Marketplace Technical Foundation

Hackathon Day 2

Template 9: Q commerce Elite food Restaurant

1. Define Technical Requirements The first step is to translate your business goals into clear technical requirements.

   Frontend Requirement:

   We will design easy to access product website for better experience with excellent UI (userinterface)

   We will design our website in Next.Js using current designing technique like tailwindcss shadcn.ui and make a fully responsive website for all type of screens. For better experience

   Our essential pages are Home ,Menu, Shop, shopping detail, cart , checkout

2. Sanity CMS as Backend:

The essential schemas in our project include:

- **Menu**: To manage the list of food items, categories, and their details.
- **Shop**: To handle shop-specific configurations or settings (e.g., operating hours, featured items).
- **Shopping Detail**: To record user-selected products during their shopping session.
- **Cart**: To manage the products added by users for purchase, including quantities and total cost.
- **Checkout**: To collect customer information, manage payment details, and process orders efficiently.

Why Sanity is Ideal for Our Project

- **Dynamic Product Management**:
  Instead of creating multiple static pages for each product, we leverage Sanity's schema design capabilities to dynamically manage and display multiple products. By using arrays in schemas, we can list and manage several products with minimal code, making the development process faster and more efficient.
- **User Data Handling**:
  Our schemas will also be designed to capture and store user data, such as shopping preferences,

orders, and cart details, securely within Sanity. This allows for seamless user interaction while maintaining robust data organization.

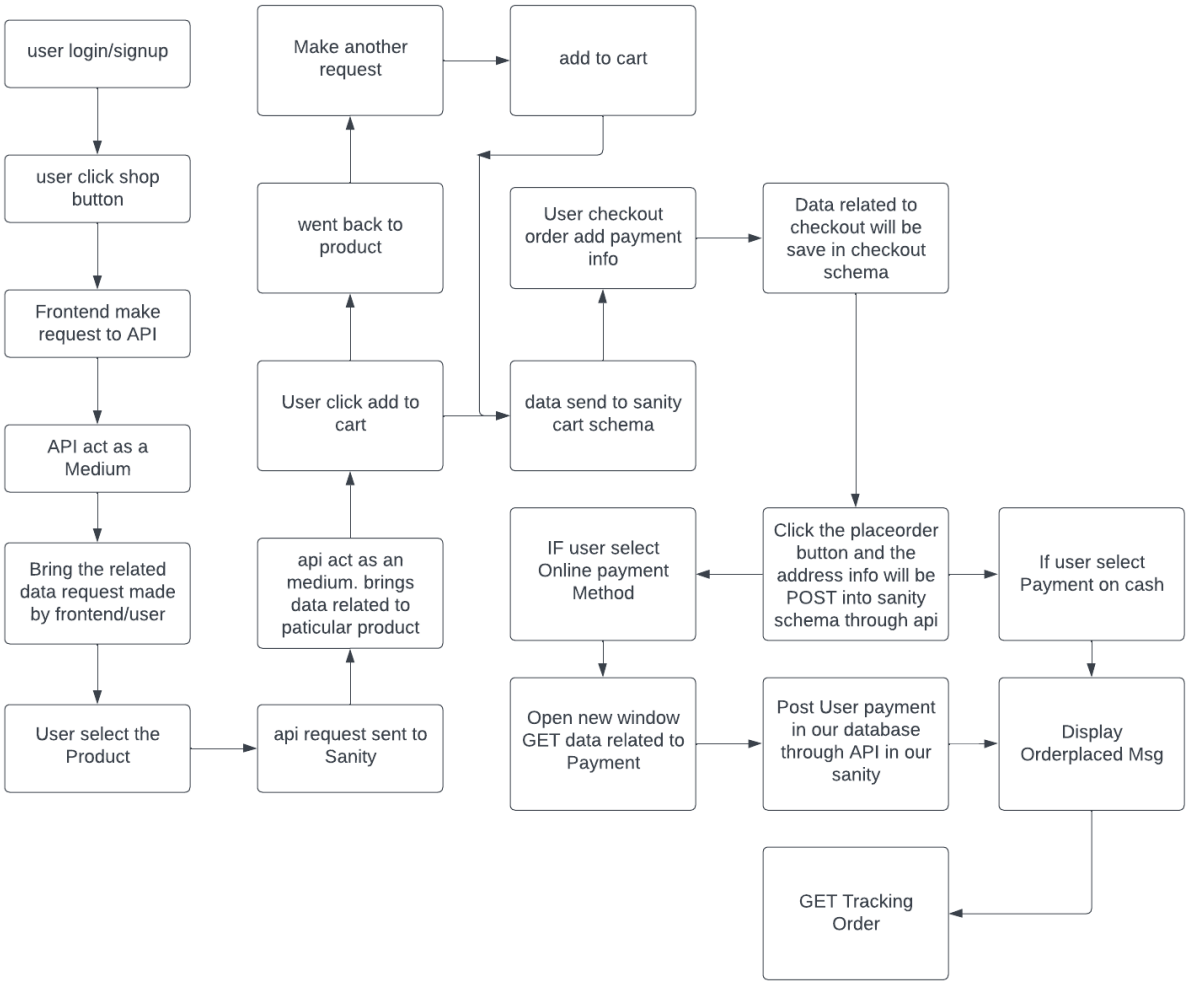
- **Scalable and Flexible Solution**:
  Sanity's real-time updates and query capabilities ensure that both the website and user data remain synchronized, providing an optimal experience for both administrators and end users.

3. Third party API

   We will use third party API for payments method like Stripe etc

4. Design System Architecture

Design System Architecture Create a high-level diagram showing how your system components interact. Use tools like pen and paper or software like Lucidchart, Figma or Excalidraw. For example, a more detailed architecture might include workflows such as:


user login/signup
user click shop button
Frontend make request to API
API act as a Medium
Bring the related data request made by frontend/user
User select the Product
api request sent to Sanity
api act as an medium. brings data related to paticular product
User click add to cart
went back to product
Make another request
add to cart
data send to sanity cart schema
User checkout order add payment info
Data related to checkout will be save in checkout schema
Click the placeorder button and the address info will be POST into sanity schema through api
IF user select Online payment Method
Open new window GET data related to Payment
Post User payment in our database through API in our sanity
If user select Payment on cash
Display Orderplaced Msg
GET Tracking Order

PLAN API REQUIREMENT

Q-commerce:
template 9
elite Food Restaurant

Following is the API
endpionts based on our
data and schema

Endpoint Name: SignUp
Method: POST
Description: Adding user info to scehma/database
Response Example: {
Name:"faizan masood",
age:"24",
location:"north karachi"
username:"faizanmasood009@gmail.com"
password:"faizan@124"}

Endpoint Name: Shop
Method: Get
Description: showcasing data(List of food items) from the schema on to the client side
Response Example: {
ProductImage:URL
Products Name:"pizza",
Product prize:"12$"

Endpoint Name: selected Item detail
Method: Get
Description: showcasing data(selected item from the shop window) from the schema on to the client side
Response Example: {
Stockdetail:In stock/ Out of stock(true or false)
Product Name:"pizza",
Product Detail:"lorem ipsum....."
Product prize:"12$"
Prodcut Rating:"stars"
Reviews Count:"no of reviews"
button:"add to cart "
catorgary:"pizza"
tag:"ourshop"
icons:"fb,insta,twit,pin"

Endpoint Name: : Add to cart
Method: Post
Description: Adding Food items into the cart
Response Example: {
Stockdetail:In stock/ Out of stock(true or false)
Product Name:"pizza",
Product prize:"12$"
Quantity:"1"
Total:"12"
Remove:" X/O"
Coupon Code
cart subtotal:"50$"
shipping charge:"00"
Total amout:"60$"

Endpoint Name: Shipping address
Method: GET
Description: Showing shipping window
Response Example: {
FirstName:
LastName:"pizza",
Email:"12$"
phone:"1"
company:"12"
Country" X/O"
choosecity:
zipcode:
address1:
address2:

Endpoint Name: Shipping address
Method: POST
Description: Adding Address Information
Response Example: {
FirstName:"faizan masood
LastName:"masood ahmed",
Email:"faizanmasood009@gmail.com"
phone:"12312321414"
company:"NIL"
Country" pakistan"
choosecity:"karachi"
zipcode:"75850"
address1: "north karachi"
address2:"north karachi"

Endpoint Name: Order Display message
Method: GET
Description: Sending order place message to Pno
Response Example: {
Message: "your order is placed

Endpoint Name: Tracking Order
Method: GET
Description: Showing order status
Response Example: {
Currect status of order: "'Order Placed', 'Preparing', 'Out for Delivery', 'Delivered"
datetime:"1/1/0000

SNAITY SCHEMAS:

USER:

```
export default {

  name: "user",
  title: "User",
  type: "document",
  fields: [
    {
      name: "name",
      title: "Name",
      type: "string",
    },
    {
      name: "email",
      title: "Email",
      type: "string",
      unique: true,
    },
```

```
    {
      name: "password",
      title: "Password",
      type: "string",
    },
  ],
};
```

PRODUCT:

```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'name',
      title: 'Name',
      type: 'string',
      validation: (Rule: any) => Rule.required()
    },
    {
      name: 'slug',
      title: 'Slug',
      type: 'slug',
      options: {
        source: 'name',
        maxLength: 90,
      },
      validation: (Rule: any) => Rule.required()
    },
    {
      name: 'descriptionOne',
      title: 'Description One',
      type: 'text',
      validation: (Rule: any) => Rule.required(),
  },
    {
      name: 'image',
      title: 'Image',
      type: 'image',
      options: {
```

```
      hotspot: true,
    },
    validation: (Rule: any) => Rule.required()
  },
  {
    name: 'thumbnailImages',
    title: 'Thumbnail Images',
    type: 'array',
    of: [{ type: 'image' }],
  },
  {
    name: 'currency',
    title: 'Currency',
    type: 'string',
    options: {
      list: [
        { title: 'USD', value: 'USD' },
        { title: 'EUR', value: 'EUR' },
        { title: 'GBP', value: 'GBP' },
        // Add more currencies as needed
      ],
      layout: 'radio', // Or 'dropdown' depending on your preference
    },
    initialValue: 'USD', // Default currency
  },

  {
    name: 'price',
    title: 'Price',
    type: 'number',
    validation: (Rule: any) => Rule.required().positive()
  },
  {
    name: "category",
    title: "Category",
    type: "reference",
    to: [{ type: "category" }],
},
{
  name: 'createdAt',
  title: 'Created At',
  type: 'datetime',
  options: {
    dateFormat: 'YYYY-MM-DD',
    timeFormat: 'HH:mm',
```

```
    },
    readOnly: true,
  },
  {
    name: 'updatedAt',
    title: 'Updated At',
    type: 'datetime',
    options: {
      dateFormat: 'YYYY-MM-DD',
      timeFormat: 'HH:mm',
    },
    readOnly: true,
  },
  {
    name: 'rating',
    title: 'Star Rating',
    type: 'number',
    description: 'The star rating for the product (1-5)',
    validation: (Rule: any) => Rule.min(1).max(5).integer(), // Ensures the
rating is between 1 and 5
    options: {
      list: [1, 2, 3, 4, 5], // Optional: Display as a dropdown in the Studio
    },
  },
  {
    name: 'description',
    title: 'Description',
    type: 'text',
    validation: (Rule: any) => Rule.required(),
},
{
  name: 'reviews',
  title: 'Reviews',
  type: 'array',
  of: [
    {
      type: 'object',
      fields: [
        { name: 'reviewer', title: 'Reviewer Name', type: 'string' },
        { name: 'rating', title: 'Rating', type: 'number' },
        { name: 'comment', title: 'Comment', type: 'text' },
      ],
      options: {
        add: {
          _key: {
```

```
            type: 'string',
            title: 'Key',
            description: 'A unique key for each review',}
        },
      },
    },
  ],
},
{
  name: 'keybenefits',
  title: 'Key Benefits',
  type: 'string',
  validation: (Rule: any) => Rule.required().max(50).warning("Keep the heading
concise.")
},
{
  name: 'benefitsList',
  title: 'Key Benefit Points',
  type: 'array',
  of: [{ type: 'string' }],
  validation: (Rule: any) =>
    Rule.required().min(1).max(5).error('You must provide 1 to 5 benefit
points.'),
},
]
};
```

```
CART: export default {

  name: 'cart',
  title: 'Cart',
  type: 'document',
  fields: [
    {
      name: 'userId',
      title: 'User ID',
      type: 'string',
      description: 'Unique identifier for a user session',
    },
    {
      name: 'Cartitems',
      title: 'Cart Items',
      type: 'array',
```

```
    of: [
      {
        type: 'object',
        fields: [
          {
            name: 'productId',
            title: 'Product ID',
            type: 'string',
          },
          {
            name: 'name',
            title: 'Product Name',
            type: 'string',
          },
          {
            name: 'image',
            title: 'Product Image',
            type: 'string',
            options: {
              hotspot: true,
            },
          },
          {
            name: 'price',
            title: 'Price',
            type: 'number',
          },
          {
            name: 'rating',
            title: 'Rating',
            type: 'number',
          },
          {
            name: 'quantity',
            title: 'Quantity',
            type: 'number',
          },
          {
            name: 'total',
            title: 'Total Price',
            type: 'number',
          },
        ],
      },
    ],
```

```
    },
    {
      name: 'createdAt',
      title: 'Created At',
      type: 'datetime',
      initialValue: () => new Date().toISOString(),
      readOnly: true,
    },
    {
      name: 'updatedAt',
      title: 'Updated At',
      type: 'datetime',
      initialValue: () => new Date().toISOString(),
      readOnly: true,
    },
  ],
};
```

```
Chef: export default {
    name: 'chef',
    title: 'Chef',
    type: 'document',
    fields: [
      {
        name: 'image',
        title: 'Chef Image',
        type: 'image',
        options: {
          hotspot: true,
        },
      },
      {
        name: 'chefName',
        title: 'Chef Name',
        type: 'string',
      },
      {
        name: 'designation',
        title: 'Designation',
        type: 'string',
      },
    ],
  };
```