API Laravel - Test des Endpoints

Ce README documente les tests des endpoints de l'API Laravel basée sur les routes définies dans `routes/api.php`. L'application est configurée avec Docker et utilise PostgreSQL.

Configuration

- Base URL : http://localhost:8000
- Version API : v1
- Authentification : Certaines routes nécessitent un token Bearer (non testé ici).

Endpoints Testés

1. GET /

- **URL** : http://localhost:8000/
- **Statut** : Redirige vers /api/documentation (probablement Swagger).
- **Réponse** : HTML de redirection.
- **Problème** : Pas un endpoint API direct.

2. GET /api/v1/users

- **URL** : http://localhost:8000/api/v1/users
- **Statut** : Échec (500 Server Error)
- **Réponse** : {"message": "Server Error"}
- **Problème** : Erreur serveur, probablement due à la configuration (APP_KEY vide, cache obsolète, ou connexion DB).

3. GET /api/v1/comptes

- **URL** : http://localhost:8000/api/v1/comptes
- **Statut** : Échec (500 Server Error)
- **Réponse** : {"message": "Server Error"}
- **Problème** : Même que ci-dessus.

PROF

4. GET /api/v1/transactions

- **URL** : http://localhost:8000/api/v1/transactions
- **Statut** : Échec (500 Server Error)
- **Réponse** : {"message": "Server Error"}
- **Problème** : Même que ci-dessus.

Endpoints Non Testés (Nécessitent Auth ou Données)

- POST /api/v1/users (créer user)
- PUT /api/v1/users/{id} (modifier user)
- DELETE /api/v1/users/{id} (supprimer user)

- GET /api/v1/users/{id} (détail user)
- POST /api/v1/comptes (créer compte)
- PUT /api/v1/comptes/{id} (modifier compte)
- DELETE /api/v1/comptes/{id} (supprimer compte)
- GET /api/v1/comptes/{id} (détail compte)
- POST /api/v1/comptes/{id}/bloquer (bloquer compte) - Nécessite auth ADMIN
- POST /api/v1/comptes/{id}/debloquer (débloquer compte) - Nécessite auth ADMIN
- POST /api/v1/transactions (créer transaction)
- GET /api/v1/transactions/{id} (détail transaction)

Endpoints Non Implémentés dans les Routes

Bien que les méthodes soient présentes dans les controllers, les routes suivantes ne sont pas définies dans `routes/api.php` :

- PUT /api/v1/transactions/{id} (modifier transaction)
- DELETE /api/v1/transactions/{id} (supprimer transaction)

Problèmes Identifiés

1. **Erreurs 500 sur tous les endpoints GET** : Probablement dues à :
    - APP_KEY vide ou non générée.
    - Cache de configuration obsolète (nécessite `php artisan config:clear`).
    - Connexion à la base de données (PostgreSQL) non configurée correctement.
    - Mode production avec APP_DEBUG=false masquant les détails.
2. **Routes avec Middleware Auth** : Les endpoints comme bloquer/debloquer nécessitent authentification, non testée ici.
3. **Routes Manquantes** : Update et delete pour transactions non routées.

Recommandations

PROF

- Corriger la configuration .env pour PostgreSQL.
- Générer APP_KEY : `docker compose exec -it app php artisan key:generate`
- Vider le cache : `docker compose exec app php artisan config:clear` et `cache:clear`
- Implémenter les routes manquantes si nécessaire.
- Tester avec authentification pour les endpoints protégés.

Fixes appliquées dans le dépôt

- Renommage des paramètres de route pour éviter l'injection automatique de modèles (route-model binding) qui causait des erreurs 500 :
    - `/{user}` -> `/{userId}`
    - `/{compte}` -> `/{compteId}`
    - `/{transaction}` -> `/{transactionId}`
- Ajout des routes manquantes pour les transactions :
    - PUT `/api/v1/transactions/{transactionId}`

    - DELETE `/api/v1/transactions/{transactionId}`
- Mise à jour des contrôleurs (`UserController`, `CompteController`, `TransactionController`) pour accepter et utiliser les identifiants (`userId`, `compteId`, `transactionId`) au lieu d'attendre des objets modèle (ce qui provoquait des TypeErrors).
- Suppression du cache des routes générées (`bootstrap/cache/routes-v7.php`) du dépôt pour éviter les conflits avec les routes mises à jour. Note : ce fichier est normalement généré par Laravel et n'est pas recommandé en VCS.

Ces corrections résolvent la majorité des erreurs 500 observées lors des appels GET de base.

Étapes recommandées après pull / changements

Après avoir récupéré ces changements sur votre machine de développement (ou dans le conteneur), exécutez :

```
# Générer APP_KEY si absent
docker compose exec -it app php artisan key:generate

# Vider le cache de configuration et des routes (important après
changement de routes)
docker compose exec -it app php artisan config:clear
docker compose exec -it app php artisan cache:clear
docker compose exec -it app php artisan route:clear

# (Optionnel) Régénérer le cache des routes en production
docker compose exec -it app php artisan route:cache
```

Ensuite, testez les endpoints :

```
curl -X GET http://localhost:8000/api/v1/users -H "Accept:
application/json"
curl -X GET http://localhost:8000/api/v1/comptes -H "Accept:
application/json"
curl -X GET http://localhost:8000/api/v1/transactions -H "Accept:
application/json"
```

PROF

Si vous continuez d'avoir des erreurs 500, activez temporairement APP_DEBUG=true dans votre `.env` (uniquement en développement) pour obtenir la stack trace et corriger les erreurs restantes.

Comment Lancer les Tests

Utilisez curl ou Postman pour tester :

```
curl -X GET http://localhost:8000/api/v1/users -H "Accept:
application/json"
```