HACKATHON 3 [DAY:2] MARKET PLACE TECHNICAL FOUNDATION

Introduction

This Furniture Website is designed to provide a user-friendly platform for buying furniture. This document outlines the technical foundation, including system architecture, API requirements, and data schemas.

Technical Requirements

1. Frontend Requirements

- Responsive design for desktop and mobile.

Essential pages:

Home
About
Contact
Product Details
Cart
Checkout
Order Confirmation

2. Backend Requirements:

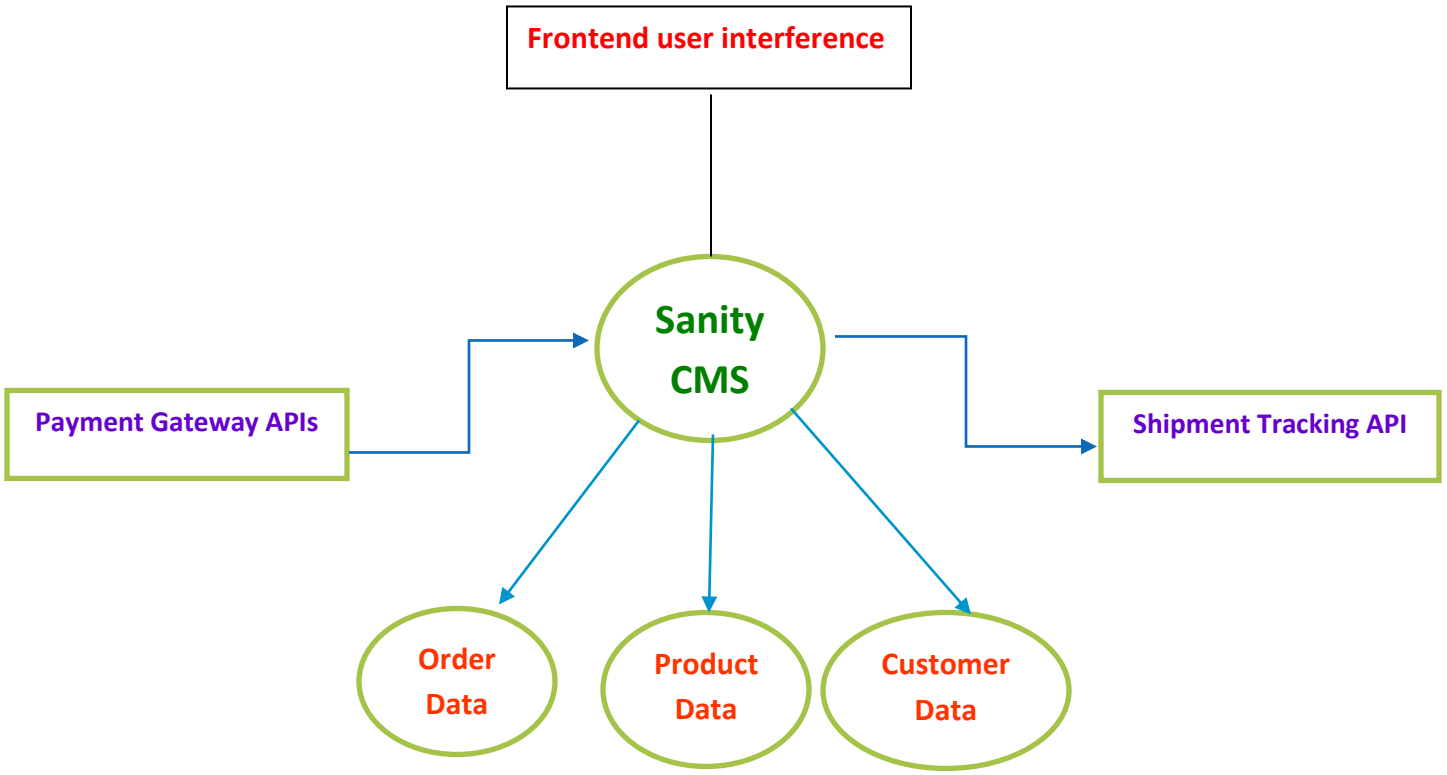
- Sanity CMS for managing products, categories, and orders.

3. API Endpoints:

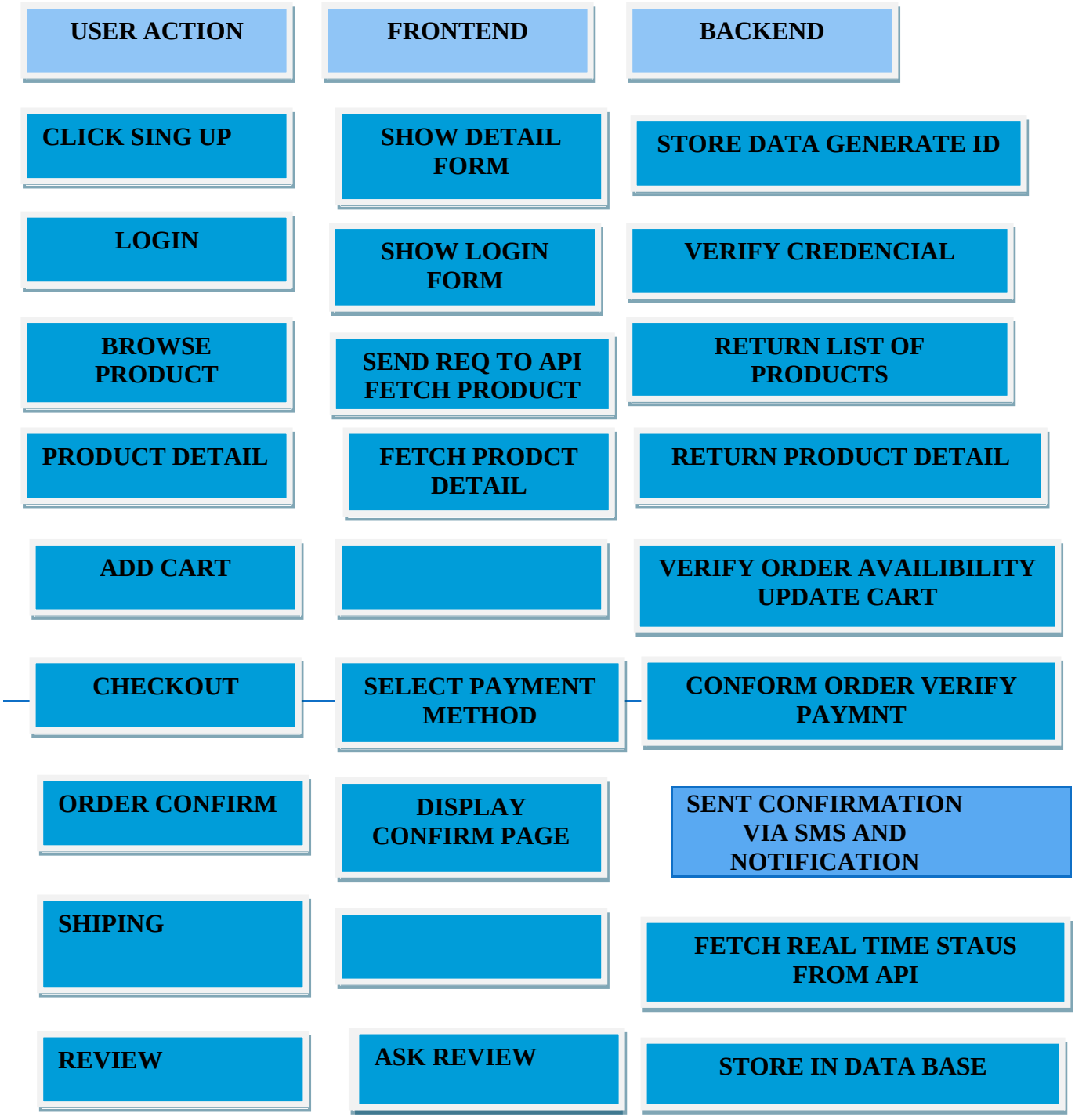
GET /products: Fetch all furniture items.

POST /orders`: Save new orders.

SYSTEM ARCHITECTURE



WORK FLOW


USER ACTION
FRONTEND
BACKEND
CLICK SING UP
SHOW DETAIL FORM
STORE DATA GENERATE ID
LOGIN
SHOW LOGIN FORM
VERIFY CREDENCIAL
BROWSE PRODUCT
SEND REQ TO API FETCH PRODUCT
RETURN LIST OF PRODUCTS
PRODUCT DETAIL
FETCH PRODCT DETAIL
RETURN PRODUCT DETAIL
ADD CART
VERIFY ORDER AVAILIBILITY UPDATE CART
CHECKOUT
SELECT PAYMENT METHOD
CONFORM ORDER VERIFY PAYMNT
ORDER CONFIRM
DISPLAY CONFIRM PAGE
SENT CONFIRMATION VIA SMS AND NOTIFICATION
SHIPING
FETCH REAL TIME STAUS FROM API
REVIEW
ASK REVIEW
STORE IN DATA BASE

Endpoint	Method	Purpose	Payload (if applicable)	Response Example
/products	GET	Fetch all available furniture products.	None	[{"id": "1", "name": "Seater", "price": 5000, "stock": 10}]
/products/:id	GET	Fetch details of a specific product.	None	{"id": "1", "name": "Seater", "price": 5000, "stock": 10, "description": "cozy and comfortable 2-seater"}
/categories	GET	Fetch all product categories.	None	[{"id": "1", "name": "sofa set Living Room"}, {"id": "2", "name": "Bedroom Set"}]
/orders	POST	Save a new order in the system.	{"customerName": "Gimini", "items": [{"productId": "1", "quantity": 2}], "totalAmount": 10000}	{"orderId": "101", "status": "Success"}
/cart	POST	Add items to the user's cart.	{"productId": "1", "quantity": 2}	{"cartId": "201", "status": "Item Added"}
/cart	GET	Fetch all items in the user's cart.	None	[{"productId": "1", "name": "Seater", "quantity": 2, "price": 5000}]
/shipment	GET	Track the shipment status of an order.	None	{"orderId": "101", "status": "In Transit", "ETA": "2 days"}

Sanity CMS Schema for furniture website

sanity schema for Products

- Name: Product

- Fields

  - Name (string)

  - Price (number)

  - Stock (number)

  - Dimensions (string)

Schema for Orders

```
  {
    name: 'status',
    type: 'string',
    title: 'Order Status',
    options: {
      list: [
        { title: 'Pending', value: 'pending' },
        { title: 'In Progress', value: 'inProgress' },
        { title: 'Completed', value: 'completed' },
        { title: 'Cancelled', value: 'cancelled' },
      ],
    },
  },
  {
    name: 'orderDate',
    type: 'datetime',
    title: 'Order Date',
  },
 ],
};
```

Schema for User Information

```
export default {
 name: 'user',
 type: 'document',
 title: 'User',
 fields: [
  {
   name: 'name',
   type: 'string',
   title: 'Full Name',
  },
  {
   name: 'email',
   type: 'string',
   title: 'Email Address',
  },
  {
   name: 'phone',
   type: 'string',
   title: 'Phone Number',
  },
  {
   name: 'address',
   type: 'object',
   title: 'Address',
   fields: [
    {
     name: 'street',
     type: 'string',
     title: 'Street',
    },
    {
     name: 'city',
     type: 'string',
     title: 'City',
    },
    {
     name: 'state',
     type: 'string',
     title: 'State',
    },
    {
     name: 'zipCode',
     type: 'string',

     title: 'ZIP Code',
    },
   ]
   name: 'createdAt',
   type: 'datetime',
   title: 'Account Created At',
  },
 ],

};
```

Conclusion

Key takeaways include:

1. **Well-Defined Technical Requirements**:

website will offer all basic functionalities.

2. **Efficient System Architecture**:

Its proper interaction flow between the frontend (Next.js), backend (Sanity CMS), and third-party APIs ensures efficient and smooth performance.

3. **Third-Party Integrations**:

Tools like Stripe for payments and shipment tracking APIs provides enhanced functionality.

4. **Scalable Data Models**:

Using Sanity CMS, schemas for products, orders, and user information have been defined to manage data effectively and allow for future scalability.

In conclusion, this technical foundation establishes a comprehensive and scalable plan for the development of the **Furniture Website**. By leveraging industry best practices, modern tools, and third-party integrations, the foundation ensures a seamless user experience and robust backend infrastructure.