**COMMUNAUTE FRANCAISE DE BELGIQUE**

**Institut des Carrieres Commerciales**

Ville de Bruxelles

Rue de la Fontaine 4 - 1000 BRUXELLES

**LIVRABLE 15**

**DOCUMENTATION API COMPLETE**

**Interface de Programmation Application FarmShop**

*Epreuve integree realisee en vue de l'obtention du titre de*

**Bachelier en Informatique de gestion**

Orientation developpement d'applications

**MEFTAH Soufiane**

Annee academique 2024-2025

# Table des matieres

Introduction

1. Architecture generale de l'API FarmShop

1.1 Standards et technologies utilisees

1.2 Structure RESTful et organisation des endpoints

1.3 Systeme d'authentification et autorisation

1.4 Gestion des erreurs et codes de reponse

2. Documentation interactive Swagger/OpenAPI

2.1 Implementation de l'interface Swagger UI

2.2 Schemas de donnees et modeles

2.3 Annotations automatisees des endpoints

2.4 Tests interactifs et validation des API

3. Endpoints produits et catalogue

3.1 CRUD complet des produits

3.2 Recherche avancee et filtrage

3.3 Gestion des stocks et disponibilites

3.4 Categories et taxonomies

4. Systeme de panier et commandes

4.1 Gestion du panier d'achat

4.2 Panier de location avec contraintes temporelles

4.3 Creation et suivi des commandes

4.4 Workflow des statuts de commande

5. Systeme de location et inspections

5.1 Contraintes de location et disponibilites

5.2 Commandes de location et gestion temporelle

5.3 Inspections et etats du materiel

5.4 Calcul des penalites et cautions

6. Gestion des utilisateurs et profils

6.1 Inscription et authentification

6.2 Profils utilisateur et preferences

6.3 Wishlist et likes

6.4 Historique et donnees personnelles GDPR

7. Systeme de paiement Stripe

7.1 Creation et confirmation des paiements

7.2 Gestion des cautions pour locations

7.3 Webhooks et synchronisation

7.4 Remboursements et annulations

8. Blog et systeme de contenu

8.1 Articles et categories de blog

8.2 Commentaires et moderation

8.3 Signalements et gestion communautaire

8.4 Statistiques et analytics

9. Systeme de messages et notifications

9.1 Messages administratifs et alertes

9.2 Notifications de stock

9.3 Messages de contact visiteurs

9.4 Newsletter et abonnements

10. Interface d'administration

10.1 Endpoints reserves aux administrateurs

10.2 Statistiques et rapports

10.3 Gestion des utilisateurs et roles

10.4 Configuration et parametrage

11. Conformite GDPR et gestion des cookies

11.1 Preferences et consentements

11.2 Export et suppression des donnees

11.3 Migration des cookies invites

11.4 Historique et auditabilite

12. Performances et optimisation

12.1 Pagination et limitations

12.2 Cache et strategies de mise en cache

12.3 Monitoring et metriques

12.4 Scalabilite et architecture

Conclusion

# Introduction

La plateforme FarmShop, specialisee dans la vente et location de produits agricoles, s'appuie sur une architecture API RESTful robuste et comprehensve. Cette documentation technique presente l'ensemble des 540+ endpoints disponibles, leur utilisation, leurs parametres et les reponses attendues.

L'API FarmShop respecte les standards OpenAPI 3.0 et beneficie d'une documentation interactive Swagger UI accessible en ligne. Elle couvre l'ensemble des fonctionnalites de la plateforme : gestion des produits, systeme de commandes classiques et de location, paiements securises, blog, messages, administration et conformite GDPR.

Cette documentation technique constitue le livrable 15 du projet et s'adresse aux developpeurs, integrateurs et equipes techniques souhaitant comprendre ou utiliser l'API FarmShop. Elle inclut des exemples concrets, les schemas de donnees complets et les meilleures pratiques d'integration.

**Documentation Swagger interactive disponible a l'adresse :**

http://127.0.0.1:8000/api/documentation

# 1. Architecture generale de l'API FarmShop

## 1.1 Standards et technologies utilisees

**Specifications techniques :**

*Standard OpenAPI 3.0.0 - Documentation automatisee et interactive*

Architecture RESTful avec ressources hierarchiques et operations HTTP standards

*Framework Laravel 10+ avec Sanctum pour l'authentification*

Package darkaonline/l5-swagger v9.0.1 pour generation automatique documentation

**Base URL de l'API :**

http://127.0.0.1:8000/api

Version actuelle : 1.0.0

**Formats de donnees supportes :**

• JSON (application/json) - Format principal pour requetes et reponses

• Form Data (multipart/form-data) - Upload de fichiers et images

• URL Encoded (application/x-www-form-urlencoded) - Formulaires simples

## 1.2 Structure RESTful et organisation des endpoints

L'API FarmShop organise ses 540+ endpoints selon une structure hierarchique claire :

**Endpoints publics (consultation libre) :**

GET /api/products - Liste des produits avec filtres avances

GET /api/categories - Catalogue des categories

GET /api/blog/posts - Articles de blog publies

POST /api/contact - Messages de contact visiteurs

**Endpoints authentifies (utilisateurs connectes) :**

GET /api/profile - Profil utilisateur

POST /api/cart/products/{id} - Ajout au panier

GET /api/orders - Historique des commandes

POST /api/stripe/payment-intent - Creation paiement

**Endpoints administrateur (acces restreint) :**

GET /api/admin/users - Gestion des utilisateurs

POST /api/admin/products - Creation de produits

GET /api/admin/orders/statistics - Rapports de ventes

## 1.3 Systeme d'authentification et autorisation

L'API utilise un systeme d'authentification hybride combine :

**Authentification par session web (auth:web) :**

• Utilisation des sessions Laravel classiques

• Protection CSRF automatique

• Gestion des cookies d'authentification

• Ideal pour interfaces web frontend

**Authentification Bearer Token (Sanctum) :**

• Tokens API personnels pour applications mobiles

• Support SPA (Single Page Applications)

• Gestion des portees et permissions

• Revocation et expiration des tokens

**Niveaux d'autorisation :**

• Visiteur : Acces lecture seule aux ressources publiques

• Utilisateur : Gestion profil, commandes, panier, wishlist

• Administrateur : Acces complet gestion plateforme

## 1.4 Gestion des erreurs et codes de reponse

L'API respecte les codes de statut HTTP standards :

**Codes de succes :**

• 200 OK : Requete reussie avec donnees

• 201 Created : Ressource creee avec succes

• 204 No Content : Operation reussie sans contenu retour

**Codes d'erreur client :**

• 400 Bad Request : Donnees invalides ou malformees

• 401 Unauthorized : Authentification requise

• 403 Forbidden : Permissions insuffisantes

• 404 Not Found : Ressource inexistante

• 409 Conflict : Conflit de donnees (stock insuffisant)

• 422 Unprocessable Entity : Erreurs de validation

**Codes d'erreur serveur :**

• 500 Internal Server Error : Erreur systeme interne

• 503 Service Unavailable : Service temporairement indisponible

# 2. Documentation interactive Swagger/OpenAPI

## 2.1 Implementation de l'interface Swagger UI

L'API FarmShop integre une documentation interactive complete basee sur Swagger UI, accessible en temps reel et synchronisee avec le code source.

**Configuration technique :**

Package : darkaonline/l5-swagger v9.0.1

URL d'acces : http://127.0.0.1:8000/api/documentation

Regeneration : php artisan l5-swagger:generate

**Fonctionnalites disponibles :**

• Interface graphique intuitive pour explorer tous les endpoints

• Tests interactifs directs depuis l'interface

• Exemples de requetes et reponses automatiques

• Validation en temps reel des parametres

• Export de la specification OpenAPI (JSON/YAML)

• Documentation des schemas de donnees complets

## 2.2 Schemas de donnees et modeles

L'API definit des schemas de donnees precis pour chaque entite :

**Modeles principaux documentes :**

• Product : Produits avec attributs vente et location

• Category / RentalCategory : Taxonomies produits

• User : Utilisateurs et profils complets

• Order / OrderLocation : Commandes achat et location

• CartItem / CartItemLocation : Articles des paniers

• BlogPost / BlogCategory : Contenu editorial

• Message : Systeme de messagerie

• SpecialOffer : Promotions et reductions

**Schemas de reponse standardises :**

• ApiResponse : Format uniforme pour toutes les reponses

• PaginatedResponse : Pagination standardisee

• ErrorResponse : Gestion coherente des erreurs

## 2.3 Annotations automatisees des endpoints

Chaque endpoint beneficie d'annotations OpenAPI completes integrees directement dans le code source PHP, garantissant la synchronisation entre documentation et implementation.

**Exemple d'annotation complete :**

@OA\Get(

path="/api/products",

tags={"Products"},

summary="Liste des produits avec filtres avances",

@OA\Parameter(name="category\_id", in="query", required=false),

@OA\Response(response=200, ref="#/components/schemas/PaginatedResponse")

)

**Informations documentees par endpoint :**

• Description fonctionnelle detaillee

• Parametres avec types, contraintes et exemples

• Codes de reponse avec schemas associes

• Exigences d'authentification et permissions

• Tags d'organisation thematique

## 2.4 Tests interactifs et validation des API

L'interface Swagger UI permet de tester directement tous les endpoints depuis le navigateur, facilitant le developpement et l'integration.

**Capacites de test :**

• Execution de requetes reelles contre l'API

• Saisie assistee des parametres avec validation

• Gestion automatique de l'authentification

• Affichage formate des reponses JSON

• Mesure des temps de reponse

• Export des requetes pour outils externes (curl, Postman)

**Validation automatique :**

• Verification des types de parametres

• Controle des valeurs obligatoires

• Validation des formats (email, date, URL)

• Verification des schemas de reponse

# 3. Endpoints produits et catalogue

## 3.1 CRUD complet des produits

L'API offre une gestion complete des produits avec operations CRUD differenciees selon les permissions utilisateur.

**Endpoints publics (consultation) :**

GET /api/products

*Liste paginee des produits actifs avec filtres avances*

Parametres : category\_id, type, stock\_status, search, sort, per\_page

Reponse : Collection paginee avec metadonnees

GET /api/products/{product}

*Details complets d'un produit avec relations*

Include : category, rental\_category, images, constraints

Cache : 60 minutes pour optimiser performances

GET /api/products/category/{category}

*Produits filtres par categorie avec pagination*

Tri automatique par popularite et stock disponible

POST /api/products/{product}/check-availability

*Verification de disponibilite en temps reel*

Parametres : quantity, start\_date, end\_date (pour locations)

Reponse : Disponibilite boolean + suggestions alternatives

**Endpoints administrateur (gestion complete) :**

POST /api/admin/products

*Creation de nouveaux produits avec validation complete*

Support upload images multiples et gestion stocks

PUT /api/admin/products/{product}

*Mise a jour complete ou partielle*

Gestion automatique historique modifications

DELETE /api/admin/products/{product}

*Suppression logique avec conservation historique commandes*

POST /api/admin/products/{id}/restore

*Restauration produits supprimes avec verification coherence*

## 3.2 Recherche avancee et filtrage

Le systeme de recherche integre des capacites avancees de filtrage et tri pour optimiser la decouverte produits.

GET /api/products/search

*Recherche textuelle intelligente avec suggestions*

**Parametres de recherche disponibles :**

• q : Terme de recherche (nom, description, tags)

• category\_id : Filtrage par categorie principale

• rental\_category\_id : Filtrage categories de location

• type : sale|rental|both - Type de produit

• price\_min / price\_max : Fourchette de prix

• rental\_price\_min / rental\_price\_max : Prix location

• stock\_status : available|low|out\_of\_stock

• is\_rental\_available : Disponible en location

**Options de tri supportees :**

• name\_asc / name\_desc : Ordre alphabetique

• price\_asc / price\_desc : Prix croissant/decroissant

• rental\_price\_asc / rental\_price\_desc : Prix location

• created\_at : Nouveautes en premier

• popularity : Tri par likes et commandes

• stock\_desc : Stock disponible en priorite

**Fonctionnalites avancees :**

• Recherche fuzzy avec tolerance aux fautes de frappe

• Suggestions automatiques de termes similaires

• Mise en evidence des termes recherches

• Filtrage intelligent par disponibilite temps reel

## 3.3 Gestion des stocks et disponibilites

Le systeme de gestion des stocks distingue le stock de vente du stock de location avec alertes automatiques et verifications en temps reel.

**Endpoints de consultation des stocks :**

GET /api/admin/products/stock/dashboard

*Tableau de bord complet des stocks avec indicateurs*

Metriques : Stock total, rotation, alertes, valorisation

GET /api/admin/products/stock/alerts

*Liste des produits en situation de stock critique*

Seuils configurables par produit avec notifications

GET /api/admin/products/stock/low

*Produits sous le seuil minimum avec suggestions reapprovisionnement*

**Gestion des stocks (administrateur) :**

POST /api/admin/products/{product}/stock/update

*Mise a jour des stocks avec traçabilite complete*

Parametres : quantity, rental\_stock, operation\_type, reason

Historique : Conservation mouvements avec horodatage

**Verifications automatiques :**

• Controle coherence stock vente/location

• Prevention survente avec reservations temporaires

• Alertes automatiques seuils critiques

• Calcul disponibilite location avec contraintes temporelles

## 3.4 Categories et taxonomies

L'organisation des produits s'appuie sur un double systeme de categorisation adapte aux besoins de vente et de location.

**Categories principales (vente) :**

GET /api/categories

*Liste complete des categories actives avec compteurs*

Include : Nombre de produits, ordre d'affichage

GET /api/categories/active

*Categories publiees uniquement avec tri par priorite*

GET /api/categories/{category}

*Details categorie avec produits associes pagines*

**Categories de location :**

GET /api/rental-categories

*Taxonomie specifique aux produits de location*

Contraintes : Durees min/max, conditions particulieres

GET /api/rental-categories/{rentalCategory}

*Details avec produits disponibles en location*

**Gestion administrative des categories :**

POST /api/admin/categories

*Creation avec gestion automatique des slugs et ordre*

PUT /api/admin/categories/{category}

*Modification avec preservation des relations produits*

PATCH /api/admin/categories/{category}/toggle-status

*Activation/desactivation avec impact automatique sur produits*

GET /api/admin/categories/stats

*Statistiques detaillees par categorie avec analytics*