**Day 3 - API Integration Report**  
**Title:** API Integration and Data Migration

**1. API Integration Process**

* Imported Required Modules:
  + createClient from @sanity/client for interacting with Sanity.
  + axios for API requests.
  + dotenv to load environment variables.
  + fileURLToPath and path for file system operations.
* **Loaded Environment Variables:**
  + Used .env.local to securely load variables like API keys or project IDs.
* const \_\_filename = fileURLToPath(import.meta.url);
* const \_\_dirname = path.dirname(\_\_filename);
* dotenv.config({ path: path.resolve(\_\_dirname, '../.env.local') });
* **Created Sanity Client:**
  + Configured the Sanity client to use environment variables for secure and flexible integration.
* const client = createClient({
* projectId: process.env.NEXT\_PUBLIC\_SANITY\_PROJECT\_ID,
* dataset: process.env.NEXT\_PUBLIC\_SANITY\_DATASET,
* useCdn: false,
* token: process.env.NEXT\_PUBLIC\_SANITY\_AUTH\_TOKEN,
* apiVersion: '2021-08-31'
* });
* **Image Upload Function:**
  + Wrote a function to upload product images to Sanity's asset library.
* async function uploadImageToSanity(imageUrl) {
* try {
* console.log(`Uploading image: ${imageUrl}`);
* const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
* const buffer = Buffer.from(response.data);
* const asset = await client.assets.upload('image', buffer, {
* filename: imageUrl.split('/').pop()
* });
* console.log(`Image uploaded successfully: ${asset.\_id}`);
* return asset.\_id;
* } catch (error) {
* console.error('Failed to upload image:', imageUrl, error);
* return null;
* }
* }
* **Data Import Function:**
  + Created a function to import data from the API into Sanity.
* async function importData() {
* try {
* console.log('Migrating data, please wait...');
* // API endpoint containing product data
* const response = await axios.get('https://template-03-api.vercel.app/api/products');
* const products = response.data.data;
* for (const product of products) {
* let imageRef = null;
* if (product.image) {
* imageRef = await uploadImageToSanity(product.image);
* }
* const existingProduct = await client.fetch(
* `\*[\_type == "product" && productName == $productName][0]`,
* { productName: product.productName }
* );
* if (existingProduct) {
* // Update the slug for existing products
* await client.patch(existingProduct.\_id).set({
* slug: {
* \_type: 'slug',
* current: product.productName.toLowerCase().replace(/\s+/g, '-'),
* },
* }).commit();
* console.log(`Updated slug for product "${product.productName}"`);
* } else {
* // Create new product if it doesn't exist
* const sanityProduct = {
* \_type: 'product',
* productName: product.productName,
* category: product.category,
* price: product.price,
* inventory: product.inventory,
* colors: product.colors || [], // Optional, as per your schema
* status: product.status,
* description: product.description,
* image: imageRef
* ? {
* \_type: 'image',
* asset: {
* \_type: 'reference',
* \_ref: imageRef,
* },
* }
* : undefined,
* slug: {
* \_type: 'slug',
* current: product.productName.toLowerCase().replace(/\s+/g, '-'),
* },
* };
* await client.create(sanityProduct);
* console.log(`Added new product "${product.productName}"`);
* }
* }
* } catch (error) {
* console.error('Error in migrating data:', error);
* }
* }
* **API Integration:**
  + Successfully integrated the API with URL:

https://template-03-api.vercel.app/api/products.

* + Imported data into Sanity and verified successful insertion.

**2. Adjustments Made to Schemas**

* Added slug field for dynamic routing in the existing schema.

slug: {

type: 'slug',

options: {

source: 'productName',

maxLength: 200,

},

},

**3. Migration Steps and Tools Used**

* **Tools Used:**
  + **Sanity Client:** For creating and managing documents in Sanity CMS.
  + **Axios:** To fetch data from the external API.
  + **Dotenv:** To load sensitive configuration details.
  + **Custom Scripts:** For data transformation and migration.
* **Steps Followed:**
* Connected to the API and fetched product data using Axios.
* Uploaded product images to Sanity CMS and obtained their asset references.
* Created new product documents in Sanity using the fetched data and schema adjustments.
* Verified successful migration by checking the Sanity Studio dashboard.

Day 3 Checklist:

Self-Validation Checklist:

API Understanding: ✔

Schema Validation: ✔

Data Migration: ✔

API Integration in Next.js: ✔

Submission Preparation: ✔

=================xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx=======================