Planing_The_Technical_Foundation (Day 2)

<u>Steps for Building a Technical Marketplace:</u>

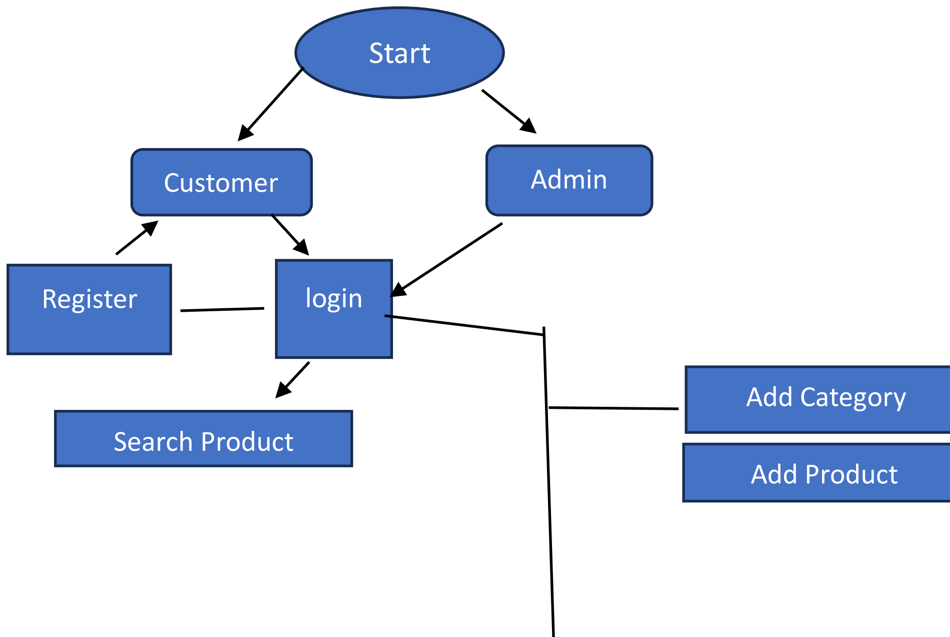
- Frontend: Next.js for building the site.
- Tailwind CSS for styling the website.
- CMS (Content Management System): Sanity to manage dynamic content like product details, descriptions, etc.
- Order Tracking: Ship Engine (real-time shipment updates)
- Payment Gateway: Stripe (securely process payments)

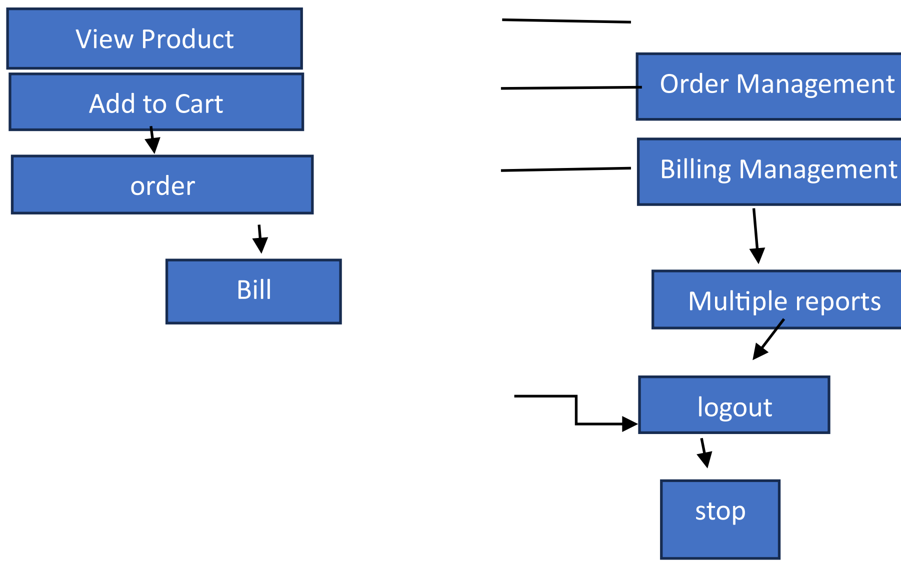
<u>Frontend (Next.js):</u>

- Client-Side Rendering (improve speed and responsiveness).
- Server-Side Rendering (SEO and preloading product pages).
- Tailwind CSS for styling the website.

<u>Website Flow :</u>

- Homepage: Shows product categories, featured items, and a search bar.
- Category Navigation: users explore products by their categories.
- Product Listings: Displays products with options to sort and filter.
- Search Bar: Helps users find specific products or sellers.
- Product Detail Page: Shows detailed info, images, reviews, and an option to add products to the cart.
- User Account: users manage their profiles, past orders, wish lists, and saved addresses.
- Shopping Cart: Displays selected products and lets users edit or apply discounts.
- Checkout Page: Collects billing, shipping, and payment details/Info.
- Order Confirmation: Confirms the order and provides tracking details.
- Delivery: Delivers the products to the customer's address.





API Endpoints (Links between different parts of the website):

- Authentication (Login/Sign Up): Register and log in users.
- Product Management: Show products, product details, create or update products for sellers.
- Categories & Filters: Get product categories and filter options (like price or brand).
- Cart & Wishlist: Add, view, update, or remove products from the cart or Wishlist.
- Orders: Create new orders, view orders made by users, and check payment status.
- Payment: Process payments and check if they were successful.
- Reviews & Ratings: Add and view reviews for products.
- Admin/Seller Management: Admins can view all users, and sellers can view orders for their products.

List of Common APIs endpoints for a market place e-commerce webistes:

- Authentication:
  - POST /Api/auth/register: Registers a new user.
  - POST /Api/auth/login: Logs a user into the system.
  - POST /Api/auth/logout: Logs a user out of the system.
  - GET /Api/auth/profile: Retrieves the user's profile details.
  - PUT /Api/auth/update: Updates the user's profile or password.
- Product Management:
  - GET /Api/products: Fetches all products (with optional filters and pagination).
  - GET /Api/products/{id}: Retrieves details of a specific product.
  - POST /Api/products: Allows sellers to add new products.
  - PUT /Api/products/{id}: Enables sellers to update product details.
  - DELETE /Api/products/{id}: Allows sellers to delete a product.
- Categories & Filters:
  - GET /Api/categories: Fetches available product categories.

    - GET /Api/categories/{id}/products: Retrieves products within a particular category.
    - GET /Api/filters: Provides available filters (e.g., price range, brand, rating).
- Cart & Wishlist:
    - POST /Api/cart: Adds an item to the user's shopping cart.
    - GET /Api/cart: Displays the contents of the cart.
    - PUT /Api/cart/{id}: Updates the quantity of an item in the cart.
    - DELETE /Api/cart/{id}: Removes an item from the cart.
    - POST /Api/Wishlist: Adds a product to the user's Wishlist.
    - GET /Api/Wishlist: Displays items in the Wishlist.
    - DELETE /Api/Wishlist/{id}: Removes an item from the wish list.
- Orders:
    - POST /Api/orders: Creates a new order.
    - GET /Api/orders: Retrieves all orders associated with the user.
- Payment:
    - POST /Api/payment: Initiates payment processing for an order.
    - GET /Api/payment/status/{id}: Fetches the payment status for a specific order.
- Reviews & Ratings:
    - POST /Api/reviews: Adds a review for a product.
    - GET /Api/reviews/{product_id}: Retrieves reviews for a particular product.
- Admin/Seller Management:
    - GET /Api/admin/users: Allows administrators to view all users.
    - GET /Api/seller/orders: Lets sellers view all orders related to their products.

Schema :

Product Schema:

```
import {defineType} from "sanity";

export const product = defineType ({

 name: "product",

 title: "Product",

 type: "document",

 fields: [

  {

   name: "title",

   title: "Product Title",

   type: "string",

   validation: (rule) => rule.required(),

  },

  {
```

```
  name: "description",
  title: "Product Description",
  type: "text",
  validation: (rule) => rule.required(),
},
{
  name: "productImage",
  title: "Product Image",
  type: "image",
  validation: (rule) => rule.required(),
},
{
  name: "price",
  title: "Product Price",
  type: "number",
  validation: (rule) => rule.required(),
},
{
  name: "tags",
  title: "Product Tags",
  type: "array",
  of: [{ type: "string" }],
},
{
  name: "discountPercentage",
  title: "Discount Percentage",
  type: "number",
},
{
  name: "isNew",
  title: "Is This a New Product?",
      type: "boolean",
    },
  ],
});
```

Order Schema:

```
export const order = {
  name: 'order',
  title: 'Customer Order',
  type: 'document',
  fields: [
    {
      name: 'orderId',
      title: 'Order ID',
      type: 'string',
      validation: (rule) => rule.required(),
    },
    {
      name: 'customer',
      title: 'Customer',
      type: 'reference',
      to: [{ type: 'customer' }],
      validation: (rule) => rule.required(),
    },
    {
      name: 'items',
      title: 'Order Items',
      type: 'array',
      of: [
        {
          type: 'object',
      fields: [
       {
         name: 'product',
         title: 'Product',
         type: 'reference',
         to: [{ type: 'product' }],
       },
       {
         name: 'quantity',
         title: 'Quantity',
         type: 'number',
         validation: (rule) => rule.min(1).required(),
       },
      ],
    },
  ],
},
{
  name: 'totalAmount',
  title: 'Total Cost',
  type: 'number',
  validation: (rule) => rule.min(0).required(),
},
{
  name: 'shippingAddress',
  title: 'Shipping Details',
  type: 'object',
  fields: [
    {
      name: 'street',
      title: 'Street',
      type: 'string',
      validation: (rule) => rule.required(),
    },
    {
      name: 'city',
      title: 'City',
      type: 'string',
      validation: (rule) => rule.required(),
    },
    {
      name: 'zipCode',
      title: 'Zip Code',
      type: 'string',
      validation: (rule) => rule.required(),
    },
  ],
},
{
  name: 'orderDate',
  title: 'Date of Order',
  type: 'datetime',
  validation: (rule) => rule.required(),
},
{
  name: 'status',
  title: 'Order Status',
  type: 'string',
  options: {
    list: ['Pending', 'Shipped', 'Delivered'],
    layout: 'dropdown',
  },
      validation: (rule) => rule.required(),
    },
  ],
};
```

Customer Schema :

```
export const customer = {
  name: 'customer',
  title: 'Customer',
  type: 'document',
  fields: [
    {
      name: 'id',
      title: 'Customer ID',
      type: 'string',
      validation: (rule) => rule.required(),
    },
    {
      name: 'name',
      title: 'Full Name',
      type: 'string',
      validation: (rule) => rule.required(),
    },
    {
      name: 'email',
      title: 'Email Address',
      type: 'string',
      validation: (rule) => rule.required().email(),
    },
    {
      name: 'phone',
      title: 'Phone Number',
      type: 'string',
      validation: (rule) => rule.regex(/^\+?[0-9]{10,15}$/).required(),
    },
    {
      name: 'address',
      title: 'Address',
      type: 'object',
      fields: [
        { name: 'street', title: 'Street', type: 'string' },
        { name: 'city', title: 'City', type: 'string' },
        { name: 'zipCode', title: 'Zip Code', type: 'string' },
      ],
    },
    {
      name: 'orders',
      title: 'Orders',
      type: 'array',
      of: [{ type: 'reference', to: [{ type: 'order' }] }],
    },
  ],
};
```

Shipment Schema:

```
export const Shipment = {
  name: 'shipment',
  title: 'Shipment',
  type: 'document',
  fields: [
    {
      name: 'shipmentId',
      title: 'Shipment ID',
      type: 'string',
      validation: (Rule) => Rule.required(),
    },
    {
      name: 'orderId',
      type: 'reference',
      to: [{ type: 'order' }],
      validation: (Rule) => Rule.required(),
    },
    {
      name: 'shippingCarrier',
      title: 'Shipping Carrier',
      type: 'string',
      validation: (Rule) => Rule.required(),
    },
    {
      name: 'trackingNumber',
      title: 'Tracking Number',
      type: 'string',
    },
    {
      name: 'shipmentDate',
      title: 'Shipment Date',
      type: 'datetime',
      validation: (Rule) => Rule.required(),
    },
    {
      name: 'shipmentStatus',
      title: 'Shipment Status',
      type: 'string',
      options: {
        list: ['Shipped', 'In Transit', 'Delivered'],
        layout: 'dropdown',
      },
      validation: (Rule) => Rule.required(),
    },
  ],
};
```

This plan builds a fast and easy-to-use online store for small businesses to sell products and manage orders.