كاشف التصيد الإلكتروني - Phishing Detector

نظرة عامة

كاشف التصيد الإلكتروني هو أداة ذكية مفتوحة المصدر تستخدم تقنيات الذكاء الاصطناعي لكشف الروابط والمواقع الخبيثة قبل أن يتم النقر عليها. تم تطوير هذه الأداة لحماية المستخدمين من محاولات التصيد الإلكتروني والمواقع الاحتيالية.

المميزات

- **كشف ذكي:** استخدام خوارزميات التعلم الآلي المتقدمة لتحليل الروابط
- **واجهة ويب سهلة الاستخدام:** واجهة عربية بسيطة وجذابة
- **واجهة سطر أوامر:** للاستخدام المتقدم والتكامل مع الأنظمة الأخرى
- **مفتوح المصدر:** متاح للجميع للاستخدام والتطوير
- **سريع ودقيق:** نتائج فورية بدقة عالية

التقنيات المستخدمة

الذكاء الاصطناعي

- **Scikit-learn**: لبناء وتدريب نماذج التعلم الآلي
- **Pandas**: لمعالجة وتحليل البيانات
- **NumPy**: للعمليات الرياضية والمصفوفات

الواجهة الخلفية (Backend)

- **Flask**: إطار عمل Python للخادم
- **Flask-CORS**: للسماح بالطلبات من مصادر مختلفة
- **Joblib**: لحفظ وتحميل النماذج المدربة

الواجهة الأمامية (Frontend)

- **React**: مكتبة JavaScript لبناء واجهات المستخدم
- **Tailwind CSS**: لتصميم واجهة عصرية ومتجاوبة
- **Shadcn/UI**: مكونات واجهة مستخدم جاهزة

- **Lucide React**: أيقونات عصرية

التثبيت والتشغيل

متطلبات النظام

- Python 3.8 أو أحدث
- Node.js 16 أو أحدث
- npm أو pnpm

تثبيت الواجهة الخلفية

```
# استنساخ المشروع
https://github.com/your-username/phishing-detector.git clone git
phishing-detector/phishing-detector-backend cd

# إنشاء بيئة افتراضية
venv venv m- python
venv/bin/activate source  # على Linux/Mac
# أو
activate\\Scripts\\venv  # على Windows

# تثبيت المتطلبات
requirements.txt r- install pip

# تشغيل الخادم
src/main.py python
```

تثبيت الواجهة الأمامية

```
phishing-detector-frontend/.. cd

# تثبيت المتطلبات
install pnpm

# بناء المشروع
build run pnpm

# نسخ الملفات إلى مجلد Flask
/phishing-detector-backend/src/static/.. */dist r- cp
```

الاستخدام

واجهة الويب

1. افتح المتصفح وانتقل إلى `http://localhost:5000`
2. أدخل الرابط الذي تريد فحصه
3. انقر على "فحص الرابط"
4. ستظهر النتيجة مع تفاصيل التحليل

واجهة سطر الأوامر

```
https://example.com cli.py python
```

كيف يعمل

استخراج الخصائص

تقوم الأداة باستخراج عدة خصائص من الرابط المدخل:

- **طول الرابط:** الروابط الطويلة غالباً ما تكون مشبوهة
- **عدد النقاط:** كثرة النقاط قد تشير لنطاقات فرعية مشبوهة
- **استخدام HTTPS**: المواقع الآمنة تستخدم HTTPS
- **طول المسار:** المسارات الطويلة قد تكون مشبوهة
- **عدد الشرطات:** كثرة الشرطات في اسم النطاق مشبوهة
- **النطاقات الفرعية:** عدد النطاقات الفرعية

نماذج التعلم الآلي

تم تدريب عدة نماذج واختبارها:

- **Naive Bayes**: نموذج احتمالي بسيط
- **Logistic Regression**: نموذج خطي للتصنيف
- **Support Vector Machine**: نموذج متقدم للتصنيف
- **Random Forest**: مجموعة من أشجار القرار

المساهمة

نرحب بالمساهمات من الجميع! يمكنك المساهمة عبر:

1. **الإبلاغ عن الأخطاء:** افتح issue جديد
2. **اقتراح مميزات:** شارك أفكارك لتحسين الأداة
3. **تطوير الكود:** أرسل Pull Request
4. **تحسين الوثائق:** ساعد في تحسين التوثيق

خطوات المساهمة

1. Fork المشروع
2. إنشاء فرع جديد (`git checkout -b feature/amazing-feature`)
3. Commit التغييرات (`git commit -m 'Add amazing feature'`)
4. Push للفرع (`git push origin feature/amazing-feature`)
5. فتح Pull Request

الترخيص

هذا المشروع مرخص تحت رخصة MIT - انظر ملف LICENSE للتفاصيل.

التواصل

- **البريد الإلكتروني:** mabbadi0@icloud.com
- **الهاتف:** 0592774301

شكر وتقدير

- شكر خاص لمجتمع المطورين مفتوحي المصدر
- البيانات المستخدمة في التدريب من مصادر عامة متاحة
- الأيقونات من Lucide Icons
- التصميم مستوحى من أفضل الممارسات في UX/UI

ملاحظة: هذه الأداة مخصصة للأغراض التعليمية والحماية. استخدمها بمسؤولية ولا تعتمد عليها كحل أمني وحيد.