Day 2 : Planning the Technical Foundation

Day 2 Goal :

Step 1 :

Define Technical Requirements

Frontend Requirements :

You need to create a clean, initiative and user friendly design, which should include the following pages

- Home Page : Display featured products , categories and promotions
- Product Listing : This page will show products based on categories , where uses can filter and sort products
- Cart: The car page allow user to view their selected items, adjust quantities and calculate the total price
- Checkout: The final page where use will fill in their address, payment details and confirm the order

Sanity CMS Integration:

Sanity CMS will be integrated for content Management which will handle product and orders

Products: Use CMS to manage product data sanity structure will allow to update and manage data

Orders: CMS will be configured to leak user orders.

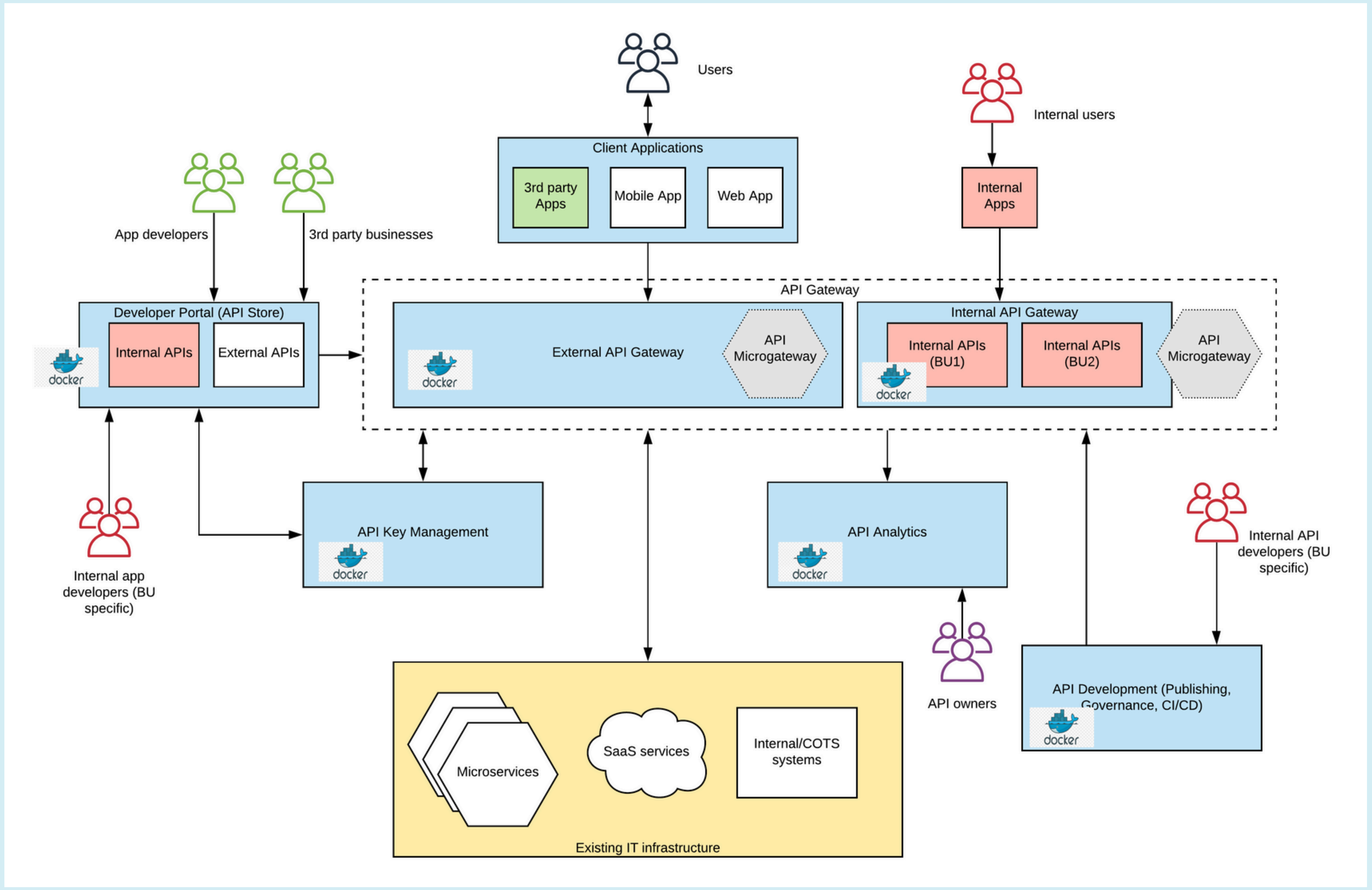
Third-Party Api’s Integration:

Need to integrate external services that are used for Payment and shipment tracking

- Payment Gateway: For integration you will connect to the payment gateway API to handle authentication, transaction and orders.
- Shipment Tracking: Need to integrate a shipment tracking services e.g (ShipEngine , AfterShip) so user can track the shipping staturs of their orders.

System Architecture

Step 2 : Diagram



Component Interactions

Frontend (Next.Js):

It will directly interact with CMS and API

Sanity CMS:

They fetch product related information from the product data api

Product Data Api:

This could be external Api that provides detailed product data

Third-Party Api:

This Api will handle shipment tracking and other relevant service

Shipment Tracking API:

This specific Api will manage shipment related data such as order delivery status

Payment Gateway:

This will handle the payment process where user select their payment method and the payment is authorized.

Step 3:
Api Requirements

Explanation:

- /products(Get): This point-end fetches a list of all product with optional filters for category pagination (limit-page).
- /product/[id](get): Fetches details of a specific product by its ID
- /product (post): Allow the creation of all product with required details(name,description,price,category).
- /product/[id](put): Update the details a specific product by it’s Id
- /product/[id](Delete):Deletes a product by it ID

Api Schema Example

```
export default {
name: "Product",
type: "document",
title: "Product",
fields: [
{
name: "name",
type: "string",
title: "Product name"
},
{
name: "price",
type: "number",
title: "Price"
},
{
name: "description",
type: "text",
title: "Description"
}
]
}
```

Step 4: Documentation

1. System Architecture

This system architecture defines the structure and interaction of different components:

- Frontend Layer:
- UI Components
- User Authentication
- Request Handling
- API Communication with Backend
- Backend Layer:
- RESTful APIs
- Business Logic
- Database Interaction
- User Management
- Database Layer:
- Relational/NoSQL Database for User Data etc.
- Third-Party Services:
- Email, SMS, Payment Gateway, etc.
- Communication:
- API Calls between Frontend and Backend
- Database Queries
- External Service Communication

2. API Requirements

Defines necessary APIs for communication between the frontend and backend:

- Authentication:
- POST /api/login: User Login
- POST /api/register: Register a User
- POST /api/logout: Logout User
- User Profile API:
- GET /api/users/profile: Get User Profile
- PUT /api/users/profile: Update User Profile
- Data Retrieval API:
- GET /api/data/items: Get List of Items
- GET /api/data/items/{id}: Get Item by ID

3. Workflow Diagrams

Shows the interaction flow between system components:

- Login Flow:
- User Enters Credentials
- Frontend Calls /api/login
- Backend Authenticates and Returns a Token
- Frontend Stores Token and Redirects

4. Data Retrieval Flow

- Frontend Requests Data via /api/data/items
- Backend Fetches Data and Responds
- Frontend Displays Data
- Error Handling Flow:
- Backend Returns Code Errors
- Frontend Displays User-Friendly Error Messages

5. Sanity Schema

Defines the data structure for the system:

User Schema Item Schema
User Item
Userid Itemid
Username Name
Password Price
Email Category
CreatedAt Stock
UpdatedAt CreatedAt
UpdatedAt

MADE BY
NABILA BANNAY KHAN