Logica di Business — SmartMaintenance

SmartMaintenance

Piattaforma SaaS multi-tenant per la gestione manutenzioni

AgriSolution S.R.L.

Generato automaticamente da BUSINESS_LOGIC.md

BUSINESS_LOGIC.md — Logica di business SmartMaintenance

Aggiornato: 2026-04-11 **Pubblico:** dev nuovi, product owner, audit/compliance

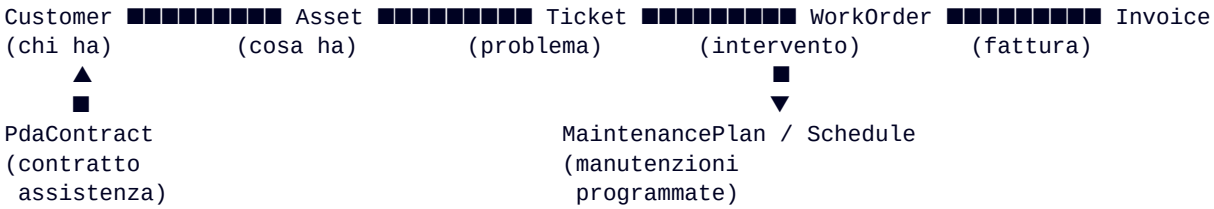
> *Spiega in dettaglio **come funziona il software**, dall'anagrafica iniziale del cliente fino alla chiusura di un task con relativa fatturazione e archiviazione storica. Compagnia di PIANO_VERIFICA_ANAGRAFICHE.md e PIANO_VERIFICA_LOGICHE.md.*

Indice

---

1. Modello concettuale

SmartMaintenance gestisce il **ciclo di vita completo di un intervento di manutenzione** in un contesto multi-tenant. Le 5 macro-entità che governano tutto il flusso:



Customer è l'anagrafica del cliente finale (chi paga il servizio). **Asset** è la cosa fisica installata presso il cliente (caldaia, condizionatore, sensore, impianto). **Ticket** è la segnalazione di un problema o di una richiesta. **WorkOrder** è l'intervento pianificato per risolvere il ticket (può anche nascere da una manutenzione preventiva, non solo da un ticket). **Invoice** è la fattura che monetizza l'intervento.

In parallelo:

- **PdaContract** = Piano di Assistenza, contratto a canone che lega un cliente a un set di prodotti/installazioni con SLA e copertura
- **MaintenancePlan** = piano di manutenzione preventiva con frequenza fissa (es. revisione caldaia ogni 12 mesi)

- **Sensor** + **SensorReading** = telemetria IoT che può generare ticket automatici quando supera soglie

---

2. Anagrafica master: il setup iniziale

2.1 Tenant onboarding

Il primo passo per usare il software è **creare il tenant**. Solo il SuperAdmin (`mario.grillo@agrisolution.it`) può farlo via `/admin/tenants/new`. Il tenant porta:

- `subdomain` (es. `lean` → `lean.fattorecreativo.it`)
- Anagrafica completa (ragione sociale, P.IVA, indirizzo, PEC, SDI)
- Brand personalizzato (logo, colori primari/secondari)
- Configurazione mail mittente custom
- Configurazione WhatsApp (numero E.164 + nome mittente)
- Soglie di escalation (ticket critico ore, WO overdue ore)

Una volta creato, il SuperAdmin invita il primo **admin del tenant** via **magic link** (UserInvitationService genera un token UUID, il sistema manda mail con un link tipo `/auth/activate?token=...`, l'utente clicca, imposta password e attiva l'account).

→ Vedi TENANTS.md per la procedura completa di setup DNS + SSL + nginx + creazione tenant.

2.2 Anagrafica clienti (Customer)

L'admin del tenant inizia caricando i propri **clienti finali**:

```
Form: /customers/new
  → first_name, last_name (se persona fisica)
  → business_name (se azienda)
  → fiscal_code, vat_number
  → indirizzo completo (address, city, province, zip_code)
  → contatti (phone, email, pec)
  → customer_type = PRIVATO / AZIENDA
  → consultant_name, agent (riferimento commerciale)
  → notes
```

Modi alternativi di popolare l'anagrafica:

1. **Inserimento manuale** uno per uno
2. **Import Excel** via script Python (`backend/scripts/import_lean.py`), utilizzato per Lean S.r.l. il 2026-04-10 → 1417 customers caricati da `GestLEAN26.xlsx` (35 sheet)
3. **API REST** `POST /api/v1/customers` (per integrazioni con CRM esterni)

■■ **Importante**: il sistema attualmente NON ha vincolo `UNIQUE` su `(tenant_id, fiscal_code)`. Questo ha causato 78 copie di RENERGIA SRL e 21 di EASY ENERGY SRL durante l'import Excel del 10 Apr 2026 (vedi audit Fase 3.3). Il fix è in `ops/dedup-customers.sql`.

2.3 Asset e installazioni (CustomerInstallation)

Per ogni Customer si registrano gli **Asset** = i beni fisici installati presso di lui che lo staff dovrà manutenere. Esempio: se Customer è "Hotel Barbieri SRL", i suoi asset sono "caldaia centrale", "condizionatore sala 1", "frigorifero cucina".

Ogni Asset ha:

- `name`, `model`, `serial_number`, `manufacturer`
- `category` (riferimento ad AssetCategory: caldaia / clima / freddo / ...)
- `customer` (chi ne è proprietario)
- `installation_address`, GPS coords
- `warranty_until`, `last_maintenance_at`
- **QR code** generato automaticamente: il tecnico lo scansiona da mobile per aprire il dettaglio asset al volo

In parallelo, **CustomerInstallation** è un'entità più ricca che lega un Customer a uno o più asset con dettagli di **installazione e contratto**: data installazione, garanzia estesa, prodotti seriali installati, riferimento PDA. Usato soprattutto per impianti complessi (caldaie a condensazione con serbatoio + accessori).

2.4 Catalogo prodotti, listini e fornitori

Prima di poter generare work orders, l'admin carica:

- **Suppliers**: i fornitori (per ricambi, prodotti, servizi terzi)
- **WarehouseItem**: gli articoli di magazzino con codice, nome, prezzo, scorta minima, punto di riordino
- **ProductFamily**: famiglie di prodotti (raggruppamento)
- **PriceList**: listini base
- **PriceCoefficient**: coefficienti applicati ai listini per ricavare il prezzo finale al cliente (es. mark-up del 30% sul costo)
- **BomHeader / BomLine**: distinte base (Bill of Materials) per prodotti composti
- **SerializedProduct**: prodotti con seriale individuale tracciato (es. ogni caldaia ha il suo seriale unico)

2.5 Contratti di assistenza (PdaContract)

Quando un cliente sottoscrive un **contratto di manutenzione a canone**, l'admin crea un **PdaContract** (Piano di Assistenza):

- `pda_number` (univoco per tenant)
- Customer di riferimento
- Periodo (start_date, end_date)
- Canone, modalità di pagamento
- Lista di prodotti/installazioni coperte (`pda_contract_products`)
- SLA: tempo massimo intervento, copertura ricambi/manodopera
- `area_manager`, `team_manager` (assegnazione organizzativa)

Un PdaContract attivo automaticamente:

1. Genera **MaintenancePlan** per ogni manutenzione preventiva pianificata (es. revisione annuale caldaia)
2. **Maintenance Schedule**: il piano genera ricorrenze concrete con date di scadenza

3 Quando una scadenza è prossima (cron job), il sistema crea un **WorkOrder DRAFT** automaticamente per l'admin/dispatcher

---

3. Apertura di un task: i 3 canali

Un "task" inizia sempre con un **Ticket**. Il ticket può nascere da 3 canali indipendenti.

3.1 Canale A — Web (interno)

Un operatore (CUSTOMER_SERVICE / ADMIN) compila il form `/tickets/new` con:

- Titolo, descrizione del problema
- Customer di riferimento
- Asset coinvolto (opzionale)
- Tipo: CORRECTIVE (correttiva) / PREVENTIVE / INSPECTION
- Priorità: LOW / MEDIUM / HIGH / CRITICAL
- Source: `WEB`

Stato iniziale: `OPEN`. `ticket_number` assegnato automaticamente: formato `TK<anno>-NNNNN` (es. `TK2026-00043`), univoco per tenant.

3.2 Canale B — WhatsApp (esterno via 2Chat)

Il cliente finale invia un messaggio WhatsApp al numero del tenant. 2Chat inoltra il payload via webhook a:

```
POST https://lean.fattorecreativo.it/api/webhooks/whatsapp
```

Il `WhatsAppWebhookController`:

1 Risolve il **tenant** dall'host header (`lean.fattorecreativo.it` → tenant 2)

2 Estrae dal payload 2Chat: - `remote_phone_number` (es. `+393687629105`) - `text` (corpo messaggio) - `contact.first_name` + `contact.last_name`

3 Costruisce un Ticket: - `title = "WhatsApp da Mario Grillo (+393687629105)"` - `description = <testo messaggio>` - `source = WHATSAPP` - `ticketType = CORRECTIVE` - `priority = MEDIUM` - `status = OPEN`

4 Salva via `TicketService.createTicket()` → genera ticket_number

5 Risponde sempre 200 a 2Chat (anche su errore, per non bloccare il provider)

42 ticket reali creati in produzione tra il 4 e l'11 Aprile 2026 via questo canale.

3.3 Canale C — IoT (alert automatico)

Un sensore IoT (es. termometro frigorifero ristorante per HACCP) invia periodicamente letture a:

```
POST https://lean.fattorecreativo.it/api/iot/data
Header: X-IoT-Api-Key: <chiave configurata>
Body: { "sensor_code": "TEMP-001", "value": 25.3, "unit": "°C" }
```

Il `IoTDataController`:

1 Verifica la API key (X-IoT-Api-Key) — fail-closed se mancante

2 Trova il Sensor dal `sensor_code`

3 Crea una `SensorReading` (sensor + valore + unità + timestamp)

4 Aggiorna `sensor.last_reading_value` e `last_reading_at`

5 **Verifica le soglie**: itera su `sensor_thresholds` con `active=true`: - Se `threshold_type=MAX` e `value > threshold.value` → **BREACH** - Se `threshold_type=MIN` e `value < threshold.value` → **BREACH**

6 Su breach: - Manda **Notification** a tutti gli ADMIN del tenant (se `notification_enabled`) - Se `auto_create_ticket=true` → crea automaticamente un Ticket: - `title = "Allarme IoT: <sensor_name> - soglia MAX superata"` - `priority = CRITICAL` se `severity=CRITICAL`, altrimenti `HIGH` - `source = IOT_ALERT` - `asset = sensor.asset` (link automatico)

---

4. Triage e assegnazione

Un Ticket appena creato ha `status = OPEN` e nessun assegnatario. L'admin/dispatcher entra su `/tickets`, vede la lista filtrata, ordina per priorità + data, e fa **triage**:

4.1 Filtro e ricerca

Lista paginata di ticket del proprio tenant, filtri per:

- Status (OPEN, ASSIGNED, IN_PROGRESS, RESOLVED, CLOSED, CANCELLED)
- Priority
- Source (WEB, WHATSAPP, EMAIL, IOT_ALERT)
- Assigned to (utente)
- Customer
- Asset
- Date range

4.2 Cambio stato

L'admin cambia lo stato del ticket via `POST /tickets/{id}/status`:

Da	A	Quando
OPEN	ASSIGNED	dopo l'assegnazione a un tecnico
OPEN	CANCELLED	richiesta non valida (spam, duplicato)
ASSIGNED	IN_PROGRESS	tecnico inizia a lavorare
IN_PROGRESS	RESOLVED	tecnico ha completato l'intervento
RESOLVED	CLOSED	admin verifica e chiude
RESOLVED	IN_PROGRESS	riapertura (problema non risolto davvero)

Ogni cambio di stato:

1 Aggiorna `tickets.status`

2 Inserisce riga in `ticket_status_history` (audit trail)

3. Log in `audit_log` con `action=STATUS_CHANGE`
4. Trigger notification al responsabile / tecnico assegnato

4.3 Assegnazione tecnico

L'admin assegna il ticket a un User (typeicamente `TECHNICIAN`). Lo fa direttamente sul ticket o **promuovendolo a WorkOrder** (vedi 5).

Sull'assegnazione il sistema:

- Setta `ticket.assigned_to = userId`
- Manda Notification al tecnico (multi-channel: email + push FCM)
- Aggiorna lo stato a `ASSIGNED`

4.4 Commenti e allegati

Durante il triage e l'esecuzione, chiunque può aggiungere:

- **TicketComment**: nota testuale (interna o cliente-visibile)
- **TicketAttachment**: foto, documento, audio (max 50MB per file)

Sono entità immutabili append-only.

---

5. Pianificazione e scheduling

Quando un ticket è ben qualificato e va eseguito sul campo, viene **promosso a WorkOrder**. È il momento in cui si decide CHI fa CHE COSA QUANDO con QUALI MATERIALI.

5.1 Creazione work order

Il dispatcher entra in `/work-orders/new` (o usa il pulsante "Crea WO" sul ticket):

- `ticket` (riferimento opzionale, vincolato al ticket di origine)
- `customer`, `asset` (auto-popolati dal ticket)
- `assigned_to` (tecnico responsabile)
- `scheduled_at` (data/ora pianificata)
- `estimated_duration_minutes`
- `description` (cosa va fatto)
- `materials` (lista di WarehouseItem da utilizzare, con qty)
- `checklist` (template di operazioni da spuntare)
- `priority`, `type`

Stato iniziale: `SCHEDULED`. `order_number` assegnato automaticamente: formato `WO<anno>-NNNNN`.

5.2 Calendar view

L'admin vede il calendario dei WO via `/calendar`. Visualizzazione tipo FullCalendar con:

- Vista mensile / settimanale / giornaliera
- Color-coding per stato e priority
- Drag & drop per ri-pianificare

5.3 Manutenzione preventiva automatica

In parallelo al canale "ticket → WO", esiste il flusso **manutenzione preventiva**:

1 Un PdaContract attivo ha asset coperti

2 Per ogni asset coperto esiste uno o più MaintenancePlan (es. "revisione caldaia annuale")

3 Il `MaintenancePlan` ha frequenza (`MONTHLY`, `QUARTERLY`, `YEARLY`, `CUSTOM_DAYS`)

4 Una `MaintenanceSchedule` viene generata con `due_date` calcolata

5 Cron job giornaliero (`@Scheduled`) controlla i schedule in scadenza: - Se `due_date - 7gg <= today` e nessun WO già generato → crea automaticamente un WorkOrder DRAFT con `type=PREVENTIVE` - Notifica admin per validazione e assegnazione

---

6. Esecuzione sul campo

Il tecnico riceve la notifica push sulla sua **app mobile Flutter** e apre il work order assegnato.

6.1 Mobile workflow (login → dettaglio WO)

```
[Login] → JSESSIONID salvato in shared_preferences
[Dashboard] → lista lavori del giorno (api/v1/work-orders/my)
[Detail WO] → vede customer, asset, descrizione, checklist
```

L'app espone (via `/api/v1/`):

- `getMyWorkOrders()` — i WO assegnati all'utente loggato
- `updateWorkOrderStatus(id, status)` — cambio stato
- Upload foto via `WorkOrderPhoto`
- Lettura QR sull'asset per aprire direttamente il dettaglio

6.2 Inizio intervento (Timer)

Il tecnico clicca "Inizio lavoro":

- `WorkOrder.started_at = now()`
- Stato → `IN_PROGRESS`
- Parte un **timer** (`work_order_timers`) che traccia il tempo effettivo
- L'app può funzionare offline: il timer continua localmente, sync alla prossima connessione

6.3 Esecuzione

Durante l'intervento il tecnico:

- Spunta voci della checklist (`work_order_checklist_items.completed_at`)

- Carica foto via mobile camera (`work_order_photos`)
- Registra materiali realmente usati (`work_order_materials` = lista di WarehouseItem + qty effettiva)
- Aggiunge note testuali

Ogni materiale aggiunto:

- Decrementa `warehouse_stock` per il proprio tenant (movimento OUT)
- Crea riga in `warehouse_movements` con `type=OUT`, `reason=WORK_ORDER`
- Se la nuova stock < `reorder_point` → notifica magazziniere

6.4 Firma cliente + upload finale

Alla fine dell'intervento il tecnico:

- Mostra al cliente la `signature_widget` Flutter
- Il cliente firma sul touch screen
- L'app salva la firma come immagine PNG
- Upload come `WorkOrderPhoto` con `type=SIGNATURE`

Il tecnico clicca "Completa":

- `WorkOrder.completed_at = now()`
- `actual_duration_minutes = completed_at - started_at`
- Stato → `COMPLETED`
- Asset.last_maintenance_at = now()
- Audit log + notification all'admin

---

7. Chiusura del work order

L'admin riceve la notifica "WO completato dal tecnico", entra in `/work-orders/{id}` e fa la **review**:

7.1 Validazione admin

L'admin verifica:

- Foto presenti e leggibili
- Firma cliente acquisita
- Materiali consumati coerenti con il tipo di intervento
- Tempo lavorato sensato
- Eventuali costi extra (`cost_entries`) — trasferte, parcheggi, ecc.

Se tutto OK:

- Stato WO → `COMPLETED` (definitivo)
- Stato del Ticket di origine → `RESOLVED`

- Possibile generare automaticamente l'**Invoice draft**

7.2 Generazione fattura draft

Su WO `COMPLETED`, dal pulsante "Genera fattura":

1. Sistema crea `Invoice` con stato `DRAFT`
2. `customer = wo.customer`
3. Generazione automatica di `invoice_lines`: - Per ogni `work_order_material` → riga con qty + prezzo da `PriceList × PriceCoefficient` - 1 riga per la **manodopera** = `actual_duration_hours × hourly_rate` (rate dal contratto PdaContract o tariffa default tenant) - 1 riga per ciascun `cost_entry` registrato
4. Calcolo IVA (default 22%, configurabile per riga)
5. Totale: imponibile + IVA = totale fattura

Stato Invoice: `DRAFT` → admin può modificare manualmente.

7.3 Chiusura ticket

Quando l'invoice è emessa (o se l'intervento è in garanzia / coperto da contratto e quindi non si fattura), l'admin può chiudere il ticket:

```
POST /tickets/{id}/status  status=CLOSED  resolution_text="..."
```

- `Ticket.closed_at = now()`
- `Ticket.resolution_text = ...` (descrizione di cosa è stato fatto)
- Audit log + notification cliente (mail/WhatsApp se attivati)
- Il ticket è ora **read-only** (non più modificabile)

---

8. Fatturazione

8.1 Issuance

L'admin entra in `/invoices/{id}`, verifica le righe, e clicca "Emetti":

- `Invoice.invoice_number` = generato (formato `<anno>/NNNN` per tenant)
- `Invoice.issued_at = now()`
- Stato → `ISSUED`
- Audit log

8.2 SDI (Italian e-invoicing)

Per le fatture verso aziende italiane è obbligatorio inviare il **XML FatturaPA** al **Sistema di Interscambio (SdI)** dell'Agenzia delle Entrate.

Il sistema genera automaticamente:

1. **SdiInvoice**: record che mantiene lo stato dell'invio SDI

2 XML FatturaPA conforme alla schema v1.2.2

3 Upload via API Aruba Fatturazione (provider intermedio)

Stati SDI:

- `PENDING` → in coda
- `SENT` → inviato a SDI
- `ACCEPTED` → SDI ha accettato e consegnato al destinatario
- `REJECTED` → SDI ha rifiutato (errore validazione)
- `DELIVERED` → consegnata al cliente

In caso di `REJECTED`:

- Notifica admin con dettagli errore
- Admin corregge la fattura, riemette
- Nuovo ciclo SDI

8.3 Pagamento

Quando il cliente paga (fuori sistema, via bonifico/POS/cassa):

- Admin marca `Invoice.paid = true`, `paid_at = now()`
- Stato → `PAID`
- Audit log + opzionale notification al cliente con ricevuta

---

9. Storico, KPI, archivio

9.1 Dashboard

`/dashboard` mostra le metriche del tenant in tempo reale:

- Tickets per status (donut)
- WO completati ultimo mese
- MTBF / MTTR (Mean Time Between Failures / To Repair)
- WO overdue
- Tickets per source (WEB / WHATSAPP / IOT)
- Top 5 customer per ticket count
- Top 5 asset più problematici

9.2 Reports

`/reports` produce export PDF/Excel:

- Report mensile interventi
- Report SLA per contratto PDA

- Report manutenzioni preventive eseguite vs pianificate
- Report fatturato per cliente
- Report HACCP (per ristorazione)
- Report AI insights (predizioni di guasto, ottimizzazioni)

9.3 Audit log

Ogni operazione critica scrive in `audit_log`:

- Login / Logout / Login_failed
- Create / Update / Delete su entità sensibili
- Cambio stato workflow
- Esportazioni dati
- Anonimizzazioni GDPR

L'admin (e SUPER_ADMIN) può consultare l'audit log su `/admin/audit`.

9.4 GDPR e diritto all'oblio

Quando un cliente esercita il diritto all'oblio (Art. 17), l'admin clicca "Cancella GDPR" sul Customer:

```
POST /customers/{id}/erase-gdpr
```

`CustomerErasureService.erase()` esegue:

- Anonimizza nome, cognome, ragione sociale, CF, P.IVA, indirizzo, telefono, email, PEC, raw_data
- Mantiene `id` e `tenant_id` per integrità FK
- Setta `active = false`
- Logga in `audit_log` con `action=ERASE_GDPR`

I record collegati (work_orders, invoices, ddt) restano per **obbligo fiscale 10 anni** ma il customer di riferimento è ora "Cliente cancellato #ID" con tutti i campi PII a null.

9.5 Retention

Job schedulato (da implementare in `DataRetentionScheduler`) che cancella periodicamente:

Tipo dato	Cancellato dopo
Notifications lette	6 mesi
Audit log	2 anni
Sensor readings raw	90 giorni
(aggregati)	5 anni

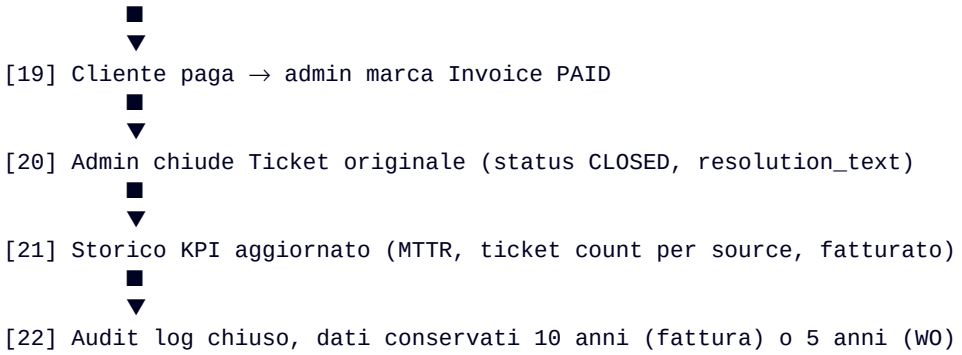
Le fatture e i contratti **non vengono mai cancellati** per obbligo fiscale.

---

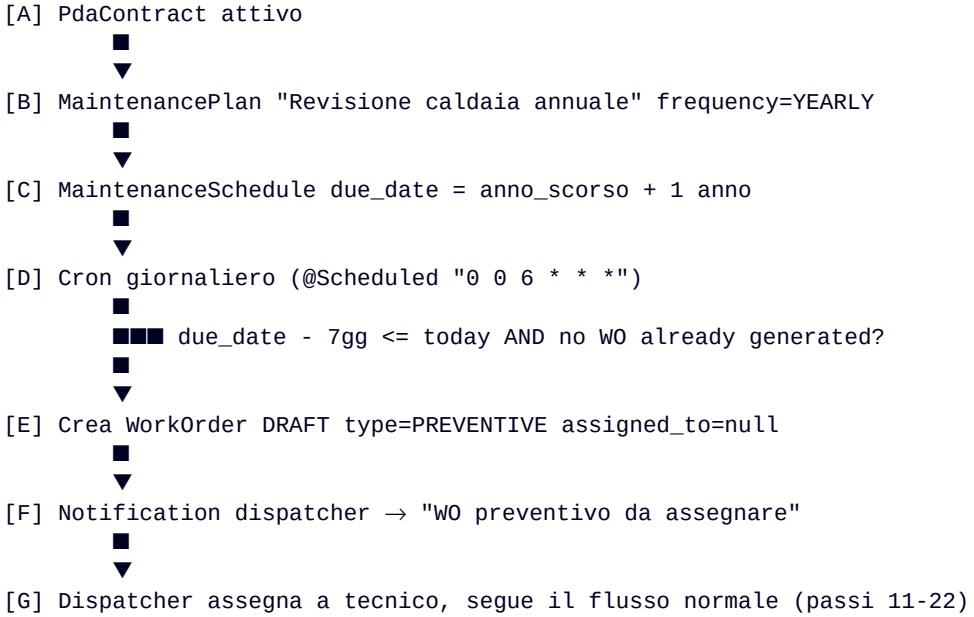
10. Diagrammi end-to-end

10.1 Flusso completo: dal cliente alla fattura

```
[1] Setup tenant + admin user (SuperAdmin)
        ■
        ▼
[2] Admin crea Customer "Hotel Barbieri SRL"
        ■
        ▼
[3] Admin crea Asset "Caldaia centrale 24kW" (collegato al customer)
        ■
        ▼
[4] Cliente attiva contratto PdaContract (revisione annuale)
        ■  ■■■ Maintenance Plan generato
        ■  ■■■ Maintenance Schedule (prossima scadenza)
        ■
        ▼
[5] Caldaia si rompe — il cliente manda WhatsApp al numero del tenant
        ■
        ▼
[6] 2Chat → POST /api/webhooks/whatsapp
        ■
        ▼
[7] WhatsAppWebhookController crea Ticket OPEN (source=WHATSAPP)
        ■
        ▼
[8] Notification all'admin (email + push)
        ■
        ▼
[9] Admin assegna Ticket → Tecnico TECHNICIAN_1
        ■  ■■■ Notification al tecnico
        ■
        ▼
[10] Admin promuove Ticket a WorkOrder (status=SCHEDULED)
         ■  ■■■ assigned_to + scheduled_at + materiali previsti
         ■
         ▼
[11] Tecnico riceve push notification sull'app mobile
         ■
         ▼
[12] Tecnico apre WO sull'app, scansiona QR caldaia, vede dettaglio asset
         ■
         ▼
[13] Tecnico clicca "Inizio lavoro" → status IN_PROGRESS, timer parte
         ■
         ▼
[14] Tecnico esegue, spunta checklist, scarica materiali da magazzino
         ■  ■■■ warehouse_movements OUT, warehouse_stock decrementato
         ■
         ▼
[15] Tecnico carica foto + firma cliente, clicca "Completa"
         ■  ■■■ WO COMPLETED, asset.last_maintenance_at = now()
         ■
         ▼
[16] Admin valida WO, clicca "Genera fattura"
         ■  ■■■ Invoice DRAFT con righe materiali + manodopera + IVA
         ■
         ▼
[17] Admin emette Invoice → status ISSUED
         ■  ■■■ SdiInvoice generata, XML FatturaPA inviato a SdI Aruba
         ■
         ▼
[18] SDI accetta → SdiInvoice ACCEPTED → cliente riceve fattura via PEC/SDI
```



10.2 Flusso parallelo: manutenzione preventiva automatica



10.3 Flusso parallelo: alert IoT con auto-ticket

```
[α] Sensor termometro frigorifero ristorante (HACCP)
        ■
        ▼
[β] Device IoT POST /api/iot/data ogni 10 minuti
        ■  Header: X-IoT-Api-Key (obbligatorio)
        ■  Body: { sensor_code: "FRIDGE-01", value: 8.5, unit: "°C" }
        ■
        ▼
[γ] IoTDataController valida API key
        ■
        ▼
[δ] Salva SensorReading + aggiorna sensor.last_reading_at
        ■
        ▼
[ε] Verifica thresholds: 8.5 > MAX (4.0)?  → BREACH
        ■
        ▼
[ζ] threshold.notification_enabled? → Notification ADMIN tenant
        ■
        ▼
[η] threshold.auto_create_ticket? → Crea Ticket
        ■  source=IOT_ALERT, priority=CRITICAL
        ■  asset = sensor.asset (frigorifero)
        ■
        ▼
[θ] Da qui in poi è il flusso normale di un ticket (triage, assegnazione,
    work order, ecc.)
```

Punti di estensione futuri

Cose previste ma non ancora implementate (vedi PIANO_QA per dettagli):

- **JWT auth per mobile** invece di JSESSIONID cookie (Fase 10.6)
- **Right-to-portability**: export GDPR del cliente in JSON (Fase 9.5)
- **Data retention scheduler** automatico (Fase 9.3)
- **Rate limiting distribuito** (oggi in-memory, OK per single-instance)
- **Sentry integration** per error tracking (Fase 8.4)
- **Multi-warehouse**: oggi un tenant ha 1 magazzino unico, in futuro N
- **Smart routing AI**: assegnazione automatica del tecnico più vicino + più qualificato per quel tipo di intervento

---

Glossario

Termine	Significato
Tenant	Cliente SaaS della piattaforma (es. Lean S.r.l.)
Customer	Cliente finale del tenant (es. Hotel Barbieri SRL)
Asset	Bene fisico installato presso il customer (caldaia, frigo)
Ticket	Segnalazione di problema o richiesta
Work Order (WO)	Intervento pianificato/eseguito sul campo
PDA	Piano di Assistenza (contratto a canone con SLA)
BoM	Bill of Materials (distinta base)
DDT	Documento Di Trasporto
SDI	Sistema di Interscambio (e-invoicing italiano)
HACCP	Hazard Analysis Critical Control Points (sicurezza alimentare)
CCP	Critical Control Point (punto critico di controllo HACCP)
MTBF	Mean Time Between Failures
MTTR	Mean Time To Repair
FCM	Firebase Cloud Messaging (push notification)
SuperAdmin	Owner della piattaforma (AgriSolution), vede tutti i tenant