**Python Project: Efficient Management of**

**E-commerce Product Information**

**\*\*Introduction\*\***

Efficient management of product information is essential for the success of e-commerce businesses. This information enables informed decision-making and facilitates effective marketing strategies. File handling plays a key role in managing such data, and Python allows us to handle various file formats commonly used in the industry.

In this project, I developed a Python-based solution to load, modify, and save data in three popular file formats:

CSV: Used to store sales values for each product over 14 days.

JSON: Used to store detailed attributes of products such as product names, prices, and brands.

TXT: Used to store textual descriptions of products.

**\*\*Problem Statement\*\***

The goal of the project was to read data from multiple file formats (.csv, .txt, .Json), perform operations on the data, and save the modified data back to the original formats.

**\*\*Dataset and File Structure\*\***

The dataset represents product information identified by a unique \*\*SKU (Stock Keeping Unit) \*\*. It consists of the following components:

1. Sales Data (sales\_data.csv) \*\*: A CSV file containing 14 days of sales data for various products. Each row represents a product identified by its SKU.

2. Product Details (product details folder) \*\*: JSON files containing detailed attributes of each product. Each file is named after the SKU (e.g., `details\_AISJDKFJW93NJ.json`).

3. Product Descriptions (product descriptions folder) \*\*: TXT files containing descriptions of each product. Each file is named after the SKU (e.g., `description\_AISJDKFJW93NJ.txt`).

**\*\*Project Tasks\*\***

**\*\*1. Environment Setup\*\***

The required packages and modules were imported:

- \*\*os\*\*: For navigating file directories.

- \*\*Json\*\*: For handling JSON files.

- \*\*csv\*\*: For handling CSV files.

**\*\*2. Loading Data\*\***

A function named `load data()` was implemented to:

- Read sales data from the `sales\_data.csv` file.

- Load product details from JSON files in the `product details` folder.

- Load product descriptions from TXT files in the `product descriptions` folder.

The data was stored in the following dictionaries:

- `sales data`

- `product details`

- `product descriptions`

**\*\*3. Adding and Updating Data\*\***

Functions were created to enable the admin to add or update product information:

- \*\*`update\_sales\_data ()`\*\*: Adds or updates sales data for a product.

- \*\*`update\_product\_details ()`\*\*: Adds or updates detailed attributes of a product.

- \*\*`update\_product\_description () `\*\*: Adds or updates a product description.

- \*\*`update () `\*\*: A comprehensive function to add or update product information, ensuring user inputs meet specified validation criteria:

- SKU must be exactly 13 characters long.

- Sales data must contain exactly 14 numeric values.

- Product details and descriptions are required fields.

**\*\*4. Saving Updated Data\*\***

The `dump data () ` function was implemented to save updated data:

- \*\*Sales Data\*\*: Overwritten in `sales\_data.csv`.

- \*\*Product Details\*\*: Stored in individual JSON files in the `product details` folder.

- \*\*Product Descriptions\*\*: Stored in individual TXT files in the `product descriptions` folder.

The function automatically creates folders and subfolders if they do not exist.

**\*\*Example Implementation\*\***

An example use case to add data for a new product:

- Product SKU: `CMWKCILOP27KF`

- Sales Data: ` [8, 14, 16, 7, 15, 21, 14, 16, 32, 29, 26, 30, 25, 22]`

- Product Details:

- Name: "Pokémon Card"

- Brand: "GameFreak"

- Model: "ScarletViolet151"

- Specifications: "Genuine, TCG, English"

- Price: "$1.99"

- Availability: "In stock"

- Product Description: "Original Pokémon TCG Pikachu card"

**\*\*Key Features\*\***

1. Seamless data handling across multiple file formats.

2. User-friendly functions for adding and updating product data.

3. Validation mechanisms to ensure data integrity.

4. Automated folder creation for saving files.

**\*\*Technologies Used\*\***

- \*\*Python\*\*: For scripting and file handling.

- \*\*Libraries\*\*: os, Json, csv.

**\*\*Conclusion\*\***

This project highlights the use of Python in managing e-commerce data efficiently. By enabling seamless handling of CSV, JSON, and TXT files, the solution simplifies the process of managing product information and ensures data integrity. This project serves as a robust foundation for more advanced e-commerce data management systems.