



SwiftStock WMS

Modern Depo Yönetim Sistemi

Hızlı • Akıllı • Güvenilir

Gerçek zamanlı stok takibi, barkod okuma ve operasyon yönetimi

SwiftStock Nedir?

SwiftStock, modern depoların ihtiyaçlarına yönelik, **tamamen yerli ve açık kaynak** bir Warehouse Management System (WMS) çözümüdür.

✓ Temel Özellikler

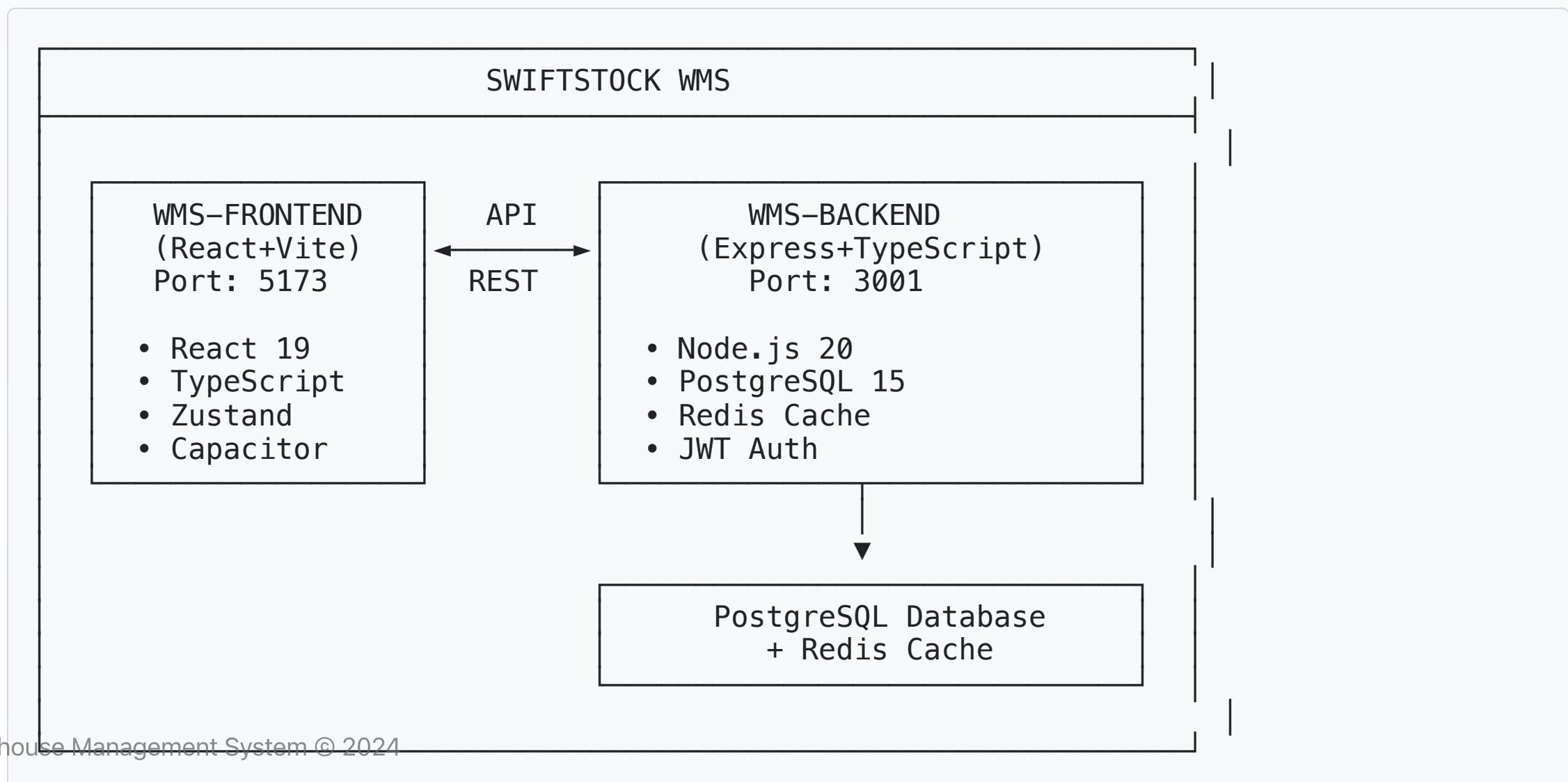
- Mobil-öncelikli tasarım
- Gerçek zamanlı stok takibi
- Barkod/QR kod okuma
- Sevkiyat yönetimi
- Sayım operasyonları
- Rol tabanlı yetkilendirme

🎨 Avantajlar

- Hızlı ve responsive
- Web ve mobil uyumlu
- Offline çalışma desteği
- Ölçeklenebilir mimari
- Kurumsal güvenlik
- Açık kaynak ve ücretsiz



Sistem Mimarisi





Teknoloji Stack

Backend

- **Runtime:** Node.js 20.x
- **Framework:** Express.js
- **Language:** TypeScript
- **Database:** PostgreSQL 15
- **Cache:** Redis 7
- **Auth:** JWT + Google OAuth
- **Validation:** Zod
- **API Docs:** Swagger/OpenAPI
- **Logging:** Winston

Frontend

- **Framework:** React 19
- **Build Tool:** Vite 7
- **Language:** TypeScript
- **State:** Zustand
- **Routing:** React Router v6
- **Mobile:** Capacitor 7
- **Camera:** MLKit Barcode Scanner
- **PWA:** Vite PWA Plugin
- **UI:** Modern CSS + Responsive

Ana Modüller

Kimlik Doğrulama

- JWT token bazlı
- Google OAuth 2.0
- Refresh token sistemi
- Rol bazlı yetkilendirme
- Oturum yönetimi

Stok Yönetimi

- Ürün CRUD işlemleri
- Lokasyon yönetimi
- Multi-warehouse desteği
- Düşük stok uyarıları
- Gerçek zamanlı envanter

Barkod İşlemleri

- 1D/2D barkod okuma
- QR kod desteği
- SKU sorgulama
- Seri numara takibi
- Kamera ve scanner

 Operasyon Modları

Mod	Kod	Açıklama	Kullanım Alanı
Giriş	IN	Ürün alım ve girişi	Tedarikçiden mal alımı, üretim girişi
Çıkış	OUT	Ürün sevk ve çıkışı	Satış siparişleri, transfer
Sayım	COUNT	Stok sayım oturumu	Periyodik sayım, spot sayım
Transfer	TRANSFER	Lokasyon değişikliği	İç depo transferleri
Paketleme	PACK	Koli/palet oluşturma	Sevkiyat hazırlığı
Sevkiyat	SHIP	Sevkiyat tamamlama	Kargo çıkışı, yükleme



Multi-Warehouse Desteği

Desteklenen Depolar

Kod	Depo	Açıklama
TUR	Türkiye	Ana merkez depo (İstanbul)
USA	Amerika	ABD şubesi deposu
FAB	Fabrika	Üretim tesisi

Özellikler:

- Depo bazlı stok takibi
- Depolararası transfer
- Merkezi raporlama
- Bağımsız lokasyon yönetimi

Lokasyon Sistemi

DEPO-KORIDOR-RAF-BÖLÜM

Örnek: TUR-A-01-05

TUR: Türkiye deposu

A : A koridoru

01 : 1. raf

05 : 5. bölüm

Avantajlar:

- Hiyerarsık yapı
- Kolay konumlandırma
- Optimize edilmiş toplama
- Hızlı arama



Güvenlik ve Yetkilendirme

Kullanıcı Rollerleri

Rol	Yetki Seviyesi	Erişim
ADMIN	● Tam yetki	Tüm işlemler + Kullanıcı yönetimi
MANAGER	● Yönetici	Stok, rapor, operasyonlar
OPERATOR	● Operatör	Temel operasyonlar, tarama
VIEWER	● Görüntüleme	Sadece okuma yetkisi

Güvenlik Önlemleri

-  **JWT Token** - Stateless authentication
-  **Refresh Token** - Otomatik oturum yenileme
-  **Rate Limiting** - DDoS koruması
-  **Helmet.js** - HTTP header güvenliği
-  **Bcrypt** - Şifreleme

Mobil Uyumluluk

Progressive Web App (PWA)

Offline Çalışma

- Service Worker desteği
- Cache-first stratejisi
- Background sync

Native Deneyim

- Ana ekran'a eklenebilir
- Tam ekran mod
- Push bildirimler

Responsive Tasarım

Capacitor Native App

Android & iOS

- Native APK/IPA build
- Google Play Store hazır
- App Store uyumlu

Kamera Entegrasyonu

- MLKit Barcode Scanner
- 1D/2D barkod okuma
- Real-time scanning
- Otomatik fokus

Barkod ve Seri Numara Takibi

Barkod Sistemleri

Desteklenen Formatlar:

- EAN-13, EAN-8
- UPC-A, UPC-E
- Code 128, Code 39
- QR Code
- Data Matrix
- PDF417

Okuma Yöntemleri:



Mobil kamera

Seri Numara Yönetimi

Özellikler:

- Seri numara kaydı
- Ürün bazlı takip
- Geçmiş kayıtları
- Garanti takibi
- RMA işlemleri

Kullanım Alanları:

- Elektronik ürünler
- Pahalı ekipmanlar



Sayım Sistemleri

Cycle Count (Periyodik Sayım)

Özellikler:

- Oturum bazlı sayım
- Çoklu kullanıcı desteği
- Real-time güncelleme
- Fark analizi
- Onay mekanizması

İş Akışı:

1. Sayım oturumu oluştur
2. Ürünleri tara/say

Sayım Raporları

Rapor Tipi	İçerik
Sayım Özeti	Toplam ürün, miktar, süre
Fark Raporu	Sistem vs Fiziki farklar
Kullanıcı Bazlı	Operatör performansı
Lokasyon Bazlı	Depo/koridor bazında
Ürün Bazlı	SKU bazında detay

Export Formatları:

-  PDF rapor
-  Excel (XLSX)

Sevkiyat ve Sipariş Yönetimi

Sipariş Toplama (Order Picking)

Toplama Stratejileri:

1. Piece Picking

- Tek sipariş bazlı
- Düşük hacim

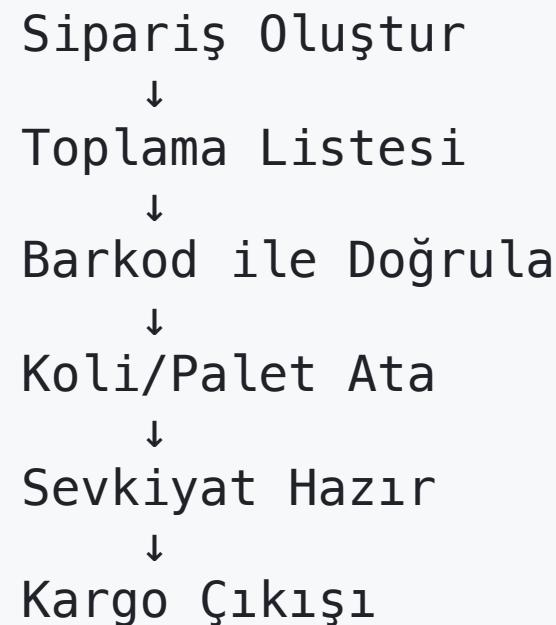
2. Batch Picking

- Çoklu sipariş birlikte
- Optimize edilmiş rota

3. Zone Picking

- Bölge bazlı toplama

Süreç Adımları:



 İade ve RMA Yönetimi

Return Merchandise Authorization (RMA) - Müşteri iadelerini ve kusurlu ürün değişimlerini yönetir.

RMA İş Akışı

- İade Talebi** - Müşteriden gelen istek
- RMA Kodu** - Benzersiz takip numarası
- Ürün Gelişisi** - Depoya iade alımı
- İnceleme** - Ürün kontrolü
- Karar** - Kabul/Red/Değişim
- Stok İşlemi** - Enyanter güncellemesi

İade Durumları

Durum	Açıklama
PENDING	İade bekliyor
RECEIVED	Ürün alındı
INSPECTED	İnceleme yapıldı
APPROVED	Onaylandı
REJECTED	Reddedildi
COMPLETED	İşlem tamamlandı



Dashboard ve Raporlama



Stok Metrikleri

- Toplam stok değeri
- Ürün çeşit sayısı
- Düşük stok uyarıları
- Stok devir hızı
- ABC analizi



Operasyon Metrikleri

- Günlük giriş/çıkış
- İşlem sayısı
- Operatör performansı
- Ortalama işlem süresi
- Hata oranları



Sevkiyat Metrikleri

- Bekleyen siparişler
- Toplanan siparişler
- Sevk edilen koliler
- On-time delivery
- Return oranı



Backend API Yapısı

17 Adet RESTful Controller

Controller	Endpoint	Durum
Auth	/api/auth/*	✓
User	/api/users/*	✓
Product	/api/products/*	✓
Location	/api/locations/*	✓
Inventory	/api/inventory/*	✓
Transaction	/api/transactions/*	✓
Scan	/api/scan/*	✓
Container	/api/containers/*	✓
Operation	/api/operations/*	✓

Controller	Endpoint	Durum
Order	/api/orders/*	✓
Shipment	/api/shipments/*	✓
Cycle Count	/api/cyclecounts/*	✓
RMA	/api/rma/*	✓
Serial	/api/serials/*	✓
Report	/api/reports/*	✓
Warehouse	/api/warehouses/*	✓



Veritabanı Şeması

Ana Tablolar (20+)

Temel Tablolar:

- users - Kullanıcılar
- products - Ürünler
- locations - Lokasyonlar
- warehouses - Depolar
- inventory - Stok kayıtları
- transactions - İşlem geçmişi
- containers - Koli/paletler
- operation_sessions - Oturum kayıtları

Gelişmiş Tablolar:

- orders - Siparişler
- order_items - Sipariş kalemleri
- shipments - Sevkiyatlar
- cycle_count_sessions - Sayım oturumları
- cycle_count_items - Sayım kalemleri
- rma_requests - İade talepleri
- serial_numbers - Seri numaraları
- audit_logs - Denetim kayıtları



Performans ve Ölçeklenebilirlik

Performans Optimizasyonları

Backend:

-  Redis caching
-  Gzip compression
-  Connection pooling
-  Query optimization
-  Composite indexes

Frontend:

-  Code splitting
-  Lazy loading

Ölçeklenebilirlik

Horizontal Scaling:

- Load balancer hazır
- Stateless API design
- Redis session store
- Database replication

Vertical Scaling:

- PostgreSQL tuning
- Node.js clustering
- Memory optimization



Güvenilirlik ve Yedekleme

Hata Yönetimi

- Try-catch blokları
- Global error handler
- Validation errors
- Graceful degradation
- Retry mekanizması
- Circuit breaker pattern

Logging

Winston Logger:

- `error.log` (kritik hatalar)
- `combined.log` (tüm loglar)

Yedekleme Stratejisi

Database Backup:

- Günlük otomatik yedek
- Incremental backups
- Point-in-time recovery
- Remote backup storage

Disaster Recovery:

- RTO: < 1 saat
- RPO: < 15 dakika
- Multi-region support



Kullanıcı Arayüzü

Modern UI/UX

Tasarım Prensipleri:

- 🎯 Kullanıcı odaklı
- 📱 Mobile-first
- ⚡ Hızlı ve responsive
- 🎨 Tutarlı renk paleti
- 🔤 Okunabilir tipografi
- ♿ Accessibility (a11y)

Renkler:

Sayfa Yapısı (10 Ana Sayfa)

Sayfa	Açıklama
 Home	Dashboard ve modül kartları
 Login	Giriş ekranı + OAuth
 Operations	Ana operasyon merkezi
 Inventory	Stok sorgulama ve arama
 Products	Ürün yönetimi (CRUD)
 Locations	Lokasyon yönetimi
 Transactions	İşlem geçmişi
 Reports	Sayımlar raporları
 Shipments	Sevkıyat yönetimi



Operations Sayfası (Modüler Mimari)

En kritik ve en gelişmiş modül - Tüm depo operasyonlarının gerçekleştirildiği merkez

Bileşen Yapısı

```
Operations/
  └── index.tsx          # Ana orchestrator
  └── types.ts            # TypeScript interface'ler

  └── components/
      ├── StatusBar.tsx    # Üst durum çubuğu
      ├── ModeSelector.tsx  # Mod seçici (IN/OUT/COUNT...)
      ├── ItemsList.tsx     # Taranan ürün listesi
      ├── CountModeView.tsx # Sayım özel görünümü
      ├── CountSummaryModal.tsx # Sayım özet modalı
      ├── HelpPanel.tsx     # Yardım paneli
      └── CameraView.tsx    # Kamera barkod okuyucu

  └── hooks/
      └── useScanner.ts     # Barkod okuma mantığı
```



API Client Mimarisi

Modüler ve Tip Güvenli

```
lib/api/
  └── index.ts
  └── client.ts

  └── auth.ts
  └── users.ts
  └── products.ts
  └── locations.ts
  └── inventory.ts
  └── transactions.ts
  └── containers.ts
  └── scan.ts
  └── serials.ts
  └── reports.ts
  └── shipments.ts

# Export hub
# Axios base client + interceptors

# Login, logout, refresh
# User CRUD
# Product operations
# Location management
# Inventory queries
# Transaction history
# Container/box operations
# Barcode scanning
# Serial number tracking
# Report generation
# Shipment management
```

Deployment ve DevOps

Hetzner Cloud Deployment

Server Gereksinimleri:

- Ubuntu 22.04 LTS
- 4GB RAM minimum
- 80GB SSD
- 2 vCPU

Kurulum Stack:

- Node.js 20.x
- PostgreSQL 15
- Redis 7
- Nginx (reverse proxy)
- PM2 (process manager)

CI/CD Pipeline

GitHub Actions:

Workflow:

1. Push to main
2. Run tests
3. Build Docker images
4. Deploy to server
5. Health check
6. Rollback if fail

Deployment Stratejisi:

- Zero-downtime deployment
- Blue-green deployment

Docker Support

Docker Compose Yapılandırması

Development Mode:

`docker-compose.dev.yml`

Services:

- wms-backend (hot reload)
- wms-frontend (Vite dev)
- postgres:15
- redis:7
- adminer (DB UI)

Production Mode:

`docker-compose.yml`

Services:

- wms-backend (optimized)
- wms-frontend (nginx)
- postgres:15-alpine
- redis:7-alpine
- nginx (reverse proxy)

Özellikler:

- Volume mounting

Optimizasyonlar:

- Multi-stage builds
- Alpine images



Test Coverage

Backend Testing

Framework: Jest + Supertest

Test Türleri:

- Unit tests
- Integration tests
- API endpoint tests
- Database tests

Komutlar:

```
npm test          # Run all tests  
npm test:watch   # Watch mode  
npm test:coverage # Coverage report
```

Frontend Testing (Hazır)

Framework: Vitest + Testing Library

Test Planı:

- Component tests
- Hook tests
- Integration tests
- E2E tests (Playwright)

CI Integration:

- GitHub Actions
- Automated testing

Mobil Uygulama (Capacitor)

Android & iOS Native App

Capacitor Konfigürasyonu:

```
{  
  "appId": "com.swiftstock.wms",  
  "appName": "SwiftStock",  
  "webDir": "dist",  
  "bundledWebRuntime": false,  
  "plugins": {  
    "BarcodeScanner": {  
      "formats": [  
        "QR_CODE", "EAN_13",  
        "EAN_8", "CODE_128"  
      ]  
    }  
  }  
}
```

Build Komutları:

```
# Android  
npm run build  
npx cap sync android  
npx cap open android  
# Build APK in Android Studio  
  
# iOS  
npm run build  
npx cap sync ios  
npx cap open ios  
# Archive in Xcode
```

Store Deployment:

→ Google Play Store hazırlık



Geliştirilecek Özellikler



Yüksek Öncelik

- Zebra Printer Entegrasyonu

- Barkod etiketi yazdırma
- ZPL dil desteği
- WiFi/USB bağlantı

- Koli İçeriği UI

- Koliye ürün ekleme
- Drag & drop interface
- Koli özet görünümü

- Production Deployment



Orta Öncelik

- Dashboard Grafikleri

- Chart.js entegrasyonu
- Real-time güncellemeler
- Interaktif grafikler

- Bildirim Sistemi

- Düşük stok uyarıları
- Push notifications
- Email alerts

- Excel Import/Export



Geliştirilecek Özellikler (devam)



Düşük Öncelik

- Multi-Tenant Support
 - Çoklu şirket desteği
 - Tenant izolasyonu
 - Merkezi yönetim
- Webhook Entegrasyonları
 - E-ticaret platformları
 - ERP sistemleri
 - 3PL entegrasyonları



İyileştirmeler

- Audit Log UI
 - Detaylı işlem geçmişi
 - Filtreleme ve arama
 - Timeline görünümü
- Dark Mode
 - Gece modu
 - Otomatik geçiş
 - Kullanıcı tercihi
- i18n (Internationalization)



Kullanım Senaryoları

Senaryo 1: Ürün Girişi (Receiving)

1. Operatör "IN" modunu seçer
2. Tedarikçiden gelen ürünü tarar
3. Sistem ürünü tanır, miktarı sorar
4. Operatör miktarı girer
5. Lokasyon seçer (örn: TUR-A-05-12)
6. İşlemi onaylar
7. Sistem stoku günceller
8. Başarılı ses ve görsel feedback

Süre: ~10 saniye/ürün

Kullanım Senaryoları (devam)

Senaryo 2: Sipariş Toplama (Picking)

1. Yönetici sevkiyat siparişi oluşturur
2. Operatör "SHIP" moduna girer
3. Sistem toplama listesini gösterir
4. Her ürün için:
 - Lokasyonu gösterir
 - Operatör ürünü tarar
 - Miktar doğrulaması yapar
5. Tüm ürünler toplandı
6. Koli numarası girilir
7. Sevkiyat tamamlanır

Süre: ~2 dakika/10 ürün

Kullanım Senaryoları (devam)

Senaryo 3: Stok Sayımı (Cycle Count)

1. Yönetici sayım oturumu başlatır
2. Sayım yapılacak lokasyonları seçer
3. Operatör "COUNT" moduna girer
4. Her lokasyonda:
 - Bulunan ürünler tarar
 - Adet bilgisi girer
5. Oturum tamamlanır
6. Sistem farkları gösterir:
 - Fazla ürünler (yeşil)
 - Eksik ürünler (kırmızı)
 - Eşit (gri)
7. Yönetici onaylar
8. Sistem stokları düzeltir

Süre: ~5 dakika/50 ürün



Kullanım İstatistikleri (Örnek)

Tipik Bir Gün

Metrik	Değer
 Toplam İşlem	1,250 adet
 Ürün Giriş	450 adet
 Ürün Çıkışı	680 adet
 Transfer	85 adet
 Sayım	35 oturum (1,800 ürün)
 Aktif Kullanıcı	12 kişi
 Ortalama İşlem Süresi	8.5 saniye
 Başarı Oranı	99.2%
 Mobil Kullanım	85%



Rekabet Avantajları

vs Ticari WMS Çözümleri

SwiftStock Avantajları:

-  Ücretsiz ve açık kaynak
-  Özelleştirilebilir
-  Modern teknoloji stack
-  Hızlı deployment
-  Yerli geliştirme
-  Kolay entegrasyon
-  Mobil-öncelikli
-  Aktif geliştirme

Karşılaştırma

Özellik	SwiftStock	Ticari WMS
Lisans	Ücretsiz	\$5K-50K/yıl
Kurulum	1 gün	1-3 ay
Özelleştirme	Kolay	Zor/Pahalı
Mobil App	Native	Genelde web
Support	Community	Paid
Güncellemeler	Sürekli	Yavaş
Cloud/On-premise	İkisi de	Genelde cloud

🤝 Destek ve Topluluk

Dokümantasyon

-  README.md - Hızlı başlangıç
-  BLUEPRINT.md - Detaylı mimari
-  API Docs - Swagger UI
-  Deployment Guide - Kurulum
-  User Manual - Kullanıcı kılavuzu

Destek Kanalları

-  GitHub Issues
-  GitHub Discussions

Açık Kaynak

Lisans: MIT License

Katkıda Bulunma:

1. Fork the repo
2. Create feature branch
3. Commit changes
4. Push to branch
5. Open Pull Request

Roadmap:

- GitHub Projects
- Issue tracking

İş Modeli ve Gelir Kaynakları

Açık Kaynak + Hizmet

Ücretsiz:

-  Kaynak kod
-  Self-hosting
-  Community support
-  Dokümantasyon

Ücretli (Opsiyonel):

-  Kurulum hizmeti
-  Özelleştirme

Hedef Müşteriler

Mikro İşletmeler:

- E-ticaret satıcıları
- Üretim atölyeleri
- Toptan satıcılar

KOBİ:

- Orta ölçek depolar
- Dağıtım merkezleri
- 3PL firmaları

Kurumsal:

Eğitim ve Onboarding

Hızlı Başlangıç

Yöneticiler İçin (30 dk):

1. Sistem tanıtımı
2. Kullanıcı oluşturma
3. Depo ve lokasyon kurulumu
4. Ürün ekleme
5. İlk işlem
6. Raporlama

Operatörler İçin (15 dk):

1. Giriş yapma
2. Mod seçimi
3. Barkod tarama
4. Miktar girişi
5. Lokasyon seçimi
6. İşlem tamamlama

Video Eğitimler: YouTube kanalında adım adım rehberler (planlanan)

Demo Ortam: test.swiftstock.com - Deneme hesabı

SEO ve Pazarlama

Anahtar Kelimeler

- Warehouse Management System
- WMS Yazılımı
- Depo Yönetim Sistemi
- Açık Kaynak WMS
- Ücretsiz Depo Programı
- Barkod Stok Takip
- Mobil Depo Uygulaması

Hedef Platformlar

- GitHub (star ve fork)

Ekran Görüntüleri (Planlanan)

Ana Sayfalar

- 1. Login Screen** - Modern giriş, Google OAuth
- 2. Dashboard** - Kartlar, istatistikler, quick actions
- 3. Operations** - Barkod okuma, mod seçimi, item listesi
- 4. Inventory** - Arama, filtreleme, stok bilgisi
- 5. Products** - Tablo görünüm, CRUD işlemleri
- 6. Reports** - Sayım sonuçları, fark analizi
- 7. Shipments** - Sevkiyat listesi, koli yönetimi
- 8. Mobile View** - Responsive tasarım örnekleri

Başarı Metrikleri (KPI)

Sistem Performansı

- ⚡ API Response Time: < 200ms
- 📊 Database Query Time: < 50ms
- 🔄 Cache Hit Rate: > 90%
- ⬆️ Uptime: > 99.9%

Kullanıcı Deneyimi

- 📱 Mobile Usage: > 80%
- ⌚ İşlem Tamamlama Süresi: < 10 saniye
- ✅ Başarılı İşlem Oranı: > 99%
- 😊 User Satisfaction: > 4.5/5

 **Gelecek Vizyonu****Kısa Vade (3-6 ay)**

-  Production deployment
-  İlk 10 müşteri
-  Mobil app launch
-  Zebra printer entegrasyon
-  Temel raporlar
-  Excel import/export

Orta Vade (6-12 ay)

-  100+ aktif kullanıcı
-  Multi-tenant support

Uzun Vade (1-2 yıl)

-  AI/ML Entegrasyonu
 - Talep tahmini
 - Optimal stok seviyesi
 - Akıllı lokasyon önerisi
-  IoT ve Otomasyon
 - RFID okuyucu
 - Akıllı raflar
 - Robotik entegrasyon
-  Blockchain



Bonus Özellikler



Ses Geri Bildirimleri

- Başarılı işlem (beep)
- Hata sesi (error)
- Uyarı sesi (warning)
- Özelleştirilebilir



Titreşim Feedback

- Mobil cihazlarda
- Haptic feedback
- İşlem doğrulama
- Native integration



Offline Mode

- Service Worker
- LocalStorage cache
- Sync when online
- Queue management



Tema Sistemi

- Light mode (default)
- Dark mode



Excel Integration

- XLSX library
- Import products



Print Support

- Thermal printer
- Label printing



KVKK ve GDPR Uyumluluğu

SwiftStock, kişisel verilerin korunması konusunda Türkiye ve AB standartlarına uygundur.

KVKK Uyumluluğu

- Veri minimizasyonu
- Şeffaflık
- Güvenlik önlemleri
- Kullanıcı hakları
- Veri silme/düzelte
- Audit logging

GDPR Özellikleri

- Right to be forgotten
- Data portability
- Consent management
- Privacy by design
- Data encryption
- Access controls



Geliştirici Deneyimi (DX)

Kolay Başlangıç

```
# 1. Repo'yu klonla  
git clone https://github.com/yourname/swiftstock.git  
cd swiftstock  
  
# 2. Bağımlılıkları yükle  
npm run install:all  
  
# 3. .env dosyasını oluştur  
cp .env.example .env  
  
# 4. Docker ile başlat  
docker-compose up -d  
  
# 5. Database'i başlat  
cd wms-backend && npm run db:init  
  
# 6. Uygulamayı çalıştır
```



Makefile Komutları

Kullanışlı komutlar:

make dev	# Hem frontend hem backend çalıştır
make dev-backend	# Sadece backend
make dev-frontend	# Sadece frontend
make build	# Production build
make test	# Tüm testleri çalıştır
make docker-up	# Docker containers başlat
make docker-down	# Docker containers durdur
make db-init	# Database'i başlat
make db-backup	# Backup al
make db-restore	# Backup'tan geri yükle
make logs	# Logları göster
make clean	# Temizle

İletişim ve Demo

Canlı Demo

 **Demo URL:** <https://demo.swiftstock.com>

Test Hesabı:

- Email: demo@swiftstock.com
- Password: Demo123!

Demo Özellikleri:

- Tam fonksiyonel
- Test verileri yüklü
- Sıfırlama: Her gece 02:00
- **Tüm modüller aktif**

İletişim

 **Email:** info@swiftstock.com

GitHub:

github.com/yourname/swiftstock

LinkedIn:

linkedin.com/company/swiftstock

 **Twitter:** @swiftstockwms

Demo Talebi

Özel demo için:

- Şirket adı
- Kullanıcı sayısı

Son Söz

SwiftStock WMS

Deponuzu Dijitalleştirin, Verimliliği Artırın

Neden SwiftStock?

- Modern ve kullanıcı dostu
- Hızlı kurulum ve kullanım
- Ölçeklenebilir mimari
- Açık kaynak ve özgür
- Aktif geliştirme

Bir Sonraki Adım

1. Demo'yu deneyin
2. GitHub'dan indirin
3. Dökümantasyonu okuyun
4. Toplulukla iletişime geçin
5. Projenize enteare edin

🎉 Teşekkürler



SwiftStock WMS

Sorularınız için hazırız!

Sunum İçeriği:

- 50+ slayt
- Detaylı teknik bilgi
- Kullanım senaryoları
- Canlı demo hazır

Sonraki Adımlar:

- Demo oturumu planla
- Teknik sorular
- Fiyatlandırma görüşmesi
- POC (Proof of Concept)

