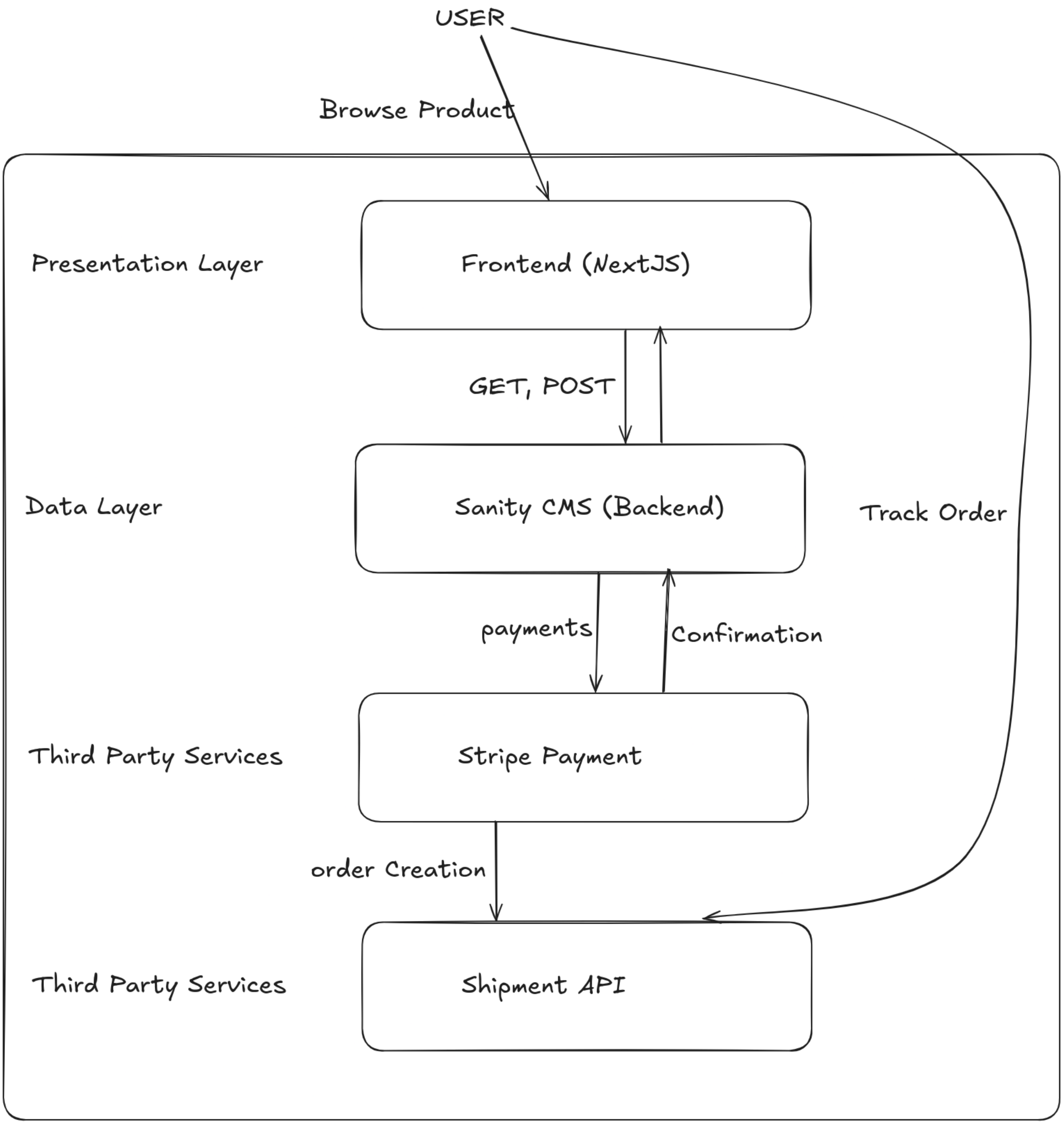
Hackathon MarketPlace Day 2: Planning the Technical Solution

Day 2 Activities: Transitioning to Technical Planning

Step 1: Define Technical Requirements

- **Frontend Requirements**
  - User Friendly interface for browsing, adding to cart and checkout.
  - NextJS for frontend
  - Responsiveness across all devices.
  - Performance should be better by loading each page within a second.
- **Backend Requirements (Sanity CMS)**
  - Management of inventory, users, orders etc using backend of Sanity.
  - Sanity is acting as backend and database also.
- **Third-Party APIs:**
  - **Shipment API** for tracking order.
  - **Stripe** as a payment service provider.

Step2: Design System Architecture

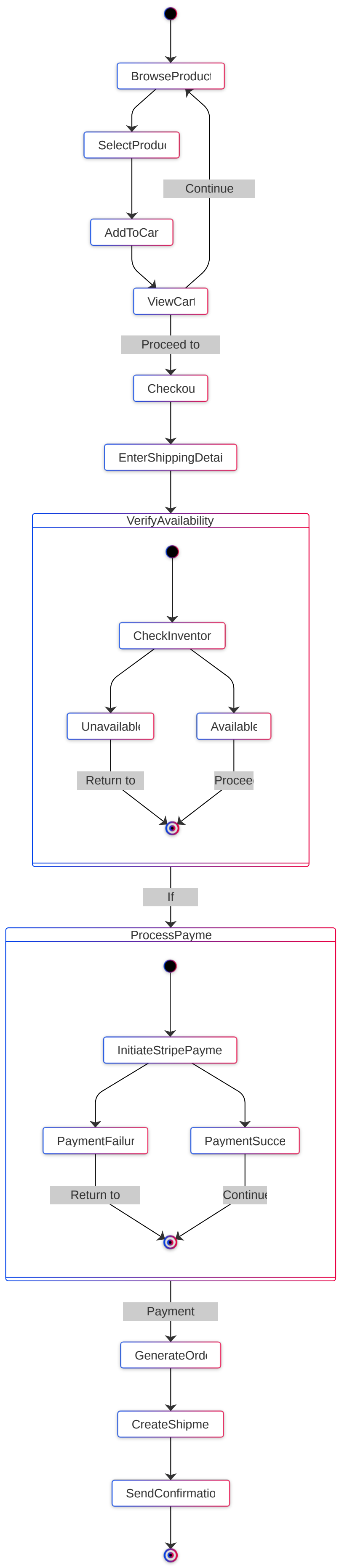


Key WorkFlows:

1. **Product Purchase:**
   - User browses and selects a product.
   - User adds items to the cart.
   - User proceeds to checkout and enters shipping details.
   - Backend verifies product availability.
   - Payment is processed using Stripe.

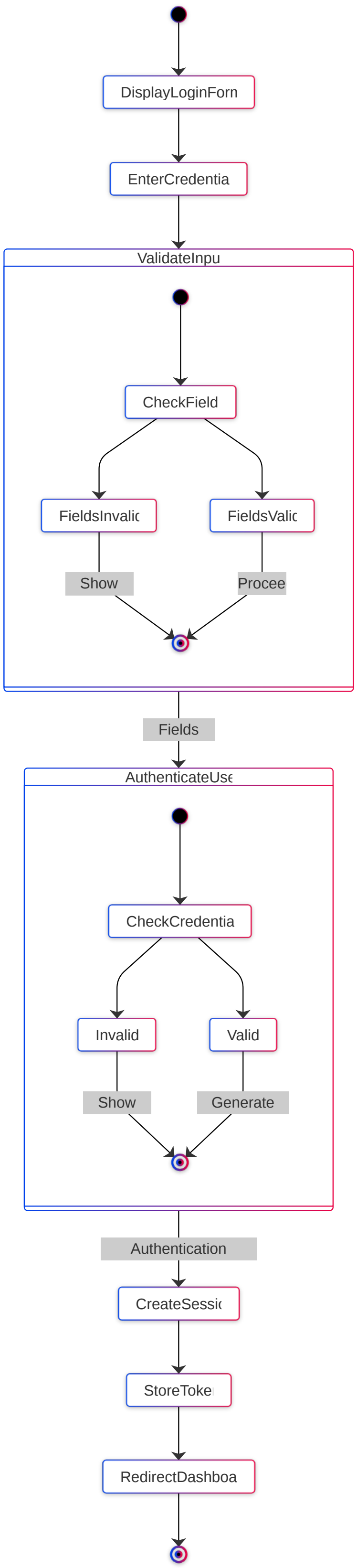
- Order is confirmed, and shipment details are generated.
- User receives an order confirmation and tracking information.

WorkFlow/Activity Diagram


BrowseProduct
SelectProdu
Continue
AddToCar
ViewCart
Proceed to
Checkou
EnterShippingDetai
VerifyAvailability
CheckInventor
Unavailable
Available
Return to
Procee
If
ProcessPayme
InitiateStripePayme
PaymentFailur
PaymentSucce
Return to
Continue
Payment
GenerateOrd
CreateShipme
SendConfirmatio

1. **User/Admin Login:**
    - User enters credentials.
    - Frontend sends credentials to backend for verification.
    - Backend authenticates user.
    - A session or token is created.
    - User gains access to the system.

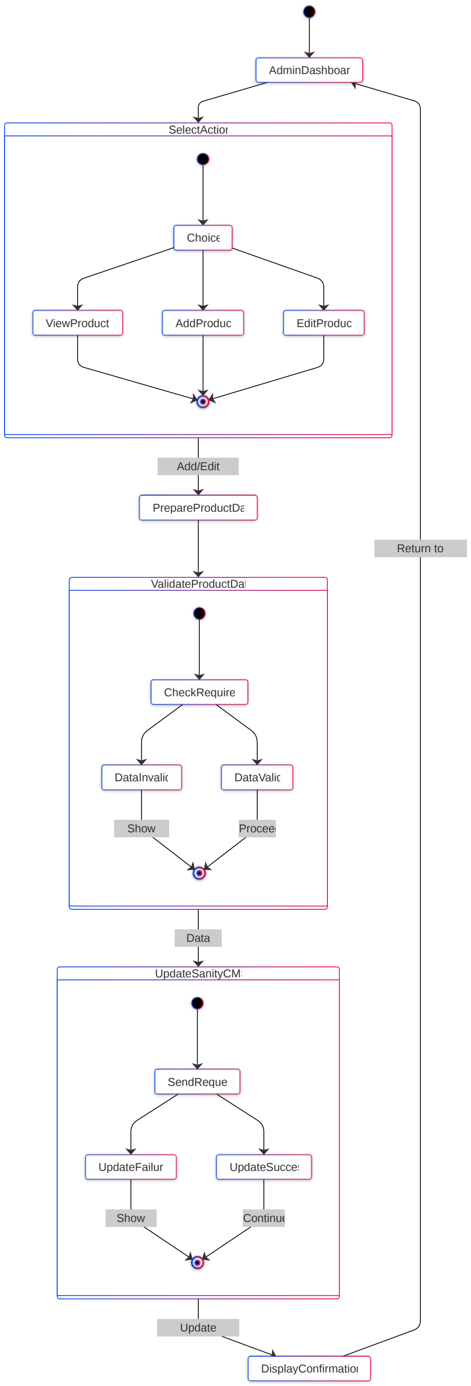
WorkFlow/Activity Diagram



1. **Admin Product Management:**
   - Admin logs into the system.

- Admin views or adds products via a management interface.
- Changes are sent to Sanity CMS for update or creation.
- Sanity CMS confirms data update.
- Admin views confirmation of the action.

WorkFlow/Activity Diagram:



Step 3: Plan API Requirements

- **Endpoint Name:** /admin
    - **Method:** GET
    - **Description:** Fetch input for Name and Password.
    - **Response:** Admin details (ID, Name, Password, Role)
- **Endpoint Name:** /products
    - **Method:** GET
    - **Description:** Fetch All available products.
    - **Response:** Product Details (ID, Name, Category, Price, Discount, stock, Description, Images, Color, Size)
- **Endpoint Name:** /my-orders
    - **Method:** GET
    - **Description:** Get all the orders of user.
    - **Response:** Order Details (Status, Tracking ID)
- **Endpoint Name:** /order
    - **Method:** POST
    - **Description:** Create a new order.
    - **Payload:** CustomerInfo, ProductDetails, Payments.
- **Endpoint Name:** /user
    - **Method: GET**
    - **Description:** Fetch details of user.
    - **Response:** User Details (ID, Name, cart Items, Info, orders).

Step 4 : Table View of API Endpoints

Endpoint	Method	Purpose	Response Example
`/admin`	GET	Fetch input for Name and Password.	`{ "adminID": "1", "name": "AdminUser", "password": "hashedpassword", "role": "SuperAdmin" }`

`/products`	GET	Fetch all available products.	`[{ "productID": "101", "name": "Wooden Table", "category": "Furniture", "price": 150, "discount": 10, "stock": true, "description": "Solid oak table", "images": ["image1.jpg", "image2.jpg"], "color": "Brown", "size": "Large" }]`
`/my-orders`	GET	Get all the orders of a user.	`[{ "orderID": "5001", "status": "Shipped", "trackingID": "TRK12345" }]`
`/order`	POST	Create a new order.	`{ "orderID": "5002", "customerID": "123", "products": [{"productID": "101", "quantity": 2}], "paymentStatus": "Paid" }`
`/user`	GET	Fetch details of a user.	`{ "userID": "123", "name": "John Doe", "cartItems": [{"productID": "101", "quantity": 1}], "info": { "email": "john@example.com" }, "orders": ["5001", "5002"] }`

Step 5: Sanity Schema Example

- **/admin**

```
// admin.schema.ts
export default {
name: "admin",
type: "document",
title: "Admin",
fields: [
{ name: "adminID", type: "string", title: "Admin ID" },
{ name: "name", type: "string", title: "Name" },
{ name: "password", type: "string", title: "Password" },
{ name: "role", type: "string", title: "Role" },
],
};
```

- **/customer OR user**

```
// customer.schema.ts
export default {
  name: "customer",
  type: "document",
  title: "Customer",
  fields: [
    { name: "customerID", type: "string", title: "Customer ID
    { name: "name", type: "string", title: "Name" },
    { name: "email", type: "string", title: "Email" },
    { name: "address", type: "string", title: "Address" },
    {
      name: "cartItems",
      type: "array",
      title: "Cart Items",
      of: [{ type: "reference", to: [{ type: "product" }] }],
    },
    { name: "contactInfo", type: "string", title: "Contact In
  ],
};
```

- **/product**

```
// product.schema.ts
export default {
  name: "product",
  type: "document",
  title: "Product",
  fields: [
    { name: "productID", type: "string", title: "Product ID"
    { name: "name", type: "string", title: "Name" },
    { name: "category", type: "string", title: "Category" },
    { name: "stockStatus", type: "string", title: "Stock Stat
    { name: "price", type: "number", title: "Price" },
    { name: "discount", type: "number", title: "Discount" },
    { name: "description", type: "text", title: "Description"
    { name: "images", type: "array", title: "Images", of: [{
    { name: "color", type: "string", title: "Color" },
    { name: "size", type: "string", title: "Size" },
  ],
};
```

- **/order**

```
// order.schema.ts
export default {
  name: "order",
  type: "document",
  title: "Order",
  fields: [
    { name: "orderID", type: "string", title: "Order ID" },
    { name: "status", type: "string", title: "Status" },
    {
      name: "product",
      type: "array",
      title: "Product",
      of: [{ type: "reference", to: [{ type: "product" }] }],
    },
    { name: "customerID", type: "reference", to: [{ type: "cu
    { name: "quantity", type: "number", title: "Quantity" },
    { name: "price", type: "number", title: "Price" },
    { name: "orderDate", type: "datetime", title: "Order Date
  ],
};
```

/cart

```
// cart.schema.ts
export default {
  name: "cart",
  type: "document",
  title: "Cart",
  fields: [
    { name: "cartID", type: "string", title: "Cart ID" },
    { name: "customer", type: "reference", to: [{ type: "cust
    {
      name: "products",
      type: "array",
      title: "Products",
      of: [{ type: "reference", to: [{ type: "product" }] }],
    },
    { name: "totalItems", type: "number", title: "Total Items
    { name: "totalPrice", type: "number", title: "Total Price
  ],
};
```