

**Sultan Moulay Slimane University
Polydisciplinary Faculty of Khouribga
Mathematics and Computer Science Department**

Major: Computer Engineering and Artificial Intelligence
Module:NoSQL Databases

MINI-PROJECT REPORT :

Multi-channel product catalog

(MongoDB, Python, Streamlit, Pandas)

Author:

[HMADNA AYMANE]

Profil GitHub:

<https://github.com/AymaneHmadna>

Academic Year: 2025-2026

Contents

1 Project Objective :	2
2 Technology Stack Description :	2
2.1 MongoDB (Database) :	2
2.2 Python & Pandas (Processing) :	2
2.3 Streamlit (User Interface) :	2
3 Infrastructure and Modeling :	3
4 Execution Process and Proofs :	3
4.1 Data Ingestion :	3
4.2 Verification in MongoDB Compass :	4
4.3 Index creation :	5
4.4 Application Launch :	6
4.5 Catalog Interface :	7
4.6 Quality workflow :	8
4.7 Manual addition of a product :	9
4.8 Manual entry validation :	10
4.9 Traceability :	10
4.10 Multi-channel export :	11
4.11 Verification of generated files :	12
5 Conclusion and perspectives :	14

1 Project Objective :

This project aims to design a **Multi-channel product catalog** system capable of centralizing, cleaning and distributing heterogeneous product data. In a modern ecommerce context, a brand must sell on several channels (Website, Marketplace, Social media).

The main objectives are :

- **Centralization** : Create a unique repository with a document oriented database (mongoDB).
- **Data Quality** : Implement a validation workflow (Draft vs Published) to prevent the diffusion of incomplete data.
- **Flexibility** : Manage products with variable attributes (Clothing vs Electronics) thanks to JSON polymorphism.
- **Multi-Channel Distribution** : Export data in the form of adapted structured feeds (nested JSON for the Web, "flattened" CSV for Marketplaces).

2 Technology Stack Description :

Each component has been selected to meet the flexibility and performance constraints of the specifications.

2.1 MongoDB (Database) :

Document oriented NoSQL database.

- **Role** : Flexible storage of product sheets.
- **Why MongoDB** ? Unlike rigid SQL, mongoDB allows storing very different products (e.g., T-shirt with "size/color" and Phone with "storage/warranty") in the same collection without multiplying empty tables.

2.2 Python & Pandas (Processing) :

- **Role** : Data manipulation and transformation for export.
- **Why Pandas** ? The `json_normalize` function is crucial to "flatten" nested JSON structures in order to generate CSV files compatible with excel.

2.3 Streamlit (User Interface) :

- **Role** : Management interface for the product manager.
- **Why Streamlit** ? Allows rapid development of an interactive CRUD (Create, Read, Update, Delete) interface with immediate data visualization.

3 Infrastructure and Modeling :

The architecture relies on three main Python scripts :

- **data.py** : Simulates data ingestion. This script generates 50 realistic products with random attributes and inserts them directly into MongoDB. This allows testing the load and indexes immediately.
- **index.py** : Administration script that creates the required indexing structures :
 - *Unique Index (SKU)* : Guarantees that no logistical duplicate is created.
 - *Text Index (Name, Brand)* : Activates the performant search engine.
- **app.py** : The heart of the system. It manages :
 - The connection to MongoDB via `pymongo`.
 - The business logic (Quality rules `check_quality`).
 - The user interface (Catalog, Add New, Export, Logs).

4 Execution Process and Proofs :

4.1 Data Ingestion :

Execution of the `data.py` script to populate the database. The script connects to mongoDB, cleans the existing collection and generates a batch of 50 products (Electronics and Clothing) with varied attributes.

```
python data.py
```

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Catalogue produits multi canaux> python data.py
Tentative de connexion...
SUCCESS: Connecté à MongoDB !
Database Cleared.
Generating 100 Products...
50 Products inserted.
PS C:\Catalogue produits multi canaux> |

```

Figure 1: Success of data insertion via the python script.

4.2 Verification in MongoDB Compass :

We verify that the documents are well stored.

The screenshot shows the MongoDB Compass application interface. On the left, there's a sidebar with 'Compass' branding, 'My Queries', 'Data Modeling', and a 'CONNECTIONS (1)' section listing 'NEO' (selected), 'Catalogue_produits_multi_canaux' (selected), 'products' (selected), and other options like 'admin', 'config', and 'local'. The main area has a dark header with 'NEO > Catalogue_produits_multi_canaux > products'. Below the header, there are tabs for 'Documents' (50), 'Aggregations', 'Schema', 'Indexes (1)', and 'Validation'. A search bar says 'Type a query: { field: 'value' } or Generate query'. There are buttons for 'Find', 'Explain', 'Reset', and 'Options'. Below the search bar, there are icons for creating a new document, deleting, and more. The results pane shows three examples of the document structure:

```

_id: ObjectId('6989204d9deed50689cc20a3')
sku: "SKU-1001"
name: "XPS 13 Plus"
category: "Electronics"
status: "DRAFT"
price: 1431
attributes: Object

_id: ObjectId('6989204d9deed50689cc20a5')
sku: "SKU-1002"
name: "Galaxy Buds2 Pro"
category: "Electronics"
status: "DRAFT"
price: 229
attributes: Object

_id: ObjectId('6989204d9deed50689cc20a7')
sku: "SKU-1003"
name: "Adidas Puffer Jacket"
category: "Clothing"
status: "PUBLISHED"
price: 152
attributes: Object

```

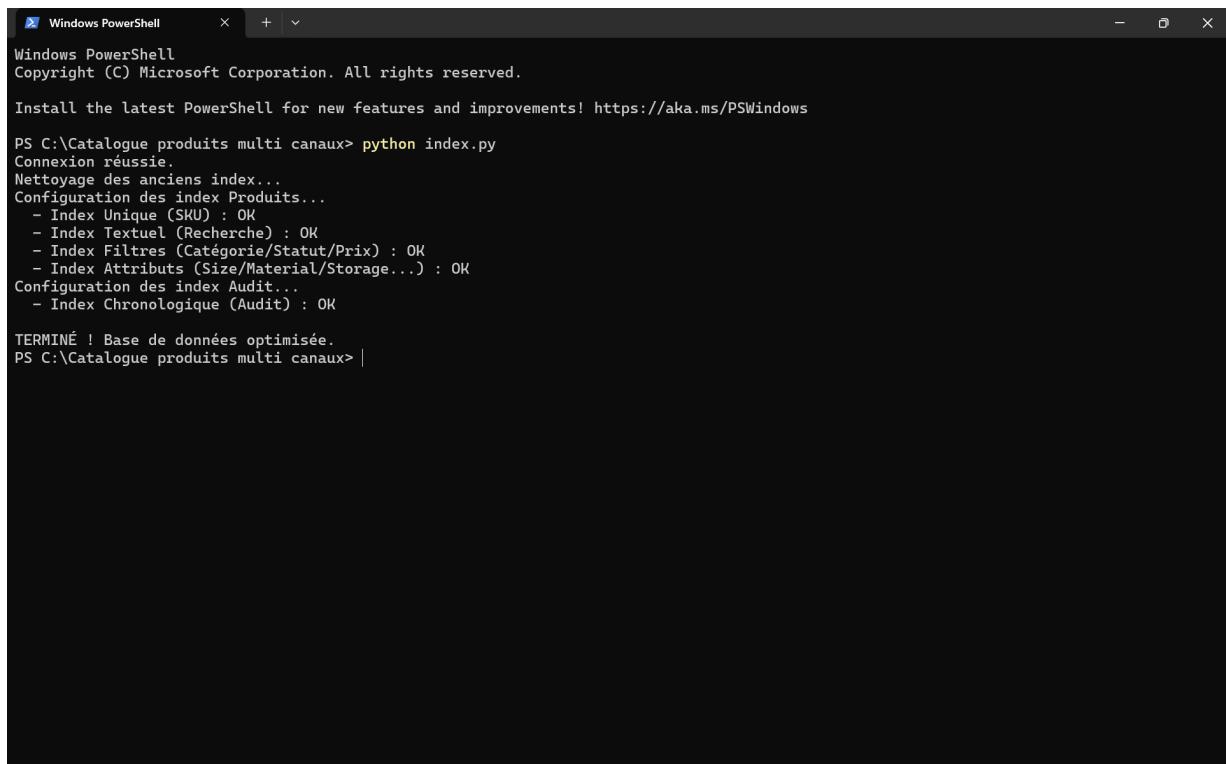
Figure 2: 50 documents inserted in the products collection.

4.3 Index creation :

Launch of the `index.py` script.

```
python index.py
```

- A **UNIQUE** index on the `sku` field to prevent duplicates.
- A **TEXT** index on `name` and `brand` for the search bar.



The screenshot shows a Windows PowerShell window titled "Windows PowerShell". The command `python index.py` is run from the path `C:\Catalogue produits multi canaux>`. The output indicates a successful execution, showing the configuration of various indexes: Unique (SKU), Textuel (Recherche), Filtres (Catégorie/Statut/Prix), Attributs (Size/Material/Storage...), and Chronologique (Audit). The process is completed with the message "TERMINÉ ! Base de données optimisée."

Figure 3: Terminal showing the successful creation of the different indexes.

The screenshot shows the MongoDB Compass interface. The left sidebar displays connections, with 'NEO' selected, showing databases like 'Catalogue_produits_multi_canaux', 'audit', and 'products'. The main area shows the 'products' collection with 50 documents. The 'Indexes' tab is active, displaying 11 indexes. The table lists the following indexes:

Name & Definition	Type	Size	Usage	Properties	Status
id	REGULAR	32.8 kB	1 (since Mon Feb 09 2026)	UNIQUE	READY
sku_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)	UNIQUE	READY
name_text_attributes.brand_text	TEXT	20.5 kB	0 (since Mon Feb 09 2026)	COMPOUND	READY
category_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
status_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
price_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
attributes.brand_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
attributes.color_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
attributes.size_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
attributes.material_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY
attributes.storage_1	REGULAR	20.5 kB	0 (since Mon Feb 09 2026)		READY

Figure 4: Visual confirmation of the indexes in compass.

4.4 Application Launch :

Starting the Streamlit server. The application is accessible locally on port 8501.

```
python -m streamlit run app.py
```

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Catalogue produits multi canaux> python index.py
Connexion réussie.
Nettoyage des anciens index...
Configuration des index Produits...
- Index Unique (SKU) : OK
- Index Textuel (Recherche) : OK
- Index Filtres (Catégorie/Statut/Prix) : OK
- Index Attributs (Size/Material/Storage...) : OK
Configuration des index Audit...
- Index Chronologique (Audit) : OK

TERMINÉ ! Base de données optimisée.
PS C:\Catalogue produits multi canaux> python -m streamlit run app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.11:8501

```

Figure 5: Starting the streamlit server.

4.5 Catalog Interface :

The home page displays the product list. We can clearly see the products in [LIVE] and those in [DRAFT]. The search bar uses the previously created text index.

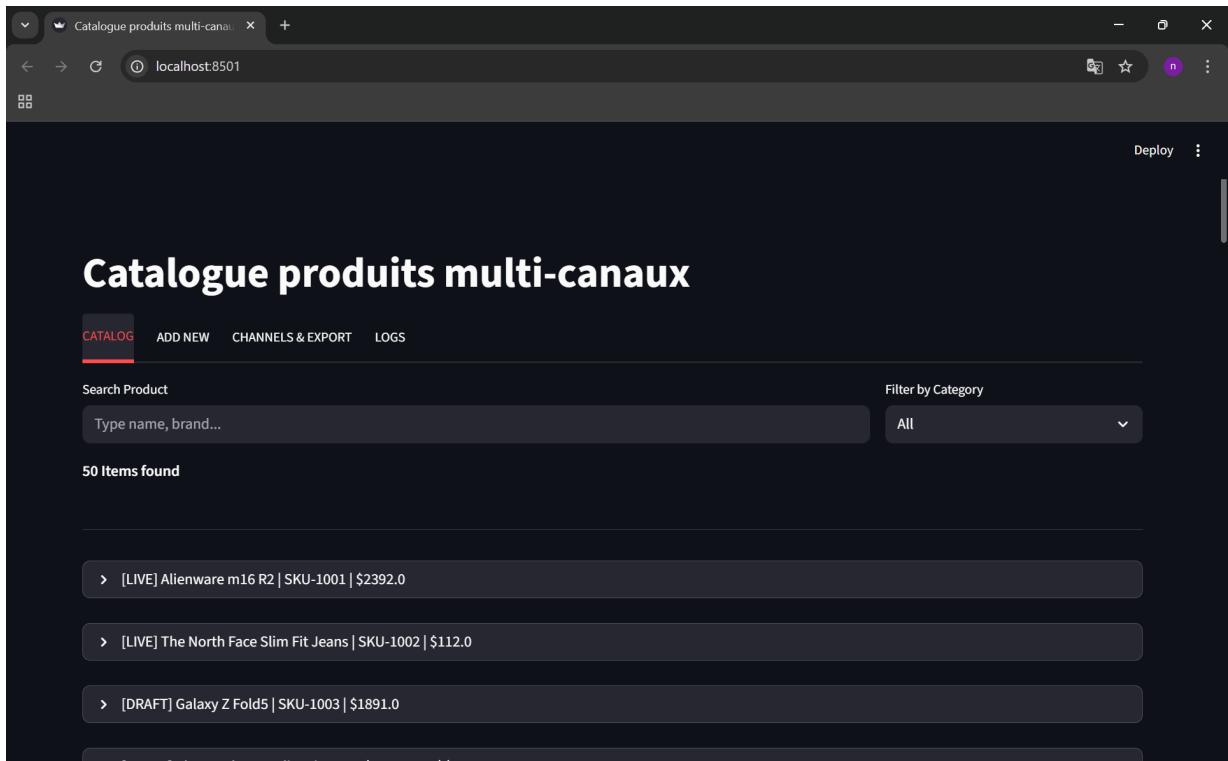


Figure 6: Global view of the multi-channel product catalog.

4.6 Quality workflow :

Demonstration of the quality rule. **Error case** : Below, an incomplete product. The "Storage" field is missing for an electronic product. The system blocks the publication and displays a red alert.

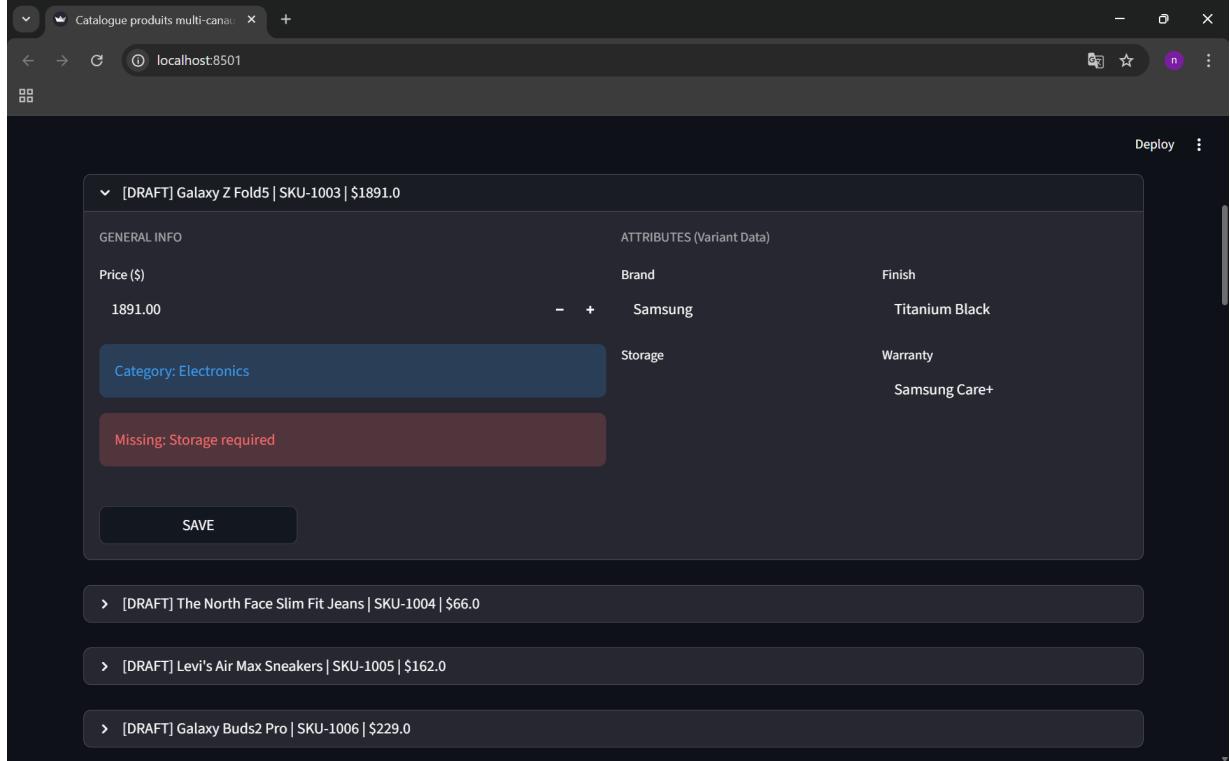


Figure 7: Failed validation : The publish button is missing, the error is explicit.

Valid case : Once the data is corrected, the red banner disappears and the "**PUBLISH TO CHANNELS**" button becomes available.

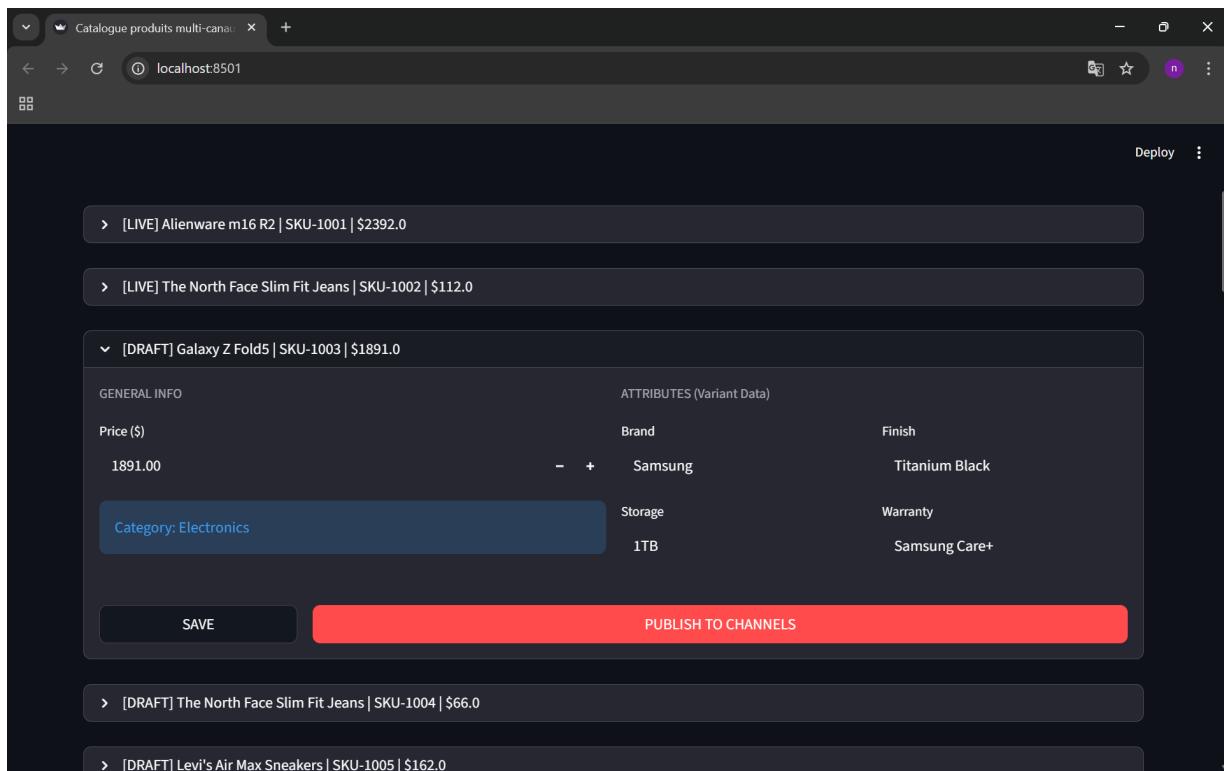


Figure 8: Successful validation : The product is ready to be broadcasted.

4.7 Manual addition of a product :

The "ADD NEW" tab allows business teams to create a product manually. The system generates a draft that will then need to be validated.

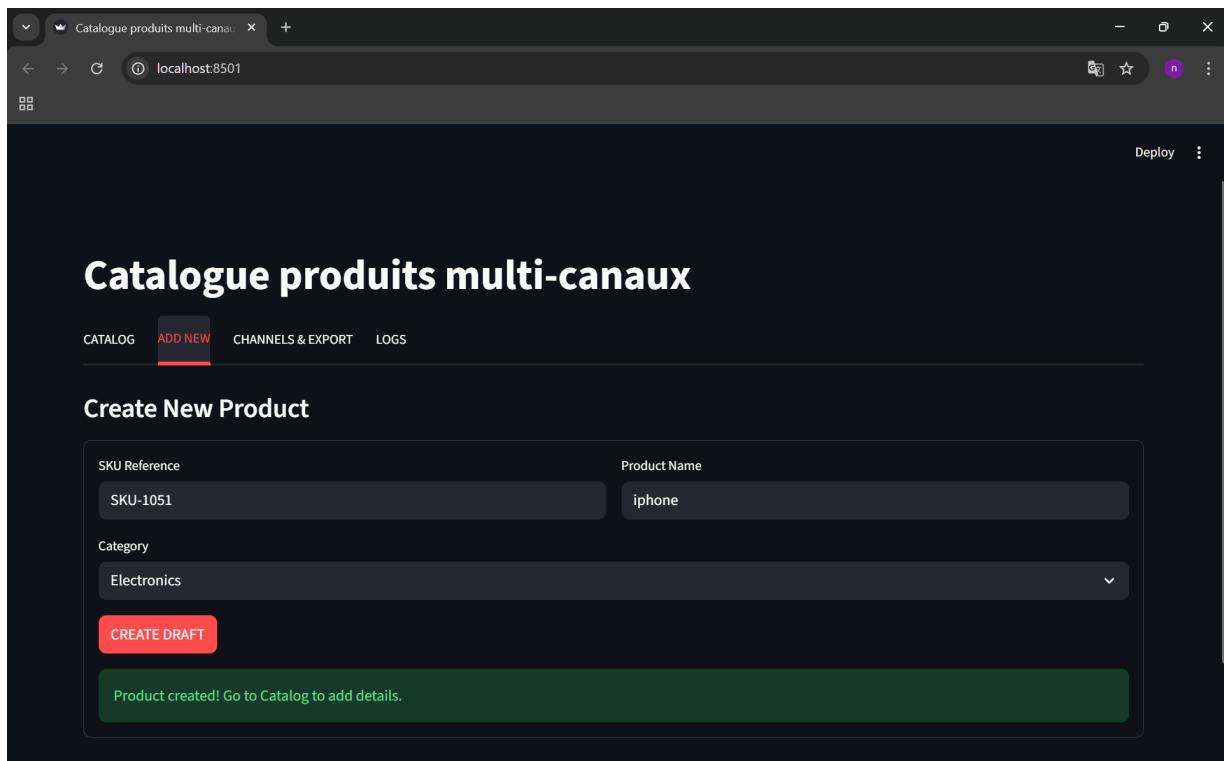


Figure 9: Creation form and success message.

4.8 Manual entry validation :

If the user tries to validate this draft without filling in the mandatory fields (Price, Storage, Warranty), the system blocks the action. We see here the previously created iphone, blocked with multiple validation errors.

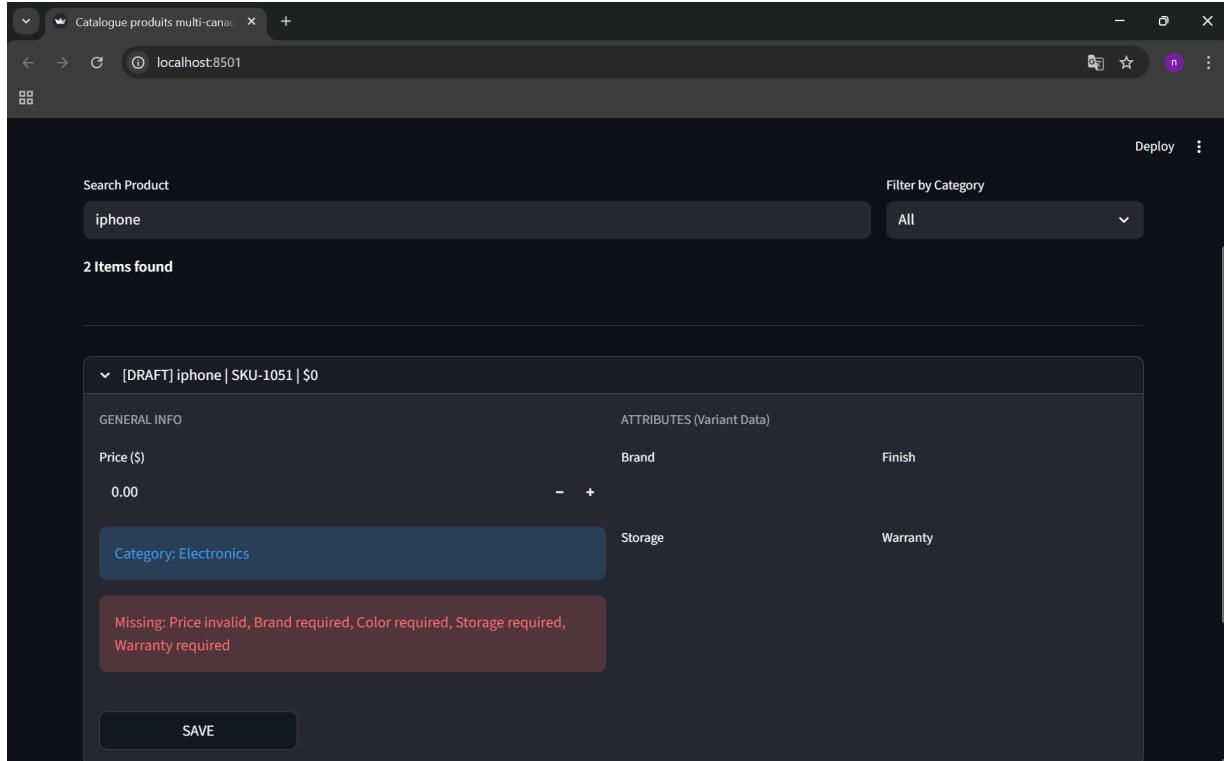


Figure 10: List of missing fields blocking publication.

4.9 Traceability :

All actions (Creation, Update, Publication) are logged. The LOGS tab allows seeing what is done and when.

The screenshot shows a dark-themed web application window titled "Catalogue produits multi-canaux". At the top, there are navigation links: "CATALOG", "ADD NEW", "CHANNELS & EXPORT", and "LOGS", with "LOGS" being the active tab. Below the navigation, the title "Audit Log & Traceability" is displayed. A table lists four log entries:

sku	action	timestamp
SKU-1003	PUBLISH	2026-02-09 03:43:06
SKU-1051	CREATE	2026-02-09 01:17:48
SKU-1003	UPDATE	2026-02-09 01:17:09
SKU-1003	UPDATE	2026-02-09 01:09:56

Figure 11: Action history (CREATE, UPDATE, PUBLISH) with timestamp.

4.10 Multi-channel export :

The "CHANNELS & EXPORT" module allows generating data feeds. The user chooses the target channel (Website, Marketplace or Social media).

The screenshot shows the same dark-themed web application window. The "CHANNELS & EXPORT" tab is now active. The title "Catalogue produits multi-canaux" is at the top. Below it, the section title "Multi-Channel Distribution" is shown. A sub-instruction "Select a sales channel to generate the appropriate data feed." is present. A dropdown menu is open, showing the following options: "E-Commerce Website (JSON)" (which is selected), "E-Commerce Website (JSON)", "Marketplace (CSV)", and "Social Media (CSV)".

Figure 12: Sales channel selection menu.

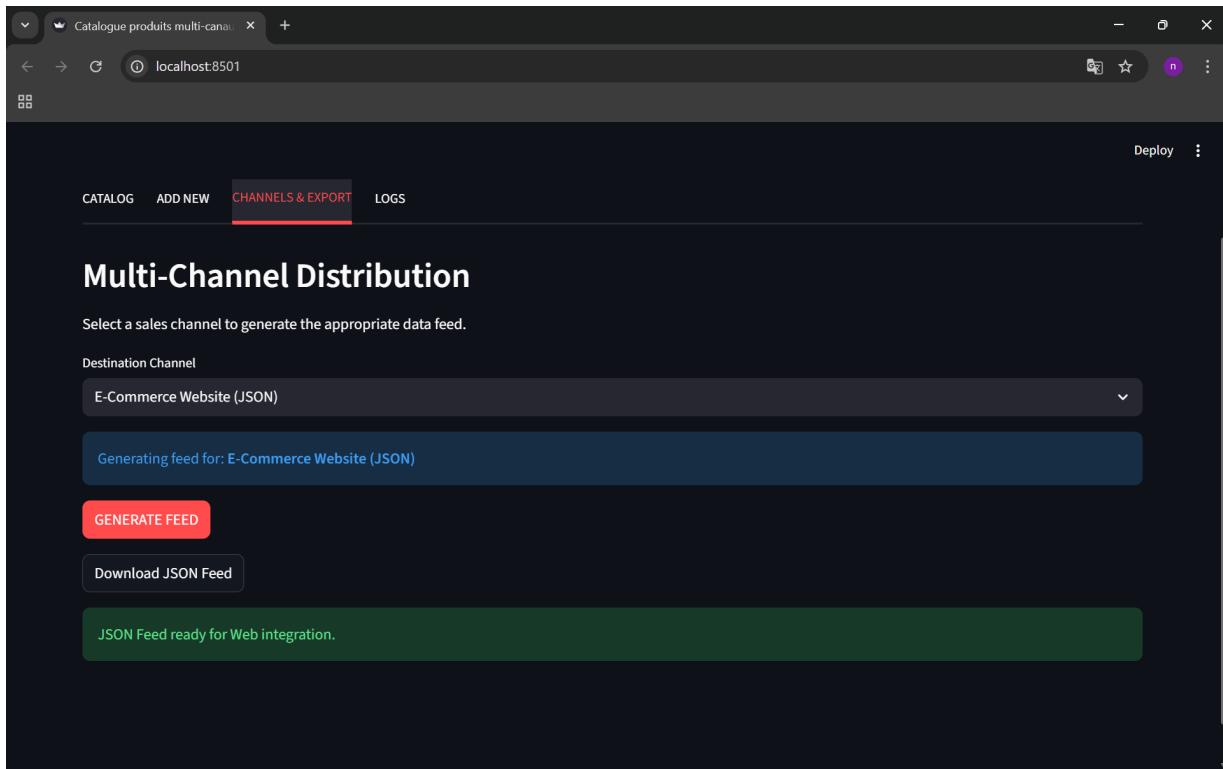


Figure 13: Successful generation of a JSON feed for the ecommerce site.

4.11 Verification of generated files :

This step validates the main objective of the project : the ability to structure data differently depending on the recipient.

- 1. Web Feed (JSON)** : The file keeps the nested structure. Technical attributes are encapsulated in the `attributes` object.

```

0 website_feed.json x
C:\Users\lilo\Downloads\{}\website_feed.json>...
1 [{"sku": "SKU-1001", "name": "Alienware m16 R2", "category": "Electronics", "status": "PUBLISHED", "price": 2392.0, "attributes": {"brand": "Dell", "color": "Silver", "storage": "2TB", "warranty": "2 Years Manufacturer"}, {"sku": "SKU-1002", "name": "The North Face Slim Fit Jeans", "category": "Clothing", "status": "PUBLISHED", "price": 112.0, "attributes": {"brand": "The North Face", "color": "Red", "size": "M", "material": "Gore-Tex"}}, {"sku": "SKU-1007", "name": "Adidas Running Shoes", "category": "Clothing", "status": "PUBLISHED", "price": 83.0, "attributes": {"brand": "Adidas", "color": "Red", "size": "39", "material": "Suede"}}, {"sku": "SKU-1009", "name": "Gucci Air Max Sneakers", "category": "Clothing", "status": "PUBLISHED", "price": 710.0, "attributes": {"brand": "Gucci", "color": "Beige", "size": "44", "material": "Synthetic Mesh"}}, {"sku": "SKU-1011", "name": "WF-1000XM Earbuds", "category": "Electronics", "status": "PUBLISHED", "price": 299.0, "attributes": {"brand": "Sony", "color": "Silver", "storage": "N/A", "warranty": "1 Year"}, {"sku": "SKU-1014", "name": "Levi's Air Max Sneakers", "category": "Clothing", "status": "PUBLISHED", "price": 125.0, "attributes": {"brand": "Levi's", "color": "Olive", "size": "42", "material": "Rubber"}}, {"sku": "SKU-1015", "name": "Galaxy Z Fold5", "category": "Electronics", "status": "PUBLISHED", "price": 1828.0, "attributes": {"brand": "Samsung", "color": "Midnight", "storage": "512GB", "warranty": "1 Year Standard"}, {"sku": "SKU-1016", "name": "AirPods Max", "category": "Electronics", "status": "PUBLISHED", "price": 549.0, "attributes": {"brand": "Apple", "color": "Space Grey", "storage": "N/A", "warranty": "AppleCare+"}}, {"sku": "SKU-1019", "name": "XPS 13 Plus", "category": "Electronics", "status": "PUBLISHED", "price": 1481.0, "attributes": {"brand": "Dell", "color": "Violent", "storage": "512GB", "warranty": "2 Years Manufacturer"}}, {"sku": "SKU-1020", "name": "Adidas Premium Cotton T-Shirt", "category": "Clothing", "status": "PUBLISHED", "price": 36.0, "attributes": {"brand": "Adidas", "color": "Olive", "size": "XXL", "material": "Wool"}}, {"sku": "SKU-1021", "name": "Galaxy Buds2 Pro", "category": "Electronics", "status": "PUBLISHED", "price": 229.0, "attributes": {"brand": "Samsung", "color": "Natural Titanium", "storage": "N/A", "warranty": "1 Year Standard"}}, {"sku": "SKU-1022", "name": "Galaxy S24 Ultra", "category": "Electronics", "status": "PUBLISHED", "price": 1353.0, "attributes": {"brand": "Samsung", "color": "Titanium Black", "storage": "512GB", "warranty": "Samsung Care+"}}, {"sku": "SKU-1023", "name": "Zara Slim Fit Jeans", "category": "Clothing", "status": "PUBLISHED", "price": 73.0, "attributes": {"brand": "Zara", "color": "Olive", "size": "S", "material": "Wool"}}, {"sku": "SKU-1029", "name": "Galaxy S24+", "category": "Electronics", "status": "PUBLISHED", "price": 1110.0, "attributes": {"brand": "Samsung", "color": "Midnight", "storage": "256GB", "warranty": "1 Year Standard"}}, {"sku": "SKU-1030", "name": "iPhone 15 Pro", "category": "Electronics", "status": "PUBLISHED", "price": 1206.0, "attributes": {"brand": "Apple", "color": "Silver", "storage": "512GB", "warranty": "1 Year"}}, {"sku": "SKU-1031", "name": "AirPods Max", "category": "Electronics", "status": "PUBLISHED", "price": 549.0, "attributes": {"brand": "Apple", "color": "Midnight", "storage": "N/A", "warranty": "AppleCare+"}}, {"sku": "SKU-1035", "name": "Galaxy Watch6 Classic", "category": "Electronics", "status": "PUBLISHED", "price": 430.0, "attributes": {"brand": "Samsung", "color": "Silver", "storage": "16GB", "warranty": "Samsung Care+"}}, {"sku": "SKU-1036", "name": "Xperia 1 V", "category": "Electronics", "status": "PUBLISHED", "price": 1294.0, "attributes": {"brand": "Sony", "color": "Natural Titanium", "storage": "512GB", "warranty": "1 Year"}}, {"sku": "SKU-1037", "name": "XPS 13 Plus", "category": "Electronics", "status": "PUBLISHED", "price": 1055.0, "attributes": {"brand": "Dell", "color": "Midnight", "storage": "1TB", "warranty": "1 Year"}}, {"sku": "SKU-1047", "name": "Apple Watch Ultra 2", "category": "Electronics", "status": "PUBLISHED", "price": 799.0, "attributes": {"brand": "Apple", "color": "Silver", "storage": "64GB", "warranty": "AppleCare+"}}, {"sku": "SKU-1048", "name": "Zara Premium Cotton T-Shirt", "category": "Clothing", "status": "PUBLISHED", "price": 25.0, "attributes": {"brand": "Zara", "color": "Black", "size": "XS", "material": "Gore-Tex"}}, {"sku": "SKU-1050", "name": "Galaxy S24 Ultra", "category": "Electronics", "status": "PUBLISHED", "price": 1371.0, "attributes": {"brand": "Samsung", "color": "Silver", "storage": "1TB", "warranty": "1 Year Standard"}}

```

Figure 14: Generated JSON file : Hierarchical structure preserved.

2. Marketplace Feed (Flattened CSV) : For marketplaces, the JSON has been "flattened". Fields like Storage, Warranty or Material have become distinct columns.

	A1	Sku	Name	Category	Price	Status	Brand	Color	Storage	Warranty	Size	Material
1	Sku	SKU-1001	Alienware	Electronic:	2392	PUBLISHED	Dell	Silver	2TB	2 Years	Manufacturer	
2		SKU-1002	The North Face	Clothing	112	PUBLISHED	The North Face	Red		M	Gore-Tex	
3		SKU-1007	Adidas	Ru Clothing	83	PUBLISHED	Adidas	Red			39	Suede
4		SKU-1009	Gucci	Air Clothing	710	PUBLISHED	Gucci	Beige			44	Synthetic Mesh
5		SKU-1011	WF-1000X	Electronic:	299	PUBLISHED	Sony	Silver	N/A	1 Year		
6		SKU-1014	Levi's	Air Clothing	125	PUBLISHED	Levi's	Olive			42	Rubber
7		SKU-1015	Galaxy Z F	Electronic:	1828	PUBLISHED	Samsung	Midnight	512GB	1 Year	Standard	
8		SKU-1016	AirPods M	Electronic:	549	PUBLISHED	Apple	Space Gre	N/A	AppleCare+		
9		SKU-1019	XPS 13 Plu	Electronic:	1481	PUBLISHED	Dell	Violet	512GB	2 Years	Manufacturer	
10		SKU-1020	Adidas Pre	Clothing	36	PUBLISHED	Adidas	Olive			XXL	Wool
11		SKU-1021	Galaxy Bu	Electronic:	229	PUBLISHED	Samsung	Natural Ti	N/A	1 Year	Standard	
12		SKU-1022	Galaxy S2	Electronic:	1353	PUBLISHED	Samsung	Titanium	512GB	Samsung	Care+	
13		SKU-1023	Zara Slim I	Clothing	73	PUBLISHED	Zara	Olive		S	Wool	
14		SKU-1029	Galaxy S2	Electronic:	1110	PUBLISHED	Samsung	Midnight	256GB	1 Year	Standard	
15		SKU-1030	iPhone 15	Electronic:	1055	PUBLISHED	Apple	Silver	128GB	AppleCare+		
16		SKU-1031	AirPods M	Electronic:	549	PUBLISHED	Apple	Midnight	N/A	AppleCare+		
17		SKU-1035	Galaxy W	Electronic:	430	PUBLISHED	Samsung	Silver	16GB	Samsung	Care+	
18		SKU-1036	Xperia 1 V	Electronic:	1206	PUBLISHED	Sony	Natural Ti	512GB	1 Year		
19		SKU-1037	XPS 13 Plu	Electronic:	1294	PUBLISHED	Dell	Midnight	1TB	1 Year		
20		SKU-1047	Apple Wai	Electronic:	799	PUBLISHED	Apple	Silver	64GB	AppleCare+		
21		SKU-1048	Zara Prem	Clothing	25	PUBLISHED	Zara	Black		XS	Gore-Tex	
22		SKU-1050	Galaxy S2	Electronic:	1371	PUBLISHED	Samsung	Silver	1TB	1 Year	Standard	
23												
24												
25												
26												

Figure 15: Detailed technical columns.

3. Social Networks Feed (Simple CSV) : For facebook/instagram, the file is simplified to

the maximum (SKU, Name, Price, Category) to lighten advertising processing.

SKU	Name	Price	Category
SKU-1001	Alienware	2392	Electronics
SKU-1002	The North	112	Clothing
SKU-1007	Adidas Ru	83	Clothing
SKU-1009	Gucci Air	710	Clothing
SKU-1011	WF-1000X	299	Electronics
SKU-1014	Levi's Air	125	Clothing
SKU-1015	Galaxy Z F	1828	Electronics
SKU-1016	AirPods M	549	Electronics
SKU-1019	XPS 13 Plu	1481	Electronics
SKU-1020	Adidas Pre	36	Clothing
SKU-1021	Galaxy Bu	229	Electronics
SKU-1022	Galaxy S2	1353	Electronics
SKU-1023	Zara Slim I	73	Clothing
SKU-1029	Galaxy S2	1110	Electronics
SKU-1030	iPhone 15	1055	Electronics
SKU-1031	AirPods M	549	Electronics
SKU-1035	Galaxy W	430	Electronics
SKU-1036	Xperia 1 V	1206	Electronics
SKU-1037	XPS 13 Plu	1294	Electronics
SKU-1047	Apple Wal	799	Electronics
SKU-1048	Zara Prem	25	Clothing
SKU-1050	Galaxy S2	1371	Electronics

Figure 16: Simplified format.

5 Conclusion and perspectives :

This project allowed the realization of a functional multi-channel product catalog answering the problem of heterogeneous data management. The use of mongoDB proved relevant to manage product polymorphism, while python facilitated the creation of custom export feeds.

Perspectives for improvement :

- **Artificial Intelligence** : Integration of an NLP model to automatically categorize products during import (e.g., detecting that "iphone" goes into "Electronics").
- **REST API** : Develop an API so that the ecommerce site can query the catalog in real time, rather than using static JSON exports.
- **Media management** : Add support for the upload and storage of product images (via Amazon S3 or GridFS).