How to Actually Deliver AI Projects (APIs, Hosting & Handover Explained)

By: Nate Herk

This guide outlines the professional lifecycle of delivering AI workflows and agents to clients, focusing on compliance, security, testing, and business structure.

THE COMPLIANCE GUIDE (n8n Sustainable Use License)

The n8n Sustainable Use License (SUL) is a "fair-code" license. It is source-available with specific commercial restrictions.

The Core Restriction

You may use n8n for your own **Internal Business Purposes**. Use is only restricted if you are selling a product or service where the value derives **entirely or substantially** from n8n functionality.

What is 100% Allowed (No Commercial License Needed)

- **Workflow Consulting:** Charging a client to build, optimize, or fix a workflow that runs on *their* n8n instance (Cloud or Self-Hosted).
- **Infrastructure Setup:** Charging a client to install and configure n8n on their own VPS (DigitalOcean, AWS, etc.).
- **Agency Operations:** Using your own n8n instance to automate your lead gen, reporting, or internal AI agents.
- **Result-Based Services:** Using n8n on your own server to process data and sending the *output* (a report, a filtered lead list, an email) to a client, provided the client never interacts with n8n.
- **Custom Node Development:** Building and selling custom n8n nodes or integrations.

What is Prohibited (Requires a Paid Embed/Enterprise License)

- **"Automation-as-a-Service":** Hosting n8n on your server and giving clients their own login/sub-account to "your" platform.
- **White-Labeling:** Rebranding the n8n UI as your own product and selling access to it.
- **External Value-Add:** Building a SaaS product where the primary "feature" is the ability for users to build or manage their own workflows.

- **Managed Instances for Resale:** Renting out n8n instances at a markup (e.g., "Get a managed n8n instance from me for $99/mo").

The "Grey Area" Warning

If you host a workflow for a client on **your** server and that workflow processes **their** data using **their** API keys, n8n generally views this as "providing n8n as a service."

- **The Safe Path:** Always have the client sign up for their own account. Even if you manage it, the billing and legal "ownership" of the instance must stay with the client.

---

1. Hosting Strategies: Who Owns the Infrastructure?

The hosting model is dictated by the software license (specifically n8n) and your business model.

Option	Model	Best For	Compliance Note
Option 1	**Client Hosted**	**Consulting / Freelance**	Each client has their own instance. You are a "builder" in their seat.
Option 2	**Agency Hosted**	**Internal Tools / Deliverables**	Clients never log in or see the UI. You deliver the *result*, not the tool.
Option 3	**SaaS / Platform**	**Productized Services**	Requires a **Commercial License**. Clients log in or use your server for their keys.

Pro Recommendation: Always aim for **Option 1**. Let the client pay for the subscription and own the environment. It removes your liability for hosting costs and license violations.

---

2. Security and Data Protection

Building "production-ready" workflows requires moving beyond basic functionality into data safety.

Technical Hardening

- **Credential Encryption:** n8n encrypts credentials at rest and only decrypts them in memory during execution.
- **Webhook Security:**
    - Always use **HTTPS**.
    - Implement **Signing Secrets** (e.g., Stripe/GitHub signatures).
    - Add **Rate Limiting** to prevent brute-force attacks on your trigger URLs.
- **AI Guardrails:** Use system prompts to prevent "jailbreaking" or prompt injection by end-users.

Privacy Compliance (GDPR/Data Laws)

1. **Data Minimization:** Only pull the specific fields needed for the automation.
2. **Access Control:** Limit who can see execution logs and payloads.
3. **Data Sovereignty:** Highlight that self-hosting n8n allows clients to keep data on their own servers (on-prem or private VPS) rather than sending it to third-party clouds.

---

3. API Key Management & Billing

The "Golden Rule": **The client owns the keys; the client pays the bill.**

- **Avoid Middleman Billing:** Charging clients for token usage creates accounting nightmares and lack of transparency.
- **Secure Transfer:** Never send keys over Slack, Email, or Text. Use encrypted vaults like **1Password** or one-time secret links.
- **Education:** Provide a **Loom video** walking the client through how to generate their own API keys (OpenAI, Anthropic, etc.) and where to paste them.

---

4. The Testing & QA Framework

Don't just check if the nodes "run." Test for quality and failure.

Phase 1: Internal QA (Engineer Mindset)

- **Black Box Testing:** Feed the workflow dozens of real-world examples (anonymized if necessary).
- **Planning for Failure:** What happens if a tool returns an error?

    - Implement **Error Workflows** to alert the team.
    - Use **"Continue on Fail"** logic where appropriate.
    - Log all failures to a Google Sheet for pattern analysis.

Phase 2: AI-Specific QA

1. **Relevance:** Does it actually answer the prompt?
2. **Tone & Safety:** Is it off-brand or toxic?
3. **Consistency:** Do similar inputs produce similar quality results?
4. **Logging:** Track tokens, tool calls, and outputs in a sheet to prove the model's efficacy to the client.

Phase 3: Client-Facing QA

- Provide a sandbox (Chatbox, Form, or Webhook) for the client to try.
- Differentiate between **Bugs** (broken logic) and **Tuning** (tone/formatting tweaks).

---

5. Professional Handover Process

A clean handover separates "Version 1" from future work and protects your reputation.

- **Production vs. Test Environments:** Duplicate the workflow. Keep one for testing and push a clean version to production.
- **Version Control:** Manually or automatically export the JSON to GitHub or Google Drive as a backup.
- **Documentation:** * **Internal:** Use "Sticky Notes" and clear naming conventions inside the workflow.
    - **External:** Provide a Loom walkthrough and a PDF map of the workflow logic.

---

6. Legal & Business Terms

To avoid "scope creep" and payment delays, define the boundaries early.

- **Definition of Done:** The project ends when the agreed-upon workflows are live, tested, and documented.
- **Maintenance Retainers:** Charge a monthly fee for bug fixes, monitoring, and updates. This *does not* include building new features.
- **Intellectual Property (IP):** * **Client Owns:** The specific implementation, prompts, and their data.
    - **You Own:** The right to reuse generic logic patterns, sub-workflows, and templates for other clients.

- **Service Level Agreements (SLA):** Set expectations for response times (e.g., 4 hours for critical outages, 48 hours for minor bugs).

Case Study: The "Assistant" Workflow

- **Kickoff:** Invited to client's n8n account during the first call.
- **Build:** Developed entirely inside the client's environment.
- **Benefit:** Handover was instantaneous; no migration or secret transfer was required.
- **Lesson:** Boundaries are key. When the client asked for more features during testing, they were added to a "Backlog" for a separate, paid Version 2 project.

Want to connect with others building and monetizing AI automation?
Become an AIS Plus Member