TARIM GIDA SÜRÜ YÖNETİM SİSTEMİ - PROJE DOKÜMANTASYONU

Sürüm: v1.1 (Production Ready) **Tarih:** 09.12.2025 **Durum:** Canlı Kullanıma Hazır

1. PROJE KÜNYESİ VE AMAÇ

Proje Adı: Tarım Gıda (Flock Planner) **Amaç:** Yumurta tavuğu tesisleri için web tabanlı, modüler, gerçek zamanlı çiftlik yönetim sistemi (ERP) ve IoT entegrasyonu.

Temel Modüller:

1. **Sürü Planlama (Planner):** Sürükle-bırak arayüzü ile civciv ve tavuk sürülerinin yaşam döngüsü planlaması.
2. **Verim Takibi (Production):** Günlük yumurta, ölüm, yem ve su tüketim verilerinin Excel benzeri grid yapısıyla işlenmesi.
3. **Kümes Durumu (Coop Status - IoT):** Su sayaçlarından gelen verilerin InfluxDB'den çekilip analiz edilmesi ve anlık izleme.

2. TEKNİK YIĞIN (TECH STACK)

- **Framework:** Next.js 16 (App Router)
- **Dil:** TypeScript
- **Stil:** Tailwind CSS v4
- **Veritabanı (Operasyonel):** Google Cloud Firestore (NoSQL)
- **Veritabanı (Zaman Serisi):** InfluxDB v2 (Sensör Verileri)
- **Authentication:** Firebase Auth
- **UI Kütüphaneleri:** * `@dnd-kit` : Sürükle bırak işlemleri için.
  - `recharts` : Grafiksel veri görselleştirme.
  - `date-fns` : Tarih manipülasyonu.
  - `lucide-react` : İkon seti.

3. VERİTABANI MİMARİSİ (SCHEMA)

Sistem hibrit bir veritabanı yapısı kullanır. Operasyonel veriler Firestore'da, sensör verileri InfluxDB'de tutulur ve senkronize edilir.

A. Firestore Şeması

1. Koleksiyon: `flocks` (Sürüler)

Bir sürünün kimlik kartıdır.

- `id` (string): UUID.
- `name` (string): Sürü adı (Örn: #05).
- `coopId` (string): Kümes ID (T1, T2 vb.).

- `initialCount` (number): Başlangıçtaki hayvan sayısı.
- `hatchDate` (Timestamp): Kuluçka tarihi.
- `type` (string): "layer" (Yumurtacı).
- `lane` (number): 0 (Civciv Hanesi) | 1 (Tavuk Hanesi).

2. Koleksiyon: `daily_logs` **(Günlük Kayıtlar)**

Verim modülünde girilen verilerdir.

- `flockId` (string): Referans ID.
- `coopId` (string): Kümes ID.
- `date` (Timestamp): Kayıt tarihi (Composite Index: flockId + date).
- `mortality` (number): Günlük ölüm.
- `eggCount` (number): Toplam yumurta.
- `feedConsumed` (number): Yem tüketimi.
- `waterConsumed` (number): Manuel girilen su tüketimi.

3. Koleksiyon: `water_readings` **(IoT Su Verileri)**

Kritik Yapı: Sensörlerden gelen verinin Firestore'daki aynasıdır. `Total` alanı veritabanında tutulmaz, yükü azaltmak için okuma anında hesaplanır.

- **Doküman ID:** `{coopId}_{ISOString}` (Örn: `T1_2025-12-08T10:00:00.000Z`)
- `coopId` (string): T1.
- `timestamp` (Timestamp): Veri zamanı.
- `b1` (number): Batarya 1 sayacı (Litre).
- `b2` (number): Batarya 2 sayacı (Litre).
- `b3` (number): Batarya 3 sayacı (Litre).
- `b4` (number): Batarya 4 sayacı (Litre).

4. Koleksiyon: `water_meta` **(Sync Takibi)**

Incremental Sync için son durum bilgisini tutar.

- **Doküman ID:** `{coopId}` (Örn: T1)
- `lastImportedTime` (Timestamp): En son başarıyla çekilen verinin zaman damgası.
- `lastSyncCheck` (Timestamp): Son kontrol zamanı.

B. InfluxDB Şeması (Raw Data)

Sensörlerden gelen ham veridir.

- **Bucket:** `kumesShieldV4Sensors`
- **Measurement:** `device_frmpayload_data_WaterMeter[1-4]` (Her batarya ayrı measurement).
- **Tag:** `device_name` (Örn: `MKR1310-K1-WaterMeter`).
- **Field:** `value` (Litre cinsinden 10 dakikalık akış).

4. MODUL DETAYLARI VE ÇALIŞMA MANTIĞI

Modül 1: Kümes Durumu (Coop Status - IoT)

Bu modül, sahadaki LoRaWAN su sayaçlarını sisteme entegre eder.

1. Veri Senkronizasyonu (Sync API):

- **Endpoint:** `/api/sync-water`
- **Mantık:**
  1. `water_meta` koleksiyonundan son kayıt tarihini kontrol eder.
  2. InfluxDB'ye sorgu atar (`range: son kayıt -> şimdi`).
  3. **Aggregation:** `aggregateWindow(every: 5m, fn: last)` kullanır. Bu, 5 dakikalık periyotlarda en son gelen veriyi alarak zamanı hizalar ve mükerrer kayıtları önler. `mean` veya `sum` kullanılmaz, veri saf kalır.
  4. **Pivot:** 4 ayrı batarya verisini tek bir satırda birleştirir.
  5. Firestore'a `water_readings` altına yazar.
  6. `water_meta` güncellenir.

2. Veri Okuma ve Görselleştirme (Read API):

- **Endpoint:** `/api/get-readings`
- **Mantık:**
  1. Tarih aralığına göre Firestore'dan veriyi çeker (Date veya Range).
  2. **Hesaplama:** `Total` alanı veritabanında olmadığı için, API yanıtı oluşturulurken `b1+b2+b3+b4` işlemi sunucu tarafında yapılır.
- **Sayfa (`page.tsx`):**
  - **Bar Grafiği:** Son 14 günün günlük toplamlarını gösterir.
  - **Çizgi Grafiği (Grid):** Seçili günün 4 bataryasının 5 dakikalık detaylı akış grafiğini gösterir.

3. Ham Veri İzleme (Raw Data):

- Doğrudan InfluxDB'ye bağlanarak `aggregateWindow` olmadan veriyi olduğu gibi listeler. Debug amaçlıdır.

Modül 2: Sürü Planlama (Planner)

- **Drag & Drop:** `@dnd-kit` kullanılarak sürüler "New Flock" alanından kümeslere sürüklenir.
- **Görselleştirme:** Kümesler sütun (`CoopColumn`), zaman ise dikey eksen (`DateSidebar`) olarak modellenmiştir.
- **Lane Mantığı:** Her kümes iki şeritlidir (Civciv ve Tavuk). Sürüler yaşlarına göre otomatik şerit değiştirmez, manuel yönetilir.

Modül 3: Verim Takibi (Production)

- **Data Grid:** Excel benzeri veri girişi sağlar.
- **Validasyon:** Gelecek tarihe veri girilemez.
- **Excel Import:** Eski verilerin sisteme toplu aktarımı için `ImportModal` kullanılır.

5. KURULUM VE ORTAM DEĞİŞKENLERİ

Aşağıdaki değişkenlerin `.env.local` dosyasında ve Vercel panelinde tanımlı olması zorunludur.

```
# Firebase Config
NEXT_PUBLIC_FIREBASE_API_KEY="..."
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN="..."
NEXT_PUBLIC_FIREBASE_PROJECT_ID="..."
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET="..."
NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID="..."
NEXT_PUBLIC_FIREBASE_APP_ID="..."

# InfluxDB Config (Coop Status Modülü İçin)
INFLUX_URL="http://[MODEM_IP]:8086" # Veya Statik IP
INFLUX_TOKEN="..."
INFLUX_ORG="..."
INFLUX_BUCKET="kumesShieldV4Sensors"
```

6. BİLİNEN KISITLAMALAR VE İPUÇLARI

1. **Sync İşlemi:** InfluxDB yerel ağda (modeme bağlı) çalışıyorsa, Vercel üzerinden "Veri Çek" butonu çalışmaz. Senkronizasyon işlemi, yerel ağa bağlı bir bilgisayardan (`localhost`) yapılmalıdır.
2. **Firestore Limitleri:** Günlük sync işlemi 5 dakikalık periyotlarla yapıldığı için ~288 kayıt/gün oluşturur. Bu, Firestore ücretsiz kota limitleri (20k/gün) içinde güvenlidir.
3. **Zaman Dilimi:** API, verileri UTC olarak kaydeder ancak okurken Türkiye saati (UTC+3) ofsetini dikkate alarak filtreleme yapar.

7. SONRAKİ ADIMLAR (ROADMAP)

- **Coop Status:** Sıcaklık ve Nem sensörlerinin de benzer mimari ile eklenmesi.
- **Mobile App:** PWA desteği ile kümes içinden veri girişinin kolaylaştırılması.
- **Alerting:** Su tüketimi belirli bir eşiğin altına düşerse Telegram/SMS bildirimi gönderilmesi.