🌿 زيتون ستور - نظام إدارة الموارد (ERP)

📋 نظرة عامة

نظام ERP متكامل لإدارة متجر الإلكترونيات "زيتون ستور" مبني بتقنيات حديثة وجاهز للعمل على السحابة.

✨ المميزات الرئيسية

- ✅ **واجهة عربية بالكامل** مع دعم RTL
- 🏢 **دعم تعدد الفروع** مع إدارة مستقلة لكل فرع
- 📦 **إدارة المخزون** مع نظام السيريال نمبر والباركود
- 🛒 **نقطة البيع** مع إنشاء وطباعة الفواتير
- 📊 **لوحة تحكم تفاعلية** مع إحصائيات ورسوم بيانية
- 👥 **نظام ولاء العملاء** مع النقاط والعروض الخاصة
- 🔔 **تنبيهات ذكية** عند انخفاض المخزون
- 💰 **إدارة المصروفات** مع التقارير المالية
- 🔐 **نظام صلاحيات متقدم** لأدوار المستخدمين
- 📱 **إشعارات SMS وEmail** عبر Twilio و SendGrid

🛠️ التقنيات المستخدمة

Backend

- Node.js + Express.js
- PostgreSQL + Sequelize ORM
- JWT للمصادقة
- Socket.io للتحديثات الفورية
- Twilio للرسائل النصية
- SendGrid للبريد الإلكتروني

Frontend

- React.js 18
- Tailwind CSS 3
- Recharts للرسوم البيانية

- React Query لإدارة البيانات
- Socket.io Client

DevOps

- Docker & Docker Compose
- Nginx كـ Reverse Proxy
- جاهز للنشر على AWS/Azure/Google Cloud

🚀 التثبيت والتشغيل

المتطلبات الأساسية

- Docker & Docker Compose
- Node.js 18+ (للتطوير المحلي)
- Git

خطوات التثبيت

1. **استنساخ المشروع**

```bash
git clone https://github.com/your-repo/zaitoon-store-erp.git
cd zaitoon-store-erp
```

2. **إعداد متغيرات البيئة**

```bash
# Backend
cp backend/.env.example backend/.env

# Frontend
cp frontend/.env.example frontend/.env
```

3. **تحديث ملف `.env` للـ Backend**

```env
```

```
NODE_ENV=production
PORT=5000

# Database
DB_HOST=postgres
DB_PORT=5432
DB_NAME=zaitoon_erp
DB_USER=zaitoon_admin
DB_PASSWORD=your-secure-password

# JWT
JWT_SECRET=your-jwt-secret-key
JWT_EXPIRE=7d

# Twilio (للرسائل النصية)
TWILIO_ACCOUNT_SID=your-twilio-sid
TWILIO_AUTH_TOKEN=your-twilio-token
TWILIO_PHONE_NUMBER=+201234567890

# SendGrid (للبريد الإلكتروني)
SENDGRID_API_KEY=your-sendgrid-key
SENDGRID_FROM_EMAIL=noreply@zaitoon-store.com

# Frontend URL
FRONTEND_URL=http://localhost:3000
```

4. **تحديث ملف `.env` للـ Frontend**

```
env

REACT_APP_API_URL=http://localhost:5000/api
REACT_APP_SOCKET_URL=http://localhost:5000
```

5. **تشغيل المشروع باستخدام Docker**

```
bash

docker-compose up -d
```

6. **إنشاء قاعدة البيانات والجداول**

```
bash
```

```
docker exec -it zaitoon_backend npm run migrate
```

التشغيل للتطوير المحلي

Backend:

```bash
cd backend
npm install
npm run dev
```

Frontend:

```bash
cd frontend
npm install
npm start
```

📂 هيكل المشروع

```
zaitoon-store-erp/
├── backend/          # خادم API
├── frontend/         # واجهة React
├── nginx/            # إعدادات Nginx
├── docker-compose.yml
└── README.md
```

👤 المستخدم الافتراضي

```
Username: admin
Password: Admin@123
```

🔑 أدوار المستخدمين والصلاحيات

الدور	الصلاحيات
admin	كامل الصلاحيات على النظام
manager	إدارة الفرع والموظفين والتقارير
warehouse	إدارة المخزون والمنتجات
cashier	نقطة البيع والفواتير
viewer	عرض التقارير فقط

🌐 النشر على السحابة

AWS

1. إنشاء EC2 Instance
2. تثبيت Docker و Docker Compose
3. نسخ المشروع وتشغيله
4. إعداد Load Balancer و Route 53

Azure

1. إنشاء Virtual Machine
2. استخدام Azure Container Instances
3. إعداد Azure Database for PostgreSQL

Google Cloud

1. استخدام Compute Engine
2. أو Google Kubernetes Engine
3. Cloud SQL لـ PostgreSQL

📊 قاعدة البيانات

الجداول الرئيسية

- `branches` - الفروع
- `users` - المستخدمين
- `products` - المنتجات
- `serial_numbers` - الأرقام التسلسلية
- `inventory` - المخزون
- `sales` - المبيعات

- `customers` - العملاء
- `loyalty_transactions` - نقاط الولاء
- `expenses` - المصروفات
- `invoices` - الفواتير

🔄 التحديثات الفورية

النظام يستخدم Socket.io لتوفير تحديثات فورية:

- تنبيهات انخفاض المخزون
- إشعارات المبيعات الجديدة
- تحديث الإحصائيات مباشرة

📈 التقارير المتاحة

- تقرير المبيعات اليومي/الأسبوعي/الشهري
- تقرير حركة المخزون
- تقرير الأرباح والخسائر
- تقرير أداء الفروع
- تقرير ولاء العملاء
- تقرير المصروفات

🔧 API Endpoints

Authentication

- `POST /api/auth/login`
- `POST /api/auth/logout`
- `GET /api/auth/profile`

Products

- `GET /api/products`
- `POST /api/products`
- `PUT /api/products/:id`
- `DELETE /api/products/:id`

Sales

- `GET /api/sales`
- `POST /api/sales`
- `GET /api/sales/:id`

Inventory

- `GET /api/inventory`
- `POST /api/inventory/scan`
- `PUT /api/inventory/transfer`

🐛 حل المشاكل الشائعة

مشكلة الاتصال بقاعدة البيانات

```bash
docker-compose down
docker-compose up -d postgres
docker-compose up -d
```

مشكلة الصلاحيات

```bash
docker exec -it zaitoon_backend npm run seed:permissions
```

إعادة بناء الحاويات

```bash
docker-compose build --no-cache
docker-compose up -d
```

📝 الترخيص

هذا المشروع مرخص تحت رخصة MIT

🤝 المساهمة

نرحب بالمساهمات! يرجى إنشاء Issue أو Pull Request

📞 الدعم

للدعم الفني: support@zaitoon-store.com

🎯 خارطة الطريق

- [ ] تطبيق الجوال (React Native)
- [ ] نظام الفواتير الإلكترونية
- [ ] التكامل مع بوابات الدفع المحلية
- [ ] نظام إدارة الموردين
- [ ] تقارير الذكاء الاصطناعي
- [ ] نظام الجرد الآلي
- [ ] التكامل مع الضرائب المصرية

🏗️ البنية التحتية

mermaid

```
graph TD
    A[المستخدم] --> B[Nginx]
    B --> C[Frontend React]
    B --> D[Backend API]
    D --> E[PostgreSQL]
    D --> F[Redis Cache]
    D --> G[Socket.io]
    D --> H[Twilio SMS]
    D --> I[SendGrid Email]
```

🔒 الأمان

- تشفير كلمات المرور باستخدام bcrypt
- JWT tokens مع انتهاء الصلاحية
- Rate limiting للحماية من DDoS
- Input validation على جميع النقاط
- SQL injection protection عبر Sequelize ORM
- CORS configuration
- Helmet.js للحماية الإضافية

📱 الباركود والسيريال نمبر

إضافة منتج بالسيريال نمبر

1. اختر "إضافة منتج" من قائمة المخزون

2. امسح الباركود أو أدخل السيريال يدوياً
3. النظام يحفظ كل قطعة بشكل منفصل
4. عند البيع، يتم ربط السيريال بالفاتورة

تتبع المنتج

- معرفة تاريخ الشراء والبيع
- معرفة العميل المشتري
- تتبع الضمان
- سجل الصيانة

💳 نظام نقاط الولاء

كيفية العمل

1. **التسجيل**: العميل يحصل على 50 نقطة ترحيبية
2. **الشراء**: 1 نقطة لكل 10 جنيه
3. **المستويات**:
   - برونزي: 0-500 نقطة
   - فضي: 501-1000 نقطة
   - ذهبي: 1001-2000 نقطة
   - بلاتيني: 2000+ نقطة
4. **المكافآت**: خصومات وعروض خاصة حسب المستوى

🎨 التخصيص

تغيير الألوان

في ملف `tailwind.config.js`:

```javascript
colors: {
  primary: {
    500: '#FFD700', // اللون الأساسي
  }
}
```

تغيير اللوجو

استبدل الملفات في:

- `frontend/public/favicon.ico`
- `frontend/src/assets/logo.png`

إضافة لغات جديدة

1. أضف ملف الترجمة في `frontend/src/locales/`
2. حدث `i18n.js`
3. أضف اللغة في قائمة الإعدادات

📊 أمثلة على الاستخدام

إضافة فرع جديد

```javascript
POST /api/branches
{
  "name": "فرع المعادي",
  "code": "MAADI",
  "address": "شارع 9، المعادي",
  "phone": "0227654321",
  "manager_name": "محمد أحمد"
}
```

إنشاء فاتورة

```javascript
```

```
POST /api/sales
{
  "customer_id": "uuid",
  "branch_id": "uuid",
  "items": [
    {
      "product_id": "uuid",
      "serial_number_id": "uuid",
      "quantity": 1,
      "unit_price": 5000
    }
  ],
  "payment_method": "cash"
}
```

🧪 الاختبار

تشغيل الاختبارات

```bash
# Backend tests
cd backend
npm test

# Frontend tests
cd frontend
npm test
```

اختبار الأداء

```bash
npm run test:performance
```

📦 النسخ الاحتياطي

نسخ قاعدة البيانات

```bash
docker exec zaitoon_db pg_dump -U zaitoon_admin zaitoon_erp > backup.sql
```

استعادة قاعدة البيانات

```bash
docker exec -i zaitoon_db psql -U zaitoon_admin zaitoon_erp < backup.sql
```

النسخ الاحتياطي التلقائي

إعداد cron job:

```bash
0 2 * * * /path/to/backup-script.sh
```

🌍 البيئات المختلفة

Development

```bash
NODE_ENV=development
DEBUG=true
```

Staging

```bash
NODE_ENV=staging
DEBUG=false
```

Production

```bash
NODE_ENV=production
DEBUG=false
ENABLE_MONITORING=true
```

📈 المراقبة والتحليل

إعداد Monitoring

- استخدام PM2 لإدارة العمليات

- تكامل مع DataDog أو New Relic
- إعداد Grafana للرسوم البيانية

Logs

- جميع السجلات في `/logs`
- تدوير يومي للملفات
- مستويات: ERROR, WARN, INFO, DEBUG

🔄 التحديثات

تحديث النظام

```bash
git pull origin main
docker-compose down
docker-compose build
docker-compose up -d
docker exec -it zaitoon_backend npm run migrate
```

التراجع عن التحديث

```bash
git checkout [previous-version]
docker-compose down
docker-compose up -d
```

🎓 الموارد التعليمية

- دليل المستخدم
- دليل المطور
- API Documentation
- فيديوهات تعليمية

🏆 الشكر والتقدير

شكر خاص لجميع المساهمين في المشاريع مفتوحة المصدر التي استخدمناها.

تم التطوير بـ ❤️ لمتجر زيتون ستور