

Roadmap TP4 Laravel – Backend Only (Mise à jour complète)

Membres exclus du TP4

****H, J, L**** → Travaillent sur le ****CMS (TP3)**** → ****Ne participent pas au TP4****

Membres actifs sur TP4

Membre	Rôle Principal	Tâches
A	**Architecture & Notifications**	Setup Laravel + Services de notification (mail, events)
C	**Base de données**	Migrations, Modèles, Seeders
G	**Authentification**	API Auth avec Sanctum + Rôles
D	**Produits**	CRUD Produits + Variantes + Images
E	**Commandes**	Gestion des commandes + Statuts
I	**Panier & Paiement**	Panier API + Paiement simulé
B	**Dashboard Admin**	API Admin + Statistiques
F	**Discounts & Rayons**	Gestion remises, codes promo, rayons (shelves)
K	**Tests & Documentation**	integration + PHPUnit + OpenAPI + Postman + README

Répartition détaillée par membre

****A – Architecture & Notifications****

****Responsabilité**** : Fondations + Communication

Tâches :

- **Setup Laravel****
 - `composer create-project laravel/laravel tp4`
 - `.env`, `APP_KEY`, `DB_*`, `MAIL_*`, `SANCTUM_STATEFUL_DOMAINS`
- **Structure API****
 - `routes/api.php` avec `Route::prefix('v1')`
 - Middleware `api`, `auth:sanctum`, `role:admin`
- **Services de notification****
 - `php artisan make:mail OrderConfirmed`
 - `php artisan make:mail PaymentFailed`
 - `php artisan make:mail DiscountApplied`
 - `php artisan make:event OrderCreated`
 - `php artisan make:listener SendOrderEmail`
 - Configuration ****Mailtrap**** ou ****log****
- **Queue simulée****
 - `QUEUE_CONNECTION=sync`
- **Documentation d'installation****
 - `README.md` complet
 - `.env.example`

```
**Livrables** :
- Projet Laravel prêt
- Notifications automatisées
- README clair

---

### **C — Base de données**
**Responsabilité** : Schéma complet + données de test

#### Tâches :
1. **Migrations** (conforme au schéma image) :
```php
users, cart, cart_item, product, product_variant,
category, discount, discount_code_usage, discount_codes,
orders, order_item, shelves, payment
```

## 2. Relations dans les modèles

- UserhasOne Cart, Cart hasMany CartItem
- Product hasMany ProductVariant, ProductVariant belongsTo Product
- Order belongsTo User, Order hasMany OrderItem
- Discount morphToMany Product, Discount hasMany DiscountCode
- Shelf hasMany Product, Product belongsTo Shelf

## 3. Factories & Seeders

- 50 produits, 10 catégories, 5 rayons, 8 utilisateurs
- 10 codes promo, 5 remises actives

### Livrables :

- Migrations complètes
- Modèles avec relations
- BDD prête à l'emploi

---

## G — Authentification (API)

**Responsabilité :** Sécurité & accès

### Tâches :

#### 1. Sanctum

```
php artisan vendor:publish --
provider="Laravel\Sanctum\SanctumServiceProvider"
```

#### 2. Endpoints

```
POST /api/v1/register
POST /api/v1/login
POST /api/v1/logout
GET /api/v1/user
```

### 3. Rôles

- `role` : USER, ADMIN, SUPERADMIN
- Middleware `role:admin`

### 4. Tests d'authentification

#### Livrables :

- Auth API sécurisée
- Tokens valides
- Protection des routes

---

## D — Gestion Produits (API)

**Responsabilité** : Catalogue complet

#### Tâches :

##### 1. Endpoints

```
GET /api/v1/products
GET /api/v1/products/{id}
POST /api/v1/products [admin]
PUT /api/v1/products/{id} [admin]
DELETE /api/v1/products/{id} [admin]
```

##### 2. Filtres & Recherche

- `?category=, ?price_min=, ?shelf=, ?search=, ?discount=`
- Pagination (`?page=, ?per_page=15`)

##### 3. Upload image

- `storage/app/public/products`
- Lien symbolique `php artisan storage:link`

#### Livrables :

- API produits complète
- Variantes (prix, couleur, stock)
- Filtres fonctionnels

---

## E — Gestion des Commandes (API)

**Responsabilité** : Cycle de commande

**Tâches :****1. Endpoints**

```
POST /api/v1/orders [checkout]
GET /api/v1/orders
GET /api/v1/orders/{id}
PUT /api/v1/admin/orders/{id}/status [admin]
```

**2. Statuts**

- pending, paid, preparing, shipped, delivered, canceled

**3. Création commande**

- Depuis panier
- Calcul total + frais de port
- Application des remises (via F)
- Décrémentation stock
- Déclenchement événement `OrderCreated`

**Livrables :**

- Commandes fonctionnelles
- Historique utilisateur
- Admin peut modifier statut

---

**I — Panier & Paiement (API)**

**Responsabilité :** Achat complet

**Tâches :****1. Panier API**

```
GET /api/v1/cart
POST /api/v1/cart/items
PUT /api/v1/cart/items/{id}
DELETE /api/v1/cart/items/{id}
DELETE /api/v1/cart
```

**2. Paiement simulé**

```
POST /api/v1/payments
```

- Numéro test 4242 4242 4242 4242 → succès
- Autre → échec

- Mise à jour `payment.status`

### 3. Checkout

- Validation stock
- Application code promo (via F)
- Création commande (via E)
- Vider panier

#### Livrables :

- Panier persistant
  - Paiement mock fiable
  - Flux d'achat complet
- 

## B — Dashboard Admin (API)

**Responsabilité :** Supervision

#### Tâches :

##### 1. Endpoints Admin

```
GET /api/v1/admin/stats
GET /api/v1/admin/orders
GET /api/v1/admin/products/low-stock
GET /api/v1/admin/discounts/usage
```

##### 2. Statistiques

- CA total, commandes jour, utilisateurs
- Produits en rupture
- Codes promo les plus utilisés

##### 3. Protégé par `role:admin`

#### Livrables :

- API admin complète
  - Stats en temps réel
- 

## F — Discounts & Rayons

**Responsabilité :** Promotions & Organisation

#### Tâches :

##### 1. Discounts

- `type:percentage, fixed`
- `start_date, end_date`
- Application auto sur produits

## 2. Codes Promo

- `code, nb_usage, max_usage`
- Table `discount_code_usage`
- Validation au checkout

## 3. Rayons (Shelves)

- `shelves : id, user_id, description`
- Produits assignés à un rayon
- Filtrage par rayon

## 4. Endpoints

```
GET /api/v1/discounts
POST /api/v1/discounts/apply [code]
GET /api/v1/shelves
GET /api/v1/shelves/{id}/products
```

### Livrables :

- Système de remise complet
- Codes promo fonctionnels
- Rayons organisés

---

## K — Tests & Documentation

**Responsabilité :** Qualité & Livraison

### Tâches :

#### 1. Tests PHPUnit

- Feature tests pour chaque endpoint
- $\geq 80\%$  coverage
- Tests d'intégration (panier → paiement → commande)

#### 2. OpenAPI (Swagger)

- `docs/openapi.yaml`
- Généré avec `l5-swagger`

#### 3. Postman Collection

- Tous les endpoints
- Variables d'environnement

#### 4. README.md

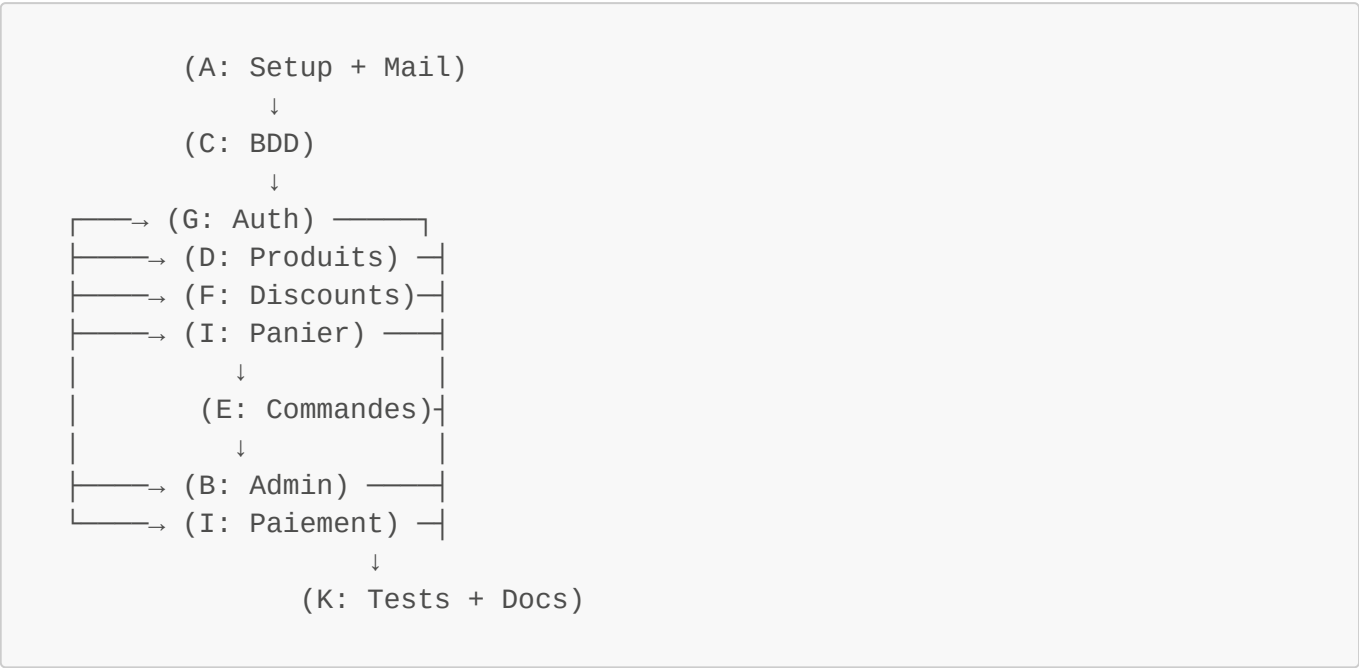
- Installation
- Comptes test (`admin@shopcart.com`, `client@shopcart.com`)
- Exemples de requêtes

### Livrables :

- Tests passant
- Documentation API complète

- Collection Postman

## Dépendances entre membres



## Checklist finale (Backend)

Tâche	Statut
Laravel installé	<input type="checkbox"/>
Migrations OK	<input type="checkbox"/>
Auth API (Sanctum)	<input type="checkbox"/>
CRUD Produits	<input type="checkbox"/>
Panier + Paiement	<input type="checkbox"/>
Commandes	<input type="checkbox"/>
Discounts & Codes	<input type="checkbox"/>
Rayons	<input type="checkbox"/>
Notifications mail	<input type="checkbox"/>
Dashboard Admin API	<input type="checkbox"/>
Tests PHPUnit ≥ 80%	<input type="checkbox"/>
OpenAPI + Postman	<input type="checkbox"/>

**Focus 100 % API — Aucun Blade public requis**  
**Prêt pour consommation par TP2 ou TP3**

---

**Bonne chance à l'équipe !**