Day 2
Plannig_Technical_Foundation
Type:
Rental E-Commerce

Define Technical Requirements:

Technical requirements for a rental e-commerce platform include:

User and Inventory Management: A robust system for user authentication, account management, product listings, availability tracking, and real-time updates on rental inventory.

Payment and Logistics Integration: Secure payment gateways, automated billing, rental duration tracking, and integration with shipping/delivery solutions for seamless order fulfillment.

Frontend requirements

Responsive Design: Ensure the platform is accessible across devices (desktop, tablet, mobile) with a user-friendly interface.

Intuitive Navigation: A clear menu structure, search bar with filters, and categories for easy browsing.

Real-Time Updates: Dynamic updates for inventory availability and pricing without needing page refreshes.

Secure User Interaction: Implement secure forms and inputs for logins, registrations, and payments.

SEO and Performance Optimization: Fast loading times and optimized code for search engine visibility.

Essential pages

Home Page: Highlight featured rentals, categories, promotions, and search functionality.

Product Listing Page: Displays available items with filters for price, category, location, and availability.

Product Detail Page: Includes item details, pricing, rental terms, reviews, and booking options.

Cart/Checkout Page: Allows users to review selected items, adjust quantities, and complete payment.

User Dashboard: Manages account details, order history, active rentals, and saved items.

Login/Sign-Up Page: For user authentication and account creation.

FAQ and Support Page: Provides answers to common queries and customer support options.

About Us and Contact Page: Details about the platform and ways to get in touch.

Technical stack

Frontend

Framework/Library:

- React.js (preferred for dynamic UI and component-based architecture)

Styling:

- CSS Frameworks: Tailwind CSS for responsive design

State Management:

- Redux, Context API for managing global state

APIs and Data Fetching:

- Axios or Fetch API for handling HTTP requests

Backend

Framework:

Node.js with APIs

Payment Gateway Integration:

- Stripe, PayPal

Cloud Storage:

- Google Cloud Storage, or Firebase for storing media files like product images

DevOps and Deployment

Hosting:

- Vercel or Netlify for frontend deployment

CI/CD:

- GitHub Actions

Monitoring:

- Google Cloud Monitoring

Backend Sanity Schema

```
// product.js schema for the rental product listing
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'title',
      type: 'string',
      title: 'Product Title'
    },
    {
      name: 'description',
      type: 'text',
      title: 'Product Description'
    },
    {
      name: 'price',
      type: 'number',
      title: 'Rental Price per Day'
    },
    {
      name: 'availability',
      type: 'boolean',
      title: 'Is Available for Rent'
    },
    {
      name: 'image',
      type: 'image',
      title: 'Product Image'
    }
  ]
}
```

Implementation steps

1. Set Up User Authentication & Authorization

- Implement user registration and login with JWT for secure sessions.
- Include role-based access control (admin, user) to manage permissions.

2. Product Management System

- Build backend APIs for CRUD operations on rental products (add, edit, delete, view).
- Create a front-end page for displaying products with availability, pricing, and details.
- Set up inventory management logic to track the status of rented items.

3. Implement Checkout & Payment Integration

- Create a shopping cart system
- Integrate a payment gateway like Stripe or PayPal for seamless transactions.
- Build a checkout page to review the order, including rental dates and final price, before payment.

Additional APIs

Rental Availability API

- **Endpoint**: GET /api/products/availability/{productId}
- **Description**: Returns the availability status of a product for specific rental dates (e.g., if the item is available during a selected rental period).
- **Parameters**: startDate, endDate
- **Response**: { "available": true/false, "nextAvailableDate": "YYYY-MM-DD" }

Order History API

- **Endpoint**: GET /api/orders/history/{userId}
- **Description**: Fetches the past rental orders for a user, including rented items, rental duration, status, and total amount.
- **Parameters**: userId
- **Response**: An array of orders with details like rental item, dates, price, and status.

Admin Product Management API

- **Endpoint**: POST /api/admin/products
- **Description**: Allows the admin to add or update products in the catalog.
- **Parameters**: title, description, price, category, availability, imageUrl
- **Response**: { "message": "Product added successfully", "productId": "12345" }

System Architecture Data FlowChart

```
+--------------------+         +------------------+          +--------------------+
|   Frontend (UI)    | <---> |   Backend API       | <---> |      Database        |
|                    |       |                     |       |                      |
| 1. Product Listing |       | 1. Fetch Products   |       | 1. Product Data      |
| 2. Add to Cart     |       | 2. Add/Update Cart  |       | 2. User Data         |
| 3. Checkout        |       | 3. Checkout Order   |       | 3. Orders            |
|                    |       | 4. User Auth        |       | 4. Payment Status    |
+--------------------+         +------------------+          +--------------------+
          |                             |                             |
          |                             |                             |
          v                             v                             v
  +--------------------+              +------------------+        +------------------+
  |  Payment Gateway     | <---> |  Order Processing| <----> | Cloud Storage      |
  |  (Stripe/PayPal)     |       |  & Transaction   |        | (Product Images) |
  |  1. Process Payment  |       |  1. Order Status |        |1. Upload Images |
  |  2. Send Confirmation|       |  2. Rental Period |        +------------------+
     +--------------------+       +------------------+
```