Date:

Day 2 Task
Hackathon 3

Planning The Technical Foundation

1. User Visit website.
at first time when User come to the website frontend frame work (next.js)

2. Fronted interacts with Sanity CMS.
data placed at Sanity (head less CMS) with images and all details.

3. Product data is displayed product.
display Sanity data on fronted dynamically.

4. User adds items to cart.
~~when User like something~~ when a User adds item to their cart, it means they are selecting products they intend to purchase later.

own Page,

furniture

you can
handle

Key Components

- Pages
- Components
- API Routes
- Static Generation
- Image Optimization
- Static Generation
- Dynamic Routing
- Styling
- State Management

Flow Chart Plan

The flow chart will include:

1. User action (e.g; browsing, adding to cart, checkout).
2. Interaction with the frontend.
3. Backend processes in Sanity CMS.
4. Third party API intergrations for shipment tracking and payment Gateway.
5. The User proceeds to checkout.
6. Order details sent to Sanity CMS.
7. Payment Gateway process payment

Payment process go through from gateway (stripe, Paypal) Customer receives receipt after confirmation.

8. Shipment tracking

After order shipment status fetches from, third party (API) informs Users about their Orders.

9. Display shipment status

Status will be shown on frontend is real time status like "In Transit" or "Delivered".

Date:

For FrontEnd, i will be using Next.Js

For BackEnd, i will be using Sanity

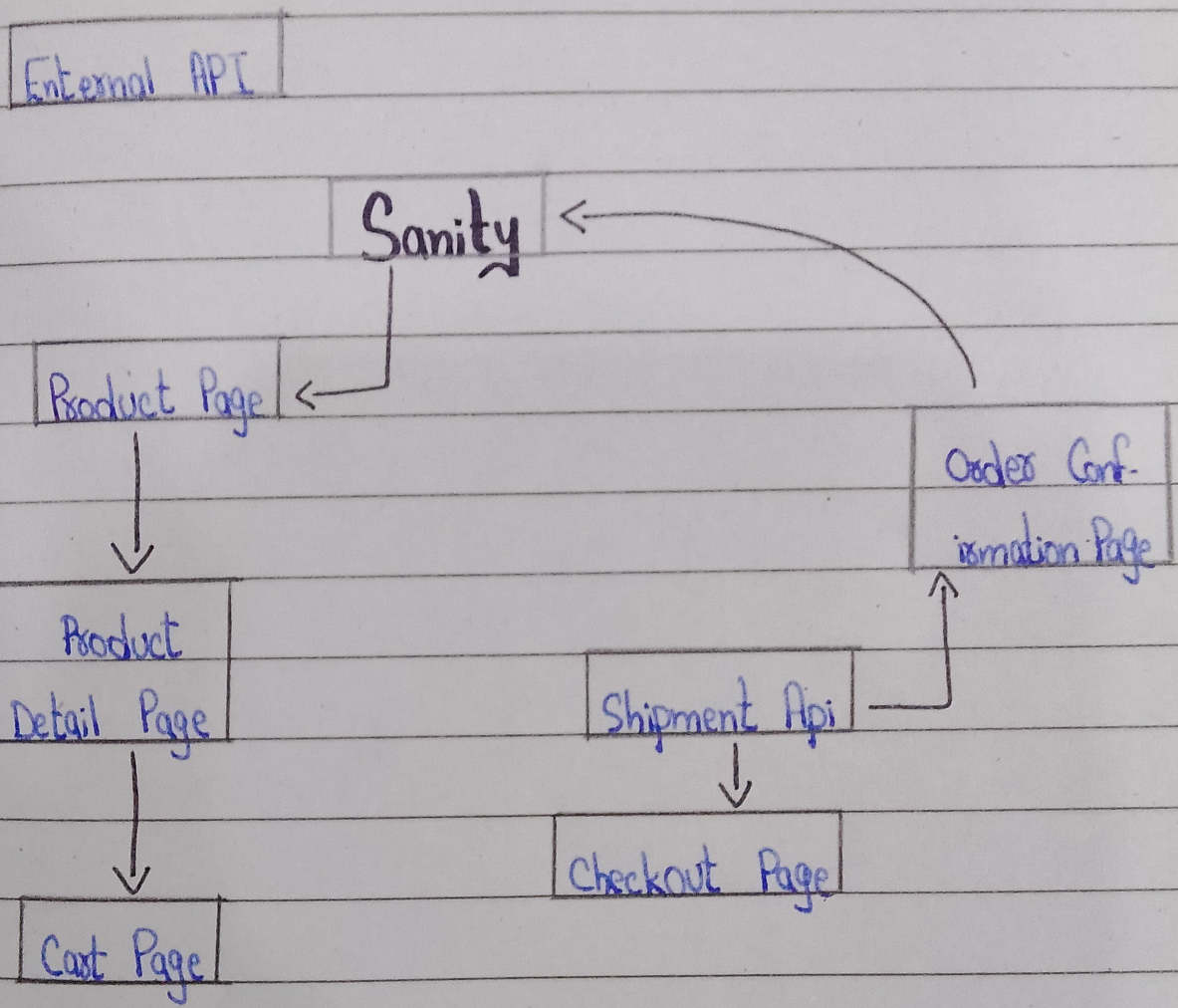
In my E-Commerce website, i have multiple page which are listed;

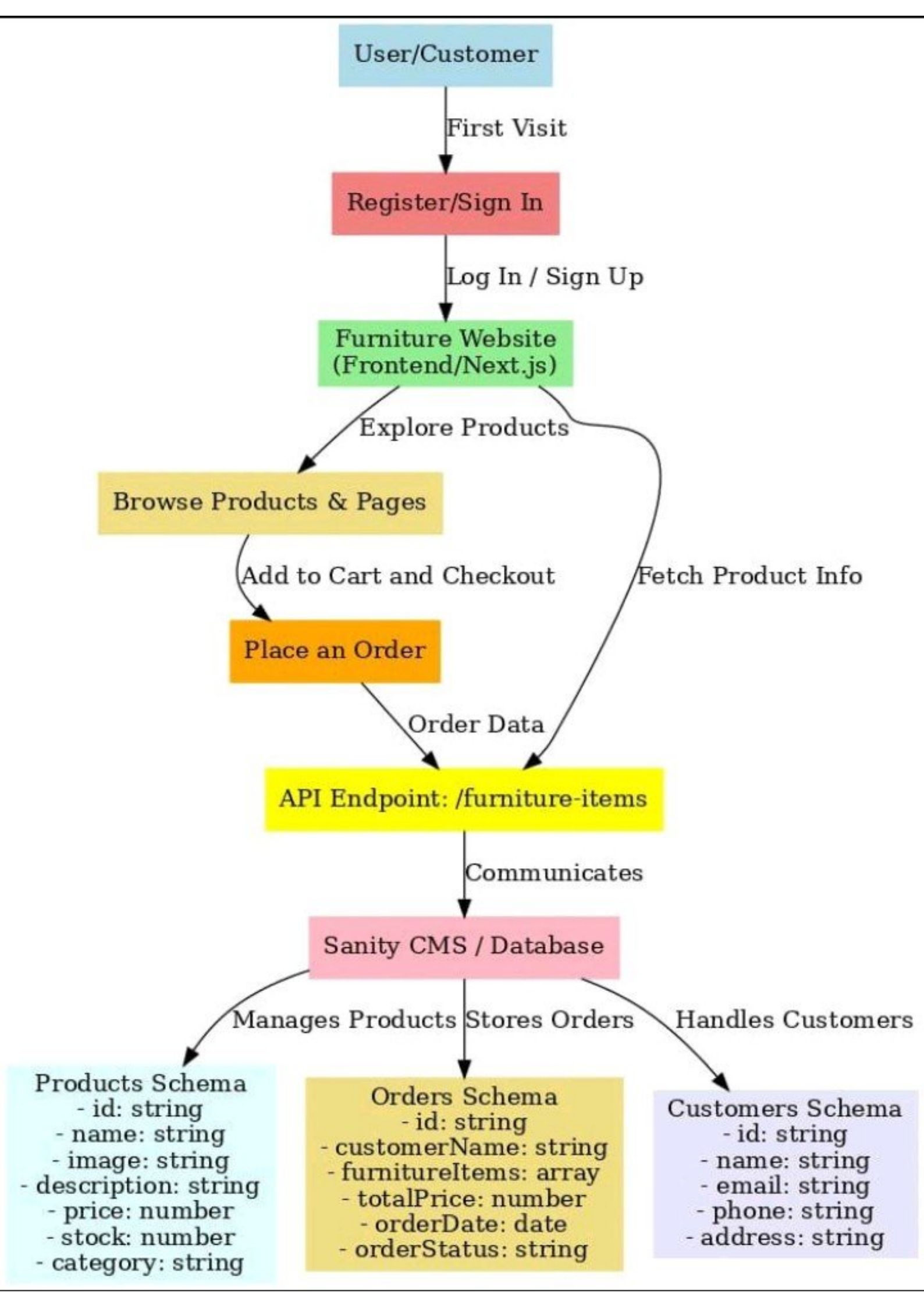
Home Page	Product Page	About Page
Cart Page	Contact Page	Product detail Page
Checkout Page	Order Confirmation Page	

Date: _____

Workflow Of My Website

Import Data Into Sanity from External API




User/Customer
First Visit
Register/Sign In
Log In / Sign Up
Furniture Website (Frontend/Next.js)
Explore Products
Browse Products & Pages
Add to Cart and Checkout
Fetch Product Info
Place an Order
Order Data
API Endpoint: /furniture-items
Communicates
Sanity CMS / Database
Manages Products
Stores Orders
Handles Customers
Products Schema
- id: string
- name: string
- image: string
- description: string
- price: number
- stock: number
- category: string
Orders Schema
- id: string
- customerName: string
- furnitureItems: array
- totalPrice: number
- orderDate: date
- orderStatus: string
Customers Schema
- id: string
- name: string
- email: string
- phone: string
- address: string