PHASED EXECUTION PLAN

THE 90-DAY DIGITAL OPERATING SYSTEM ROADMAP

A Structured Blueprint for Consolidating Fragmented SaaS Subscriptions, Integrating Data Pipelines, and Executing Governed Workflows.

OBJECTIVE

To transition growing enterprises away from isolated, disconnected software subscriptions (e.g., forms, calendar plugins, spreadsheets) into a unified, secure database core. This process maintains operational continuity while saving over 90% in seat fees.

METHODOLOGY

Designed as a three-phase structure: Map and Stabilize (Days 1–30), Consolidate and Connect (Days 31–60), and Govern and Optimize (Days 61–90). Each phase implements specific data logic checks and validation gates.

EXECUTIVE SCOPE

This playbook outlines the technical requirements, PostgreSQL schemas, PL/pgSQL audit triggers, n8n routing configurations, and human override logic required to execute a secure operating system migration.

THE PHILOSOPHY OF PHASED MIGRATION

Rebuilding an organization's digital systems overnight introduces unacceptable business risks. Disrupting live customer touchpoints, email routing, or invoice processing damages cash flow and team morale. This roadmap rejects "big bang" deployments, advocating instead for a phased transition that stabilizes legacy systems while introducing new, unified databases in parallel.

THE THREE PRINCIPLES OF SYSTEM MIGRATION

1. Parallel Integration: Never disconnect a legacy tool (e.g., Calendly or Typeform) until its database equivalent has been active and monitored for a minimum of 14 days.

2. Schema-Driven Operations: Design and index database tables before creating integrations. Do not route unstructured payloads into key tables without strict validation rules.

3. Human Verification boundaries: Automated actions should not write directly to key database tables. The system must restrict the LLM to outputting structured JSON payloads, requiring manual admin review before updates occur.

WHY TRADITIONAL REBUILDS FAIL

Most companies start migrations by choosing a software vendor, asking: *"Which tool should we purchase?"* This approach leads to fragmented configurations. When tools are connected without a central database, data becomes scattered across separate platforms (CMS, Calendly, CRM, billing).

This roadmap prioritizes data centralization, ensuring your team works from a single source of truth.

PHASE 1: DAYS 1–30 (MAP & STABILIZE)

The primary objective of the first phase is to map all existing tools and data pathways without changing the live software stack. This mapping identifies manual copy-paste work, integration bottlenecks, and data security risks.

ACTIONS & PRIORITIES

During Phase 1, teams inventory all software subscriptions, noting duplicate seats and calculating total spend. They trace how data moves from public contact forms to internal spreadsheets and client emails, documenting every manual copy-paste action.

This step highlights the friction points in your current workflows, showing which integrations require immediate stabilization.

DELIVERABLE

A comprehensive stack inventory and a prioritized backlog. This inventory maps every tool to its business utility, tracking monthly costs and data dependencies.

ESTABLISHING SYSTEM GOVERNANCE BOUNDARIES

Before deploying any automated workflow, teams must establish clear guidelines. This includes auditing employee usage of third-party AI assistants, identifying unapproved browser extensions (shadow AI), and defining rollback parameters for all automated database updates.

PHASE 1 CHECKLIST & TOOL INVENTORY

Complete these mapping steps before writing database migrations or configuring automation nodes.

- [ ] **Document All Subscriptions:** List every software tool, monthly cost, and active team licenses.
- [ ] **Map Data Streams:** Trace how data moves from leads submitting forms to final client invoicing.
- [ ] **Audit Manual Tasks:** Note every task where team members copy-paste customer data between systems.
- [ ] **Establish Security Policies:** Define which data fields (e.g. BSN, emails) must be scrubbed before hitting external APIs.
- [ ] **Select Core Metrics:** Choose 3-5 operational metrics (e.g. lead response times) to monitor throughout the migration.

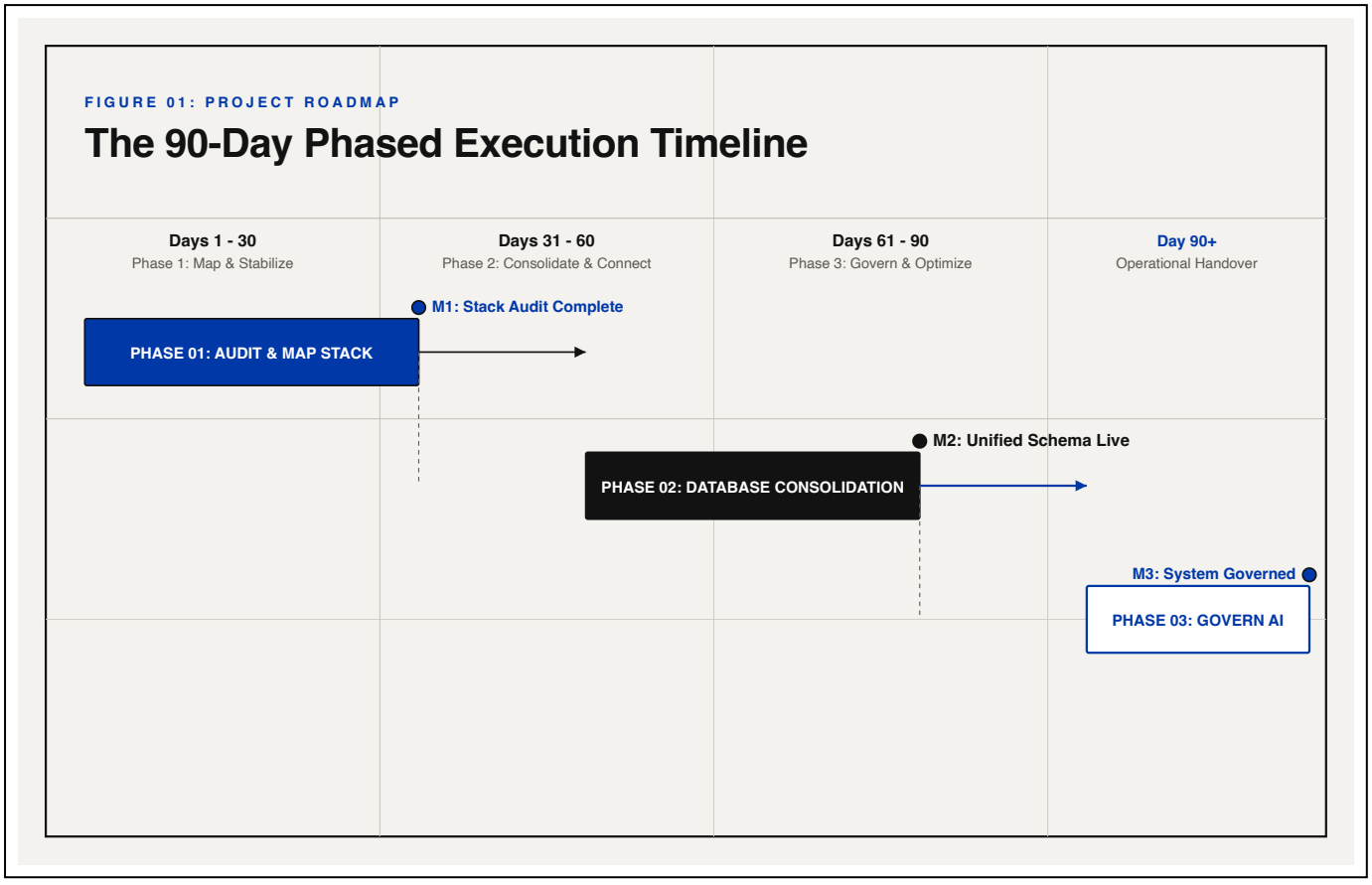
STACK COST & DEPENDENCY INVENTORY

SYSTEM LAYER	CURRENT TOOL	MONTHLY FEE	PRIMARY DATA SILO
Public CMS	WordPress	€45 / mo	Lead contact emails
Scheduler	Calendly	€15 / mo / seat	Appointment booking notes
Automation Hub	Zapier	€99 / mo	Disconnected webhook logs
Client CRM	HubSpot	€79 / mo / seat	Customer contact histories
Billing / Ledger	Excel Spreadsheet	€12 / mo	Manual invoice lists

Fig 1

FIGURE 01: THE 90-DAY EXECUTION TIMELINE

The chart below illustrates the overlapping execution windows of the mapping, database consolidation, and AI governance phases.



Note that Phase 2 (Database Consolidation) begins while Phase 1 is wrapping up. Similarly, Phase 3 (AI Governance) starts midway through Phase 2, ensuring that all data schemas and integrations are active before enforcing security policies.

PHASE 2: DAYS 31–60 (CONSOLIDATE & CONNECT)

During the second phase, organizations build their database core in parallel with their legacy tools. The goal is to route all data streams into a single database core, bypassing seat upgrade fees.

CONSOLIDATING DATA STREAMS

Teams configure n8n integrations to route lead data, appointment schedules, and transaction records directly into PostgreSQL tables (core_profiles, core_bookings, core_ledgers).

This removes the need for manual copy-pasting, ensuring that when an appointment is scheduled, the associated invoice and customer profile automatically sync.

SCHEMA MAPPING

By using foreign key constraints, the database serves as the single source of truth, preventing the duplicate records typical of multi-SaaS setups.

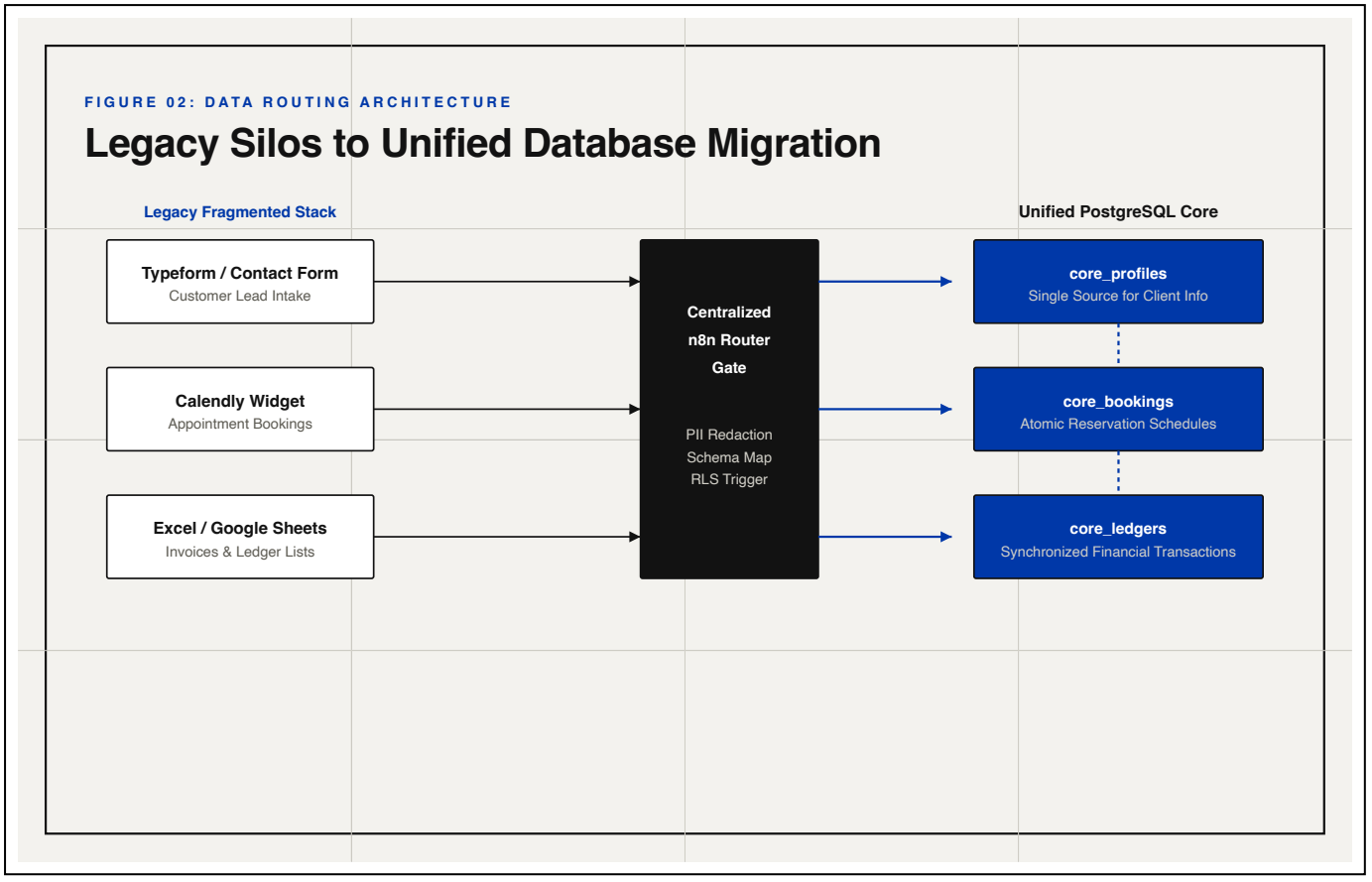
ROW-LEVEL SECURITY (RLS) POLICIES

Storing all client profiles, bookings, and ledger entries in a single database requires strict security. Row-Level Security (RLS) ensures that customers can access only their own records, while internal team members have access to the tables they need, keeping data private and secure.

FIGURE 02: LEGACY SILOS TO UNIFIED DATABASE MIGRATION

Fig 2

The routing matrix below maps the transition from fragmented SaaS tools to a single, unified database schema.



Rather than chaining tools together, all intakes flow through the central n8n gateway router, which performs safety checks before writing transactions to the database.

C2

DATABASE MIGRATION: CORE RELATIONS

Below is the SQL migration script to configure unified profile, booking, and ledger tables in PostgreSQL, with RLS policy setup.

```
-- Initialize relational database core
CREATE TABLE core_profiles (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    email VARCHAR(255) UNIQUE NOT NULL,
    full_name VARCHAR(255) NOT NULL,
    created_at TIMESTAMP WITH TIME ZONE DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE core_bookings (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    profile_id UUID REFERENCES core_profiles(id) ON DELETE CASCADE,
    booking_time TIMESTAMP WITH TIME ZONE NOT NULL,
    status VARCHAR(50) DEFAULT 'pending',
    created_at TIMESTAMP WITH TIME ZONE DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE core_ledgers (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    booking_id UUID REFERENCES core_bookings(id) ON DELETE CASCADE,
    amount_eur NUMERIC(10, 2) NOT NULL,
    paid_flag BOOLEAN DEFAULT FALSE,
    updated_at TIMESTAMP WITH TIME ZONE DEFAULT CURRENT_TIMESTAMP
);

-- Enable Row-Level Security
ALTER TABLE core_profiles ENABLE ROW LEVEL SECURITY;

-- Policy: Customers see only their own profile details
CREATE POLICY customer_select_policy ON core_profiles
    FOR SELECT USING (auth.uid() = id);
```

PHASE 3: DAYS 61–90 (GOVERN & OPTIMIZE)

Once data streams flow through the central database, the final phase introduces AI governance controls, ensuring all external queries are safe, audited, and cost-controlled.

MANAGING AI WORKFLOWS

The system intercepts outbound queries to public LLM endpoints, scrub PII (such as emails or Dutch BSN numbers), and logs request metadata to postgres audit tables.

This logging captures token usage, error rates, and calculated cost in EUR, helping operations track spend and prevent surprises.

AUDIT CONTROL

By using a central proxy gate, the company maintains a complete audit trail of AI usage, ensuring compliance with European and Dutch data protection rules.

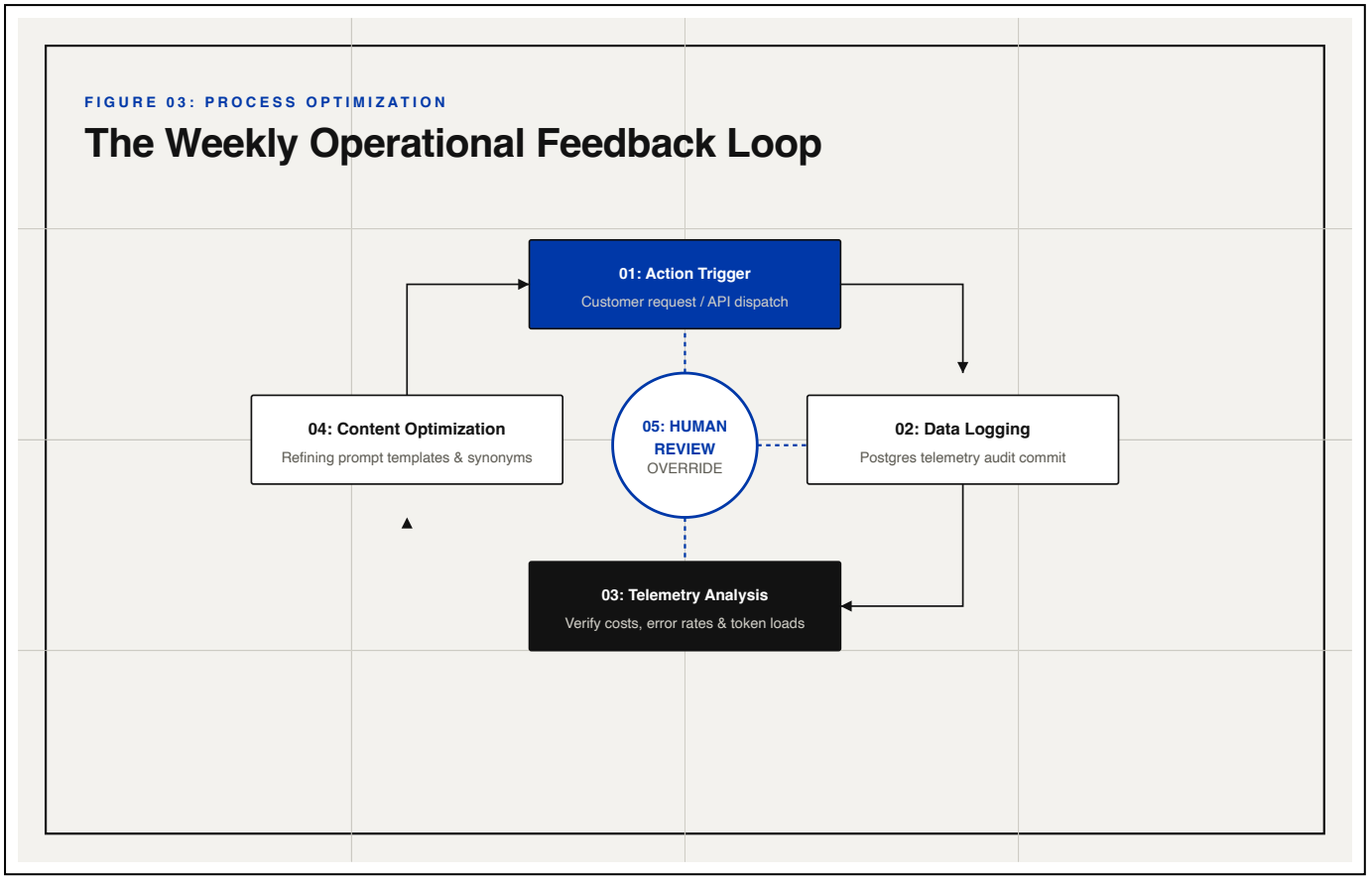
ESTABLISHING FEEDBACK LOOPS

Weekly reviews of these logs help identify routing errors and prompt drift. This feedback loop allows team members to update prompt templates and synonyms, improving output quality and system efficiency.

Fig 3

FIGURE 03: THE WEEKLY OPERATIONAL FEEDBACK LOOP

The process flow diagram below details the weekly telemetry analysis and human review cycles.



All automated actions flow back into the human review gate. Team members override, edit, or approve generated logs, ensuring quality control before final delivery.

DATABASE MIGRATION: TELEMETRY LOGS

Below is the SQL table migration code and Postgres trigger to log API telemetry and send automated Slack alerts if costs exceed thresholds.

```
-- Database table to capture proxy-level AI telemetry
CREATE TABLE telemetry_audit_log (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    model_requested VARCHAR(100) NOT NULL,
    token_cost_eur NUMERIC(8, 6) NOT NULL,
    latency_ms INTEGER,
    log_time TIMESTAMP WITH TIME ZONE DEFAULT CURRENT_TIMESTAMP
);

-- Trigger procedure to verify cost spikes
CREATE OR REPLACE FUNCTION check_cost_spikes()
RETURNS TRIGGER AS $$
BEGIN
    -- Trigger warning if a single API query cost exceeds €0.25
    IF NEW.token_cost_eur > 0.25 THEN
        PERFORM pg_notify('slack_cost_alert', json_build_object(
            'model', NEW.model_requested,
            'cost', NEW.token_cost_eur,
            'time', NEW.log_time
        )::TEXT);
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_check_cost_spikes
AFTER INSERT ON telemetry_audit_log
FOR EACH ROW EXECUTE FUNCTION check_cost_spikes();
```

APPENDIX A: 12-MONTH FINANCIAL PROJECTIONS

Migrating your software architecture from fragmented SaaS systems to a unified database core impacts your bottom line. Below is a financial projection based on an initial €60,000 investment.

YEAR PHASE	FRAGMENTED STACK COSTS	UNIFIED DATABASE COSTS	NET SAVINGS
Month 1-3	€3,105 / month (seat license totals)	€6,135 / month (inc. migration setup)	- €3,030
Month 4-6	€3,105 / month	€135 / month (Supabase + n8n hosting)	€2,970 / mo
Month 7-12	€3,550 / month (with seat growth)	€135 / month	€3,415 / mo
Year 1 Total	€38,515	€19,620 (inc. migration cost)	€18,895

BREAK-EVEN HORIZON

Based on our financial models, the migration achieves its break-even point in month 6. By year 3, direct subscription savings exceed €85,000, while reducing administrative labor hours by 24 hours per week.

APPENDIX B: DIAGNOSTIC TROUBLESHOOTING

Use these diagnostic steps if you encounter issues during the migration or scaling phases.

1. WEBHOOK PAYLOAD MISMATCH

Symptom: Incoming lead data from public forms fails to write to the core_profiles table.
Resolution: Check the n8n execution log to verify that the request body conforms to the database's schema rules, and check that required fields are present.

2. ROW-LEVEL SECURITY ACCESS DENIED

Symptom: Internal team members are unable to view booking logs in the custom interface.
Resolution: Verify the user's JWT token payload. Ensure that the role claim matches 'staff' as defined in your database policies.

3. TELEMETRY ALERT LOOPS

Symptom: Slack cost notifications trigger continuously.
Resolution: Review the database's cost thresholds. Check that API queries are cache-enabled to prevent duplicate queries, and check the trigger threshold value.

4. API RATE LIMITS

Symptom: Integrations fail during peak traffic windows.
Resolution: Enforce queue limits in your n8n workflows, enabling automatic retries with exponential backoff on all API calls.

SYSTEM VERIFICATION & REFERENCES

The migration is complete once all steps in the checklist below are verified.

- [ ] **Database Schema Confirmed:** Verify that all core tables are active with valid foreign key constraints.
- [ ] **Security Policies Audited:** Run automated checks to verify that RLS policies prevent unauthorized data exposure.
- [ ] **n8n Pipelines Active:** Verify that lead forms, calendar schedulers, and invoicing tables sync correctly.
- [ ] **AI Telemetry Logged:** Confirm that the proxy gate logs all outgoing query metadata, token usage, and costs.
- [ ] **Slack Cost Alerts Verified:** Test cost alert notify functions to confirm alerts trigger above thresholds.

REFERENCES

Osterwalder, A. & Pigneur, Y. (2010). *Business Model Generation*. John Wiley & Sons.

European Union (2016). *General Data Protection Regulation (GDPR)*. Regulation (EU) 2016/679.

Swiss Design Standard (2026). *Minimalist Grid Systems for Complex Corporate Reports*. Zurich Print Press.