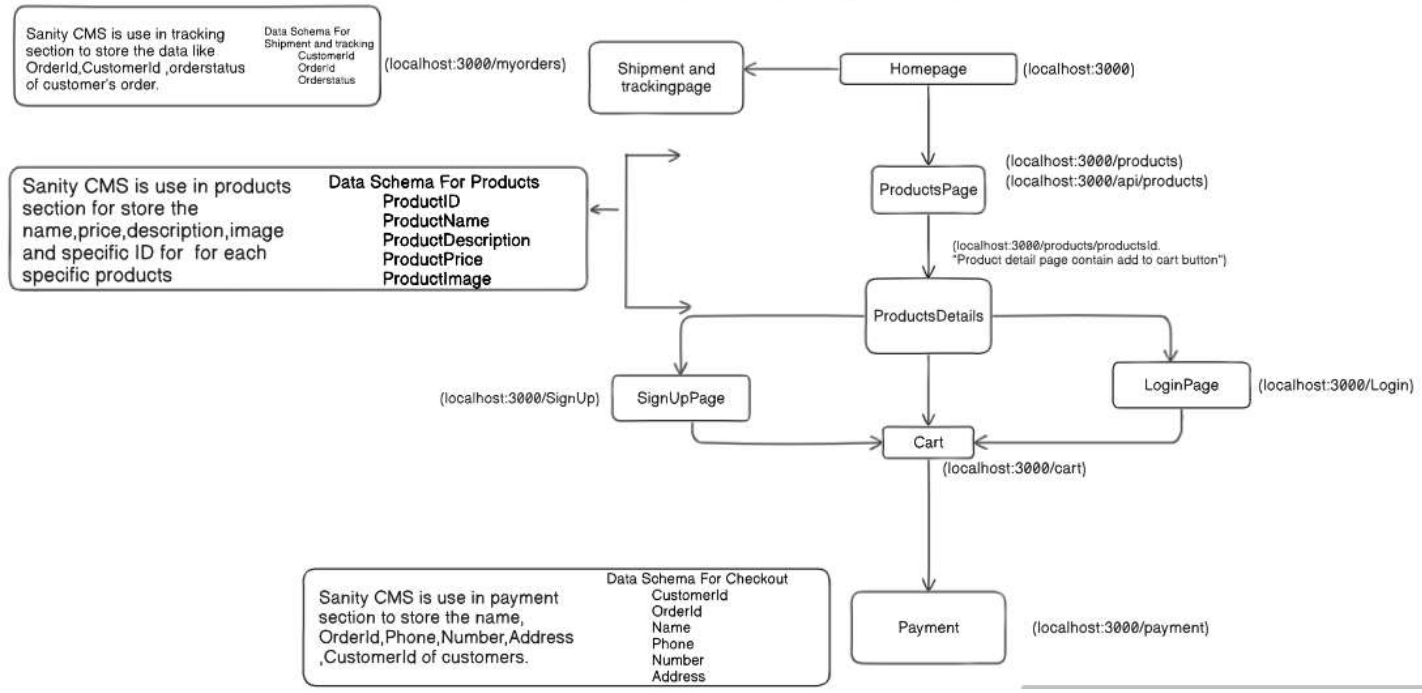
WORKFLOW


Sanity CMS is use in tracking section to store the data like OrderId,CustomerId ,orderstatus of customer's order.
Data Schema For Shipment and tracking
CustomerId
OrderId
Orderstatus
(localhost:3000/myorders)
Shipment and trackingpage
Homepage
(localhost:3000)
Sanity CMS is use in products section for store the name,price,description,image and specific ID for for each specific products
Data Schema For Products
ProductID
ProductName
ProductDescription
ProductPrice
ProductImage
ProductsPage
(localhost:3000/products)
(localhost:3000/api/products)
(localhost:3000/products/productsId.
"Product detail page contain add to cart button")
ProductsDetails
(localhost:3000/SignUp)
SignUpPage
LoginPage
(localhost:3000/Login)
Cart
(localhost:3000/cart)
Sanity CMS is use in payment section to store the name, OrderId,Phone,Number,Address ,CustomerId of customers.
Data Schema For Checkout
CustomerId
OrderId
Name
Phone
Number
Address
Payment
(localhost:3000/payment)

System Architecture for E-Commerce Platform

1. Frontend (Next.js + Tailwind CSS + ShadCN UI):

- **Next.js:** The frontend is built with Next.js for SSR (Server-Side Rendering), API routes, and optimized performance. The pages will include Home, About, products, products details , shopping cart, checkout, shipment and tracking etc.
- **Tailwind CSS:** For styling, you'll use Tailwind CSS, a utility-first CSS framework, to ensure a responsive and customizable design. This helps speed up the development process and keeps your UI consistent.
- **ShadCN UI**: This UI component library will be used for reusable and customizable UI elements, such as buttons, modals, forms, and cards. It's built on top of Tailwind CSS, so the two are compatible.

Frontend Flow:

- The user interacts with the UI for browsing, selecting products, adding them to the cart, and checking out.
- Dynamic product data like prices, descriptions, and availability will be fetched from Sanity.

2. Backend (Sanity CMS):

- **Sanity:** Your backend CMS (Content Management System) will be Sanity. It will store your product data (titles, images, descriptions, prices), manage categories, and store other content like reviews or blogs.
- It is used as a database for store customers' personal information like name, email, address, phone, etc.
- It is also useful in the delivery section where we can store information of where we have to delivered products
- Sanity API: You'll fetch the product data from Sanity in your Next.js app using the Sanity API. This will allow you to dynamically update the frontend content without needing to redeploy.

3. Payment (Stripe):

- **Stripe:** Stripe will handle all payment processing. I can integrate Stripe's payment gateway for secure transactions.
- For checkout, Stripe will be used to securely handle card information and create payment intents for the transactions.

- It'll use Next.js API routes to handle server-side logic, such as creating a payment session and confirming payment.

Payment Flow:

- User completes checkout and enters payment details.
- Your app communicates with the Stripe API to process the payment.
- After successful payment, Stripe will notify your system (via webhooks) to confirm the transaction.

4. Shipping & Tracking (ShipEngine):

- **ShipEngine:** ShipEngine will handle the shipment and tracking functionalities. You'll integrate it into your backend to calculate shipping costs, create shipping labels, and provide real-time tracking updates.
- You can fetch live shipping rates based on product weights, sizes, and destination.
- After an order is placed and payment is successful, you'll use ShipEngine to generate shipping labels and manage tracking info.

Shipping Flow:

- Once the user places an order and Stripe confirms payment, you create a shipping request via the ShipEngine API.
- ShipEngine will provide tracking information that can be sent to the user.

5. Deployment (Vercel):

- Vercel: Entire platform will be deployed on Vercel. Vercel is optimized for Next.js and provides automatic scaling, fast deployment, and easy integration with your GitHub repository.
- You'll set up the necessary environment variables for your Sanity, Stripe, and ShipEngine integrations.
- Vercel's serverless functions (API routes in Next.js) will handle backend tasks such as payment processing, shipping requests, and data fetching from Sanity.

API ENDPOINTS

1) EndPoint: api/products

Method: GET

Description : Retrieves a list of available products.

sample response: [

{

"ProductId":1,

"ProductName":" The Golden Glow",

"ProductPrice":"200,

"Productdescription":" The Golden Glow",

"ProductImage":" The Golden Glow.png",

}

]

2) EndPoint: api/products/:productid

Method: GET

Description :Get a specific product according to specific ID

sample response: [

{

"ProductId":1,

"ProductName":" The Golden Glow",

"ProductPrice":"200,

"Productdescription":" The Golden Glow",

"ProductImage":" The Golden Glow.png",

}

]

3) EndPoint: api/signup

Method : POST

Payload: {

"name": "Test User",

"username": "testuser",

"email": "testuser@example.com",

"password": "securepassword123"

}

response : {

"userId": 1,

"token": "faketoken",

"username": "testuser",

"email": testuser@example.com

}

4) Endpoint: api/login

Method: POST

Payload: {

"username": "testuser",

"password": "securepassword123"

}

Response: {

"userId": 1,

"token": "faketoken",

"username": "testuser",

"email": "testuser@example.com"

}

5) Endpoint: api/order

Method : POST

Header: {

"Authorization": "Bearer <Token>"

}

Payload: {

"userId": 1,

"products": [

{

```
    "productId": 1,
    "quantity": 2
   },
   {
    "productId": 2,
    "quantity": 1
   }
  ],
  "totalPrice": 1299.97,
  "address": "123 Main Street, City, Country",
  "status": "Pending"
}
Response: {
  "id": 101,
  "status": "Pending",
  "message": "Order placed successfully"
}
```

6) Endpoint:api/myOrders

Method :GET

Header {

Authorization: Bearer <Token>

}

```
response : [
 {
  "id": 101,
  "userId": 1,
  "products": [
   {
    "productId": 1,
    "quantity": 2,
    "name": "Modern Sofa",
    "price": 499.99
   },
   {
    "productId": 2,
    "quantity": 1,
    "name": "Wooden Dining Table",
    "price": 799.99
   }
  ],
  "totalPrice": 1299.97,
  "address": "123 Main Street, City, Country",
  "status": "Pending",
  "placedAt": "2025-01-17T10:00:00Z"
 }
```

]

7) Endpoint: api/myOrders/:orderId

Method: GET

Header {

Authorization: Bearer <Token>

}

Response: {

"id": 101,

"userId": 1,

"products": [

{

"productId": 1,

"quantity": 2,

"name": "Modern Sofa",

"price": 499.99

},

{

"productId": 2,

"quantity": 1,

"name": "Wooden Dining Table",

"price": 799.99

```
    }
  ],
  "totalPrice": 1299.97,
  "address": "123 Main Street, City, Country",
  "status": "Pending",
  "placedAt": "2025-01-17T10:00:00Z",
  "updatedAt": "2025-01-17T11:00:00Z"
}
```