نسخة المنزل - نظام الخواجه لإدارة العملاء (CRM)

نظرة عامة

نظام إدارة العملاء الشامل "الخواجه" مصمم لدعم الشركات في إدارة عملائها، والمبيعات، والإنتاج، والمخزون بكفاءة عالية. النظام يدعم الآن إدارة قواعد البيانات المتعددة وعمليات النسخ الاحتياطي والاستعادة.

المميزات الرئيسية

إدارة العملاء

- إضافة وتعديل بيانات العملاء
- تتبع تاريخ التعاملات مع كل عميل
- إدارة معلومات الاتصال والعناوين
- تصنيف العملاء حسب الأولوية

إدارة الطلبات والمبيعات

- إنشاء وتتبع الطلبات
- إدارة حالات الطلبات (جديد، قيد التنفيذ، مكتمل)
- حساب الأسعار والخصومات
- تتبع المدفوعات والمستحقات

إدارة المخزون

- تتبع المواد الخام والمنتجات النهائية
- إدارة حركات المخزون (دخول، خروج، تحويل)
- تنبيهات عند انخفاض المخزون
- تقارير المخزون المفصلة

نظام الإنتاج

- جدولة عمليات الإنتاج
- تتبع مراحل التصنيع
- إدارة الموارد والعمالة
- تقارير الإنتاجية

المعاينات والتركيبات

- جدولة المعاينات الفنية
- تتبع حالة التركيبات
- إدارة فرق العمل
- تقارير الأداء

إدارة قواعد البيانات المتقدمة

- **نظام النسخ الاحتياطية المجدولة**: إنشاء نسخ احتياطية تلقائية بجدولة مرنة (كل ساعة، يومياً، أسبوعياً، شهرياً)

- **إدارة متعددة لقواعد البيانات**: دعم SQLite و PostgreSQL
- **استعادة النسخ الاحتياطية**: استعادة كاملة أو جزئية للبيانات
- **تصدير واستيراد البيانات**: دعم تنسيقات متعددة (JSON، SQL، CSV)
- **مراقبة الأداء**: تتبع حجم قواعد البيانات وأداء العمليات

التقارير والإحصائيات

- تقارير المبيعات والأرباح
- تحليل أداء العملاء
- إحصائيات الإنتاج والمخزون
- تقارير مالية شاملة

الأمان والنسخ الاحتياطي

- نسخ احتياطية مجدولة مع حد أقصى 24 نسخة
- حذف تلقائي للنسخ القديمة
- تشفير البيانات الحساسة
- نظام صلاحيات متقدم

المتطلبات التقنية

- Python 3.11+
- Django 4.2
- PostgreSQL أو SQLite
- Redis (للقنوات والويب سوكت)
- Railway (للنشر)

إعداد المشروع

1. استنساخ المستودع

```
git clone https://github.com/zakeetahawi/homeupdate.git
cd homeupdate
```

2. إنشاء بيئة افتراضية

```
python -m venv venv
source venv/bin/activate  # على Linux/macOS
venv\Scripts\activate  # على Windows
```

3. تثبيت التبعيات

```
pip install -r requirements.txt
```

4. قاعدة البيانات

في بيئة التطوير المحلية، يمكنك استخدام SQLite أو PostgreSQL.

محليًا PostgreSQL لاستخدام:

```
createdb crm_system
```

في بيئة الإنتاج (Railway):

يتم إنشاء قاعدة بيانات PostgreSQL تلقائيًا عند إضافة خدمة PostgreSQL إلى مشروعك على Railway.

5. الترحيلات

```
python manage.py migrate
```

عادةً لن تحتاج إلا إذا غيرت النماذج أو أردت تحديث قاعدة البيانات.

6. إنشاء مستخدم مدير (Superuser)

```
python manage.py createsuperuser
```

أو يمكنك تسجيل الدخول مباشرة إذا كان لديك مستخدم بالفعل في قاعدة البيانات الحالية.

7. تشغيل الخادم

```
python manage.py runserver
```

استخدام ميزة النسخ الاحتياطية المجدولة

إنشاء جدولة نسخ احتياطية جديدة

1. انتقل إلى قسم "إدارة قواعد البيانات" في النظام
2. اختر "جدولة النسخ الاحتياطية"
3. انقر على "إنشاء جدولة جديدة"
4. املأ البيانات المطلوبة:
    - **اسم الجدولة**: اسم وصفي للجدولة
    - **نوع النسخة الاحتياطية**:
        - `full`: نسخة كاملة من جميع البيانات
        - `customers`: بيانات العملاء فقط
        - `users`: بيانات المستخدمين فقط

        - `settings`: إعدادات النظام فقط
    - **التكرار**:
        - `hourly`: كل ساعة
        - `daily`: يومياً
        - `weekly`: أسبوعياً
        - `monthly`: شهرياً
    - **وقت التنفيذ**: الساعة والدقيقة المطلوبة
    - **الحد الأقصى للنسخ**: عدد النسخ الاحتياطية المحفوظة (الافتراضي: 24)

إدارة النسخ الاحتياطية

- **عرض النسخ الاحتياطية**: يمكنك عرض جميع النسخ الاحتياطية مع تفاصيلها
- **تنزيل النسخة الاحتياطية**: انقر على أيقونة التنزيل لحفظ النسخة محلياً
- **استعادة النسخة الاحتياطية**: انقر على أيقونة الاستعادة لاستعادة البيانات
- **حذف النسخة الاحتياطية**: انقر على أيقونة الحذف لإزالة النسخة

الميزات المتقدمة

- **الحذف التلقائي**: يتم حذف النسخ القديمة تلقائياً عند تجاوز الحد الأقصى
- **مراقبة الأداء**: تتبع حجم النسخ الاحتياطية ووقت إنشائها
- **التنبيهات**: إشعارات عند فشل إنشاء النسخة الاحتياطية
- **دعم تنسيقات متعددة**: JSON مضغوط، SQL، وتنسيقات أخرى

التكوين

- لتكوين إعدادات المشروع قم بتعديل `crm/settings.py`
- لتكوين متغيرات البيئة في بيئة التطوير استخدم ملف `.env`
- كمرجع لإعداد متغيرات البيئة في الإنتاج استخدم ملف `.env.production`

النشر على الإنترنت

1. استخدام Cloudflare Tunnel

يمكنك تشغيل التطبيق على الإنترنت مباشرة من جهازك باستخدام:

```
./تشغيل_CRM_اونلاين.bat
```

2. إعداد قاعدة البيانات للإنتاج

1. تأكد من تشغيل PostgreSQL
2. قم بإنشاء قاعدة بيانات جديدة للإنتاج
3. قم بتحديث إعدادات قاعدة البيانات في النظام

3. ترحيل البيانات (اختياري)

إذا كنت تريد نقل البيانات من قاعدة بيانات إلى أخرى:

1. قم بتصدير البيانات:

```
python manage.py dumpdata --exclude auth.permission --exclude
contenttypes > data.json
```

2. قم باستيراد البيانات:

```
python manage.py loaddata data.json
```

استكشاف الأخطاء وإصلاحها

مشاكل شائعة وحلولها

1. مشكلة ترحيل accounts.fix_user_model_swap

إذا ظهرت رسالة تحذيرية حول ترحيل غير مطبق:

```
You have 1 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): accounts.
```

الحل: هذا التحذير طبيعي ولا يؤثر على عمل النظام. تم تجاوز هذا الترحيل تلقائياً لتجنب المشاكل.

2. مشكلة عدم تطابق إصدارات PostgreSQL

إذا ظهرت رسالة خطأ:

```
pg_dump: error: server version: 17.4; pg_dump version: 14.17
```

الحل: النظام يتعامل مع هذه المشكلة تلقائياً ويستخدم طريقة بديلة لإنشاء النسخ الاحتياطية.

3. مشكلة ترميز الأحرف العربية

إذا واجهت مشاكل في عرض الأحرف العربية:

الحل: تأكد من:

- استخدام UTF-8 في قاعدة البيانات
- تعيين `LANGUAGE_CODE = 'ar'` في settings.py
- استخدام متصفح يدعم UTF-8

4. مشكلة فشل النسخ الاحتياطية

إذا فشل إنشاء النسخة الاحتياطية:

الحل:

1. تحقق من صلاحيات الكتابة في مجلد النسخ الاحتياطية
2. تأكد من وجود مساحة كافية على القرص
3. تحقق من اتصال قاعدة البيانات

5. مشكلة عدم عمل الجدولة

إذا لم تعمل النسخ الاحتياطية المجدولة:

الحل:

1. تأكد من تشغيل الخادم بشكل مستمر
2. تحقق من إعدادات المنطقة الزمنية
3. راجع سجلات النظام للأخطاء

نصائح للأداء الأمثل

1. تحسين قاعدة البيانات

- قم بتنظيف البيانات القديمة بانتظام
- استخدم فهارس مناسبة للاستعلامات المتكررة
- راقب حجم قاعدة البيانات

2. إدارة النسخ الاحتياطية

- اختر التكرار المناسب حسب أهمية البيانات
- احتفظ بنسخ احتياطية خارجية للبيانات الحساسة
- اختبر عملية الاستعادة بانتظام

3. الأمان

- استخدم كلمات مرور قوية
- فعّل HTTPS في بيئة الإنتاج
- راجع سجلات الوصول بانتظام

المساهمة

1. قم بعمل fork للمشروع
2. أنشئ branch جديد (`git checkout -b feature/AmazingFeature`)
3. التزم بتغييراتك (`git commit -m 'Add some AmazingFeature'`)
4. ادفع إلى البرانش (`git push origin feature/AmazingFeature`)
5. افتح طلب سحب

الإصدارات والتحديثات

الإصدار الحالي: v2.0.0

تاريخ الإصدار: ديسمبر 2024

الميزات الجديدة:

- ☑ نظام النسخ الاحتياطية المجدولة مع دعم تكرارات مختلفة
- ☑ إدارة متقدمة لقواعد البيانات مع دعم SQLite و PostgreSQL
- ☑ واجهة محسنة للإجراءات باستخدام أيقونات واضحة
- ☑ نظام حذف تلقائي للنسخ القديمة (حد أقصى 24 نسخة)
- ☑ دعم تنسيقات متعددة للنسخ الاحتياطية
- ☑ معالجة تلقائية لمشاكل ترميز الأحرف العربية
- ☑ حل مشكلة عدم تطابق إصدارات PostgreSQL

التحسينات:

- 🔧 تحسين أداء عمليات النسخ الاحتياطي
- 🔧 واجهة مستخدم محسنة ومتجاوبة
- 🔧 رسائل خطأ أكثر وضوحاً
- 🔧 تحسين نظام الترحيلات التلقائية

إصلاح الأخطاء:

- 🩹 إصلاح مشكلة ترميز الأحرف العربية في النسخ الاحتياطية
- 🩹 إصلاح مشكلة تنفيذ الترحيلات مرتين
- 🩹 إصلاح مشكلة تسلسل كائنات datetime إلى JSON
- 🩹 إصلاح مشكلة عدم تطابق إصدارات pg_dump

الإصدارات السابقة:

v1.5.0 (نوفمبر 2024)

- إضافة نظام إدارة قواعد البيانات الأساسي
- دعم استيراد وتصدير البيانات
- تحسينات في واجهة المستخدم

v1.0.0 (أكتوبر 2024)

- الإصدار الأولي من نظام إدارة العملاء
- إدارة العملاء والطلبات
- نظام المخزون والإنتاج

خطة التطوير المستقبلية:

v2.1.0 (مخطط - يناير 2025)

- 🔮 إضافة نظام التنبيهات والإشعارات
- 🔮 تحسين تقارير الأداء والإحصائيات
- 🔮 دعم النسخ الاحتياطية السحابية (Google Drive, AWS S3)

v2.2.0 (مخطط - فبراير 2025)

- ⚽ نظام إدارة المستخدمين المتقدم
- ⚽ واجهة برمجة تطبيقات (API) شاملة
- ⚽ تطبيق الهاتف المحمول

الترخيص

جهات الاتصال

- **المطور**: Zakee Tahawi
- **البريد الإلكتروني**: zakeetahawi@gmail.com
- **رابط المشروع**: https://github.com/zakeetahawi/homeupdate
- للحصول على المساعدة GitHub على issue **الدعم الفني**: يمكنك فتح

شكر وتقدير

- شكر خاص لمجتمع Django على الأدوات الرائعة
- شكر لفريق PostgreSQL على قاعدة البيانات الموثوقة
- شكر لمنصة Railway على خدمة الاستضافة المتميزة