next8n

# MASTER ARCHITECTURE DIAGRAM

Workflow Automation Delivery Framework

ENTERPRISE EDITION

**Version:** 2.0

**Date:** December 28, 2025

**Author:** Mirza Iqbal

**Contact:** mirza.iqbal@next8n.com

# Table of Contents

## ADR-003: Test/Production Separation

# Master Architecture Diagram

## Complete Workflow Automation Delivery System

# 1. High-Level System Architecture

```
flowchart TB
    subgraph CLIENT["CLIENT DOMAIN"]
        subgraph CLIENT_INFRA["Client Infrastructure"]
            N8N_PROD[" n8n Production<br/>Client-Owned Instance"]
            N8N_TEST[" n8n Test Environment<br/>Staging/Development"]
            CLIENT_DB[(Client Databases<br/>CRM, ERP, etc.)]
            CLIENT_APPS["Client Applications<br/>Email, Calendar, etc."]
        end

        subgraph CLIENT_CREDS["Client Credentials"]
            API_KEYS[" API Keys<br/>OpenAI, Anthropic, etc."]
            OAUTH["OAuth Tokens<br/>Google, Microsoft, etc."]
            SECRETS[" Secrets Vault<br/>Passwords, Tokens"]
        end

        CLIENT_TEAM["Client Team<br/>Users & Stakeholders"]
    end

    subgraph CONSULTANT[" CONSULTANT DOMAIN"]
        subgraph DEV_ENV["Development Environment"]
            CONSULTANT_N8N[" Consultant n8n<br/>Internal Use Only"]
            TEMPLATES[" Template Library<br/>Reusable Components"]
            DEV_TOOLS["Dev Tools<br/>Testing, Debugging"]
        end

        subgraph DELIVERY["Delivery Assets"]
            DOCS[" Documentation<br/>Guides, Videos"]
            EXPORTS[" Workflow Exports<br/>JSON Backups"]
            TRAINING["Training Materials<br/>Loom Videos"]
        end

        CONSULTANT_TEAM[" Consultant/Agency"]
    end

    subgraph EXTERNAL["EXTERNAL SERVICES"]
        AI_PROVIDERS[" AI Providers<br/>OpenAI, Anthropic, etc."]
        INTEGRATIONS["Integrations<br/>Slack, HubSpot, etc."]
        WEBHOOKS["Webhook Sources<br/>Stripe, GitHub, etc."]
    end

    %% Relationships
    CONSULTANT_TEAM -->|"Invited as Team Member"| N8N_PROD
    CONSULTANT_TEAM -->|"Develops & Tests"| N8N_TEST
    TEMPLATES -->|"Deploy Patterns"| N8N_PROD
    DOCS -->|"Delivered to"| CLIENT_TEAM

    N8N_PROD <-->|"Connects via"| CLIENT_DB
    N8N_PROD <-->|"Integrates with"| CLIENT_APPS
    N8N_PROD -->|"Uses"| API_KEYS
    N8N_PROD -->|"Authenticates via"| OAUTH
```

```
    AI_PROVIDERS <-->|"API Calls"| N8N_PROD
    INTEGRATIONS <-->|"Data Sync"| N8N_PROD
    WEBHOOKS -->|"Triggers"| N8N_PROD

    N8N_PROD -->|"Backup to"| EXPORTS

    style CLIENT fill:#e1f5fe,stroke:#01579b
    style CONSULTANT fill:#f3e5f5,stroke:#4a148c
    style EXTERNAL fill:#fff3e0,stroke:#e65100
```

## 2. Hosting Models Comparison

```
flowchart LR
    subgraph MODEL1[" RECOMMENDED: Client Hosts"]
        direction TB
        C1_CLIENT["Client"] -->|"Owns & Pays"| C1_N8N["n8n Instance"]
        C1_CONSULTANT["Consultant"] -->|"Invited as User"| C1_N8N
        C1_N8N -->|"Client Pays"| C1_APIS["API Usage"]
        style MODEL1 fill:#c8e6c9,stroke:#2e7d32
    end

    subgraph MODEL2[" INTERNAL: Consultant Hosts Own"]
        direction TB
        C2_CONSULTANT["Consultant"] -->|"Owns for Internal"| C2_N8N["n8n Instance"]
        C2_N8N -->|"Powers"| C2_SERVICE["Service Delivery"]
        C2_SERVICE -->|"Deliverable to"| C2_CLIENT["Client"]
        C2_CLIENT -.->|"Never Sees n8n"| C2_N8N
        style MODEL2 fill:#fff9c4,stroke:#f9a825
    end

    subgraph MODEL3["REQUIRES LICENSE: SaaS Model"]
        direction TB
        C3_CONSULTANT["Consultant"] -->|"Hosts as Product"| C3_N8N["n8n Platform"]
        C3_CLIENTS["Multiple Clients"] -->|"Access"| C3_N8N
        C3_N8N -->|"Requires"| C3_LICENSE["Commercial License"]
        style MODEL3 fill:#ffcdd2,stroke:#c62828
    end
```

# 3. Data Flow Architecture

```
flowchart TB
    subgraph INPUT["INPUT LAYER"]
        WEBHOOK["Webhook Triggers"]
        SCHEDULE["Scheduled Triggers"]
        MANUAL["Manual Triggers"]
        APP_TRIGGER["App Triggers"]
    end

    subgraph PROCESSING[" PROCESSING LAYER"]
        subgraph VALIDATION["Validation"]
            AUTH_CHECK["Authentication Check"]
            SIGNATURE_VERIFY["Signature Verification"]
            RATE_LIMIT["Rate Limiting"]
        end

        subgraph TRANSFORMATION["Data Transformation"]
            PARSE["Parse & Extract"]
            ENRICH["Enrich Data"]
            NORMALIZE["Normalize Format"]
        end

        subgraph AI_LAYER["AI Processing"]
            LLM_CALL["LLM API Call"]
            PROMPT_MGMT["Prompt Management"]
            RESPONSE_PARSE["Response Parsing"]
        end
    end

    subgraph OUTPUT["OUTPUT LAYER"]
        CRM_UPDATE["CRM Updates"]
        EMAIL_SEND["Email/Notifications"]
        DATA_STORE["Data Storage"]
        REPORT_GEN["Report Generation"]
    end

    subgraph MONITORING[" MONITORING LAYER"]
        EXEC_LOG["Execution Logs"]
        ERROR_TRACK["Error Tracking"]
        USAGE_METRICS["Usage Metrics"]
        ALERT_SYSTEM["Alert System"]
    end

    INPUT --> VALIDATION
    VALIDATION --> TRANSFORMATION
    TRANSFORMATION --> AI_LAYER
    AI_LAYER --> OUTPUT

    PROCESSING --> MONITORING
    OUTPUT --> MONITORING

    style INPUT fill:#e3f2fd
```

```
    style PROCESSING fill:#f3e5f5
    style OUTPUT fill:#e8f5e9
    style MONITORING fill:#fff8e1
```

# 4. Security Architecture

```
flowchart TB
    subgraph PERIMETER[" PERIMETER SECURITY"]
        HTTPS["HTTPS Encryption"]
        WAF["Web Application Firewall"]
        DDOS["DDoS Protection"]
    end

    subgraph ACCESS["ACCESS CONTROL"]
        RBAC["Role-Based Access"]
        MFA["Multi-Factor Auth"]
        SSO["Single Sign-On"]
        SESSION["Session Management"]
    end

    subgraph DATA["DATA SECURITY"]
        subgraph AT_REST["At Rest"]
            CRED_ENCRYPT["Credential Encryption<br/>AES-256"]
            DB_ENCRYPT["Database Encryption"]
        end

        subgraph IN_TRANSIT["In Transit"]
            TLS["TLS 1.3"]
            SIGNED["Signed Payloads"]
        end

        subgraph IN_MEMORY["In Memory"]
            DECRYPT_RUNTIME["Runtime Decryption Only"]
            SECURE_HEAP["Secure Memory Handling"]
        end
    end

    subgraph COMPLIANCE[" COMPLIANCE"]
        GDPR["GDPR Compliance"]
        DATA_MIN["Data Minimization"]
        AUDIT_LOG["Audit Logging"]
        RETENTION["Data Retention Policy"]
    end

    PERIMETER --> ACCESS
    ACCESS --> DATA
    DATA --> COMPLIANCE

    style PERIMETER fill:#ffebee
    style ACCESS fill:#e8eaf6
    style DATA fill:#e0f2f1
    style COMPLIANCE fill:#fff3e0
```

# 5. Project Delivery Pipeline

```
flowchart LR
    subgraph PHASE1["Phase 1: Discovery"]
        DISCOVERY["Discovery Call"]
        SCOPE["Scope Definition"]
        CONTRACT["Contract Signing"]
    end

    subgraph PHASE2["Phase 2: Setup"]
        KICKOFF["Kickoff Call"]
        ENV_SETUP["Environment Setup"]
        CRED_CONFIG["Credential Config"]
    end

    subgraph PHASE3["Phase 3: Development"]
        BUILD["Build Workflows"]
        INTERNAL_QA["Internal QA"]
        ITERATE["Iterate & Refine"]
    end

    subgraph PHASE4["Phase 4: Testing"]
        CLIENT_TEST["Client Testing"]
        FEEDBACK["Feedback Collection"]
        REFINEMENT["Final Refinements"]
    end

    subgraph PHASE5["Phase 5: Delivery"]
        HANDOVER["Handover"]
        TRAINING["Training"]
        DOCS_DELIVER["Documentation"]
    end

    subgraph PHASE6["Phase 6: Support"]
        GO_LIVE["Go Live"]
        MONITOR["Monitor"]
        MAINTAIN["Maintain"]
    end

    PHASE1 --> PHASE2 --> PHASE3 --> PHASE4 --> PHASE5 --> PHASE6

    style PHASE1 fill:#e3f2fd
    style PHASE2 fill:#e8f5e9
    style PHASE3 fill:#fff3e0
    style PHASE4 fill:#fce4ec
    style PHASE5 fill:#f3e5f5
    style PHASE6 fill:#e0f7fa
```

# 6. Credential Flow Architecture

```
sequenceDiagram
    participant Client as  Client
    participant Vault as Secret Vault
    participant N8N as  n8n
    participant API as External API

    Note over Client,API: Secure Credential Setup Flow

    Client->>Client: Signs up for API service
    Client->>Client: Generates API key
    Client->>Vault: Stores key in secure vault
    Vault->>Client: Generates one-time share link
    Client->>N8N: Pastes credential (or shares link)

    Note over N8N: Key encrypted at rest

    N8N->>N8N: Stores encrypted credential

    Note over Client,API: Runtime Execution Flow

    N8N->>N8N: Workflow triggered
    N8N->>N8N: Decrypts credential in memory
    N8N->>API: Makes API call with credential
    API->>N8N: Returns response
    N8N->>N8N: Clears credential from memory

    Note over N8N: Key never stored in plaintext
```

# 7. Error Handling & Recovery Architecture

```
flowchart TB
    subgraph DETECTION[" ERROR DETECTION"]
        EXEC_FAIL["Execution Failure"]
        TIMEOUT["Timeout"]
        API_ERROR["API Error"]
        VALIDATION_FAIL["Validation Failure"]
    end

    subgraph CLASSIFICATION[" ERROR CLASSIFICATION"]
        CRITICAL[" Critical<br/>System Down"]
        HIGH[" High<br/>Partial Failure"]
        MEDIUM[" Medium<br/>Degraded Performance"]
        LOW[" Low<br/>Minor Issue"]
    end

    subgraph RESPONSE[" RESPONSE ACTIONS"]
        subgraph IMMEDIATE["Immediate"]
            RETRY["Auto Retry"]
            FALLBACK["Fallback Logic"]
            GRACEFUL["Graceful Degradation"]
        end

        subgraph NOTIFY["Notification"]
            ALERT_TEAM["Alert Team"]
            LOG_ERROR["Log to Sheet"]
            SLACK_NOTIFY["Slack Notification"]
        end

        subgraph RECOVER["Recovery"]
            MANUAL_FIX["Manual Intervention"]
            ROLLBACK["Rollback"]
            RESTART["Restart Workflow"]
        end
    end

    DETECTION --> CLASSIFICATION
    CRITICAL --> ALERT_TEAM
    CRITICAL --> MANUAL_FIX
    HIGH --> FALLBACK
    HIGH --> LOG_ERROR
    MEDIUM --> RETRY
    MEDIUM --> SLACK_NOTIFY
    LOW --> LOG_ERROR

    style DETECTION fill:#ffebee
    style CLASSIFICATION fill:#fff3e0
    style RESPONSE fill:#e8f5e9
```

# 8. Maintenance & Monitoring Architecture

```
flowchart TB
    subgraph MONITORING[" MONITORING SYSTEMS"]
        EXEC_MONITOR["Execution Monitor"]
        ERROR_MONITOR["Error Monitor"]
        USAGE_MONITOR["Usage Monitor"]
        COST_MONITOR["Cost Monitor"]
    end

    subgraph LOGGING[" LOGGING SYSTEMS"]
        EXEC_LOG["Execution History"]
        ERROR_LOG["Error Log<br/>(Google Sheet)"]
        AI_LOG["AI Response Log"]
        AUDIT_LOG["Audit Trail"]
    end

    subgraph ALERTING["ALERTING"]
        EMAIL_ALERT["Email Alerts"]
        SLACK_ALERT["Slack Alerts"]
        SMS_ALERT["SMS (Critical)"]
    end

    subgraph MAINTENANCE[" MAINTENANCE"]
        BACKUP["Automated Backups"]
        CLEANUP["Log Cleanup"]
        UPDATE["Version Updates"]
        HEALTH["Health Checks"]
    end

    MONITORING --> ALERTING
    MONITORING --> LOGGING
    LOGGING --> MAINTENANCE

    style MONITORING fill:#e3f2fd
    style LOGGING fill:#e8f5e9
    style ALERTING fill:#ffebee
    style MAINTENANCE fill:#fff3e0
```

# Architecture Decision Records

## ADR-001: Client-Hosted Infrastructure

**Decision**: Clients host their own n8n instances
**Rationale**:
- Complies with n8n licensing
- Client owns data and credentials
- Clean separation of concerns
- No billing complexity

## ADR-002: Credential Ownership

**Decision**: Clients own and pay for all API credentials
**Rationale**:
- Transparent billing
- No markup disputes
- Client retains control
- Easier handover

## ADR-003: Test/Production Separation

**Decision**: Maintain separate test and production environments
**Rationale**:
- Safe testing without production impact
- Validate updates before deployment
- Professional development practices

---

**Next**: See `02-hosting-decision-tree.md` for detailed hosting decisions.

---