RAPPORT DE PROJET

ElectroManage

PAGE DE GARDE :

Titre du projet : Développement d'une Application Web de Gestion de Produits electroniques

Réalisé par : RABII Souhail

Encadré par : Mr M.Ghazouani

Filière : 3IIR G6

Date : 05 Janvier 2026

Établissement : EMSI (École Marocaine des Sciences de l'Ingénieur)

RÉSUMÉ

Ce projet consiste en la conception et le développement d'une application web de gestion de stock pour des produits électroniques ("Eproducts"). L'objectif principal était de mettre en place un système CRUD (Create, Read, Update, Delete) complet permettant aux administrateurs de gérer l'inventaire efficacement. L'application a été réalisée en utilisant le framework PHP Symfony, couplé à une base de données relationnelle gérée via Doctrine ORM. Les résultats obtenus offrent une interface fonctionnelle pour l'ajout, la modification, la suppression et la visualisation des produits, répondant ainsi aux besoins de digitalisation du processus de gestion.

TABLE DES MATIÈRES

1. INTRODUCTION

1.1 Contexte

Dans un environnement commercial de plus en plus numérisé, la gestion manuelle des stocks (papier ou tableurs simples) devient rapidement obsolète et source d'erreurs. Les entreprises ont besoin d'outils fiables et centralisés pour suivre l'état de leurs produits en temps réel.

1.2 Problématique

Comment concevoir une solution web robuste et évolutive permettant de centraliser les informations des produits (nom, marque, prix, quantité, date de fabrication) et de faciliter leur mise à jour par les gestionnaires ?

1.3 Objectifs

- Développer une architecture MVC (Modèle-Vue-Contrôleur) respectant les bonnes pratiques.
- Assurer la persistance des données via une base de données relationnelle. Fournir une
- interface utilisateur intuitive pour les opérations de gestion courantes.

2. ANALYSE ET CONCEPTION

2.1 Cahier des Charges Fonctionnel

Le système doit permettre de :

- **Lister** l'ensemble des produits disponibles avec leurs détails.
- **Ajouter** un nouveau produit en spécifiant son nom, sa marque, son prix, sa quantité et sa date de fabrication.
- **Modifier** les informations d'un produit existant.
- **Supprimer** un produit obsolète ou erroné.

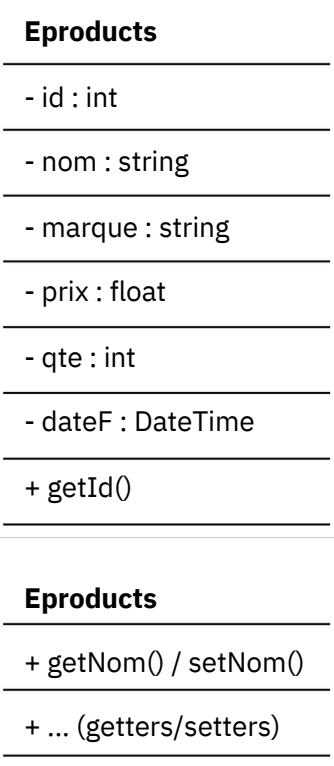
2.2 Technologies Utilisées

- **Langage de programmation :** PHP , HTML
- **Framework Backend :** Symfony 7
- **Base de données :** MySQL / MariaDB
- **ORM (Object-Relational Mapping) :** Doctrine
- **Moteur de template :** Twig
- **Serveur Web :** Apache (via XAMPP)

2.3 Modélisation des Données (UML)

L'entité principale du projet est `Eproducts`.

Diagramme de Classe (Simplifié) et Annexes :




```php
#[ORM\Entity(repositoryClass: EproductsRepository::class)]
class Eproducts
{
    #[ORM\Id] #[ORM\GeneratedValue] #[ORM\Column]
    private ?int $id = null;

    #[ORM\Column(length: 255)]
    private ?string $nom = null;

    #[ORM\Column(length: 255)]
    private ?string $marque = null;

    #[ORM\Column]
    private ?float $prix = null;

    #[ORM\Column]
    private ?int $qte = null;

    #[ORM\Column(type: Types::DATE_MUTABLE)]
    private ?\DateTime $dateF = null;

    //... Getters et Setters ...
}
```

3. RÉALISATION

3.1 Architecture du Projet

Le projet suit la structure standard de Symfony :

- src/Entity/ : Contient la classe Eproducts mappée à la base de données.
- src/Repository/ : Contient EproductsRepository pour les requêtes personnalisées.
- src/Controller/ : Contient la logique métier dans Controller.php.
- `templates/` : Contient les vues Twig (`home.html.twig`, `ajout.html.twig`,etc.).

3.2 Étapes de Développement

1. **Initialisation :** Création du projet Symfony et configuration de la connexion base de données dans le fichier .env.
2. **Création de la Base de Données :** Utilisation de la commande php bin/console `doctrine:database:create`.
3. **Création de l'Entité :** Génération de l'entité Eproducts avec ses attributs via php bin/console `make:entity`.
4. **Migration :** Mise à jour du schéma de base de données avec php bin/console `doctrine:migrations:migrate`.
5. **Logique Contrôleur :** Implémentation des méthodes index, ajout, modifier, et supprimer dans `Controller.php`.
    - *Note technique :* Le traitement des formulaires a été réalisé en récupérant directement les données de la requête ($request->request->get(...)) pour une gestion explicite des données.

3.3 Extraits de Code

Extrait du Contrôleur (Ajout d'un produit) :

```
publicfunction ajoutProduit(EntityManagerInterface $entityManager, Request
$request): Response
{
    $item = new Eproducts();
    $item->setNom($request->request->get("nom"));
    $item->setMarque($request->request->get("marque"));
    //... hydratation de l'objet ...

    $entityManager->persist($item);
    $entityManager->flush();

    return $this->redirectToRoute('index');
```

4. RÉSULTATS ET TESTS

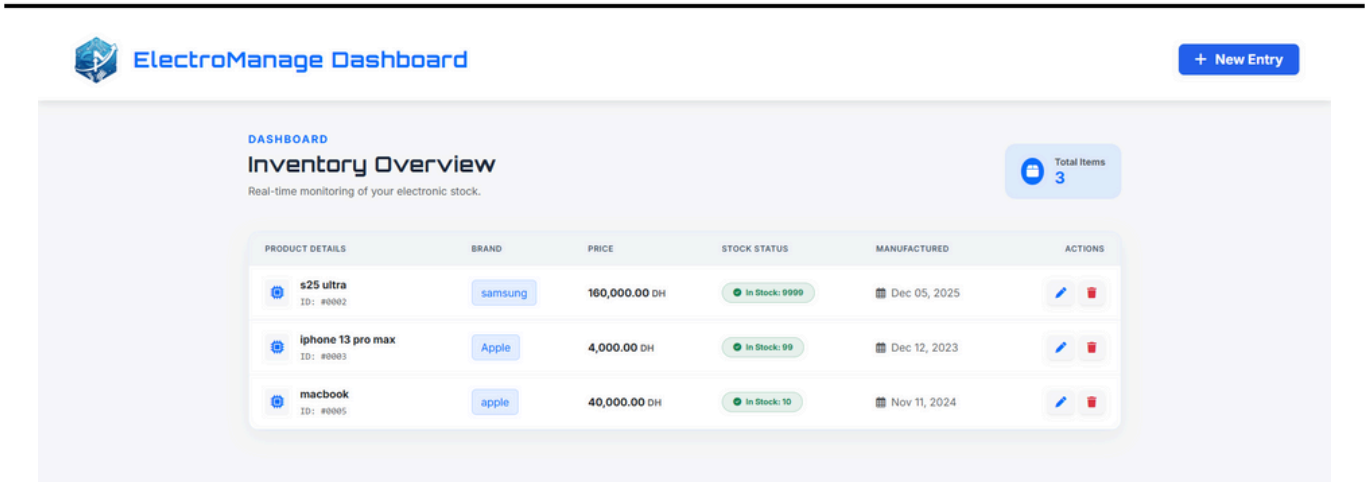
4.1 Fonctionnalités Obtenues

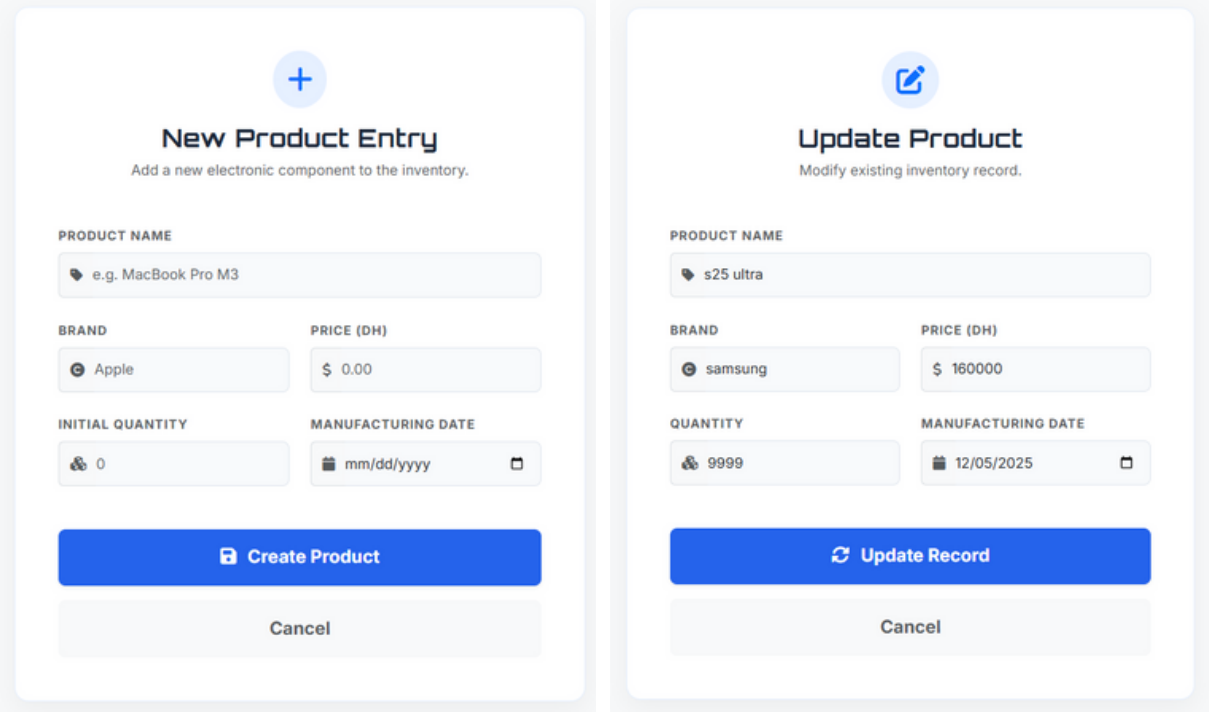
L'application est fonctionnelle et accessible via le navigateur.

Page d'accueil : Affiche un tableau listant tous les produits enregistrés en base de données.
Formulaire d'ajout : Permet de saisir les détails d'un nouveau produit.
Actions : Des boutons "Modifier" et "Supprimer" sont présents pour chaque ligne du tableau.

4.2 Tests Réalisés

Test d'insertion : Ajout d'un produit "Laptop HP" -> Le produit apparaît bien dans la liste et dans la table eproducts de la base de données.
Test de modification : Changement du prix d'un produit -> La mise à jour est immédiate.
Test de suppression : Suppression d'un produit -> Il disparaît de la liste.
Validation basique : Vérification que les champs numériques (prix, qte) sont bien traités (casting en float et int dans le contrôleur).





5. CONCLUSION ET PERSPECTIVES

5.1 Bilan

Ce projet a permis de mettre en pratique les concepts fondamentaux du développement web avec Symfony. L'architecture MVC a facilité la séparation des préoccupations, rendant le code maintenable. L'utilisation de Doctrine a grandement simplifié les interactions avec la base de données.

5.2 Améliorations Possibles

Pour aller plus loin, plusieurs évolutions sont envisageables :

- **Sécurité :** Mise en place d'un système d'authentification (Login/Register) pour restreindre l'accès à l'administration.
- **Validation :** Utilisation du composant Symfony Form et Validator pour une gestion plus robuste des erreurs de saisie (côté serveur et client).
- **Design :** Amélioration de l'interface utilisateur avec un framework CSS moderne ou des composants JavaScript dynamiques.
- **Pagination :** Mise en place d'une pagination pour gérer un grand nombre de produits.

6. BIBLIOGRAPHIE

1. **Documentation Officielle Symfony** - *symfony.com/doc*
2. **Cours de PHP/Symfony (3IIR)** - *Support de cours EMSI*
3. ***Les* atelier 1, 2 et 3** - *Apprenez à développer avec Symfony*
4. **ChatGPT pour L'assistance**