Maketplace Technical Foundation - Ecommerce

1.Technical Requirements

Frontend

Technology Stack

- **FrameWork:** Next.js

A React-based framework that enables server-side rendering (SSR) and static site generation (SSG), perfect for building high-performance and SEO-friendly applications.

- **UI Library: shadcn/ui**

Provides pre-built, customizable UI components that work seamlessly with Tailwind CSS.

- **Styling Framework: Tailwind CSS**

A utility-first CSS framework for fast and responsive design.

Pages to Create

1.Home Page

- Features: Display promotional banners, featured products, and categories.
- Libraries/Features to Use:
- shadcn/ui components like Card, Button, and Carousel.
- Tailwind CSS for responsive design.

2.Product Listing Page

- Features: Show all products with options for filtering (category, price range) and sorting (price, popularity).
- Libraries/Features to Use:
- shadcn/ui components like Select, Input, and Grid.
- Use dynamic routes (pages/products/[category].tsx) for category-based listing.

3.Product Details Page

- Features: Show detailed product information (image gallery, price, description, and stock status).
- Libraries/Features to Use:
- shadcn/ui components like Modal, Carousel, and Tabs.
- Image optimization using Next.js <Image> component.

4.Cart Page

- Features: View and update selected product quantities with a summary of the total price.
- Libraries/Features to Use:
- shadcn/ui components like Table and Badge.
- Persist cart data in localStorage or context.

5.Checkout Page

- Features: Collect user information (address, payment details) and process payments.
- Libraries/Features to Use:
- shadcn/ui components like Form and Input.
- Integrate a payment gateway like Stripe.

6.Rental Listing Page

- Features: Display all rentable helicopters with filters for availability and pricing.
- Libraries/Features to Use:
- Dynamic routes (pages/rentals/[id].tsx) for individual helicopter details.
- shadcn/ui components like Card and Filter.

7.Rental Details Page

- Features: Show detailed information about the selected helicopter, including pricing and rental terms.
- Libraries/Features to Use:
- shadcn/ui components like Modal and Accordion.

8.Rental Cart Page

- Features: View selected rental items and finalize booking details.
- Libraries/Features to Use:
- shadcn/ui components for form handling and summary display.

Sanity CMS

Schema Design for Data Management:

Product Schema:

```
export default {
 name: 'product',
 type: 'document',
 fields: [
  { name: 'name', type: 'string', title: 'Product Name' },
  { name: 'price', type: 'number', title: 'Price' },
  { name: 'stock', type: 'number', title: 'Stock Level' },
  { name: 'description', type: 'text', title: 'Description' },
  { name: 'image', type: 'image', title: 'Product Image' }
 ]
};
```

Rental Schema:

```
export default {
name: 'rental',
type: 'document',
fields: [
{ name: 'name', type: 'string', title: 'Helicopter Name' },
{ name: 'rentalPrice', type: 'number', title: 'Rental Price per Hour' },
{ name: 'availability', type: 'boolean', title: 'Availability' },
{ name: 'description', type: 'text', title: 'Description' },
{ name: 'image', type: 'image', title: 'Helicopter Image' },
{ name: 'rentalTerms', type: 'text', title: 'Rental Terms' }
]
};
```

Customer Schema:

```
export default {
 name: 'customer',
 type: 'document',
 fields: [
  { name: 'name', type: 'string', title: 'Customer Name' },
  { name: 'email', type: 'string', title: 'Email Address' },
  { name: 'address', type: 'string', title: 'Shipping Address' }
 ]
};
```

Order Schema:

```
\export default {
 name: 'order',
 type: 'document',
 fields: [
 { name: 'customer', type: 'reference', to: [{ type: 'customer' }], title:
'Customer' },
 { name: 'products', type: 'array', of: [{ type: 'reference', to: [{ type:
'product' }] }], title: 'Products' },
 { name: 'total', type: 'number', title: 'Total Amount' },
 { name: 'status', type: 'string', title: 'Order Status' }
 ]
};
```

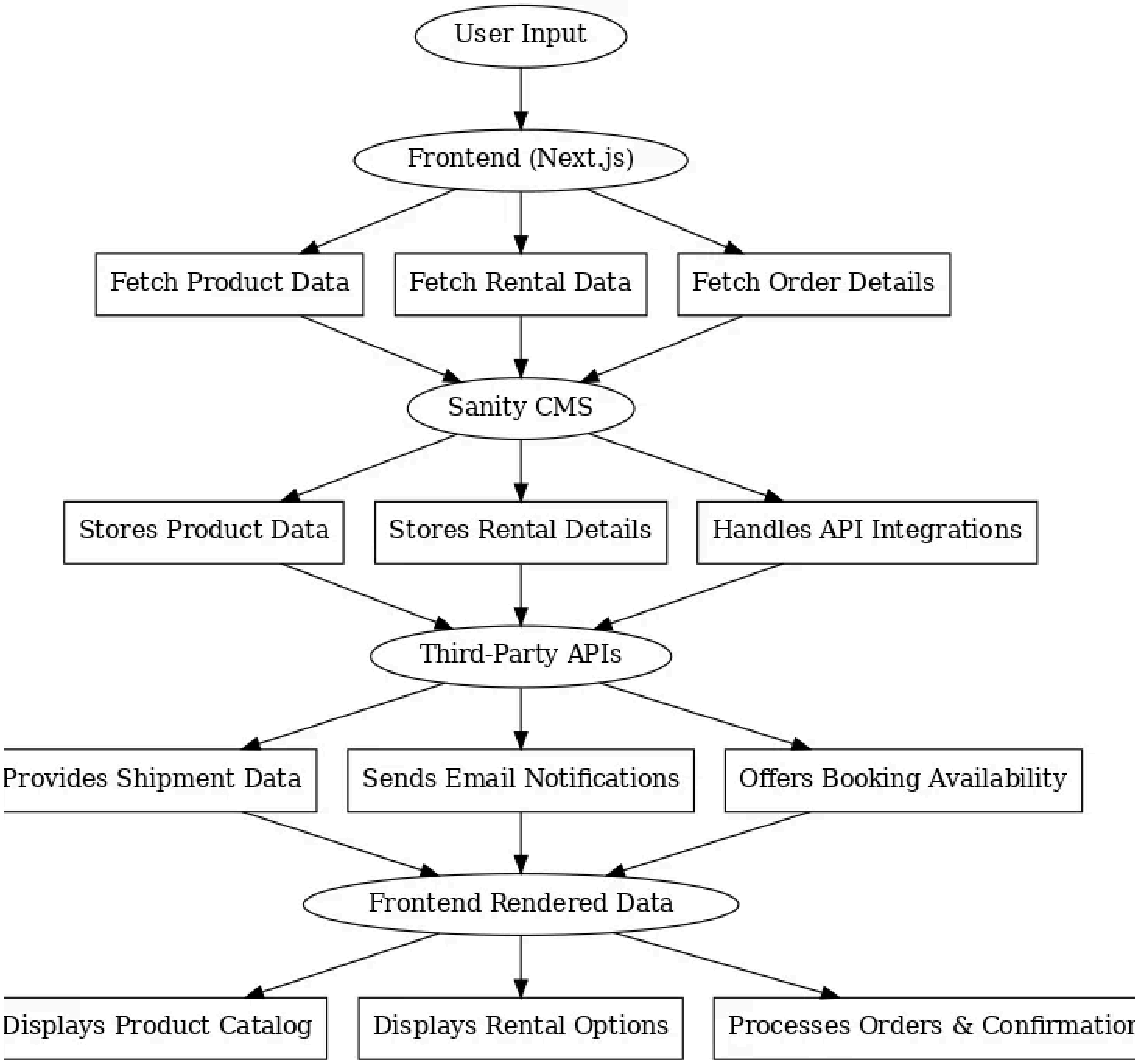
Rental Order Schema:

```
export default {
 name: 'rentalOrder',
 type: 'document',
 fields: [
 { name: 'customer', type: 'reference', to: [{ type: 'customer' }], title: 'Customer' },
 { name: 'helicopter', type: 'reference', to: [{ type: 'rental' }], title: 'Helicopter' },
 { name: 'rentalDuration', type: 'number', title: 'Rental Duration (Hours)' },
 { name: 'total', type: 'number', title: 'Total Amount' },
 { name: 'status', type: 'string', title: 'Rental Order Status' }
 ]
};
};
```

2. System Architecture Design

System Overview Diagram:

```
                    +--------------------+
                    |    User Input     |
                    +--------------------+
                              |
                              v
                  +----------------------+
                  |  Frontend (Next.js)  |
                  +----------------------+
                              |
           +-------+-----------------------------+
                   |                     |
                   v                     v
       +--------------------+    +--------------------+
      | Fetch Product Data |    | Fetch Rental Data  |
       +--------------------+    +--------------------+
                        |              |
                        v              v
             +-------------+      +-------------+
             | Sanity CMS  |      | Sanity CMS  |
             +-------------+      +-------------+
                        |              |
                +-------+-------+          |
                   |        |          |
                   v        v          v
       +--------------------+    +--------------------+
        | API Integration   |    | Third-party APIs  |
      | Shipment + Booking |      + Send Notification  |
```



Workflows:

Product Data Fetching:

1. Frontend requests product data from Sanity CMS.
2. Sanity CMS retrieves and returns the data in JSON format.

Rental Data Fetching:

1. Frontend requests rental helicopter data from Sanity CMS.
2. Sanity CMS retrieves and returns the data in JSON format.

Order Processing:

1. Frontend sends order details to the API endpoint.
2. API saves the order in Sanity CMS.
3. Notification and payment processing APIs are triggered.

Rental Booking Processing:

1. Frontend sends rental booking details to the API endpoint.
2. API saves the booking in Sanity CMS.
3. Notification and payment processing APIs are triggered.

Shipment Integration:

1. Frontend fetches the shipment status from the Shipment Tracking API.
2. Status is displayed to the user on the Order Details page.

3. API Endpoints Plan

Planned Endpoints:

/products **(GET)**

- **Description: Fetch all product details.**
- **Response Example:**

```
[
 {
  "id": "1",
  "name": "Sofa",
  "price": 200.0,
  "stock": 50
 }
]
```

/rentals **(GET)**

- **Description: Fetch all available rental helicopters.**
- **Response Example:**

```
[
 {
  "id": "1",
  "name": "Chairs",
  "rentalPrice": 50,
  "availability": true
 }
]
```

/orders **(POST)**

- **Description: Save new order details.**
- **Request Example:**

```
{
  "customerId": "123",
  "products": ["1", "2"],
  "total": 40.0
}
```

/rentalOrders **(POST)**

- **Description: Save new rental booking details.**
- **Request Example:**

```
{
  "customerId": "123",
  "Chairs": "1",
  "rentalDuration": 5,
  "total": 250
}
```

/shipment **(GET)**

- **Description: Fetch the shipment status for an order.**
- **Response Example:**

```
{
  "orderId": "123",
  "status": "In Transit"
}
```