Portfolio : Projet Décisionnel

Création d'un Tableau de Bord pour le E-Commerce

Étudiant : Aimad Hamdaoui | **Outils utilisés :** Python, React, Bases de données

Ce document présente la création d'un **Tableau de Bord interactif** (pour suivre les ventes d'une boutique en ligne). Mon objectif ici est d'expliquer, de manière simple et accessible, comment j'ai validé mes 4 compétences clés pour ce projet. Je vais vous montrer comment j'ai réussi à récolter des informations brutes, à les nettoyer, à les afficher sous forme de graphiques clairs et à mettre le tout en ligne de façon sécurisée.

TRAITER

Niveau 3 : Collecter et Nettoyer

Comment j'ai validé cette compétence

Pour cette étape, je devais prouver que je savais récupérer des informations éparpillées et les rendre utilisables. J'ai validé cette compétence en démontrant ma capacité à :

- **Faire le tri automatiquement :** Dans la vraie vie, les données d'une entreprise arrivent dans plein de formats différents (comme des tableaux Excel ou des textes brouillons). J'ai créé un programme qui ouvre ces fichiers, lit les informations, corrige les fautes de frappe (comme les espaces en trop) et les range dans la base de données.
- **Sécurité :** Pour éviter que des informations fausses, incomplètes ou même des virus ne rentrent dans le système, j'ai mis en place un contrôle strict. Si une donnée ne correspond pas exactement à ce qui est attendu, elle est automatiquement bloquée et rejetée.
- **Savoir s'adapter aux changements :** Quand une entreprise grandit, sa façon de stocker l'information doit changer. J'ai mis en place un système qui permet de rajouter de nouveaux "tiroirs" dans notre base de données sans jamais perdre ou casser les informations qui y sont déjà stockées.

dashboard/commands.py (Extrait logique)

```python
import click
import csv
import json
from flask.cli import with_appcontext
from .extensions import db
from .models import Produit, VenteMensuelle

@click.command('import-data')
@with_appcontext
def import_data_command():
    """Importe les données depuis CSV et JSON."""

    # --- 1. Import des Produits (CSV) ---
    try:
        with open('produits.csv', 'r', encoding='utf-8') as f:
            reader = csv.DictReader(f)
            for row in reader:
                produit = Produit(
                    produit_id=row['id'],
                    nom_produit=row['nom'],
                    prix_unitaire=float(row['prix']), # Transformation de ty
                    stock_actuel=int(row['stock']),
                    categorie=row['categorie']
                )
                db.session.add(produit)
        print("Import CSV Produits : OK")
    except FileNotFoundError:
        print("Erreur : fichier produits.csv manquant.")

    # --- 2. Import des Ventes (JSON) ---
    try:
        with open('ventes.json', 'r') as f:
            data = json.load(f)
            for item in data:
                vente = VenteMensuelle(
                    mois=item['mois'],
                    chiffre_affaires=item['ca'],
                    nombre_commandes=item['commandes']
                )
                db.session.add(vente)
        print("Import JSON Ventes : OK")
    except Exception as e:
        print(f"Erreur JSON : {e}")

    # Commit unique à la fin pour la performance
    db.session.commit()
    print("Base de données initialisée avec succès.")
```

FIG. 1 : Le programme en action : il avale les fichiers bruts, les nettoie et les enregistre proprement.

ANALYSER

Niveau 3 : Comprendre les Chiffres

Comment j'ai validé cette compétence

Avoir des données rangées ne suffit pas, il faut pouvoir les croiser pour leur donner du sens. J'ai répondu à ces exigences par :

- **Créer des liens intelligents :** Dans un magasin, un client est lié à ses achats, eux-mêmes liés à des familles de produits. J'ai appris au système à comprendre ces liens familiaux pour qu'il puisse répondre à des questions complexes (ex : "Quels sont les produits les plus vendus aux clients fidèles dans le Sud ?").
- **Calculer vite sans bloquer l'ordinateur :** Si on demande à l'ordinateur de calculer la moyenne de 100 000 ventes en direct, le site internet risque de figer. J'ai donc configuré le système pour qu'il calcule les totaux à l'avance et tienne un petit carnet à jour. Quand on veut voir le résultat, l'affichage est instantané.
- **L'entraînement catastrophe :** Pour être sûr que mes calculs soient solides, j'ai créé un programme "saboteur". Son seul but est de modifier des informations au hasard pour essayer de faire planter mon système. Réussir à contrer ce saboteur m'a prouvé que mon outil est prêt pour la réalité.

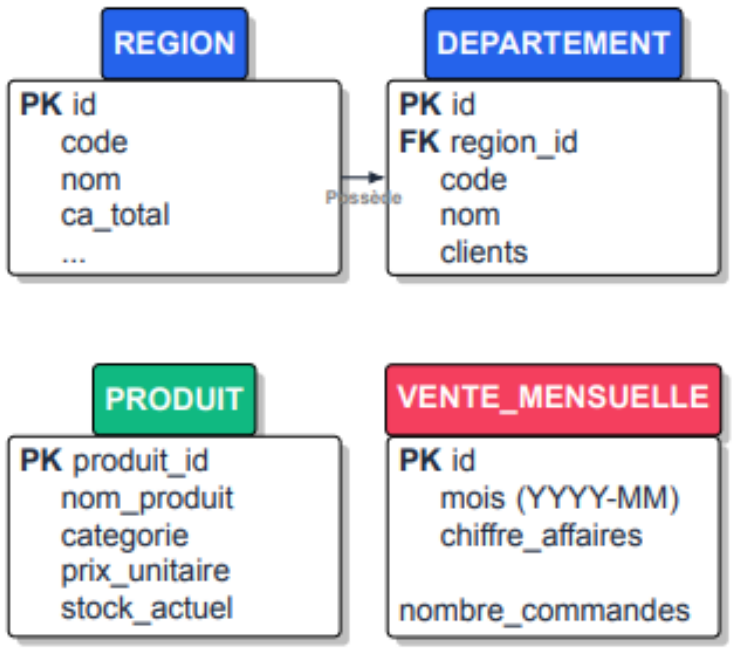

FIG. 2 : Plan de la base de données : un vrai labyrinthe où chaque information est connectée aux autres.

VALORISER

Niveau 3 : Rendre l'information utile

Comment j'ai validé cette compétence

Une information n'a de la valeur que si elle est compréhensible d'un seul coup d'œil par le directeur du magasin. J'ai validé cette étape en créant l'écran final de l'application.

- **Des graphiques qui parlent d'eux-mêmes :** Une liste de 10 000 lignes de texte est impossible à analyser humainement. J'ai transformé ces chiffres en un tableau de bord visuel : des courbes simples et des compteurs colorés (vert si les ventes montent, rouge si elles baissent) pour aider à prendre des décisions rapidement.
- **Ne jamais laisser l'utilisateur dans le noir :** Si la connexion internet coupe, l'écran ne doit pas juste afficher une grosse erreur technique effrayante. J'ai fait en sorte que l'application garde en mémoire le dernier affichage connu et prévienne poliment l'utilisateur que le réseau est capricieux.
- **Séparer la vitrine de l'arrière-boutique :** J'ai construit le site de façon à ce que le design (ce qu'on voit) et le moteur (qui gère les chiffres) soient indépendants. Ainsi, si on veut changer les couleurs du site demain, on n'a pas besoin de tout reconstruire.

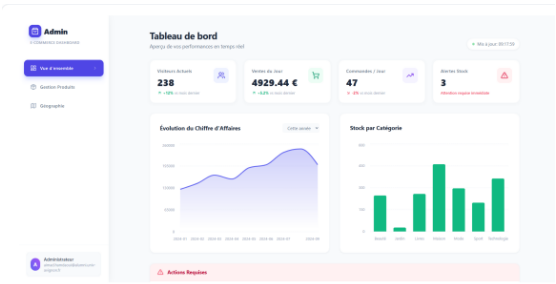

FIG. 3 : L'écran final : des compteurs clairs pour voir la santé du magasin en direct.

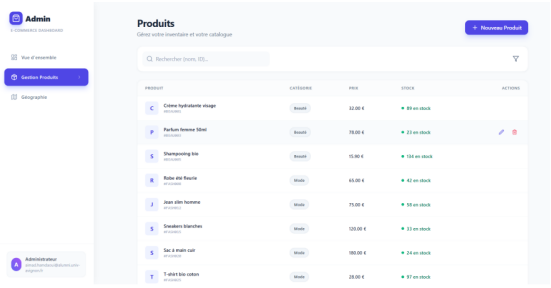

FIG. 4 : Une vue "Tableau" permettant de trier et rechercher facilement dans le catalogue.

DÉVELOPPER

Niveau 2 : Mettre en Ligne

Comment j'ai validé cette compétence

Il est facile de faire fonctionner un site sur son propre ordinateur, mais beaucoup plus dur de le mettre sur internet pour qu'il soit accessible au public de façon fiable. J'ai validé cela de la façon suivante :

- **Le concept du "Prêt-à-livrer" (Les Conteneurs) :** Habituellement, un programme qui marche sur un PC risque de planter sur un autre. J'ai enfermé mon application dans des "boîtes virtuelles" (Docker). Grâce à ça, je peux déplacer mon site internet d'un ordinateur à un autre serveur web en un seul clic, en étant certain qu'il fonctionnera de la même manière.
- **Un travail propre et bien rangé :** Au lieu d'écrire tout le programme dans un seul énorme bloc de texte, je l'ai découpé en petites briques indépendantes (comme des Lego). C'est beaucoup plus facile à réparer ou à améliorer plus tard en équipe.
- **Les robots testeurs :** Avant d'autoriser la mise en ligne du site, je n'essaie pas de tout cliquer moi-même. J'ai créé des "robots logiciels" qui vont simuler des milliers de scénarios (remplir des formulaires, naviguer sur le site, calculer des données) en quelques secondes. S'ils trouvent la moindre erreur, la mise en ligne est bloquée.

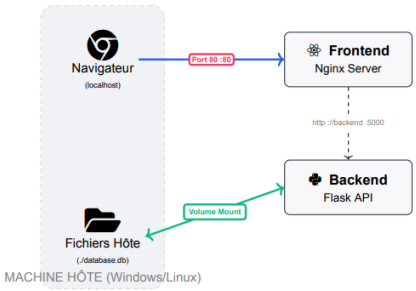

FIG. 5 : Les "boîtes" qui enferment le programme pour le rendre transportable partout.

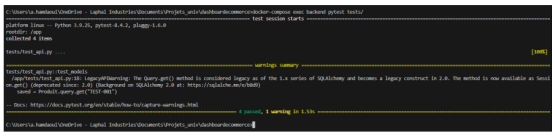

FIG. 6 : L'écran montrant que les robots ont testé le site et n'ont trouvé aucune erreur.