

سجل تحديثات نظام الأساس المحاسبي

يحتوي هذا الملف على سجل شامل لجميع التحديثات والإضافات التي تمت على النظام، مرتبة من الأحدث إلى الأقدم.

الإصدار 1.2 - (17 نوفمبر 2025)

الميزات الجديدة ✨

• نظام دليل الحسابات الذكي

- تم بناء نظام دليل حسابات مرّن وذكي، وليس صارماً وغيبياً!
- دعم الحسابات الرئيسية (للترتيب والتقارير) و الحسابات الفرعية (للترحيل الفعلي).
- ترقيم مرّن: يمكن استخدام أي أرقام وتعديلها في أي وقت.
- هيكلية شجرية متعددة المستويات بدون حدود.
- كل حساب يمكن أن يكون له حساب أب (رئيسي أو فرعي).

• أنواع الحسابات القابلة للتخصيص

- جدول `account_types`: كل مؤسسة تُعرّف أنواع الحسابات الخاصة بها (أصول، خصوم، إيرادات، مصروفات...).
- لا توجد أنواع مفروضة مسبقاً - مرونة كاملة!
- كل نوع له طبيعة (مدين/دائن).

• نظام الحسابات التحليلية

- جدول `analytical_account_types`: أنواع الحسابات التحليلية (صندوق، بنك، صراف، محفظة، مورد، عميل...).
- جدول `analytical_accounts`: الحسابات التحليلية الفعلية (مثل: صندوق بغداد، بنك الرافدين...).
- الربط الذكي: الحسابات الفرعية في الدليل تُربط بنوع تحليلي، ثم تُضاف الحسابات التحليلية الفعلية.

- **مثال:** حساب فرعي "الصندوق الرئيسي" (نوع: صندوق) ← حسابات تحليلية: "صندوق بغداد", "صندوق البصرة".

• تحسين دليل النظام 🎨:

- إعادة كتابة دليل النظام بطريقة **ملونة وذكية وممتعة**.
- إضافة رموز تعبيرية (Emojis) ومخططات توضيحية.
- شرح مفصل لكل خطوة في سير العمل.
- أمثلة عملية واقعية.

التحسينات ➡

• تحديث Models:

- تم إنشاء Models `AccountType`, `AnalyticalAccountType`, `AnalyticalAccount`
- تم تحديث `Account Model` ليدعم `is_main` (رئيسي/فرعي) و `analytical_account_type_id`
- إضافة Scopes متقدمة (`mainAccounts`, `subAccounts`, `ofType`, `ofAnalyticalType`)

• تحديث Migrations:

- migration شامل لإنشاء/تحديث جداول: `account_types`, `analytical_account_types`, `analytical_accounts`, `accounts`
- إضافة حقل `is_main` لتمييز الحسابات الرئيسية عن الفرعية.
- إضافة حقل `analytical_account_type_id` للحسابات الفرعية.

• تحديث Controllers:

- تم إنشاء `AccountTypeController` و `AnalyticalAccountTypeController`
- جاهز لإكمال الواجهات في المرحلة القادمة.

التوثيق 📖

- دليل النظام **المحدّث**: شرح شامل ومفصل بطريقة ملونة وممتعة.
 - **سجل التحديثات**: تحديث CHANGELOG بجميع التغييرات.
-

الميزات الجديدة ✨

• إضافة نظام المؤسسات والوحدات:

- تم بناء نظام متكامل لإدارة المؤسسات والوحدات، مع دعم الهيكلية الهرمية الكاملة: **الوحدة** ← **المؤسسة** ← **الفرع**.
- كل مؤسسة تعمل كنظام محاسبي مستقل تماماً، مع دليل حسابات وبيانات منفصلة.
- تم إنشاء جداول `companies` و `units` وتحديث جدول `branches`.
- تم إنشاء `CompanyController` و `UnitController` مع جميع العمليات (CRUD).
- تم إنشاء واجهات احترافية لإدارة المؤسسات والوحدات.

• إضافة صفحة دليل النظام:

- تم إنشاء صفحة "الدليل" داخل النظام لعرض فكرة النظام والهيكلية وسجل التحديثات.
- تحتوي الصفحة على تبويين: "دليل النظام" و "تحديثات النظام".
- إمكانية تحميل دليل النظام وسجل التحديثات بصيغة PDF.

التحسينات

• تحديث العلاقات في Models:

- تم تحديث `Company`, `Unit`, `Branch` Models لتعكس الهيكلية الجديدة.
- تم إضافة علاقات `Has-Many-Through` لتسهيل الوصول إلى البيانات عبر المستويات الهرمية.

• تحديث الـ Migrations:

- تم إنشاء migration جديد لتعديل الهيكلية وإضافة الحقول المطلوبة (`unit_id`, `company_id`).

إصلاح الأخطاء

- **تعديل الهيكلية الأولية:** تم تعديل الهيكلية من (المؤسسة ← الوحدة) إلى (الوحدة ← المؤسسة) بناءً على المتطلبات الجديدة.

الميزات الجديدة ✨

• بناء نظام سندات الصرف والقبض:

- تم بناء نظام متكامل لإدارة سندات الصرف والقبض باستخدام Laravel.
- دعم أنواع الدفع (نقدي/بنكي) وحالات متعددة للسند (مسودة، قيد الانتظار، معتمد، مرفوض).
- توليد تلقائي لأرقام السندات (PAY/REC).
- واجهات عربية احترافية (RTL) مع فلاتر بحث متقدمة.

• إنشاء `VoucherController`:

- تم إنشاء Controller متكامل بـ 11 دالة لإدارة جميع عمليات السندات.
- AJAX endpoint لتصفية الحسابات ديناميكياً حسب الفرع وطريقة الدفع.

• إنشاء واجهات السندات:

- `index`: عرض قائمة السندات مع فلاتر.
- `create`: نموذج إنشاء سند جديد.
- `edit`: نموذج تعديل السند.
- `show`: عرض وطباعة السند.

• إنشاء جدول `vouchers`:

- تم إنشاء migration لجدول السندات بجميع الحقول المطلوبة.

إصلاح الأخطاء 🐛

- **مشكلة `deleted_at`**: تم حل مشكلة عدم وجود عمود `deleted_at` عن طريق إزالة `SoftDeletes` من `Voucher Model`.
- **مشكلة `Branch Model`**: تم حل مشكلة عدم وجود `Branch Model` عن طريق إنشائه وإضافة جدول `branches`.