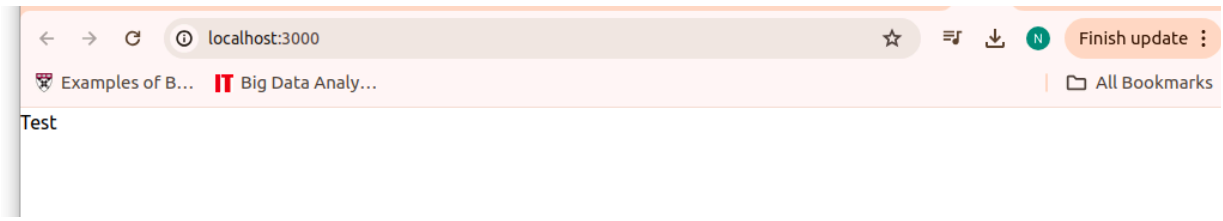
E-commerce - Front

npm create next-app .

Github repository: https://github.com/nthapa000/ecommerce-front



component/header.js

For better font we will use google fonts

Choosed Roboto:Normal

<style>

@import

url('https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,900&display=swap');

</style>

Also import the css classes

Global.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;


@import
url('https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300
```

```
;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,900&display=swap'
);

.roboto-thin {
 font-family: "Roboto", sans-serif;
 font-weight: 100;
 font-style: normal;
}

.roboto-light {
 font-family: "Roboto", sans-serif;
 font-weight: 300;
 font-style: normal;
}

.roboto-regular {
 font-family: "Roboto", sans-serif;
 font-weight: 400;
 font-style: normal;
}

.roboto-medium {
 font-family: "Roboto", sans-serif;
 font-weight: 500;
 font-style: normal;
}

.roboto-bold {
 font-family: "Roboto", sans-serif;
 font-weight: 700;
 font-style: normal;
}

.roboto-black {
 font-family: "Roboto", sans-serif;
 font-weight: 900;
 font-style: normal;
}

.roboto-thin-italic {
```

```
 font-family: "Roboto", sans-serif;
 font-weight: 100;
 font-style: italic;
}

.roboto-light-italic {
 font-family: "Roboto", sans-serif;
 font-weight: 300;
 font-style: italic;
}

.roboto-regular-italic {
 font-family: "Roboto", sans-serif;
 font-weight: 400;
 font-style: italic;
}

.roboto-medium-italic {
 font-family: "Roboto", sans-serif;
 font-weight: 500;
 font-style: italic;
}

.roboto-bold-italic {
 font-family: "Roboto", sans-serif;
 font-weight: 700;
 font-style: italic;
}

.roboto-black-italic {
 font-family: "Roboto", sans-serif;
 font-weight: 900;
 font-style: italic;
}


body{
 @apply p-0 m-0
 }
```

Components/Header.js

```
import Link from "next/link";
import Center from "./Center";

export default function Header() {
 return (
   <header className="bg-[#222] ">
     <Center>
       <div className="headerDiv">
         <Link className="text-white " href={"/"}>
           Ecommerce
         </Link>
         <nav className="headerNav">
           <Link className="navLink" href={"/"}>Home</Link>
           <Link className="navLink" href={"/products"}>All
products</Link>
           <Link className="navLink"
href={"/categories"}>Categories</Link>
           <Link className="navLink" href={"/account"}>Account</Link>
           <Link className="navLink" href={"/cart"}>Cart (0)</Link>
         </nav>
       </div>
     </Center>
   </header>
 );
}
```

Global.css

```
/* Header.js */
.decorationNone{
 text-decoration: none;
}

.center{
 max-width: 800px;
 margin: 0 auto;
 padding: 0 20px;
```

```
}

.headerDiv{
 display: flex;
 justify-content: space-between;
 padding: 20px 0;
}

.navLink{
   color: #aaa;
}

.headerNav{
 display: flex;
 gap: 15px;
}
```

Make sure you import this component in the Homepage

```
import Featured from "@/components/Featured";
import Header from "@/components/Header";

export default function Homepage(){
 return (
   <div className="roboto-regular">
     <Header/>
     <Featured/>
   </div>
 )
}
```

Now Lets work on the Featured product component , Here we will feature a single product

```
import Center from "./Center";

export default function Featured() {
 return (
   <div className="FeaturedDiv">
```

```
     <Center>
       <div className="FeaturedWrapper">
         <div className="FeaturedWrapperDiv">
           <div>
             <h1 className="FeaturedTitle">Pro Anywhere</h1>
             <p className="FeaturedDisc">
               Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Nam
               vitae dui nisi. Fusce suscipit purus sed mauris
consectetur, ac
               posuere turpis facilisis. Vestibulum molestie dui nec est
               volutpat, vitae venenatis.
             </p>
             <div className="FeaturedButtonWrapper pt-2">
               <button className="primaryBtn  ButtonWhite
ButtonWhiteOutline">
                 Read More
               </button>
               <button className="primaryBtn FeaturedButtonPrimary">
                 <svg
                   xmlns="http://www.w3.org/2000/svg"
                   viewBox="0 0 24 24"
                   fill="currentColor"
                   className="w-5 h-5 mr-1 "
                 >
                   <path d="M2.25 2.25a.75.75 0 0 0 0 1.5h1.386c.17 0
.318.114.362.278l2.558 9.592a3.752 3.752 0 0 0-2.806 3.63c0
.414.336.75.75.75h15.75a.75.75 0 0 0 0-1.5H5.378A2.25 2.25 0 0 1 7.5
15h11.218a.75.75 0 0 0 .674-.421 60.358 60.358 0 0 0 2.96-7.228.75.75 0 0
0-.525-.965A60.864 60.864 0 0 0 5.68 4.509l-.232-.867A1.875 1.875 0 0 0
3.636 2.25H2.25ZM3.75 20.25a1.5 1.5 0 1 1 3 0 1.5 1.5 0 0 1-3 0ZM16.5
20.25a1.5 1.5 0 1 1 3 0 1.5 1.5 0 0 1-3 0Z" />
                 </svg>
                 Add to cart
               </button>
             </div>
           </div>
         </div>
         <div className="FeaturedWrapperDiv">
           {/* Here will be image */}
```

```
            <img
              className="FeaturedImg"

src="https://ecommerce-1-prototype.s3.amazonaws.com/1720810262701.jpeg"
            ></img>
          </div>
        </div>
      </Center>
    </div>
 );
}
```

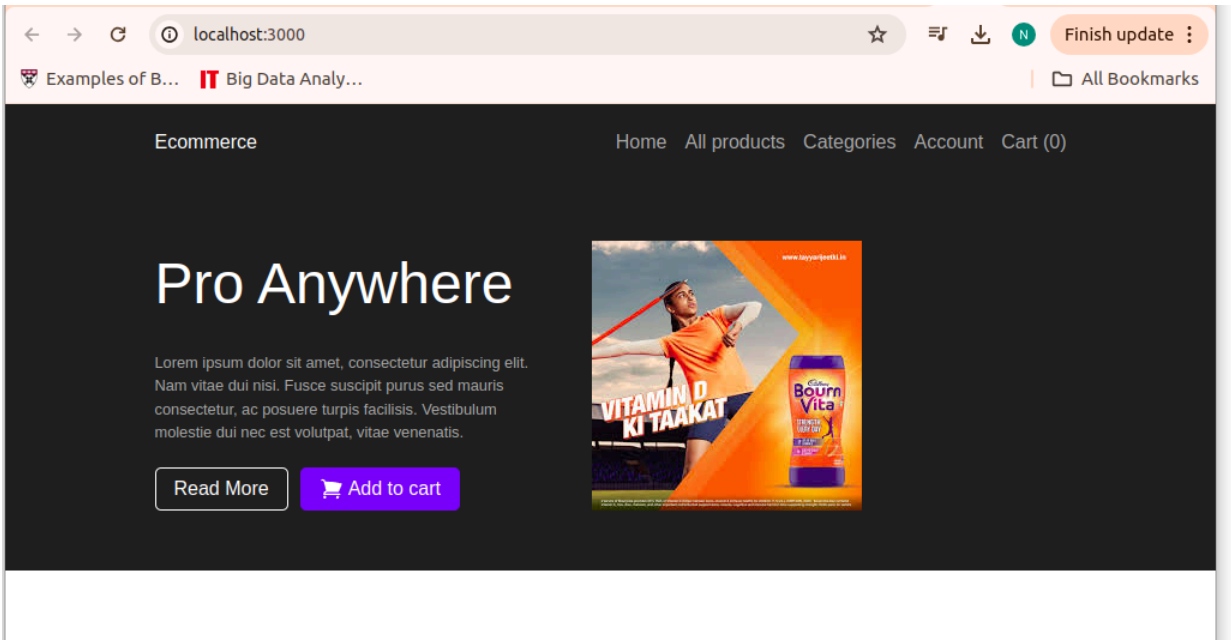
Global.css

```
/* Featured Product */

.FeaturedDiv{
 background-color: #222;
 color: #fff;
 padding: 50px 0;
}

.FeaturedTitle{
 margin: 0;
 font-weight: normal;
 font-size:3rem;
 padding-bottom: 20px;
}

.FeaturedDisc{
 color: #aaa;
 font-size: 0.8rem;
}

.FeaturedWrapper{
 display: grid;
 grid-template-columns: .9fr 1.1fr;
 gap: 40px;
}
```

```
.FeaturedImg{
 max-width: 100%;
}

.FeaturedWrapperDiv{
 display: flex;
 align-items: center;
}

.primaryBtn{
 border: 0;
 padding: 5px 15px;
 border-radius: 5px;
 cursor: pointer;
 display: inline-flex;
 align-items: center;
}

.ButtonWhite{
 background-color: #fff;
 color: #000;
}

.ButtonWhiteOutline{
 background-color:transparent;
 color: #fff;
 border: 1px solid #fff;
}

.FeaturedButton{
 font-size: 1.2rem;
 padding: 10px 20px;
}

.FeaturedButtonPrimary{
 background-color: #5542F6;
 border: 1px solid #5542F6;
 color: #fff;
}
```

```
.FeaturedButtonWrapper{
 display: flex;
 gap: 10px;
 margin-top: 12px;
}
```

It looks like this



Now lets get these information from db currently it is hardcoded and then pass it to Featured component

Repository :

So first lets connect to Database

lib/mongoose.js

Copy the same code from the e-commerce admin

```
import mongoose from "mongoose";

export function mongooseConnect(){
   const uri = process.env.MONGODB_URI;
   if(mongoose.connection.readyState === 1){
```

```
        return mongoose.connection.asPromise();
    } else{
        const uri = process.env.MONGODB_URI;
        return mongoose.connect(uri);
    }
}
```

npm install mongoose

Now we can connect in API but before that we need models

Also copy the Product model from admin

We can maintain a mono repo where we have models for both admin dashboard and front part both in a common file

model/Product.js

```
const {model, Schema, models, default: mongoose } = require("mongoose");

const ProductSchema = new Schema({
   title: {type:String, required:true},
   description: String,
   price: {type:Number, required:true},
   images:[{type:String}],
   category:{type:mongoose.Types.ObjectId, ref:'Category'},
   properties:{type:Object}
});

export const Product =models.Product || model('Product',ProductSchema);
```

.env

```
MONGODB_URI="mongodb+srv://ecommerce:eRBOLJoWlcnKGOOa@cluster0.hi7b9ah.mon
godb.net/?retryWrites=true&w=majority&appName=Cluster0"
```

Now we will use this in getServerSideProps function in index.js

```
import Featured from "@/components/Featured";
import Header from "@/components/Header";
import { mongooseConnect } from "@/lib/mongoose";
import { Product } from "@/model/Product";

export default function Homepage({product}) {
 console.log(product);
 return (
   <div className="roboto-regular">
     <Header />
     <Featured />
   </div>
 );
}

export async function getServerSideProps() {
 const FeaturedProductId = "668e4d317cc6e41ed18d8c4a";
 await mongooseConnect();
 const product = await Product.findById(FeaturedProductId);
 return {
   props: { product:JSON.stringify(product) }
 };
}
```

But this way we can see that our product is a string

```
[HMR] connected                                                    websocket.js:46
{"_id":"668e4d317cc6e41ed18d8c4a","title":"Bournvita","description":"Shak  index.js:7
ti hi Bhakti hai","price":10,"__v":0,"category":"6690db0ff5b89a113adafb11","images":
["https://ecommerce-1-prototype.s3.amazonaws.com/1720810262701.jpeg"],"properties":
{"Palm Oil ":"No"}}
```

We will parse it back

```
   props: { product:JSON.parse(JSON.stringify(product)) }
```

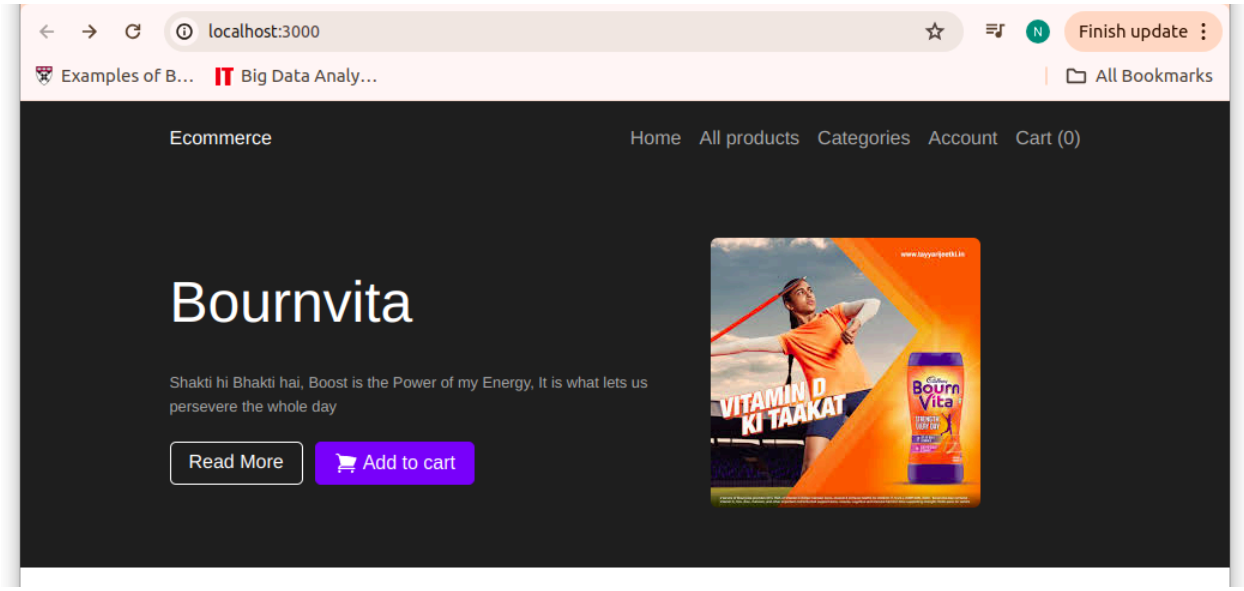
Now it is an object

```
index.js:7
▶ {_id: '668e4d317cc6e41ed18d8c4a', title: 'Bournvita', description: 'Shakti hi Bhakti hai', price: 10, __v: 0, …}
index.js:7
▼ {_id: '668e4d317cc6e41ed18d8c4a', title: 'Bournvita', description: 'Shakti hi Bhakti hai', price: 10, __v: 0, …} i
    category: "6690db0ff5b89a113adafb11"
    description: "Shakti hi Bhakti hai"
  ▶ images: ['https://ecommerce-1-prototype.s3.amazonaws.com/1720810262701.jpeg']
    price: 10
  ▼ properties:
      "Palm Oil ": "No"
    ▶ [[Prototype]]: Object
    title: "Bournvita"
    __v: 0
    _id: "668e4d317cc6e41ed18d8c4a"
  ▶ [[Prototype]]: Object
```

Now lets make changes on Featured Component

```
import Link from "next/link";
import Center from "./Center";

export default function Featured({product}) {
 return (
   <div className="FeaturedDiv">
     <Center>
       <div className="FeaturedWrapper">
         <div className="FeaturedWrapperDiv">
           <div>
             <h1 className="FeaturedTitle">{product.title}</h1>
             <p className="FeaturedDisc">
               {product.description}
             </p>
             <div className="FeaturedButtonWrapper pt-2">
               <Link href={'/products/'+product._id} className="primaryBtn
ButtonWhite ButtonWhiteOutline">
                 Read More
               </Link>
               <button className="primaryBtn FeaturedButtonPrimary">
                ……..
                 Add to cart
```

```
                </button>
              </div>
            </div>
          </div>
          <div className="FeaturedWrapperDiv">
            {/* Here will be image */}
            <img
              className="FeaturedImg rounded-md"
              src={product.images[0]}
            ></img>
          </div>
        </div>
      </Center>
    </div>
  );
}
```



https://github.com/nthapa000/ecommerce-front/commits/main/

Recent Product

Component/NewProduct.js

Change the Product.js model at frontend and admin page to and update all the product in admin

```
const {model, Schema, models, default: mongoose } = require("mongoose");

const ProductSchema = new Schema({
   title: {type:String, required:true},
   description: String,
   price: {type:Number, required:true},
   images:[{type:String}],
   category:{type:mongoose.Types.ObjectId, ref:'Category'},
   properties:{type:Object}
},{
   timestamps:true,
});

export const Product =models.Product || model('Product',ProductSchema);
```

index.js

```
import Featured from "@/components/Featured";
import Header from "@/components/Header";
import NewProducts from "@/components/NewProduct";
import { mongooseConnect } from "@/lib/mongoose";
import { Product } from "@/model/Product";

export default function Homepage({featuredProduct,newProducts}) {
 // console.log(product);
 return (
   <div className="roboto-regular">
     <Header />
     <Featured  product={featuredProduct}/>
     <NewProducts products={newProducts}/>
   </div>
 );
}

export async function getServerSideProps() {
 const FeaturedProductId = "668e4d317cc6e41ed18d8c4a";
```

```
 await mongooseConnect();
 const featuredProduct = await Product.findById(FeaturedProductId);
 // -1 descending, last updated product is basically the latest product
 const newProducts = await
Product.find({},null,{sort:{'_id':-1},limit:10});
 return {
   props: {
     featuredProduct:JSON.parse(JSON.stringify(featuredProduct)),
     newProducts:JSON.parse(JSON.stringify(newProducts))
   }
 };
}
```

We can see that all the product are arranged in descending order of their updated

```
▶ Array(5)                                                    index.js:8
                                                              index.js:8
▼ Array(5) i
  ▶ 0: {_id: '669404d4ca915962ff59cf88', title: 'Power Bank', description: '10000 mAh
  ▶ 1: {_id: '6691727a54f9076bd42484b5', title: 'Motorolla Edge 50 pro', description:
  ▶ 2: {_id: '668e67255e49a86eade474c1', title: 'Choco Cookies', description: '100 Per
  ▶ 3: {_id: '668e54415e49a86eade4742e', title: 'Cricket ball', description: 'Red in C
  ▶ 4: {_id: '668e4d317cc6e41ed18d8c4a', title: 'Bournvita', description: 'Shakti hi B
    length: 5
  ▶ [[Prototype]]: Array(0)
>
```

Components/Product.js

```
export default function NewProducts({products}){
   return(
       <div className="NewProductGrid">
           {products?.length>0 && products.map(product => (
               <div>{product.title}</div>
           ))}
       </div>
```

```
    )
}
```

Peanut Butter	Boat Immortal 1300	Power Bank
Motorolla Edge 50 pro	Choco Cookies	Cricket ball
Bournvita		

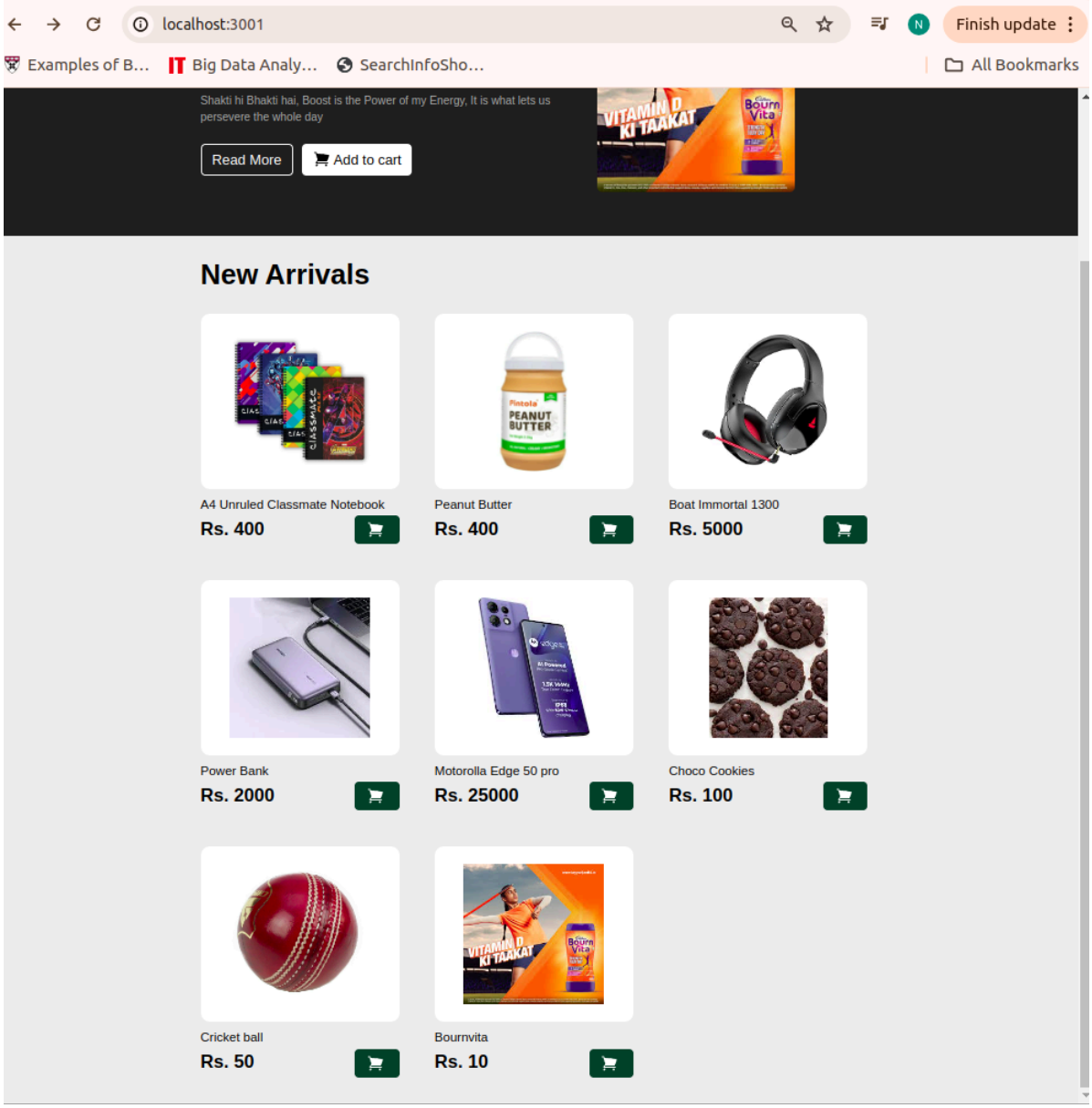
Now we will center this grid

```
import Center from "./Center";

export default function NewProducts({ products }) {
 return (
   <Center>
     <div className="NewProductGrid">
       {products?.length > 0 &&
         products.map((product) => <div>{product.title}</div>)}
     </div>
   </Center>
 );
}
```

Now we will make product box component

```
import Center from "./Center";
import ProductBox from "./ProductBox";

export default function NewProducts({ products }) {
 return (
   <Center>
     <h2 className="NewProductTitle">New Arrivals</h2>
     <div className="NewProductGrid">
       {products?.length > 0 &&
         products.map((product) => (
           <ProductBox {...product} />
         ))}
```

```
      </div>
    </Center>
  );
}
```



Product Box Component

```
import Link from "next/link"
```

```
export default function ProductBox({ _id, title, description, price,
images }) {
 const url = '/product/'+_id;
 return (
   <div className="ProductBoxWrapper">
     <Link href={url} className="ProductBox">
       <div>
         <img className="ProductBoxImage" src={images[0]} alt="" />
       </div>
     </Link>
     <div className="ProductInfoBox">
       <Link href={url} className="ProductBoxTitle">{title}</Link>
       <div className="ProductPriceRow">
         <div className="ProductBoxPrice">Rs. {price}</div>
         <div>
           <button className="primaryBtn FeaturedButtonPrimary ">
             <svg
               xmlns="http://www.w3.org/2000/svg"
               viewBox="0 0 24 24"
               fill="currentColor"
               className="w-5 h-5 mr-1 "
             >
               <path d="M2.25 2.25a.75.75 0 0 0 0 1.5h1.386c.17 0
.318.114.362.278l2.558 9.592a3.752 3.752 0 0 0-2.806 3.63c0
.414.336.75.75.75h15.75a.75.75 0 0 0 0-1.5H5.378A2.25 2.25 0 0 1 7.5
15h11.218a.75.75 0 0 0 .674-.421 60.358 60.358 0 0 0 2.96-7.228.75.75 0 0
0-.525-.965A60.864 60.864 0 0 0 5.68 4.509l-.232-.867A1.875 1.875 0 0 0
3.636 2.25H2.25ZM3.75 20.25a1.5 1.5 0 1 1 3 0 1.5 1.5 0 0 1-3 0ZM16.5
20.25a1.5 1.5 0 1 1 3 0 1.5 1.5 0 0 1-3 0Z" />
             </svg>
           </button>
         </div>
       </div>
     </div>
   </div>
 );
}
```

Global.css

```
/* New Product */

.NewProductGrid{
 display: grid;
 grid-template-columns:1fr 1fr 1fr ;
 gap: 40px;
 padding-top: 30px;
}

.NewProductTitle{
 margin-top: 20px;
 font-size: 2rem;
 font-weight: 600;
 margin-bottom: -10px;
}
/* Product Box */

.ProductBox{
 background-color:#fff;
 padding: 20px;
 text-align: center;
 display: flex;
 /* height: 180px; */
 align-items: center;
 justify-content: center;
 border-radius: 10px;
}

.ProductBoxImage{
 max-width: 100%;
 max-height: 160px;
}

.ProductBoxWrapper{

}
```

```
.ProductBoxTitle{
   font-weight: normal;
   font-size: .9rem;
   margin: 0;
}

.ProductInfoBox{
 margin-top: 5px;
}

.ProductPriceRow{
 display: flex;
 align-items: center;
 justify-content: space-between;
 margin-top: 2px;
}

.ProductBoxPrice{
 font-size: 1.3rem;
 font-weight:600 ;
}

.ProductBoxButton{
 background-color: transparent;
 border: 1px solid #5542F6;
 color: #fff;
}
```

Github Repository at the moment;
https://github.com/nthapa000/Cantilever/tree/d3813f739407a783057cc1d09a9d5a24e39486c9/e-commerce/e-commerce-front

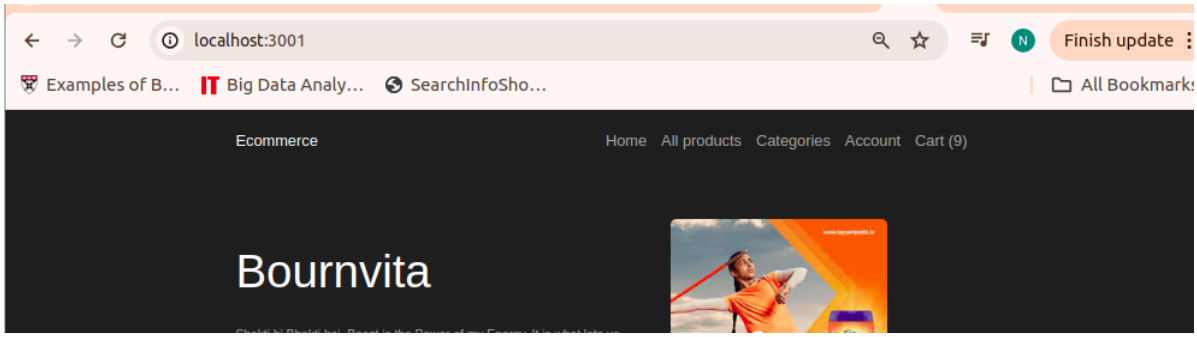
Now we will work on Feature Add to Cart

CartContext.js

```
import {  createContext, useState } from "react";

export const CartContext = createContext({});

export function CartContextProvider({children}){
   const [cartProducts,setCartProducts] = useState([])
   function addProduct(productId){
       setCartProducts(prev => [...prev,productId])
   }
   return(
       <CartContext.Provider
value={{cartProducts,setCartProducts,addProduct}}>
           {children}
       </CartContext.Provider>
   )
}
```

This will increase the count of no. of product in the Cart

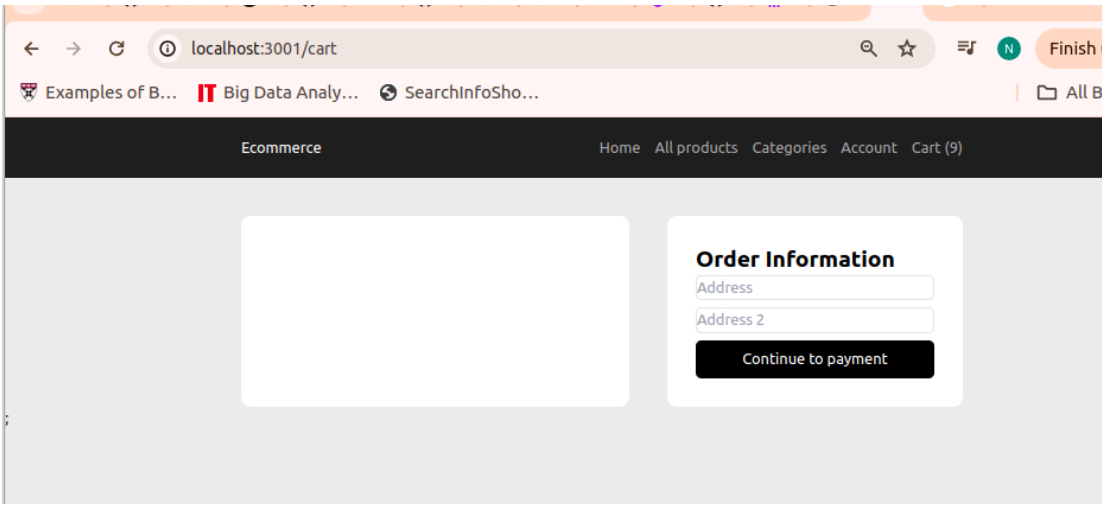
```
 const {addProduct} = useContext(CartContext)
 function addFeature(){
   addProduct(product._id);
 }
```



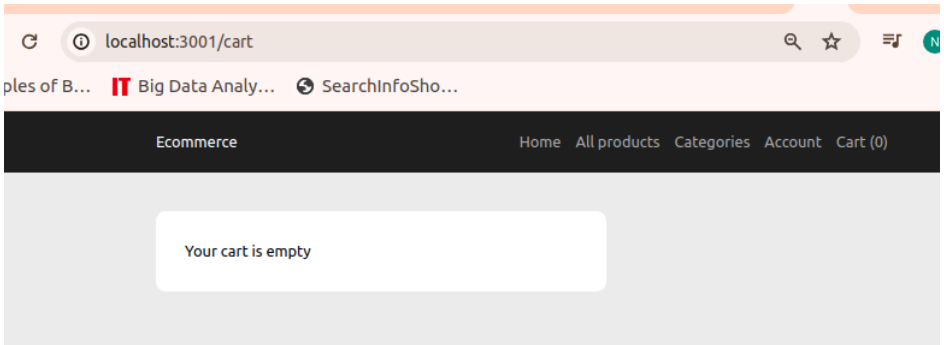
Now lets make the cart page

We will see continue to payment only if we have some products in the Cart

When there is some product in cart



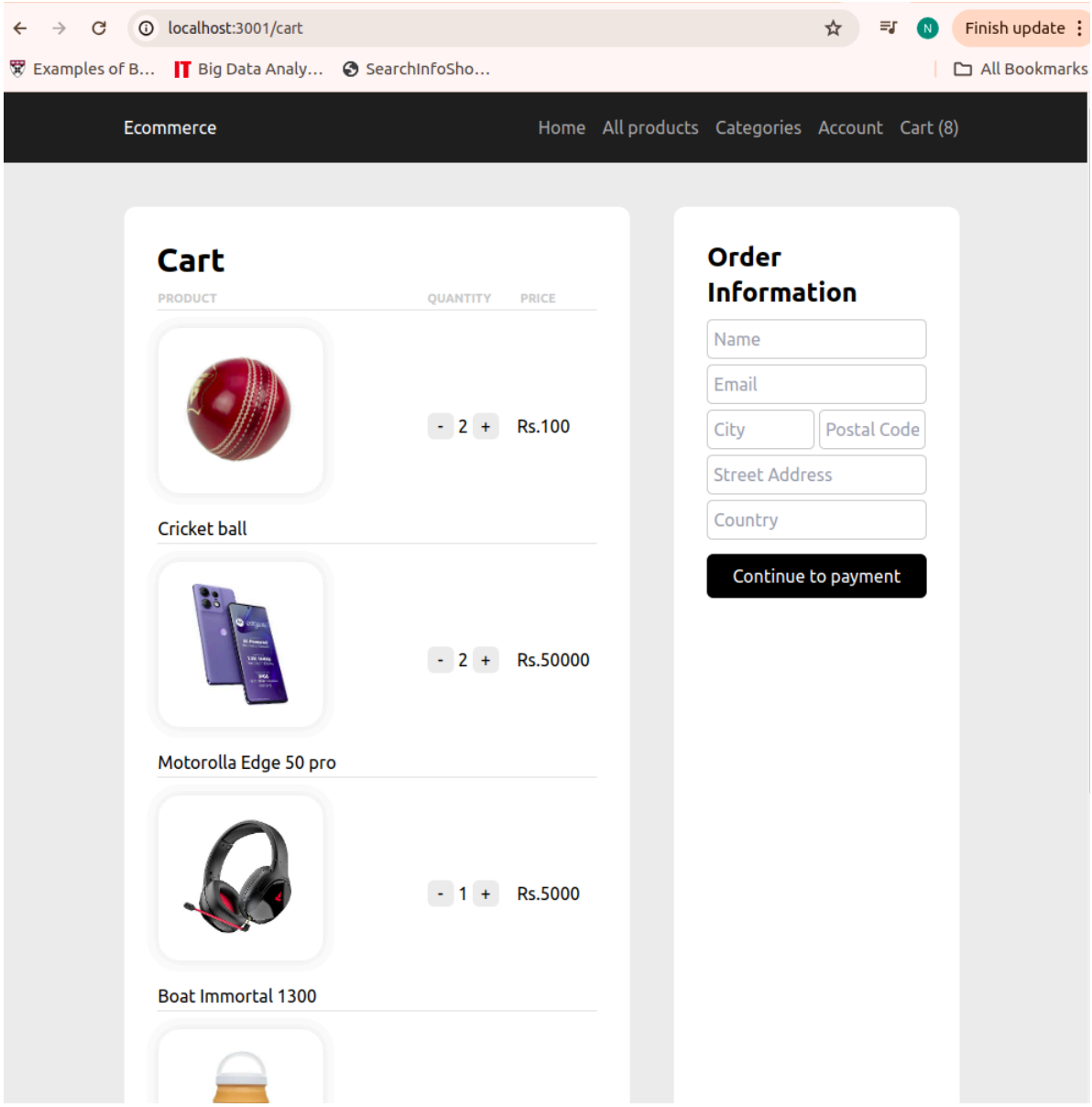
No product in cart



Make an api to fetch all the products from productId and install axios

```
import { mongooseConnect } from "@/lib/mongoose";
import { Product } from "@/model/Product";

export default async function handle(req,res){
   await mongooseConnect();
   const ids = req.body.ids;
   res.json(await Product.find({_id:ids}))
}
```

Final look of Cart



Card Component

```
import { CartContext } from "@/components/CardContext";
import Center from "@/components/Center";
import Header from "@/components/Header";
import axios from "axios";
import { useContext, useEffect, useState } from "react";

export default function CartPage() {
```

```
 const { cartProducts, addProduct, removeProduct } =
useContext(CartContext);
 const [products, setProducts] = useState();
 const [name, setName] = useState("");
 const [email, setEmail] = useState("");
 const [city, setCity] = useState("");
 const [postalCode, setPostalCode] = useState("");
 const [streetAddress, setStreetAddress] = useState("");
 const [country, setCountry] = useState("");

 useEffect(() => {
   if (cartProducts.length > 0) {
     axios.post("/api/cart", { ids: cartProducts }).then((response) => {
       setProducts(response.data);
     });
   }else{
      setProducts([])
   }
 }, [cartProducts]);

 function moreOfThisProduct(id) {
   addProduct(id);
 }

 function lessOfThisProduct(id) {
   removeProduct(id);
 }

 let total = 0;
 for (const productId of cartProducts) {
   const price = products?.find((p) => p._id === productId)?.price || 0;
   total += price;
 }
 return (
   <>
     <Header />
     {/* Displaying the products */}
     <Center>
       <div className="CartColumnsWrapper">
         <div className="CartBox">
```

```
            <h2 className="font-bold text-3xl mb-2">Cart</h2>
            {!cartProducts?.length && <div>Your cart is empty</div>}
            {/* converted cartProducts to boolean */}
            {products?.length > 0 && (
              <table className="CartTable">
                <thead>
                  <tr>
                    <th>Product</th>
                    <th className="pr-5">Quantity</th>
                    <th className="pl-1">Price</th>
                  </tr>
                </thead>
                <tbody>
                  {products.map((product) => (
                    <tr>
                      <td>
                        <div className="CartProductInfoCellImagesBox">
                          <img
                            className="CartProductInfoCellImages"
                            src={product.images[0]}
                            alt=""
                          />
                        </div>
                        {product.title}
                      </td>
                      <td>
                        <button
                          onClick={() => lessOfThisProduct(product._id)}
                          className="bg-[#eee] rounded-md w-6 mr-1"
                        >
                          -
                        </button>
                        {cartProducts.filter((id) => id ===
product._id).length}
                        <button
                          onClick={() => moreOfThisProduct(product._id)}
                          className="bg-[#eee] rounded-md w-6 ml-1"
                        >
                          +
                        </button>
```

```
                    </td>
                    <td>
                      Rs.
                      {cartProducts.filter((id) => id === product._id)
                        .length * product.price}
                    </td>
                  </tr>
                ))}
                <tr>
                  <td></td>
                  <td></td>
                  <td>Rs. {total}</td>
                </tr>
              </tbody>
            </table>
          )}
        </div>
        {!!cartProducts?.length && (
          <div className="CartBox">
            <h2 className="font-bold text-2xl mb-2">Order
Information</h2>
            <input
              className="border rounded-md mb-2 CartInput"
              type="text"
              placeholder="Name"
              value={name}
              onChange={(ev) => setName(ev.target.value)}
            />
            <input
              className="border rounded-md CartInput"
              type="text"
              placeholder="Email"
              value={email}
              onChange={(ev) => setEmail(ev.target.value)}
            />
            <div className="flex gap-1">
              <input
                className="border rounded-md CartInput"
                type="text"
                placeholder="City"
```

```
                value={city}
                onChange={(ev) => setCity(ev.target.value)}
              />
              <input
                type="text"
                placeholder="Postal Code"
                className="border rounded-md CartInput"
                value={postalCode}
                onChange={(ev) => setPostalCode(ev.target.value)}
              />
            </div>
            <input
              type="text"
              placeholder="Street Address"
              className="border rounded-md CartInput"
              value={streetAddress}
              onChange={(ev) => setStreetAddress(ev.target.value)}
            />
            <input
              type="text"
              placeholder="Country"
              className="border rounded-md CartInput"
              value={country}
              onChange={(ev) => setCountry(ev.target.value)}
            />
            <button className="rounded-md p-2  bg-black text-white  mt-2
w-[100%] block text">
              Continue to payment
            </button>
          </div>
        )}
      </div>
    </Center>
  </>
 );
}
```

Repository:
https://github.com/nthapa000/Cantilever/tree/41db621b613e93a50aca357f36c24b89705719d5

We will make an api for checkout

```
import { mongooseConnect } from "@/lib/mongoose";
import { Product } from "@/model/Product";

export default async function handler(req,res){
   if (req.method !=='POST'){
       res.json('Should be a POST request');
       return;
   }
   const {name,email,city,postalCode,streetAddress,country,products} =
req.body;

   await mongooseConnect()
   const productsIds = products.split(',');
   const uniqueIds = [...new Set(productsIds)];
   const productsInfos = await Product.find({_id:uniqueIds});

   let line_items = [];
   for(const productId of uniqueIds){
       const productInfo = productsInfos.find(p => p._id.toString() ===
productId);
       const quantity = productsIds.filter(id => id === productId)?.length
|| 0
       if(quantity>0 && productInfo){
           line_items.push({
               quantity,
               price_data: {
                   currency:'Rs',
                   product_data:{name:productInfo.name},
                   unit_amount: quantity*productInfo.price
               }
           })
       }
```

```
    }
    res.json({line_items})
}
```


```
{
  "line_items": [
    {
      "quantity": 2,
      "price_data": {
        "currency": "Rs",
        "product_data": {

        },
        "unit_amount": 800
      }
    },
    {
      "quantity": 1,
      "price_data": {
        "currency": "Rs",
        "product_data": {

        },
        "unit_amount": 400
      }
    },
    {
      "quantity": 2,
      "price_data": {
        "currency": "Rs",
        "product_data": {

        },
        "unit_amount": 50000
      }
    },
    {
      "quantity": 1,
      "price_data": {
        "currency": "Rs",
        "product_data": {

        },
        "unit_amount": 5000
      }
    },
    {
      "quantity": 2,
      "price_data": {
        "currency": "Rs",
        "product_data": {

        },
        "unit_amount": 100
      }
    }
  ]
```


We want to save it as order inside our Database:

models/Order.js

```
const { Schema, model, models } = require("mongoose");

const OrderSchema = new Schema({
   line_items:Object,
   name:String,
   email:String,
   city:String,
   postalCode:String,
   streetAddress:String,
   country:String,
   paid:Boolean

})

export const Order = models?.Order || model('Order',OrderSchema);
```

Now to implement Stripe we will make a test account in Stripe