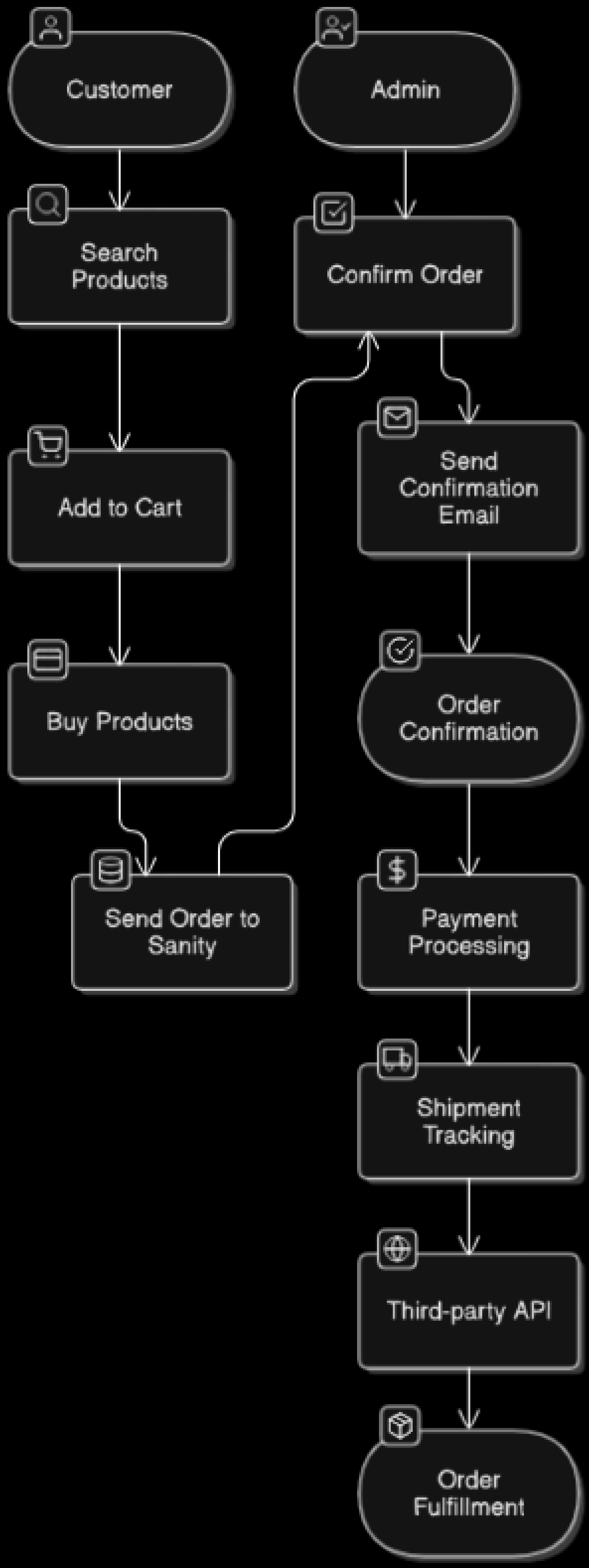
E-commerce Flowchart with Next.js and Sanity



Flowchart Explanation:

Customer Flow:

1. **Search Products**: The customer starts by searching for products.
2. **Add to Cart**: The customer adds selected products to the shopping cart.
3. **Buy Products**: The customer proceeds to purchase the products in their cart.
4. **Send Order to Sanity**: The order details are sent to the Sanity CMS backend for processing.

Admin Flow (Triggered by Customer's "Buy Products" Action):

1. **Payment Processing**: The system processes the customer's payment, likely through an external payment gateway.
2. **Shipment Tracking**: The system triggers shipment tracking using a third-party shipping API.
3. **Third-party API**: Interaction with external services such as payment gateways and shipping providers.
4. **Order Fulfillment**: The order is packaged and shipped.
5. **Order Confirmation**: The system generates an order confirmation after processing the payment and shipment.
6. **Send Confirmation Email**: A confirmation email is sent to the customer confirming their order.
7. **Confirm Order**: The admin confirms the order in the system, signaling that the transaction and shipping have been processed.

Relationship to Sanity Backend ER Diagram:

The flowchart represents the flow of actions, while the ER diagram depicts the relationships between the entities and their attributes. Below is how the flowchart's actions relate to the entities and relationships in the ER diagram:

Entities in the ER Diagram:

1. **Product**: Represents the product being sold (attributes: name, description, price, stock, images, etc.).
2. **Order**: Represents a customer's purchase, including attributes such as order date, order status, total amount, shipping address, etc.
3. **Customer**: Contains customer-related information (attributes: name, email, address, etc.).

Relationships in the ER Diagram:

- **Customer - Order (One-to-Many)**: One customer can place multiple orders.
- **Order - Product (Many-to-Many)**: An order can contain multiple products, and a product can appear in multiple orders. This is implemented through a join table or array of references in Sanity.

Sanity Schema and ER Diagram Connection:

- The **Product schema** in Sanity corresponds to the **Product** entity in the ER diagram, with its fields (name, description, price, etc.) representing the attributes of the Product entity.
- Similarly, the **Order schema** and **Customer schema** in Sanity correspond to the **Order** and **Customer** entities, respectively, in the ER diagram.

Simplified ER Diagram:

```
Customer
+------------+     1..*     +---------+     *..*     +---------+
| customerId |------------| orderId |------------| productId |
| name       |            | orderDate|            | name      |
| email      |            | status   |            | price     |
| address    |            | totalAmount|          | description|
+------------+            +---------+            +---------+
                           |
                           |
                           +---------+
                           | OrderItem| (Join Table/Array of References)
                           | orderId  |
                           | productId|
                           | quantity |
                           +---------+
```

Explanation of the Diagram:

- **Customer - Order (1..*)**: One customer can place many orders.
- **Order - Product (..)**: An order can contain many products, and products can appear in multiple orders. This relationship is facilitated through the **OrderItem** join table, which holds the `orderId`, `productId`, and `quantity`.
- **OrderItem**: This table acts as a join between the **Order** and **Product** entities, allowing the system to store the details of each product in an order, such as the quantity purchased.