro an Ecommerce Sneaker Store

Introduction

This project report covers the 'Metro' Sneaker Store Ecommerce application. It solves the stakeholders issue to grow and scale a small business. It covers the technical details of the full MERN stack architecture, UI design and testing process. It integrates Stripe which is a leading global payment gateway to the testing level. Overall the project is a tech demo to highlight skills taught by Jo Batkin and Gareth Wootton (IOD) from the AUT industry accredited certificate in Software Engineering.

Purpose

As the world becomes more digital, ecommerce has become an essential part of our digital economy. There is a need for efficient and user-friendly ecommerce stores that provide a seamless shopping experience to customers. The project aims to create an online store that is intuitive, easy to use, and provides a personalized shopping experience for the customers. This project will help businesses expand their reach by enabling them to showcase their products on a digital platform and sell them directly to consumers.

Industry

The Industry has become oversaturated with template stores such as shopify. Custom built E-commerce stores have more customisation, lower fees, a better branding reputation and more flexible designing. It allows a small business to grow and scale to generate more growth using a bigger online presence.

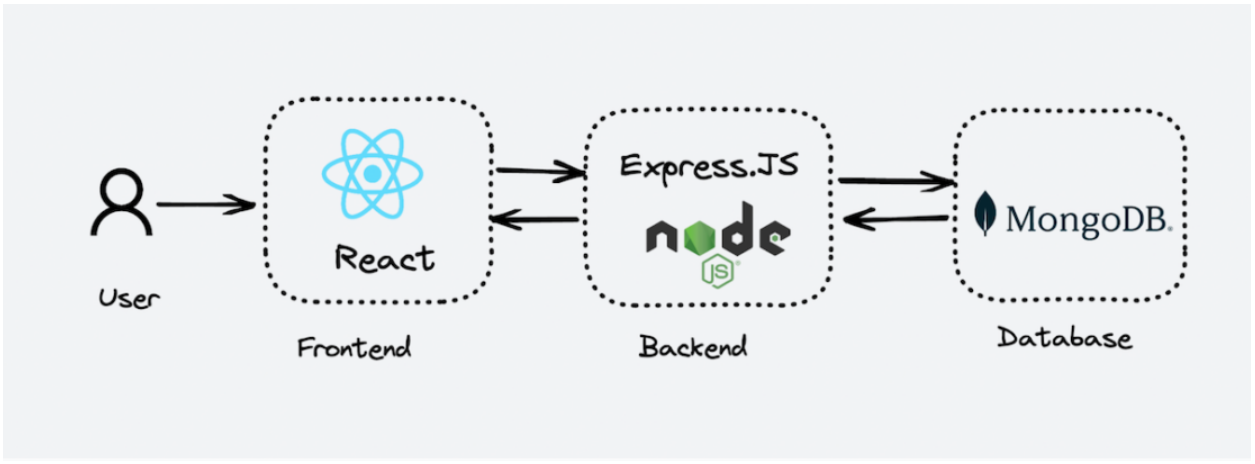
Stakeholders

Some sneaker stores only use Instagram to resell sneakers.Sneaker resellers are generally small businesses that grow fast quickly.They generally expand into other niches as well such as clothing.Most resellers in today's age need a website with a reliable backend to expand and handle orders. Social media such as IG and FB marketplace does not provide a scalable solution.The project aims to grow the stakeholder from a small business to a larger business where they can scale.The Project aim is to create an online store that is intuitive, easy to use and provides a personalized shopping experience to the customers.

Product Description

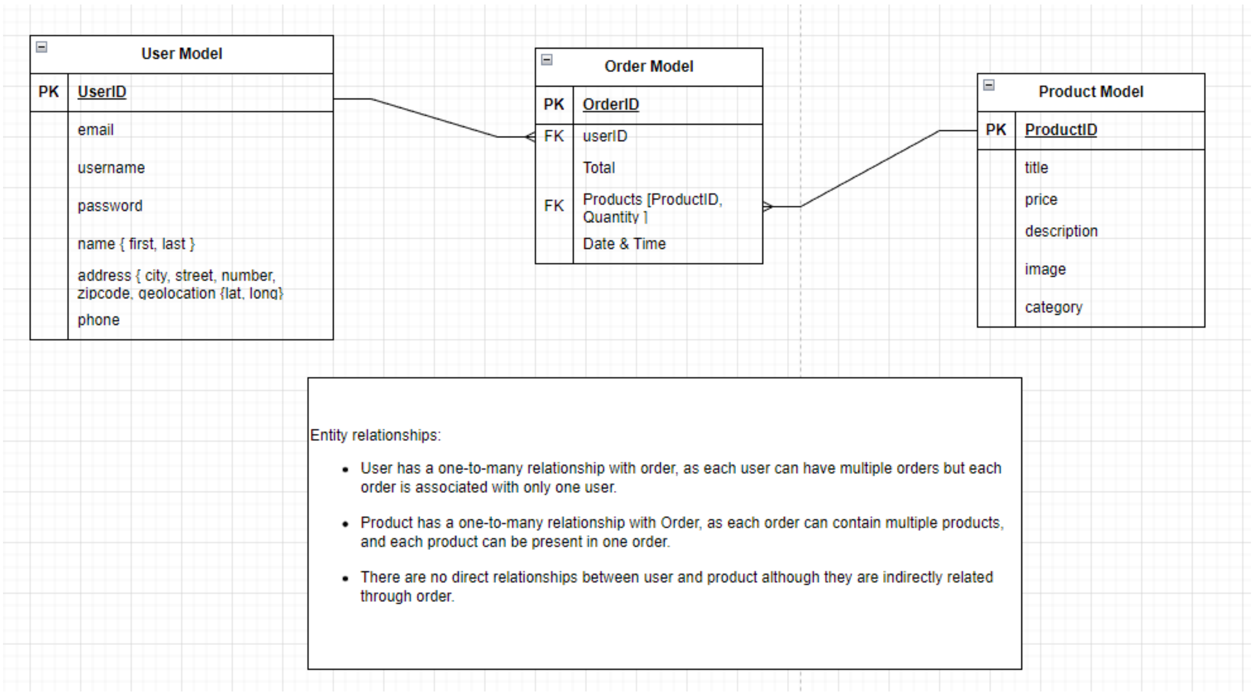
Architecture Diagram

Overall the MERN Stack architecture with MongoDB, Express, React and Node.



Backend Architecture

The backend is using a MVC architecture between 3 models with several functions that implement CRUD operations (create, read, update, delete) for data stored in a MongoDB database.

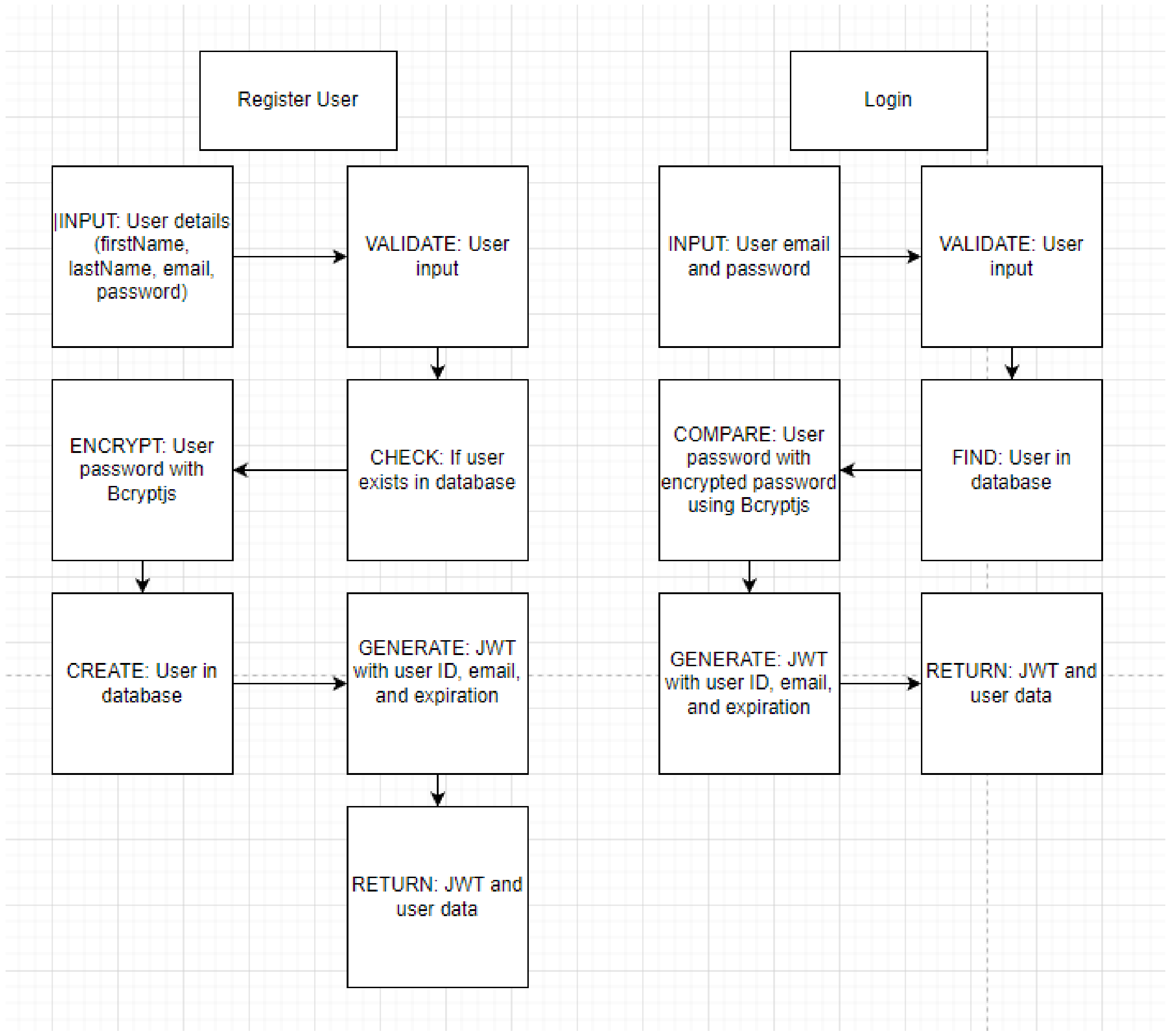


Entity relationships:

- User has a one-to-many relationship with order, as each user can have multiple orders but each order is associated with only one user.
- Product has a one-to-many relationship with Order, as each order can contain multiple products, and each product can be present in one order.
- There are no direct relationships between user and product although they are indirectly related through order.

User architecture

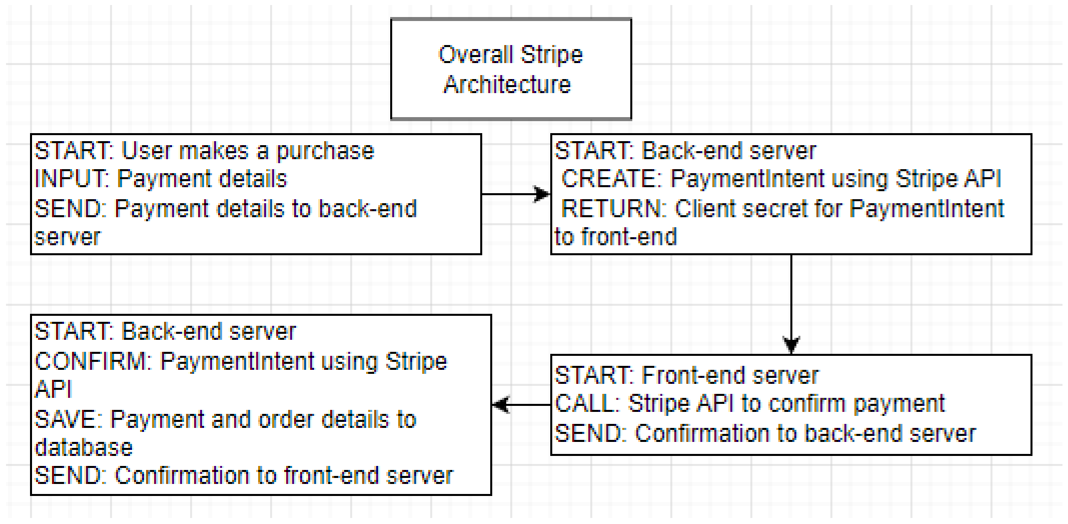
To make the login process secure, the user functions integrate JSON Web Tokens(JWT) for user authentication and authorization.

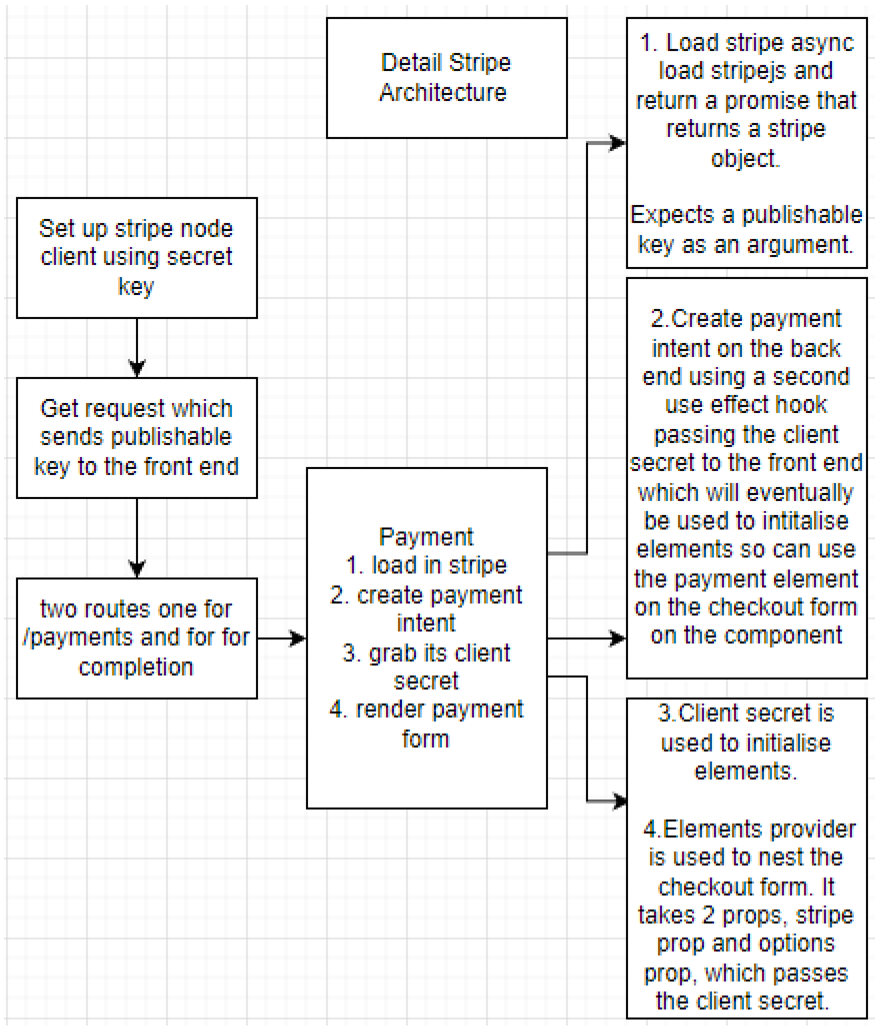

Register User
|INPUT: User details (firstName, lastName, email, password)
VALIDATE: User input
CHECK: If user exists in database
ENCRYPT: User password with Bcryptjs
CREATE: User in database
GENERATE: JWT with user ID, email, and expiration
RETURN: JWT and user data
Login
INPUT: User email and password
VALIDATE: User input
FIND: User in database
COMPARE: User password with encrypted password using Bcryptjs
GENERATE: JWT with user ID, email, and expiration
RETURN: JWT and user data

Order Architecture

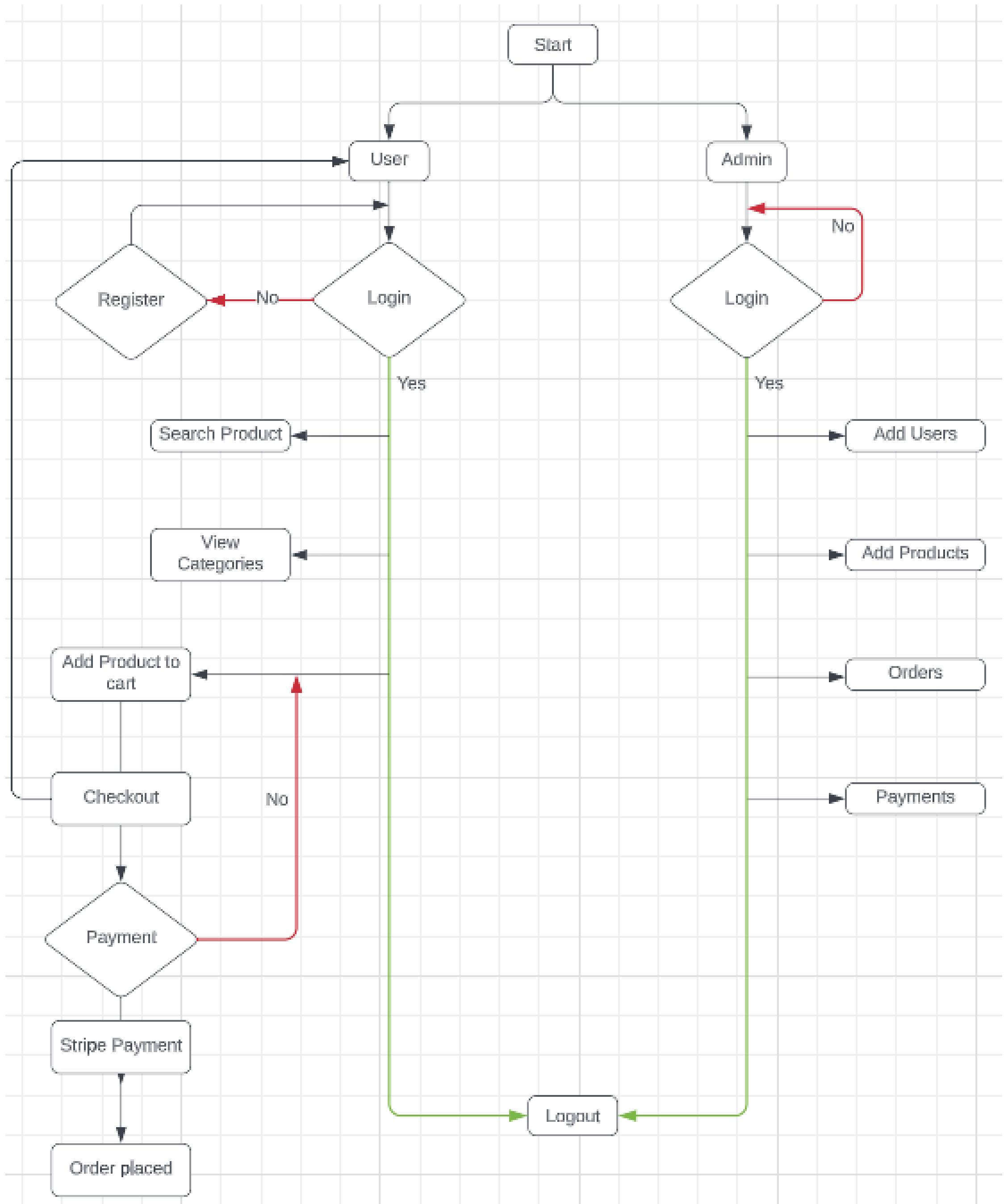
To correctly generate an order the order needs to pass an object Id from both the User and the product object ID. This is to provide a security measure that a user that has registered and a product that is in the database can only be created into an order.

Stripe Architecture


Overall Stripe Architecture
START: User makes a purchase
INPUT: Payment details
SEND: Payment details to back-end server
START: Back-end server
CREATE: PaymentIntent using Stripe API
RETURN: Client secret for PaymentIntent to front-end
START: Front-end server
CALL: Stripe API to confirm payment
SEND: Confirmation to back-end server
START: Back-end server
CONFIRM: PaymentIntent using Stripe API
SAVE: Payment and order details to database
SEND: Confirmation to front-end server


Detail Stripe Architecture
Set up stripe node client using secret key
Get request which sends publishable key to the front end
two routes one for /payments and for for completion
Payment
1. load in stripe
2. create payment intent
3. grab its client secret
4. render payment form
1. Load stripe async load stripejs and return a promise that returns a stripe object.
Expects a publishable key as an argument.
2.Create payment intent on the back end using a second use effect hook passing the client secret to the front end which will eventually be used to intitalise elements so can use the payment element on the checkout form on the component
3.Client secret is used to initialise elements.
4.Elements provider is used to nest the checkout form. It takes 2 props, stripe prop and options prop, which passes the client secret.

Product Design

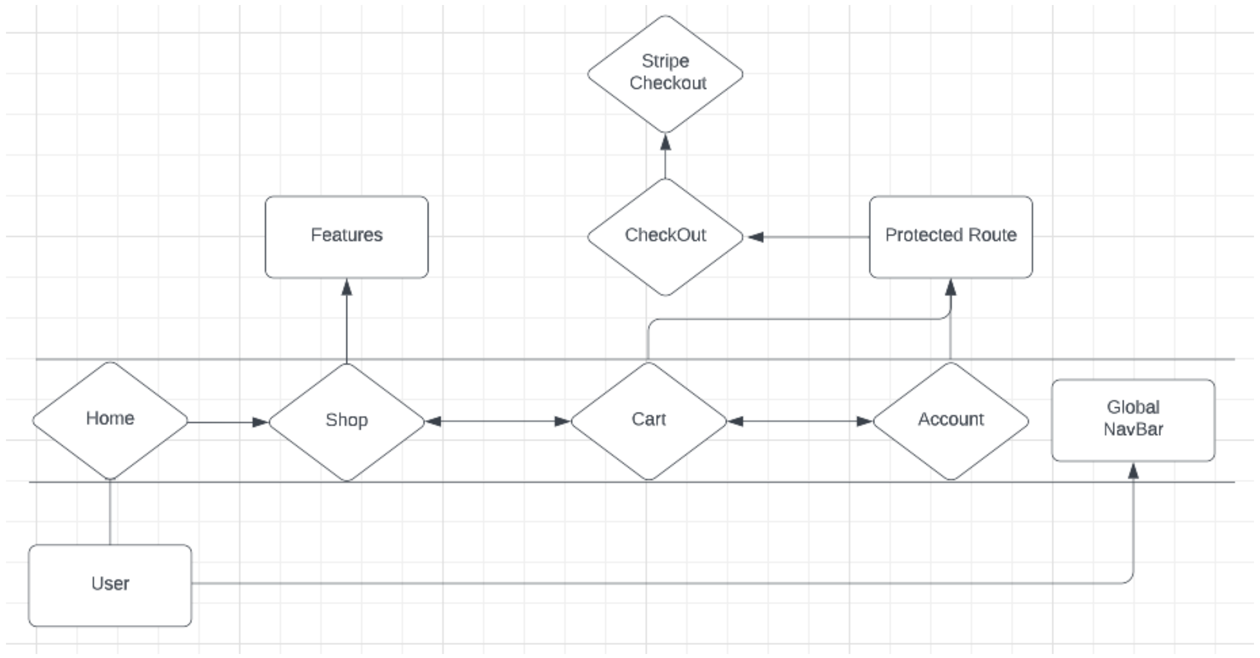


UserStories

#	User Story Title	User Story Description	Priority	Additional Notes
1	Admin logs in to create new products	Admin logs in, goes to the admin page, inserts a new image, price, details of the product and lists it for sale so the public can see it.	**High**	**Admin: Create**
2	Admin logs in to update an existing product	Admin logs in, goes to the admin page, inserts an updated image, price, details of the product for a sale/event	**High**	**Admin: Update**
3	Admin logs in to Delete an existing product	Admin logs in, goes to the admin page deletes a product from the page as it is out of stock	**High**	**Admin: delete**
4	User goes onto the site, decides they like the item and proceed to purchase it	User adds to cart, proceeds to checkout which is a protected route. Gets prompted to make a log in where user creates account, if already has an account check if is logged in and proceed to payment screen	**High**	**User: log in**
5	User goes onto the site, adds it to the cart, decides they do not like the item and want to delete it	User adds to the card, proceeds to shop more, and adds another item to the cart. Proceeds to check out but realizes they do not want to purchase one one of the items so deletes it from the cart.	**High**	**User: Remove Item**
6	User adds to the cart and want to change the quantity of items to purchase	User adds to the card, proceeds to shop more, and wants to add more of the same item to the cart. So clicks to increase quantity in the cart	**medium**	**User: increase Item**
7	User logs can see what they have ordered and update profile	User logs in goes to profile page and can modify details and see orders	**Low**	**User:Profile**

8	Admin logs in and can see all the orders that have been made	Admin Logs in and sees all order that have been made	**Low**	**Admin:Orders**

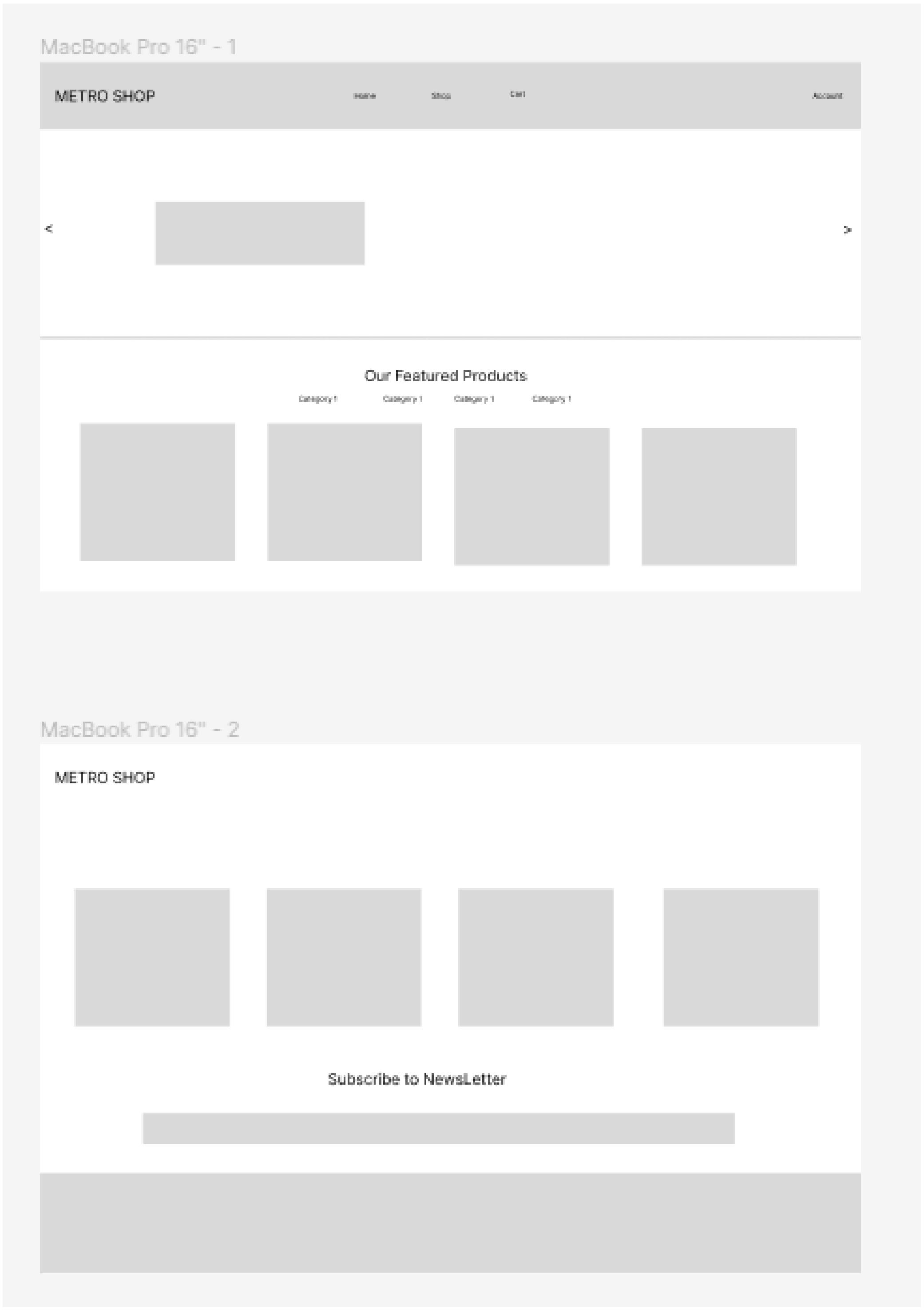
User Flow



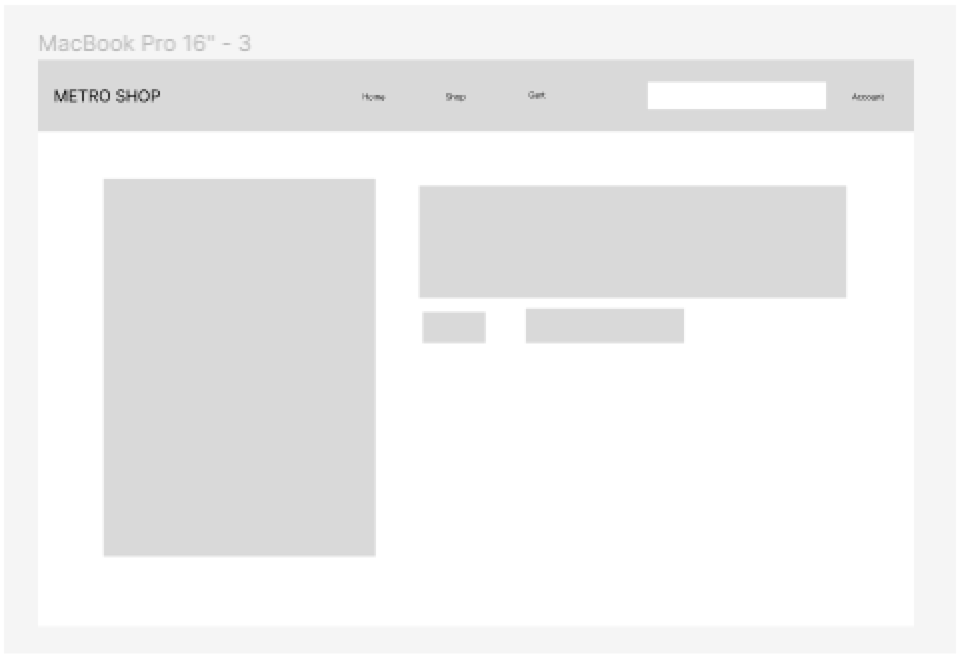
Users that click on the link are directed to the homepage with global nav.
Users can navigate through home, shop, cart and account.
Shop has features such as category button, search, and add to cart.
Cart has features like remove item, add/subtract quantity and purchase
Account has Login and register.
Check out is a protected route that requires users to have an account.
The user will go through the stripe checkout prompts.

Wireframe Design - Figma

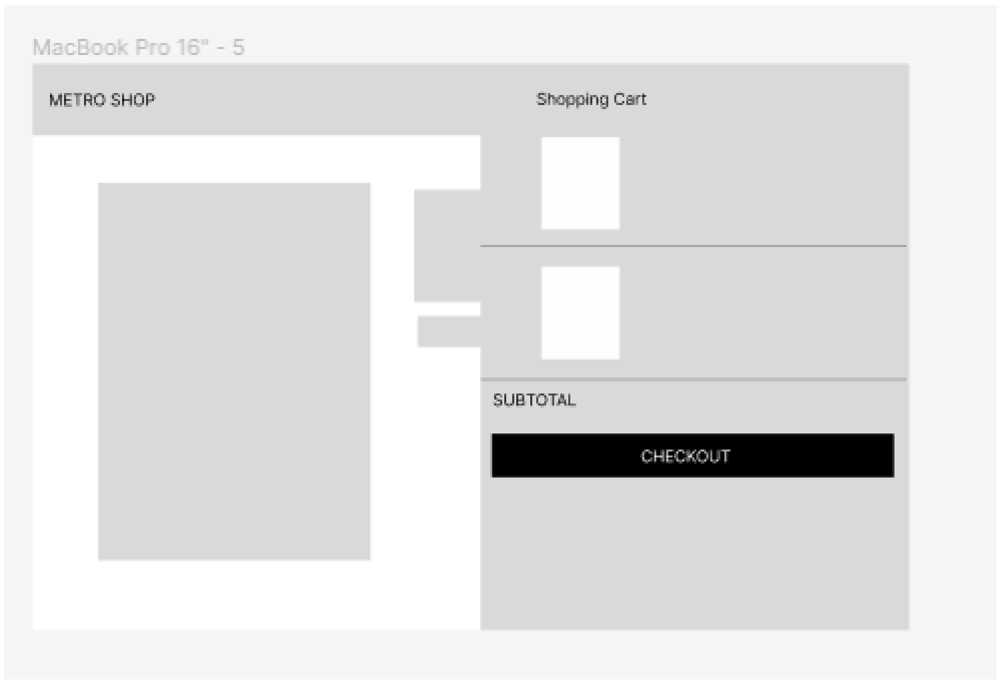
Home Page



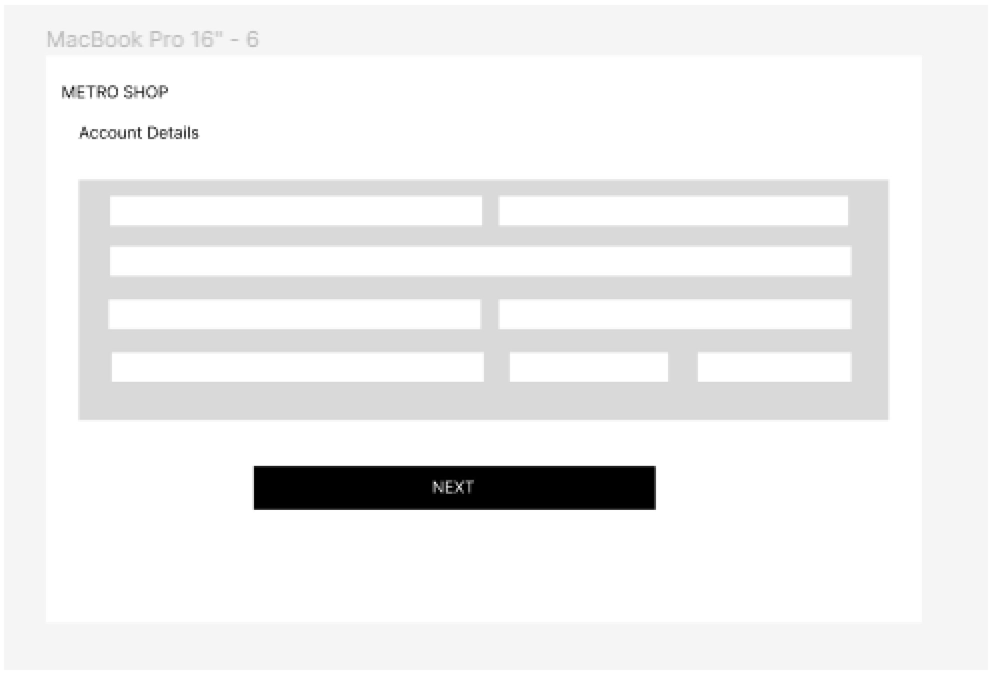
Item Page

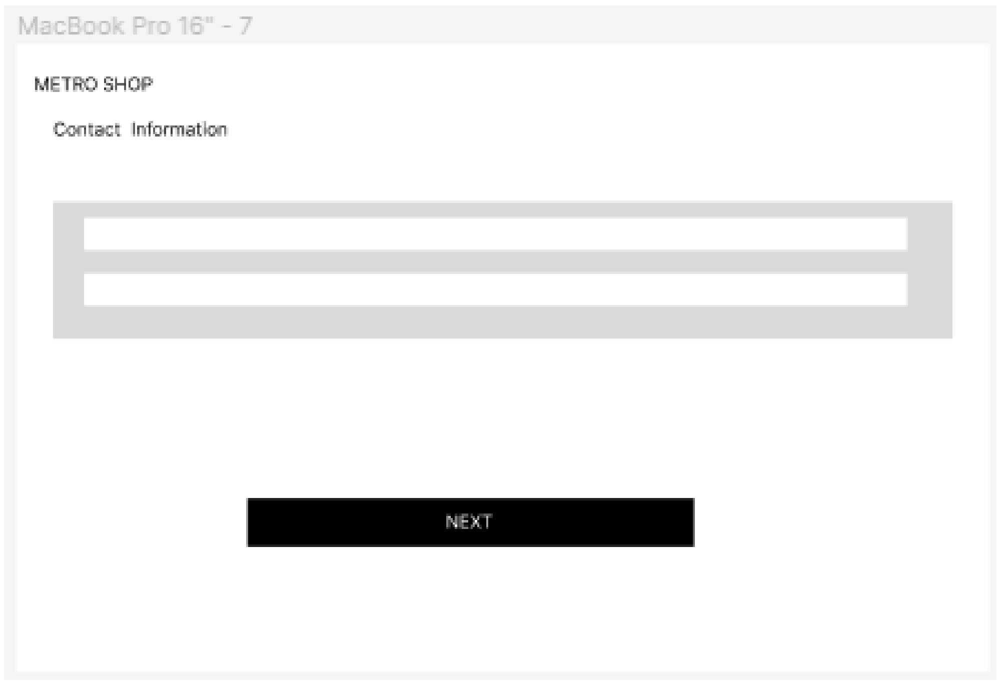


Shopping Cart

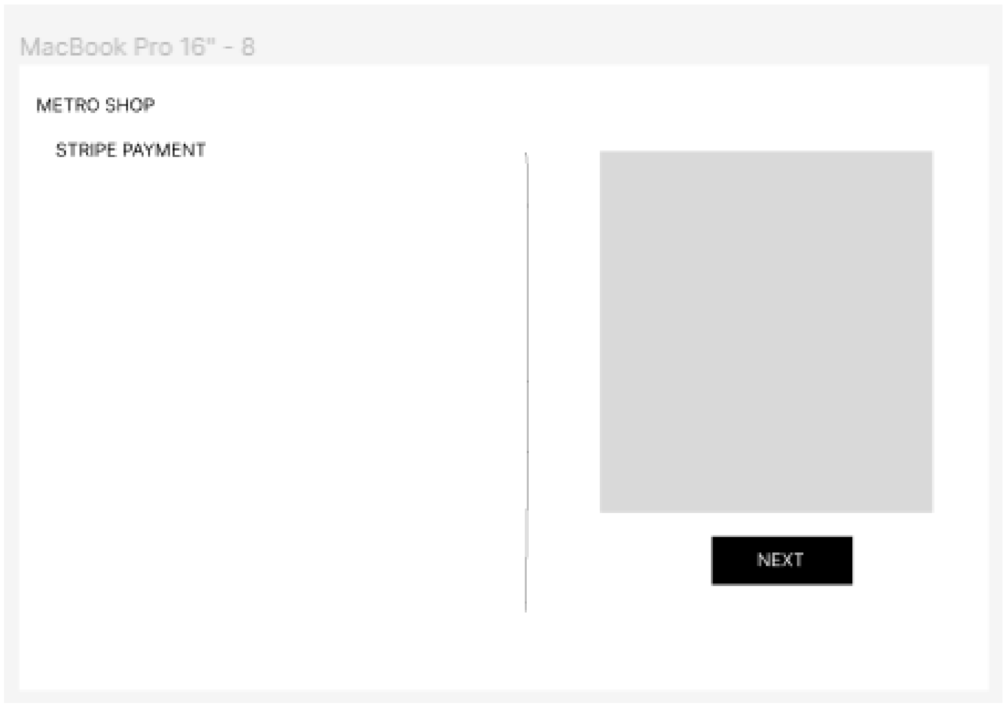


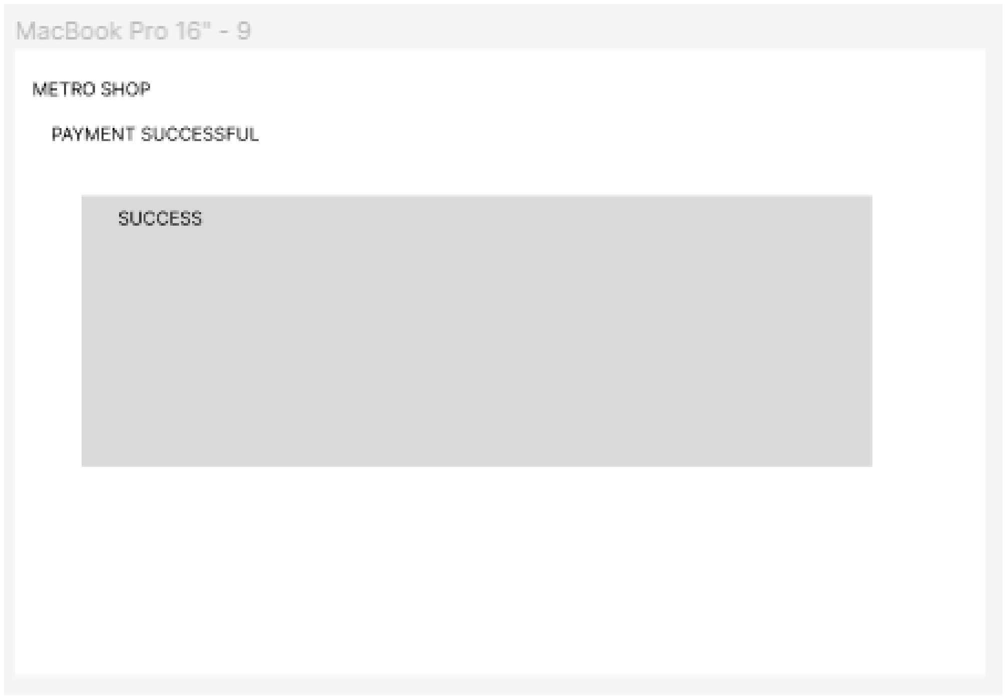
Checkout Details





Stripe Payment





Open Questions/Out of Scope
out of scope

- Using Stripe outside of the Test environment.
- Incorporating google maps api and actual address checking. Currently the user can send to anywhere even a made up address.
- Profile picture images.
- HTTPS integration for security .
- Refunds.
- Ability to become a reseller. Only have admin and user privileges currently.

Non-functional Requirements
What are the key security requirements?

- Login.
- storage of personal details.
- Inactivity timeout after 3 hrs.
- Data encryption.
- Web page loading as fast as possible.
- Images loading as fast as possible.

How many transactions should be enabled at peak time?

- Stripe can handle large amount of volume and to secure checkout by secure means

How easy to use does the software need to be?

- Very user friendly so that users and admin can use it without external help and any user can navigate through most of the site.

How quickly should the application respond to user requests?

- As fast as possible using things like local storage and a reliable backend to fetch data.

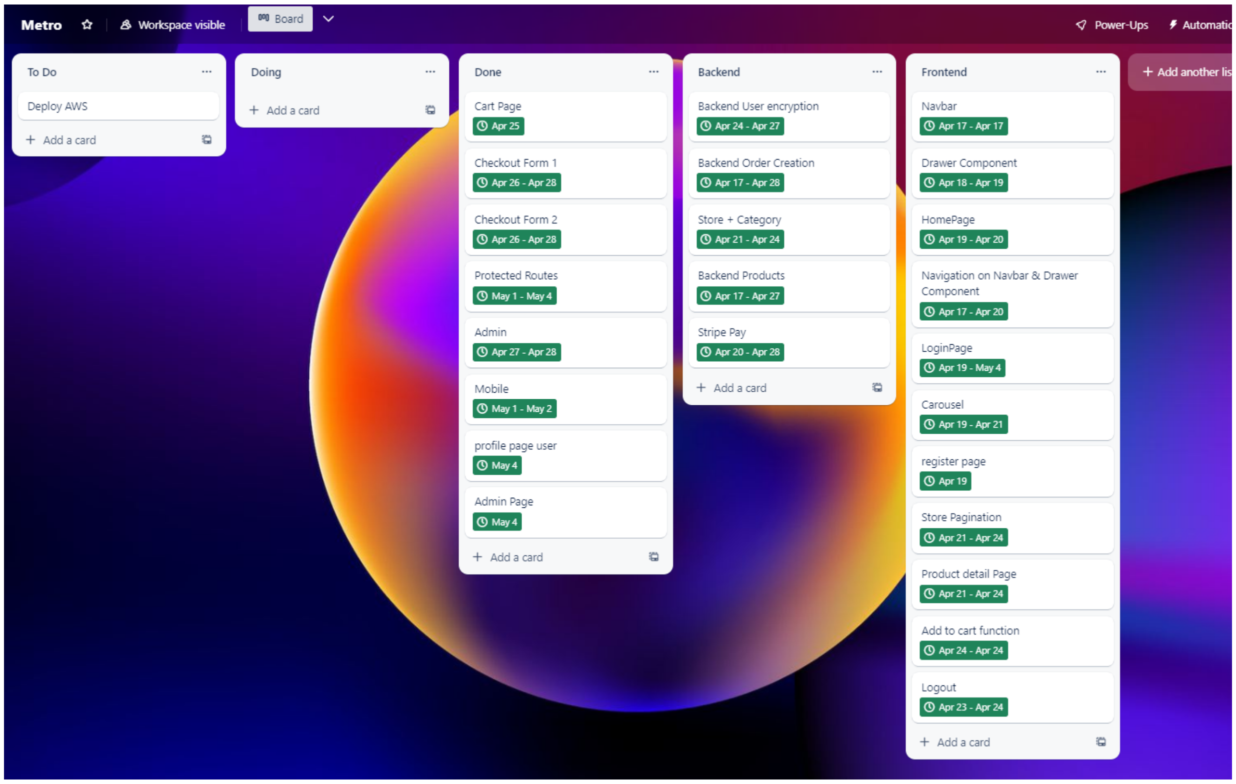
How reliable must the application be? (e.g. mean time between failures)

- Very reliable so that if there is any problem it can be solved by refresh or the user can deal with it

Does the software conform to any technical standards to ease maintainability?

- Yes, clean code to conform with the clean code guidelines
- Use Rest API
- Using Stripe to confirm with global security standards

Project Planning
Used a trello board to keep on track of development. Also created backend diagrams and user flow diagrams and user stories before I started. This way I had a reference on the direction.



To Do
- Deploy AWS
- + Add a card

Doing
- + Add a card

Done
- Cart Page — Apr 25
- Checkout Form 1 — Apr 26 - Apr 28
- Checkout Form 2 — Apr 26 - Apr 28
- Protected Routes — May 1 - May 4
- Admin — Apr 27 - Apr 28
- Mobile — May 1 - May 2
- profile page user — May 4
- Admin Page — May 4
- + Add a card

Backend
- Backend User encryption — Apr 24 - Apr 27
- Backend Order Creation — Apr 17 - Apr 28
- Store + Category — Apr 21 - Apr 24
- Backend Products — Apr 17 - Apr 27
- Stripe Pay — Apr 20 - Apr 28
- + Add a card

Frontend
- Navbar — Apr 17 - Apr 17
- Drawer Component — Apr 18 - Apr 19
- HomePage — Apr 19 - Apr 20
- Navigation on Navbar & Drawer Component — Apr 17 - Apr 20
- LoginPage — Apr 19 - May 4
- Carousel — Apr 19 - Apr 21
- register page — Apr 19
- Store Pagination — Apr 21 - Apr 24
- Product detail Page — Apr 21 - Apr 24
- Add to cart function — Apr 24 - Apr 24
- Logout — Apr 23 - Apr 24
- + Add a card

Testing Strategy
Front end
What were steps undertaken to achieve product quality?

- To achieve a high product quality consistent testing during the development cycle had to be done. I used two methods including End-to-End Testing and Integration Testing.

How was each feature of the application tested?

- End-to-End Testing: This involves testing the application as a whole from a user and admin perspective. I tested each feature as I was in the process of the development cycle. Found issues along the way which would then lead to new edge cases being found.
- Integration Testing: Tested how different components of the application interact with each other, specifically the front end interacting with the backend-api. I used tools like compass CRUD operations to test this.

How did you handle edge cases?

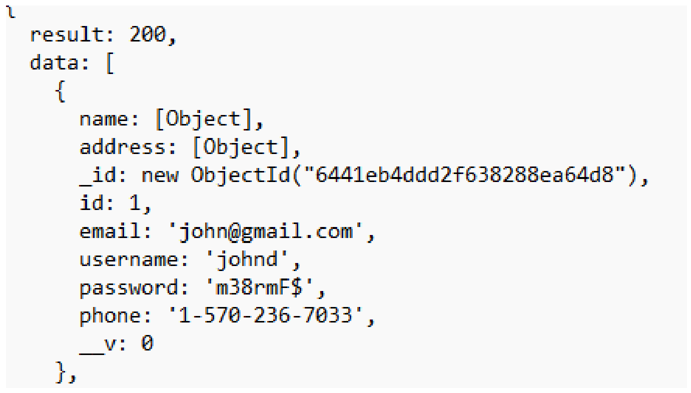
- Checking Input validation, making sure the correct input and data is being passed. I had a particularly hard time with this as my checkout process involved passing information through multiple forms. To solve this I had to retrieve and modify data multiple times

where sometimes I would lose data. It was important to test and find a consistent way to retrieve and pass data through the forms. This data is then vital for Stripe integration.
- Error handling, making catches and console logs to make sure the data and what data was being passed through.

Test user stories to see correct behavior.

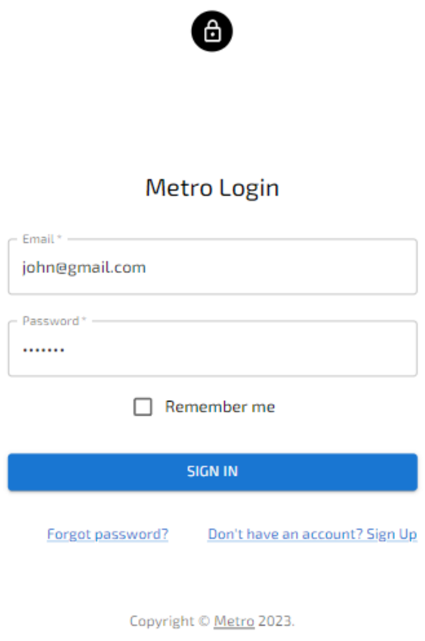
Test case 1 - Admin logs in to create new products

Admin Login



```
{
  result: 200,
  data: [
    {
      name: [Object],
      address: [Object],
      _id: new ObjectId("6441eb4ddd2f638288ea64d8"),
      id: 1,
      email: 'john@gmail.com',
      username: 'johnd',
      password: 'm38rmF$',
      phone: '1-570-236-7033',
      __v: 0
    },
```

Admin Login Page

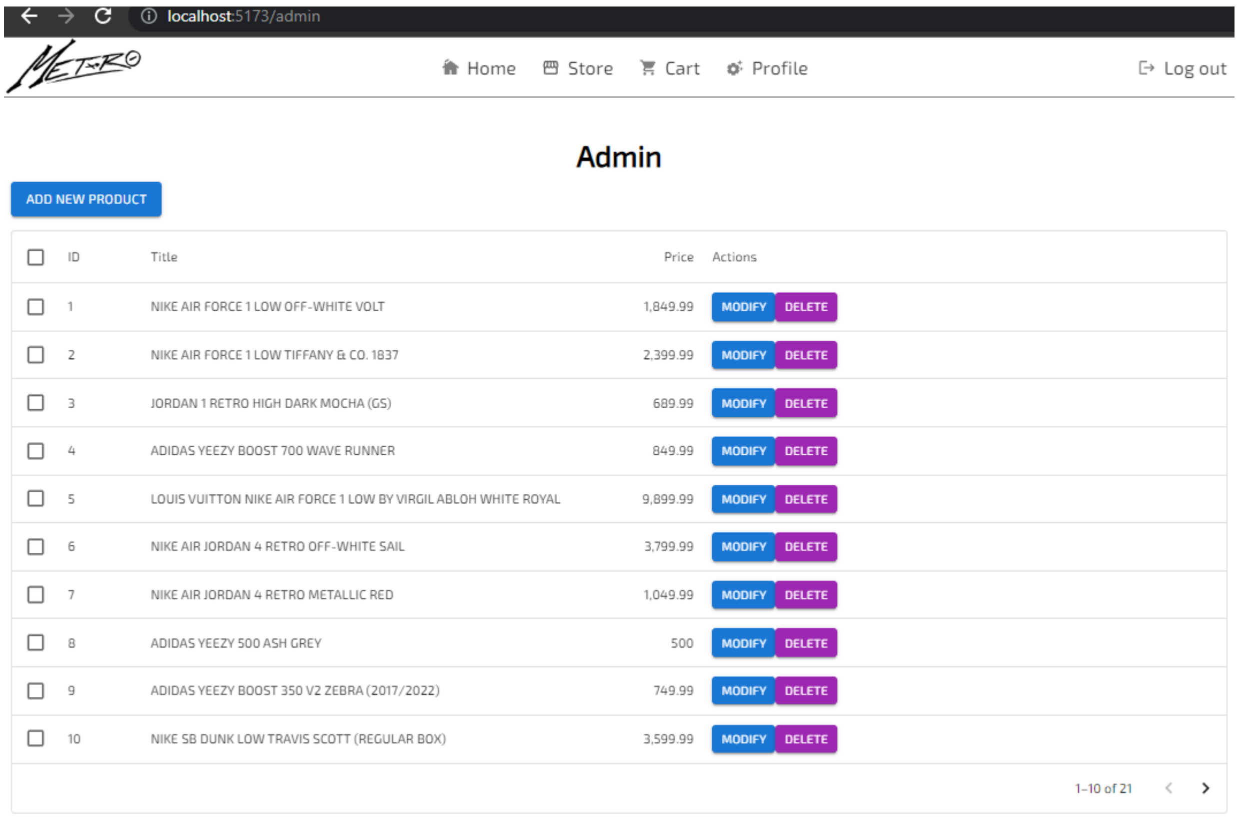


Metro Login

Email *: john@gmail.com

Password *: •••••••

Admin Page



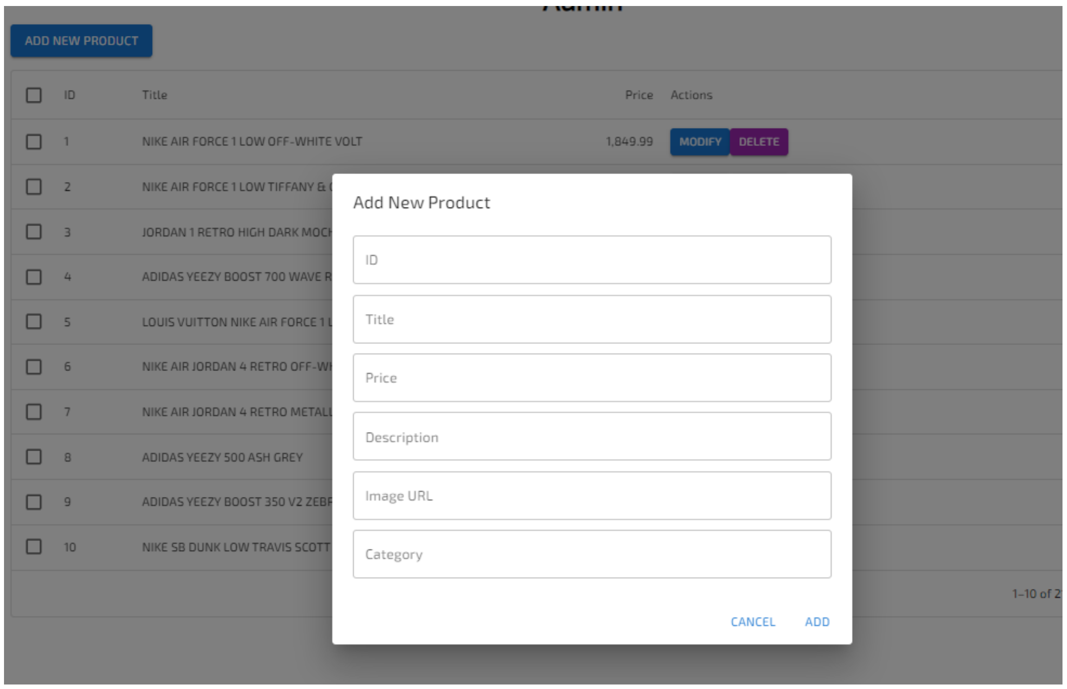
Admin

ADD NEW PRODUCT

	ID	Title	Price	Actions
☐	1	NIKE AIR FORCE 1 LOW OFF-WHITE VOLT	1,849.99	MODIFY DELETE
☐	2	NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837	2,399.99	MODIFY DELETE
☐	3	JORDAN 1 RETRO HIGH DARK MOCHA (GS)	689.99	MODIFY DELETE
☐	4	ADIDAS YEEZY BOOST 700 WAVE RUNNER	849.99	MODIFY DELETE
☐	5	LOUIS VUITTON NIKE AIR FORCE 1 LOW BY VIRGIL ABLOH WHITE ROYAL	9,899.99	MODIFY DELETE
☐	6	NIKE AIR JORDAN 4 RETRO OFF-WHITE SAIL	3,799.99	MODIFY DELETE
☐	7	NIKE AIR JORDAN 4 RETRO METALLIC RED	1,049.99	MODIFY DELETE
☐	8	ADIDAS YEEZY 500 ASH GREY	500	MODIFY DELETE
☐	9	ADIDAS YEEZY BOOST 350 V2 ZEBRA (2017/2022)	749.99	MODIFY DELETE
☐	10	NIKE SB DUNK LOW TRAVIS SCOTT (REGULAR BOX)	3,599.99	MODIFY DELETE

1–10 of 21

Add New Product



Add New Product

ID

Title

Price

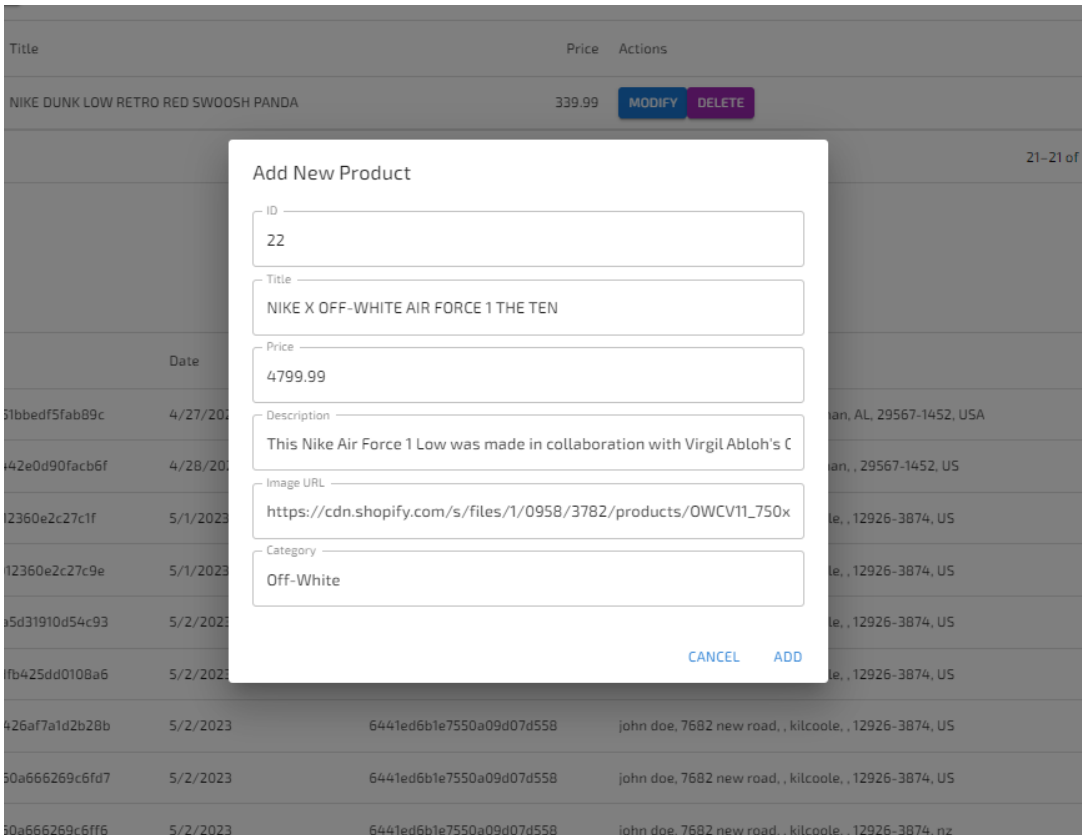
Description

Image URL

Category

CANCEL ADD

Fills out Details



Title	Price	Actions
NIKE DUNK LOW RETRO RED SWOOSH PANDA	339.99	MODIFY DELETE

21–21 of

Add New Product

ID: 22

Title: NIKE X OFF-WHITE AIR FORCE 1 THE TEN

Price: 4799.99

Description: This Nike Air Force 1 Low was made in collaboration with Virgil Abloh's C

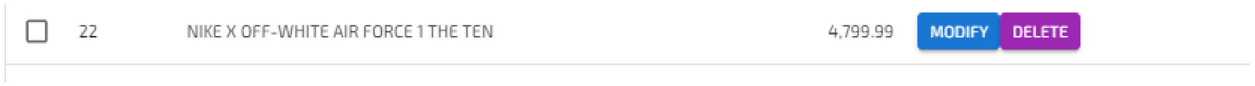
Image URL: https://cdn.shopify.com/s/files/1/0958/3782/products/OWCV11_750x

Category: Off-White

CANCEL ADD

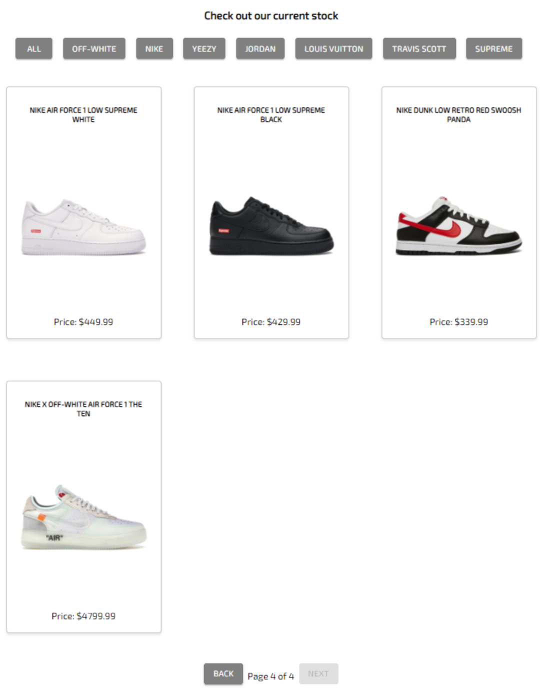
	Date		
51bbedf5fab89c	4/27/202		ian, AL, 29567-1452, USA
42e0d90facb6f	4/28/202		ian, , 29567-1452, US
12360e2c27c1f	5/1/2023		le, , 12926-3874, US
12360e2c27c9e	5/1/2023		le, , 12926-3874, US
a5d31910d54c93	5/2/2023		le, , 12926-3874, US
fb425dd0108a6	5/2/2023		le, , 12926-3874, US
426af7a1d2b28b	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, US
50a666269c6fd7	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, US
50a666269c6ff6	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, nz

Creates product



☐	22	NIKE X OFF-WHITE AIR FORCE 1 THE TEN	4,799.99	MODIFY DELETE

Visible to public


Check out our current stock

ALL | OFF-WHITE | NIKE | YEEZY | JORDAN | LOUIS VUITTON | TRAVIS SCOTT | SUPREME

NIKE AIR FORCE 1 LOW SUPREME WHITE
Price: $449.99

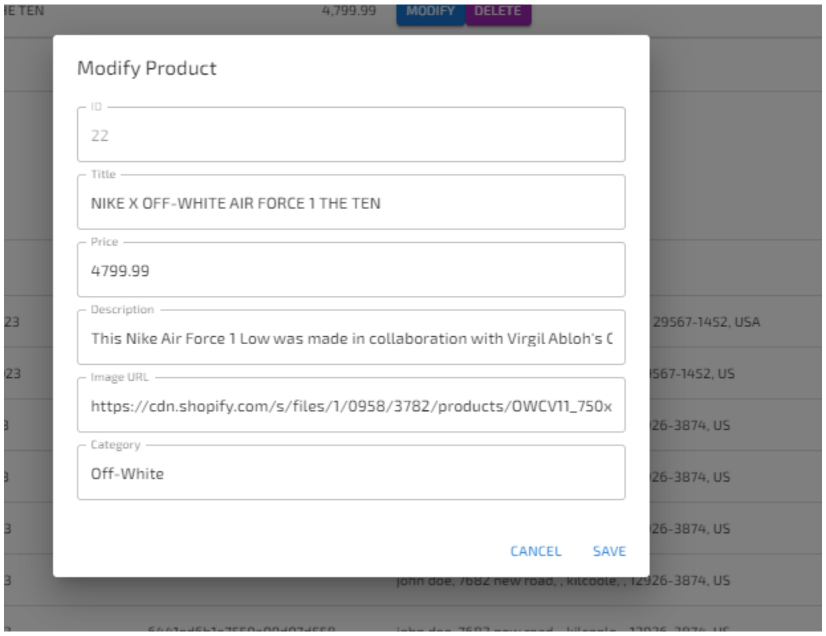
NIKE AIR FORCE 1 LOW SUPREME BLACK
Price: $429.99

NIKE DUNK LOW RETRO RED SWOOSH PANDA
Price: $339.99

NIKE X OFF-WHITE AIR FORCE 1 THE TEN
Price: $4799.99

BACK Page 4 of 4 NEXT

Test case 2 - Admin logs in to update new products


Modify Product

ID: 22

Title: NIKE X OFF-WHITE AIR FORCE 1 THE TEN

Price: 4799.99

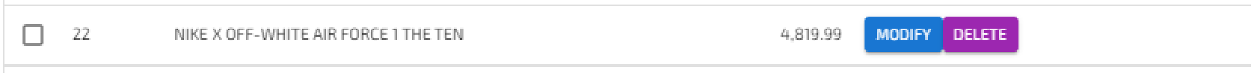
Description: This Nike Air Force 1 Low was made in collaboration with Virgil Abloh's C

Image URL: https://cdn.shopify.com/s/files/1/0958/3782/products/OWCV11_750x

Category: Off-White

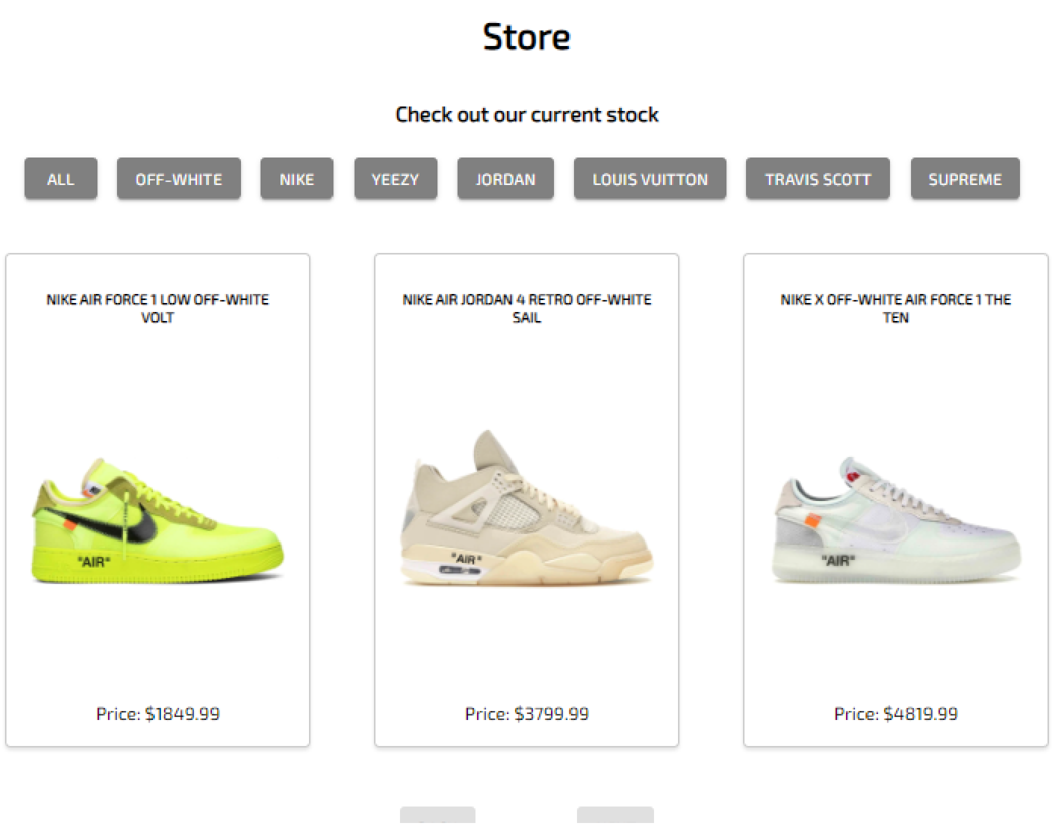
CANCEL SAVE

Price is updated



☐	22	NIKE X OFF-WHITE AIR FORCE 1 THE TEN	4,819.99	MODIFY DELETE

Change price from 4799.99 to 4819.99
Price is visible to public



Store

Check out our current stock

NIKE AIR FORCE 1 LOW OFF-WHITE VOLT

Price: $1849.99

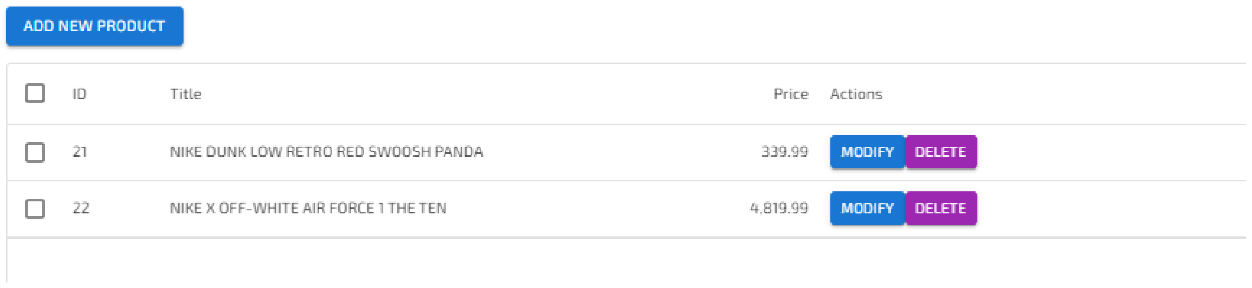
NIKE AIR JORDAN 4 RETRO OFF-WHITE SAIL

Price: $3799.99

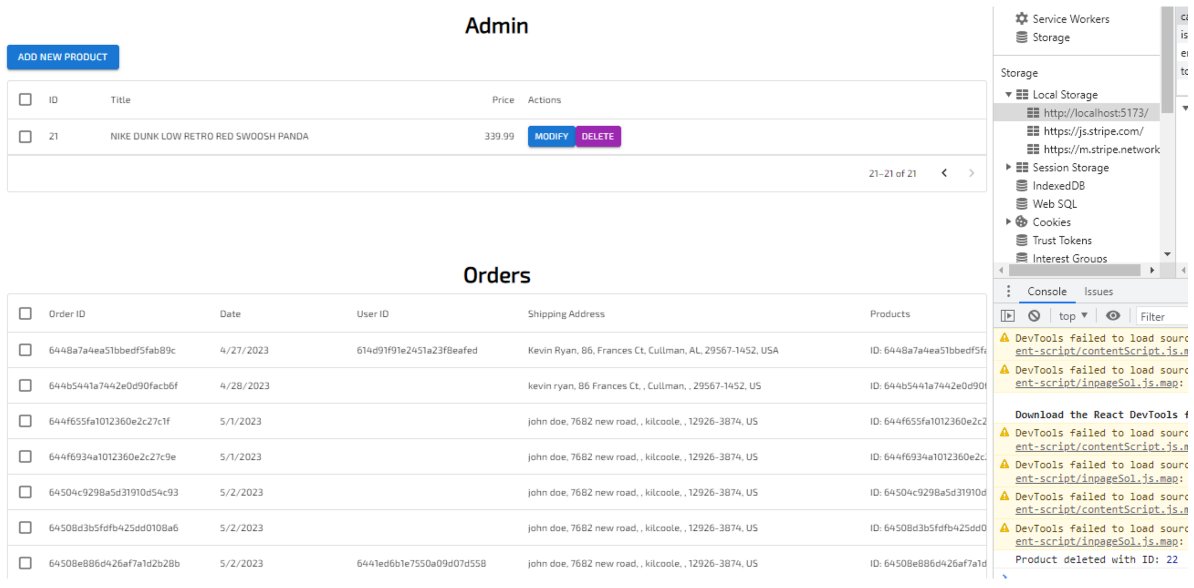
NIKE X OFF-WHITE AIR FORCE 1 THE TEN

Price: $4819.99

Test case 3 - Admin logs in to delete a product



	ID	Title	Price	Actions
☐	21	NIKE DUNK LOW RETRO RED SWOOSH PANDA	339.99	MODIFY DELETE
☐	22	NIKE X OFF-WHITE AIR FORCE 1 THE TEN	4,819.99	MODIFY DELETE



Admin

	ID	Title	Price	Actions
☐	21	NIKE DUNK LOW RETRO RED SWOOSH PANDA	339.99	MODIFY DELETE

21–21 of 21

Orders

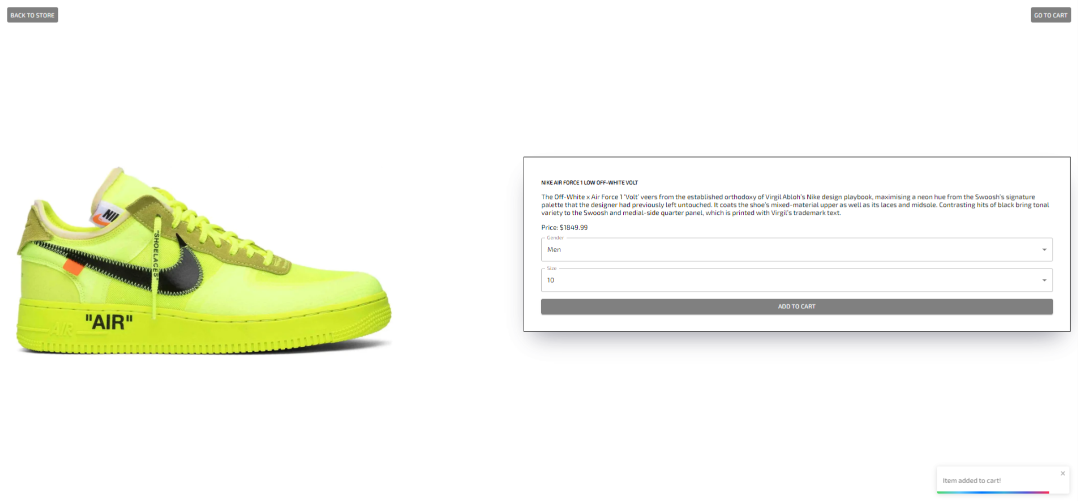
	Order ID	Date	User ID	Shipping Address	Products
☐	6448a7a4ea51bbedf5fab89c	4/27/2023	614d91f91e2451a23f8eafed	Kevin Ryan, 86, Frances Ct, Cullman, AL, 29567-1452, USA	ID: 6448a7a4ea51bbedf5fa
☐	644b5441a7442e0d90facb6f	4/28/2023		kevin ryan, 86 Frances Ct, , Cullman, , 29567-1452, US	ID: 644b5441a7442e0d90f
☐	644f655fa1012360e2c27c1f	5/1/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 644f655fa1012360e2c2
☐	644f6934a1012360e2c27c9e	5/1/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 644f6934a1012360e2c
☐	64504c9298a5d31910d54c93	5/2/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64504c9298a5d31910d
☐	64508d3b5fdfb425dd0108a6	5/2/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64508d3b5fdfb425dd0
☐	64508e886d426af7a1d2b28b	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64508e886d426af7a1d

```
DevTools failed to load sourc
ent-script/contentScript.js.m
DevTools failed to load sourc
ent-script/inpageSol.js.map:

Download the React DevTools f
DevTools failed to load sourc
ent-script/contentScript.js.m
DevTools failed to load sourc
ent-script/inpageSol.js.map:
DevTools failed to load sourc
ent-script/contentScript.js.m
DevTools failed to load sourc
ent-script/inpageSol.js.map:
Product deleted with ID: 22
```

Test case 4 - User goes onto the site, decides they like the item and proceeds to purchase it

Adds to cart



NIKE AIR FORCE 1 LOW OFF-WHITE VOLT

The Off-White x Air Force 1 'Volt' veers from the established orthodoxy of Virgil Abloh's Nike design playbook, maximising a neon hue from the Swoosh's signature palette that the designer had previously left untouched. It coats the shoe's mixed-material upper as well as its laces and midsole. Contrasting hits of black bring tonal variety to the Swoosh and medial-side quarter panel, which is printed with Virgil's trademark text.

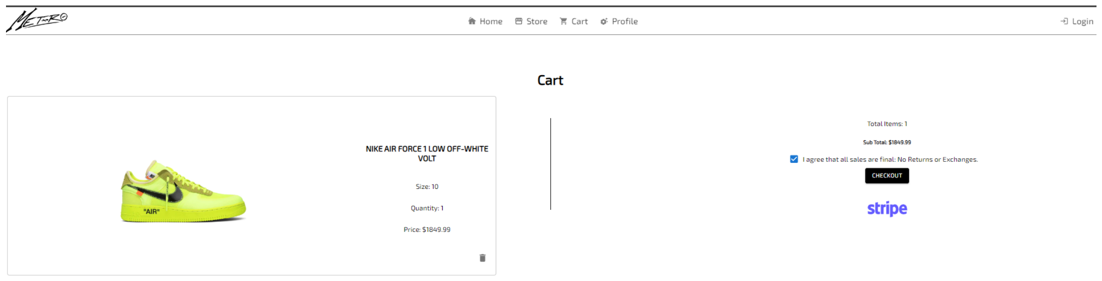
Price: $1849.99

Gender: Men

Size: 10

Item added to cart!

Proceeds to Checkout



Cart

NIKE AIR FORCE 1 LOW OFF-WHITE VOLT

Size: 10

Quantity: 1

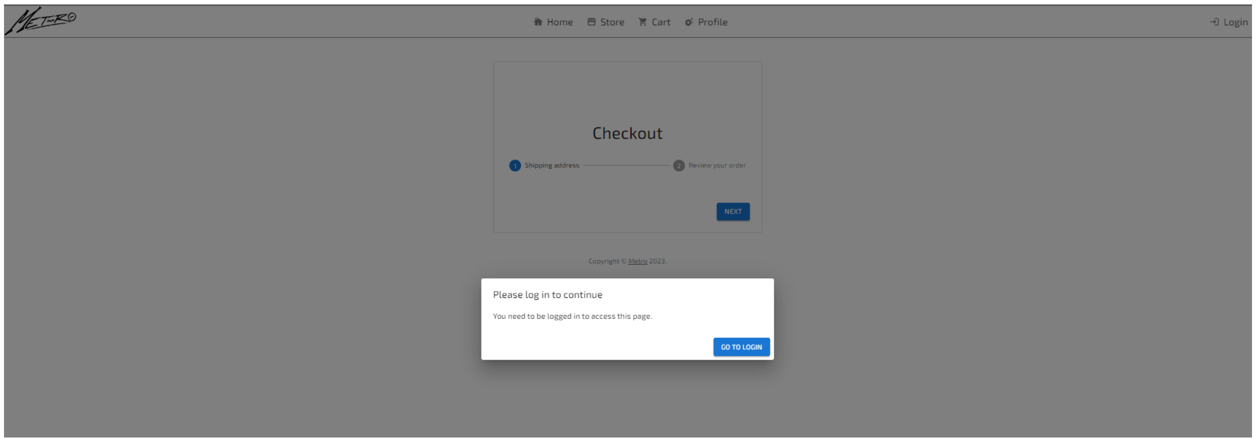
Price: $1849.99

Total Items: 1

Sub Total: $1849.99

☑ I agree that all sales are final: No Returns or Exchanges.

Gets prompted for login



Checkout

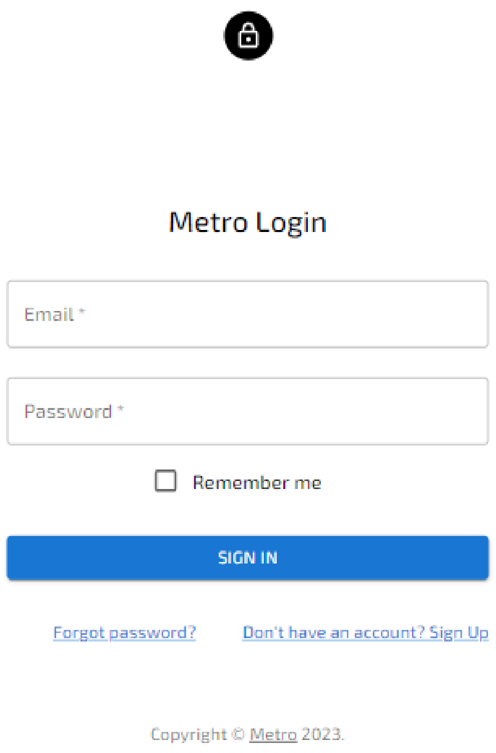
1 Shipping address — 2 Review your order

Copyright © Metro 2023.

Please log in to continue

You need to be logged in to access this page.

Goes to login page



Metro Login

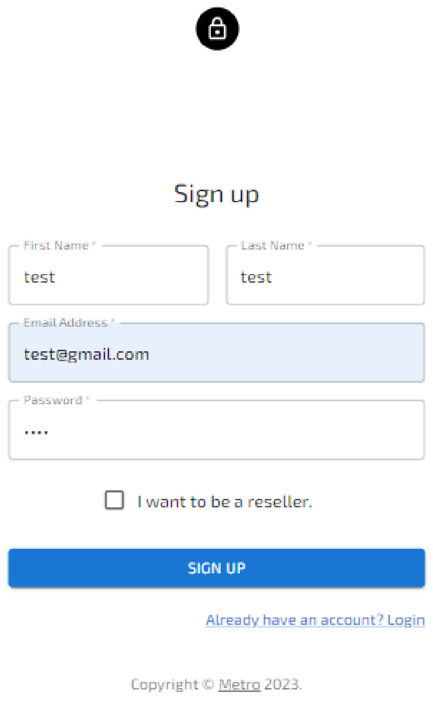
Email *

Password *

Remember me

Copyright © Metro 2023.

Doesn’t have an account so proceeds to register

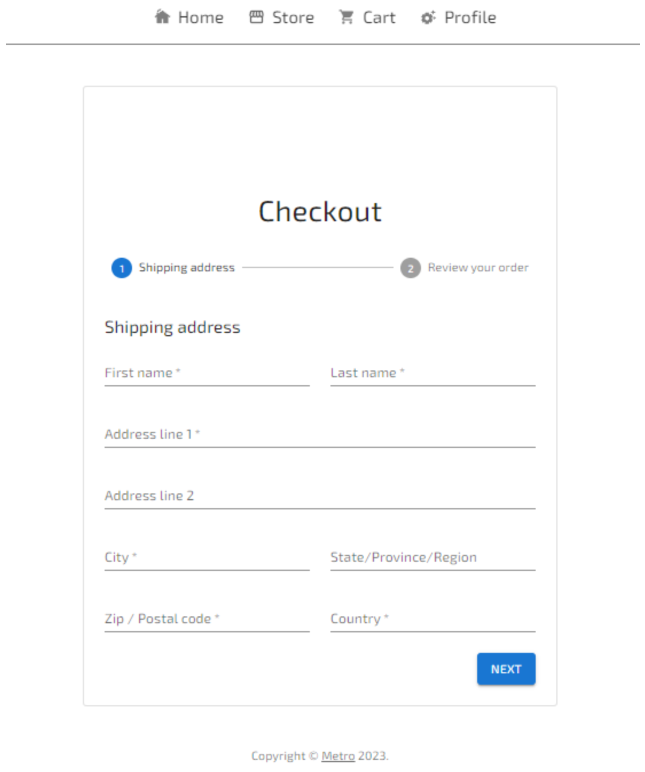


Sign up

First Name *	test
Last Name *	test
Email Address *	test@gmail.com
Password *	••••

☐ I want to be a reseller.

Can proceed to checkout



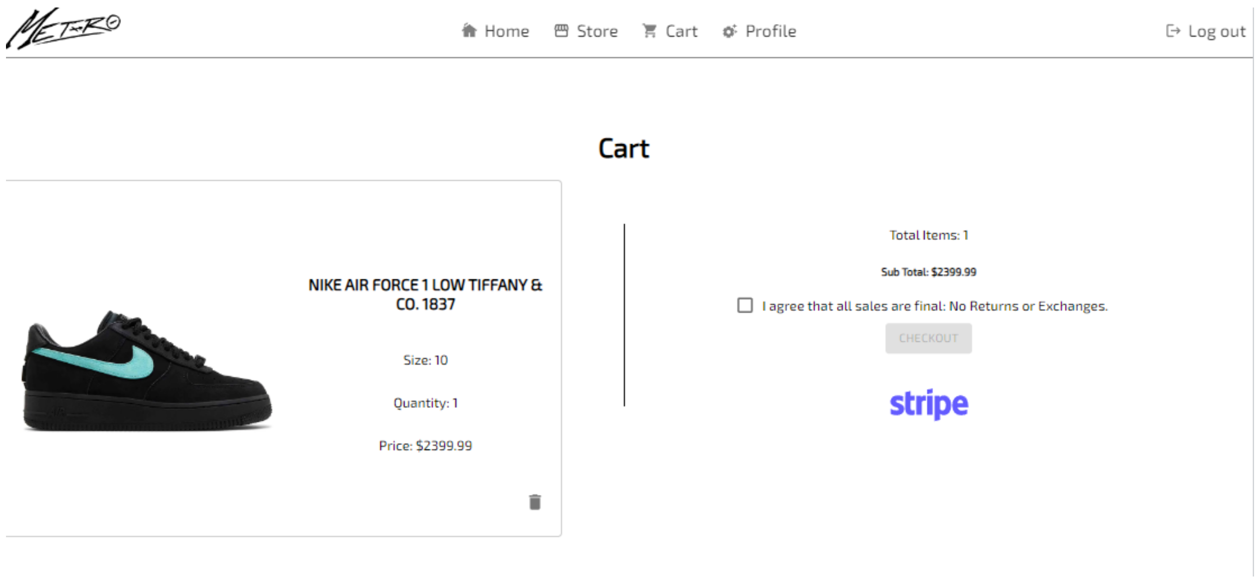
Checkout

First name *	
Last name *	
Address line 1 *	
Address line 2	
City *	
State/Province/Region	
Zip / Postal code *	
Country *	

Found error new user is not updating

Had to sort out the data being passed through.Nested it so it would pass through.

Test case 5 - user remove item
Logged in



Cart

NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837

Size: 10

Quantity: 1

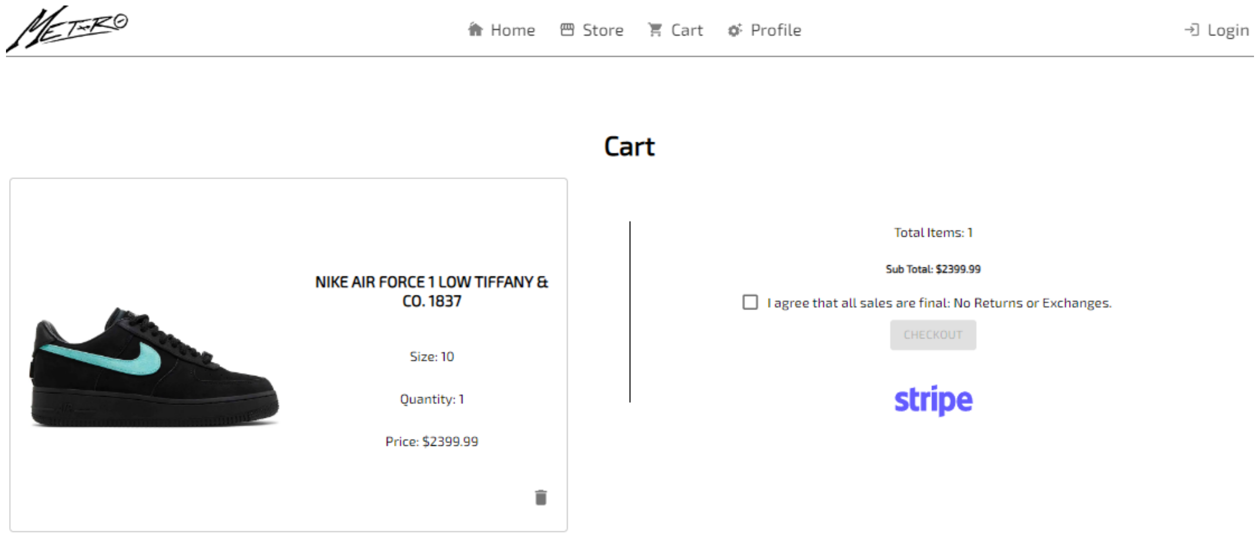
Price: $2399.99

Total Items: 1

Sub Total: $2399.99

I agree that all sales are final: No Returns or Exchanges.

Logged Out



Cart

NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837

Size: 10

Quantity: 1

Price: $2399.99

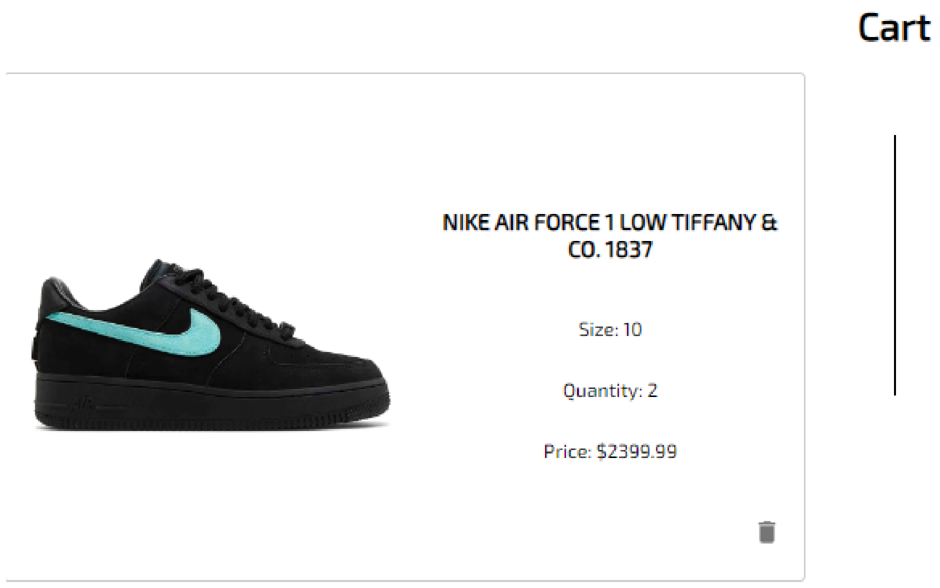
Total Items: 1

Sub Total: $2399.99

I agree that all sales are final: No Returns or Exchanges.

Works as expected

Test Case 6 - user change quantity



Cart

NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837

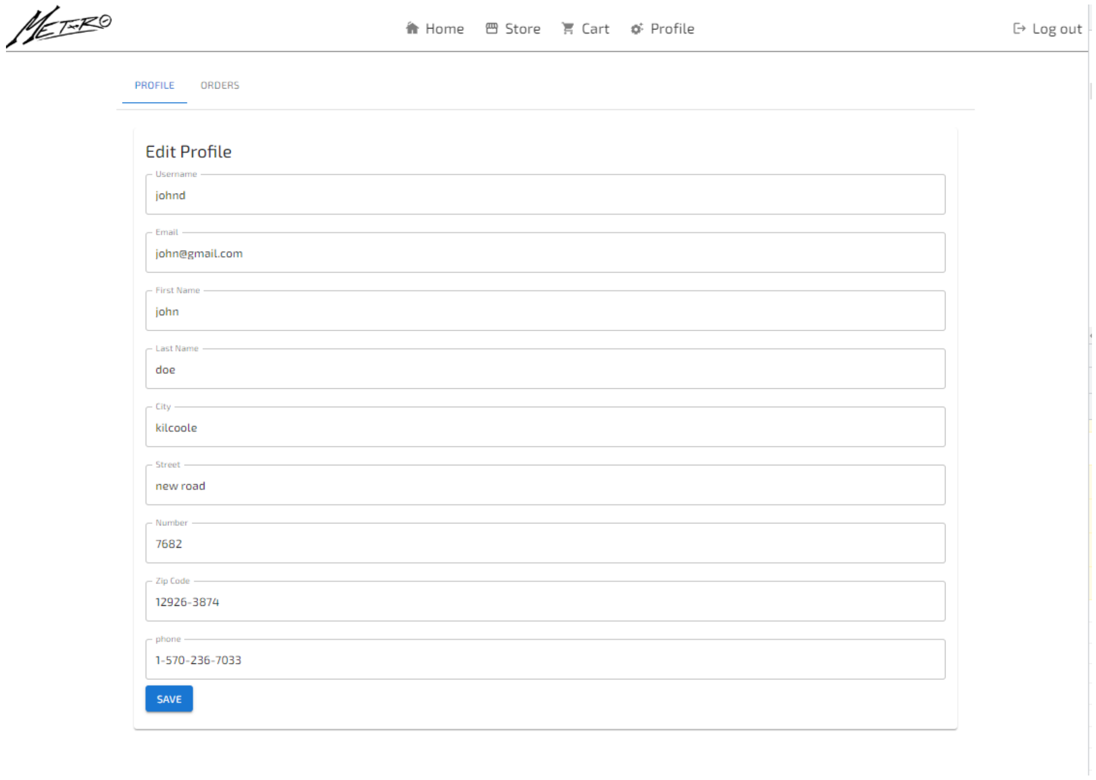
Size: 10

Quantity: 2

Price: $2399.99

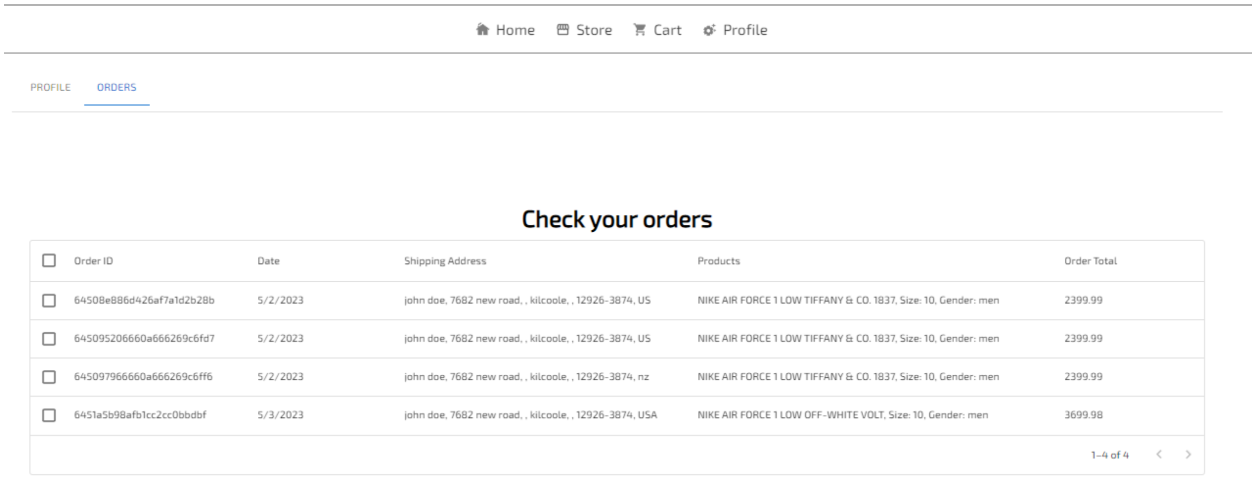
No toggle in size or quantity in cart but the user can still add more than one.
Found bug where if user adds same ID different sizes it deletes the both items with same ID.
Area for much improvement.

Test Case 7 - user can see orders and update profile



Edit Profile

Username	johnd
Email	john@gmail.com
First Name	john
Last Name	doe
City	kilcoole
Street	new road
Number	7682
Zip Code	12926-3874
phone	1-570-236-7033



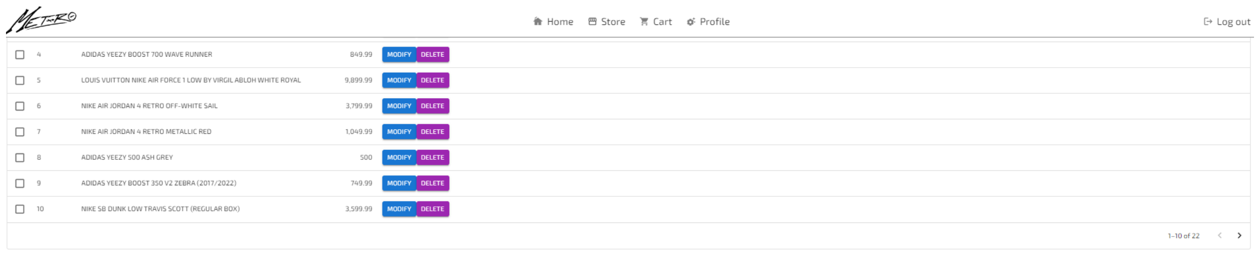
Check your orders

Order ID	Date	Shipping Address	Products	Order Total
64508e886d426af7a1d2b28b	5/2/2023	john doe, 7682 new road, , kilcoole, , 12926-3874, US	NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837, Size: 10, Gender: men	2399.99
645095206660a666269c6fd7	5/2/2023	john doe, 7682 new road, , kilcoole, , 12926-3874, US	NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837, Size: 10, Gender: men	2399.99
645097966660a666269c6ff6	5/2/2023	john doe, 7682 new road, , kilcoole, , 12926-3874, nz	NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837, Size: 10, Gender: men	2399.99
6451a5b98afb1cc2cc0bbdbf	5/3/2023	john doe, 7682 new road, , kilcoole, , 12926-3874, USA	NIKE AIR FORCE 1 LOW OFF-WHITE VOLT, Size: 10, Gender: men	3699.98

1–4 of 4

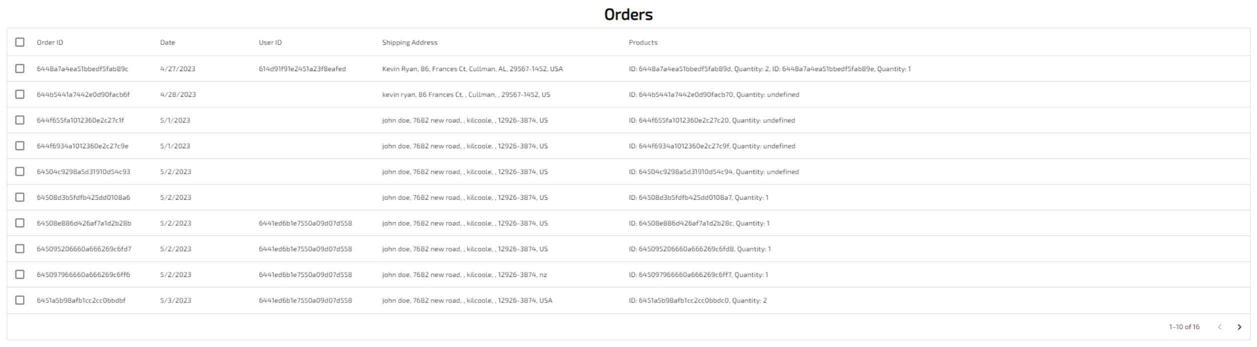
Had a bug in orders, fix when user login created object _id token as well to fetch orders from back end.

Test Case 8- admin logs in and sees all orders that can be made.



☐	4	ADIDAS YEEZY BOOST 700 WAVE RUNNER	849.99	MODIFY DELETE
☐	5	LOUIS VUITTON NIKE AIR FORCE 1 LOW BY VIRGIL ABLOH WHITE ROYAL	9,899.99	MODIFY DELETE
☐	6	NIKE AIR JORDAN 4 RETRO OFF-WHITE SAIL	3,799.99	MODIFY DELETE
☐	7	NIKE AIR JORDAN 4 RETRO METALLIC RED	1,049.99	MODIFY DELETE
☐	8	ADIDAS YEEZY 500 ASH GREY	500	MODIFY DELETE
☐	9	ADIDAS YEEZY BOOST 350 V2 ZEBRA (2017/2022)	749.99	MODIFY DELETE
☐	10	NIKE SB DUNK LOW TRAVIS SCOTT (REGULAR BOX)	3,599.99	MODIFY DELETE

1–10 of 22

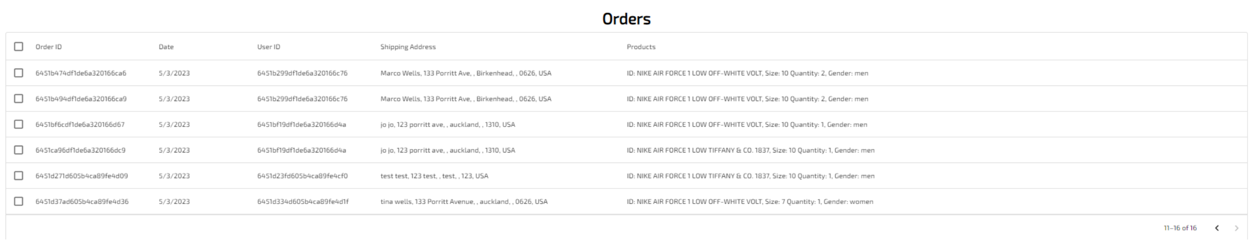


Orders

☐	Order ID	Date	User ID	Shipping Address	Products
☐	6448a7a4ea51bbedf5fab89c	4/27/2023	614d91f91e2451a23f8eafed	Kevin Ryan, 86, Frances Ct, Cullman, AL, 29567-1452, USA	ID: 6448a7a4ea51bbedf5fab89d, Quantity: 2, ID: 6448a7a4ea51bbedf5fab89e, Quantity: 1
☐	644b5441a7442e0d90facb6f	4/28/2023		kevin ryan, 86 Frances Ct, , Cullman, , 29567-1452, US	ID: 644b5441a7442e0d90facb70, Quantity: undefined
☐	644f655fa1012360e2c27c1f	5/1/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 644f655fa1012360e2c27c20, Quantity: undefined
☐	644f6934a1012360e2c27c9e	5/1/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 644f6934a1012360e2c27c9f, Quantity: undefined
☐	64504c9298a5d31910d54c93	5/2/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64504c9298a5d31910d54c94, Quantity: undefined
☐	64508d3b5fdfb425dd0108a6	5/2/2023		john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64508d3b5fdfb425dd0108a7, Quantity: 1
☐	64508e886d426af7a1d2b28b	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 64508e886d426af7a1d2b28c, Quantity: 1
☐	645095206660a666269c6fd7	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, US	ID: 645095206660a666269c6fd8, Quantity: 1
☐	645097966660a666269c6ff6	5/2/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, nz	ID: 645097966660a666269c6ff7, Quantity: 1
☐	6451a5b98afb1cc2cc0bbdbf	5/3/2023	6441ed6b1e7550a09d07d558	john doe, 7682 new road, , kilcoole, , 12926-3874, USA	ID: 6451a5b98afb1cc2cc0bbdc0, Quantity: 2

1–10 of 16

Found bug not showing correct ID so changed to product title

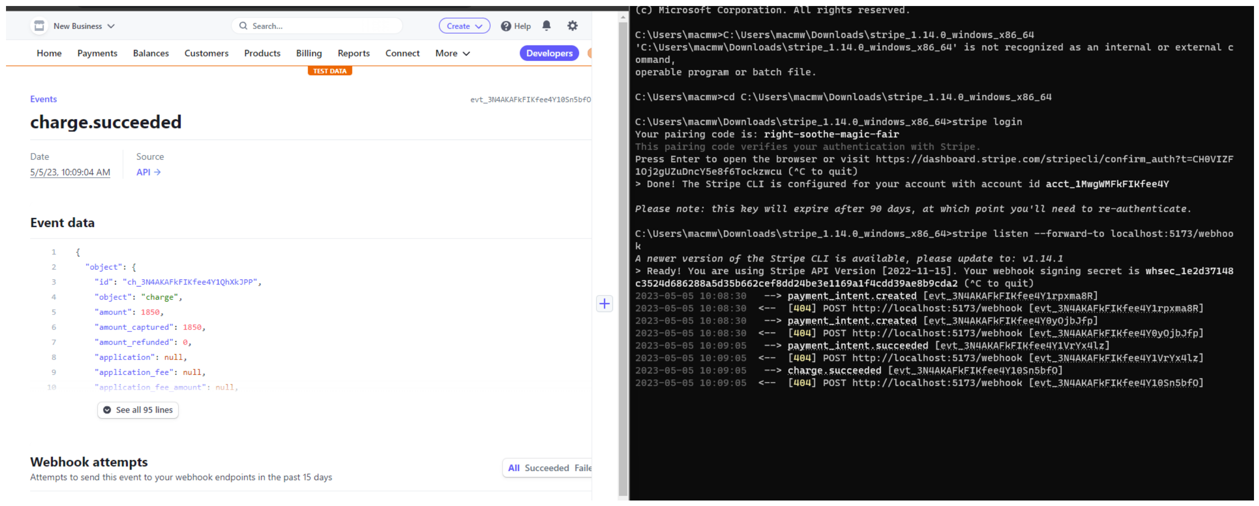


Orders

☐	Order ID	Date	User ID	Shipping Address	Products
☐	6451b474df1de6a320166ca6	5/3/2023	6451b299df1de6a320166c76	Marco Wells, 133 Porritt Ave, , Birkenhead, , 0626, USA	ID: NIKE AIR FORCE 1 LOW OFF-WHITE VOLT, Size: 10 Quantity: 2, Gender: men
☐	6451b494df1de6a320166ca9	5/3/2023	6451b299df1de6a320166c76	Marco Wells, 133 Porritt Ave, , Birkenhead, , 0626, USA	ID: NIKE AIR FORCE 1 LOW OFF-WHITE VOLT, Size: 10 Quantity: 2, Gender: men
☐	6451bf6cdf1de6a320166d67	5/3/2023	6451bf19df1de6a320166d4a	jo jo, 123 porritt ave, , auckland, , 1310, USA	ID: NIKE AIR FORCE 1 LOW OFF-WHITE VOLT, Size: 10 Quantity: 1, Gender: men
☐	6451ca96df1de6a320166dc9	5/3/2023	6451bf19df1de6a320166d4a	jo jo, 123 porritt ave, , auckland, , 1310, USA	ID: NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837, Size: 10 Quantity: 1, Gender: men
☐	6451d271d605b4ca89fe4d09	5/3/2023	6451d23fd605b4ca89fe4cf0	test test, 123 test, , test, , 123, USA	ID: NIKE AIR FORCE 1 LOW TIFFANY & CO. 1837, Size: 10 Quantity: 1, Gender: men
☐	6451d37ad605b4ca89fe4d36	5/3/2023	6451d334d605b4ca89fe4d1f	tina wells, 133 Porritt Avenue, , auckland, , 0626, USA	ID: NIKE AIR FORCE 1 LOW OFF-WHITE VOLT, Size: 7 Quantity: 1, Gender: women

11–16 of 16

Testing Stripe
Running a webhook



evt_3N4AKAFkFIKfee4Y10Sn5bf0

charge.succeeded

Date: 5/5/23, 10:09:04 AM
Source: API →

Event data

```
{
  "object": {
    "id": "ch_3N4AKAFkFIKfee4Y1QhXkJPP",
    "object": "charge",
    "amount": 1850,
    "amount_captured": 1850,
    "amount_refunded": 0,
    "application": null,
    "application_fee": null,
    "application_fee_amount": null,
```

Webhook attempts
Attempts to send this event to your webhook endpoints in the past 15 days

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\macmw>C:\Users\macmw\Downloads\stripe_1.14.0_windows_x86_64
'C:\Users\macmw\Downloads\stripe_1.14.0_windows_x86_64' is not recognized as an internal or external c
ommand,
operable program or batch file.

C:\Users\macmw>cd C:\Users\macmw\Downloads\stripe_1.14.0_windows_x86_64

C:\Users\macmw\Downloads\stripe_1.14.0_windows_x86_64>stripe login
Your pairing code is: right-soothe-magic-fair
This pairing code verifies your authentication with Stripe.
Press Enter to open the browser or visit https://dashboard.stripe.com/stripecli/confirm_auth?t=CH0VIZF
1Oj2gUZuDncY5e8f6Tockzwcu (^C to quit)
> Done! The Stripe CLI is configured for your account with account id acct_1MwgWMFkFIKfee4Y

Please note: this key will expire after 90 days, at which point you'll need to re-authenticate.

C:\Users\macmw\Downloads\stripe_1.14.0_windows_x86_64>stripe listen --forward-to localhost:5173/webhoo
k
A newer version of the Stripe CLI is available, please update to: v1.14.1
> Ready! You are using Stripe API Version [2022-11-15]. Your webhook signing secret is whsec_1e2d37148
c3524d686288a5d35b662cef8dd24be3e1169a1f4cdd39ae8b9cda2 (^C to quit)
2023-05-05 10:08:30   --> payment_intent.created [evt_3N4AKAFkFIKfee4Y1rpxma8R]
2023-05-05 10:08:30  <--  [404] POST http://localhost:5173/webhook [evt_3N4AKAFkFIKfee4Y1rpxma8R]
2023-05-05 10:08:30   --> payment_intent.created [evt_3N4AKAFkFIKfee4Y0yOjbJfp]
2023-05-05 10:08:30  <--  [404] POST http://localhost:5173/webhook [evt_3N4AKAFkFIKfee4Y0yOjbJfp]
2023-05-05 10:09:05   --> payment_intent.succeeded [evt_3N4AKAFkFIKfee4Y1VrYx4lz]
2023-05-05 10:09:05  <--  [404] POST http://localhost:5173/webhook [evt_3N4AKAFkFIKfee4Y1VrYx4lz]
2023-05-05 10:09:05   --> charge.succeeded [evt_3N4AKAFkFIKfee4Y10Sn5bf0]
2023-05-05 10:09:05  <--  [404] POST http://localhost:5173/webhook [evt_3N4AKAFkFIKfee4Y10Sn5bf0]
```

This way we can see what data is being passed through stripe.

Backend testing

Using postman requests to test the backend.

User

```
//TEST

//GET
//http://127.0.0.1:8000/api/users/
//retrive list of all users
//http://127.0.0.1:8000/api/users/1
//get user 1 info

//POST
// http://127.0.0.1:8000/api/users/create
/*
{
    "id":"11",
    "email":"mwz@gmail.com",
    "username":"macmwz",
    "password":"1234",
    "name":{
        "firstname":"Marco",
        "lastname":"Wells"
        },
    "address":{
    "city":"Auckland",
    "street":"street ave",
    "number": "123",
    "zipcode":"1234",
    "geolocation":{
        "lat":"12.23.43",
        "long":"1.2.3.4"
        }
    },
    "phone":"09 123 4567"
}
*/
//PUT
//http://127.0.0.1:8000/api/users/update/11
```

```
/*
{
    "id":"11",
    "email":"CHANGED@gmail.com",
    "username":"macmwz",
    "password":"1234567",
    "name":{
        "firstname":"Marco",
        "lastname":"Wells"
        },
    "address":{
    "city":"Auckland",
    "street":"street ave",
    "number": "123",
    "zipcode":"1234",
    "geolocation":{
        "lat":"12.23.43",
        "long":"1.2.3.4"
        }
    },
    "phone":"09 123 4567"
}

   */

//DELETE

// http://127.0.0.1:8000/api/users/delete/11

//WORKING AFTER TESTING

//http://127.0.0.1:8000/api/users/register

/*
{
    "email": "mwzzzzz@gmail.com",
    "password": "12345",
    "firstName": "Marcos",
    "lastName": "Wellss"
}
```

```
*/
```

Products

```

//TEST
//GET

// http://127.0.0.1:8000/api/product  retrive all
// http://127.0.0.1:8000/api/products/1 retrive 1 product id

//post

// http://127.0.0.1:8000/api/products/create
/*
{
    "id": 100,
    "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
    "price": 109.95,
    "description": "Your perfect pack for everyday use and walks in the
forest. Stash your...",
    "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg",
    "category": "men's clothing",
    "__v": 0
}
 */

//PUT
//http://127.0.0.1:8000/api/product/update/100

/*
{
    "id": 100,
    "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
    "price": 120,
    "description": "Your perfect pack for everyday use and walks in the
forest. Stash your...",
    "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg",
    "category": "men's clothing",
    "__v": 0
}
*/
//delete
//http://127.0.0.1:8000/api/products/delete/100

//WORKING AFTER TESTING
```

Orders

```
//TEST
//GET

//http://localhost:8000/api/orders

//http://localhost:8000/api/orders/64362fade8d168035a4cb92f

//POST

//http://127.0.0.1:8000/api/orders/create

/*
{
  "id": 1,
  "userId": "64362039ff6373d8692cc6f7",
  "date": "2023-04-12T12:34:56.789Z",
  "products": [
    {
      "productId": "64376f90ab8111f192d252cf",
      "quantity": 2
    },
    {
      "productId": "64376f90ab8111f192d252d0",
      "quantity": 1
    }
  ]
}


{
  "_id": "64362039ff6373d8692cc6f8",
  "userId": "64362039ff6373d8692cc6f7",
  "date": "2023-04-12T12:34:56.789Z",
  "products": [
    {
      "productId": "64376f90ab8111f192d252cf",
      "quantity": 2
    },
    {
      "productId": "64376f90ab8111f192d252d0",
      "quantity": 1
    }
  ]
}


{
  "_id": "NEW USER ID",
  "userId": "6437ac1012827d41e7a3e018",
  "date": "2023-04-12T12:34:56.789Z",
  "products": [
    {
      "productId": "64376f90ab8111f192d252cf",
      "quantity": 2
    },
    {
      "productId": "64376f90ab8111f192d252d0",
      "quantity": 1
    }
  ]
}

 */
//delete
//http://localhost:8000/api/orders/delete/
//Problems with this one
```

Implementation

What were the considerations for deploying the software?

- Use AWS to deploy the back end and front end.

End-to-end solution

How well did the software meet its objectives?

- The project met all front end and back end requirements. The order creation has a bug where the order is generated first and then the stripe payment is sent through second. If it was to be a fully functional checkout system then I would need to retrieve back a payment success and then create the order. I would have to debug using webhooks and do more stripe integration for it to be a fully functional site. Apart from this, the font end and backend are fully functional

References
Where is the code used in the project? (link to GitHub)
https://github.com/redm3/Capstone

● What are the resources used in the project? (libraries, APIs, databases, tools, etc)
For backend:
MongoDB
Compass

```
    "@stripe/react-stripe-js": "^2.1.0",
    "@stripe/stripe-js": "^1.52.1",
    "axios": "^1.3.6",
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "express": "^4.18.2",
    "mongoose": "^7.0.4",
    "stripe": "^12.1.1"
```

Data was originally pulled from https://fakestoreapi.com/ Using a one-off function for users & products which is commented out in userController and productController respectively.The function automatically encrypts the users passwords.

Manually grant the user to be an admin by setting the state to true or false inside mongoDB.

For frontend:

```
    "@emotion/react": "^11.10.8",
    "@emotion/styled": "^11.10.8",
    "@mui/icons-material": "^5.11.16",
    "@mui/lab": "^5.0.0-alpha.128",
    "@mui/material": "^5.12.2",
    "@mui/x-data-grid": "^6.2.1",
    "@reduxjs/toolkit": "^1.9.3",
    "@stripe/react-stripe-js": "^2.1.0",
    "@stripe/stripe-js": "^1.52.1",
    "axios": "^1.3.6",
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "express": "^4.18.2",
    "formik": "^2.2.9",
    "framer-motion": "^10.12.4",
    "isotope-layout": "^3.0.6",
    "jsonwebtoken": "^9.0.0",
    "jwt-decode": "^3.1.2",
    "material-ui-dropzone": "^3.5.0",
    "mongoose": "^7.0.4",
    "nanoid": "^4.0.2",
    "nodemon": "^2.0.22",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-dropzone": "^14.2.3",
    "react-hook-form": "^7.43.9",
    "react-isotope": "^1.0.7",
    "react-multi-carousel": "^2.8.3",
    "react-particles-js": "^3.6.0",
    "react-redux": "^8.0.5",
    "react-responsive-carousel": "^3.2.23",
    "react-router-dom": "^6.10.0",
    "react-stripe-checkout": "^2.6.3",
    "react-toastify": "^9.1.2",
    "stripe": "^12.1.1",
    "styled-components": "^5.3.9",
    "uuid": "^9.0.0",
    "yup": "^1.1.0"
```

Note that packages such as yup,uuid,formik, isotope, particles, dropzone etc were not actually used.