Day 3 - API Integration & Data Migration

Objective:

Integrate APIs and migrate data into Sanity CMS to build a functional marketplace backend, preparing for real-world client requirements.

Key Learning Outcomes:

1. Integrate APIs into Next.js.
2. Migrate data from APIs to Sanity CMS.
3. Utilize eCommerce platform data.
4. Validate schemas for compatibility.

API Overview:

Provided read-only APIs for schema validation and migration.:

- Use these APIs for data population.
- Import data manually.
- Utilize external APIs or data sources.

Steps:

1. Understand the API - Review documentation and identify key endpoints.
2. Validate & Adjust Schema - Align Sanity CMS schema with API data structure.
3. Data Migration Methods:
    - Use API scripts for fetching and transforming data.
    - Import manually via JSON.
    - Utilize external platform APIs.
4. API Integration in Next.js:
    - Create utility functions.
    - Render data in components.
    - Test endpoints using Postman.

Best Practices:

- Backup data before migration.
- Validate imported data and adjust schemas accordingly.
- Use `.env` for API keys.
- Follow clean coding practices.
- Document and test thoroughly.

Expected Output:

1. Populated Sanity CMS with imported data.

2. Functional API integration displaying products, categories, etc.

Submission Requirements:

- Report detailing API integration, schema adjustments, and migration steps.
- Screenshots of API calls, populated CMS, and frontend display.
- Code snippets for API integration and migration scripts.

Issues & Solutions

1. ReferenceError: `__dirname` is not defined

Cause:

- `__dirname` is unavailable in ES modules (ESM) when `"type": "module"` is set in `package.json`.

Solution:

- Use `import.meta.url` instead of `__dirname`:
- `import { fileURLToPath } from 'url';`
- `import { dirname } from 'path';`
-
- `const __filename = fileURLToPath(import.meta.url);`
  `const __dirname = dirname(__filename);`

---

2. Missing `tmp` Directory for Image Storage

Cause:

- The script attempts to save images to `C:\tmp\image`, but the directory does not exist.

Solution:

- Ensure the `tmp` directory exists before saving files:
- `import fs from 'fs';`
- `const dir = './tmp';`
- `if (!fs.existsSync(dir)) {`
- `    fs.mkdirSync(dir);`
  `}`

---

3. Undefined Image Property during Import

Cause:

- The `image_url` may be incorrect or `imageRef` is not assigned properly.

Solution:

1. **Debug the `image_url` before upload:**

```
console.log("Uploading Image URL:", image_url);
```

2. **Check the `uploadImageToSanity` function:**

```
console.log("Image Reference:", imageRef);
```

3. **Ensure `imageRef` is assigned correctly:**

```
if (!imageRef) {
    console.error("Error: Image reference is undefined");
}
```

6. **Verify image schema:**

```
image: {
    _type: "image",
    asset: {
        _type: "reference",
        _ref: asset._id,
     },
}
```

13. **Confirm image URL format:**
    - Ensure URLs are properly formatted before processing.

**4. CommonJS vs. ESM Compatibility Issue with **`@sanity/client`

Cause:

- TypeScript files are compiled as CommonJS modules, but `@sanity/client` is an ESM package.

Solution:

Convert to ESM properly

1. **Update `package.json`:**

```
{
  "type": "module"
}
```

4. **Fix imports in `sanityClient.ts`:**

```
import { createClient } from "@sanity/client";
```

5. **Update `tsconfig.json`:**

```
{
  "compilerOptions": {
    "module": "ESNext",
    "moduleResolution": "node"
  }
}
```

 Alternative Fix (Stick with CommonJS):

- Remove `"type": "module"` from `package.json`.
- Use `require()` instead of `import`:

```
const sanityClient = require("@sanity/client");
```

Final Thoughts

By addressing these issues, i successfully import data into Sanity without encountering `__dirname` errors, missing image uploads, or module format conflicts.