LeadGen OS

Automating lead generation & outreach with AI agents

Technical Blueprint — Investor-Ready Edition

Agentic AI • v1.0 — Beta Launch 2025

Table of Contents

1. Executive Summary

LeadGen OS is a launch-ready website and backend for Agentic AI’s core business development. This blueprint documents the current v1.0 (Beta) — the Astro site, FastAPI backend, n8n webhook integration, and docs that render into investor-ready PDFs.

Objectives: Launch a professional Render-hosted site, document all Agent deliverables (1→7), and expose a minimal backend for automation. Scope is strictly launch-focused (no future roadmap included).

2. Agent Build Sequence (1→7)

#	Agent	Role	Deliverables
1	Architect	Lock PRD, schemas, repo	PRD.md, schema files, repo map, env template
2	Planner	Create dev roadmap	Kanban board, task breakdown, milestones
3	Backend	Implement logic layer	FastAPI router stubs, validators, migrations
4	n8n	Integrate core workflows	Workflow JSONs, exposed webhooks, registry
5	Frontend	UI & UX flows	Page structure, component tree, state diagram
6	QA	Test automation + validation	Acceptance test matrix, coverage report
7	Writer	Final documentation	Docs site, README, onboarding

3. PRD (Current v1.0 Scope)

Problem & Goals
Present Agentic AI capabilities, enable minimal backend for n8n, and document agent deliverables.

Scope (v1.0)
• Astro site (landing + agents + deliverables)
• FastAPI backend (/health, /agents, /webhooks/n8n/:name)
• Docs repo with CI PDF builds
• Render deploy via render.yaml
Non-goals: auth, DB, advanced workflows

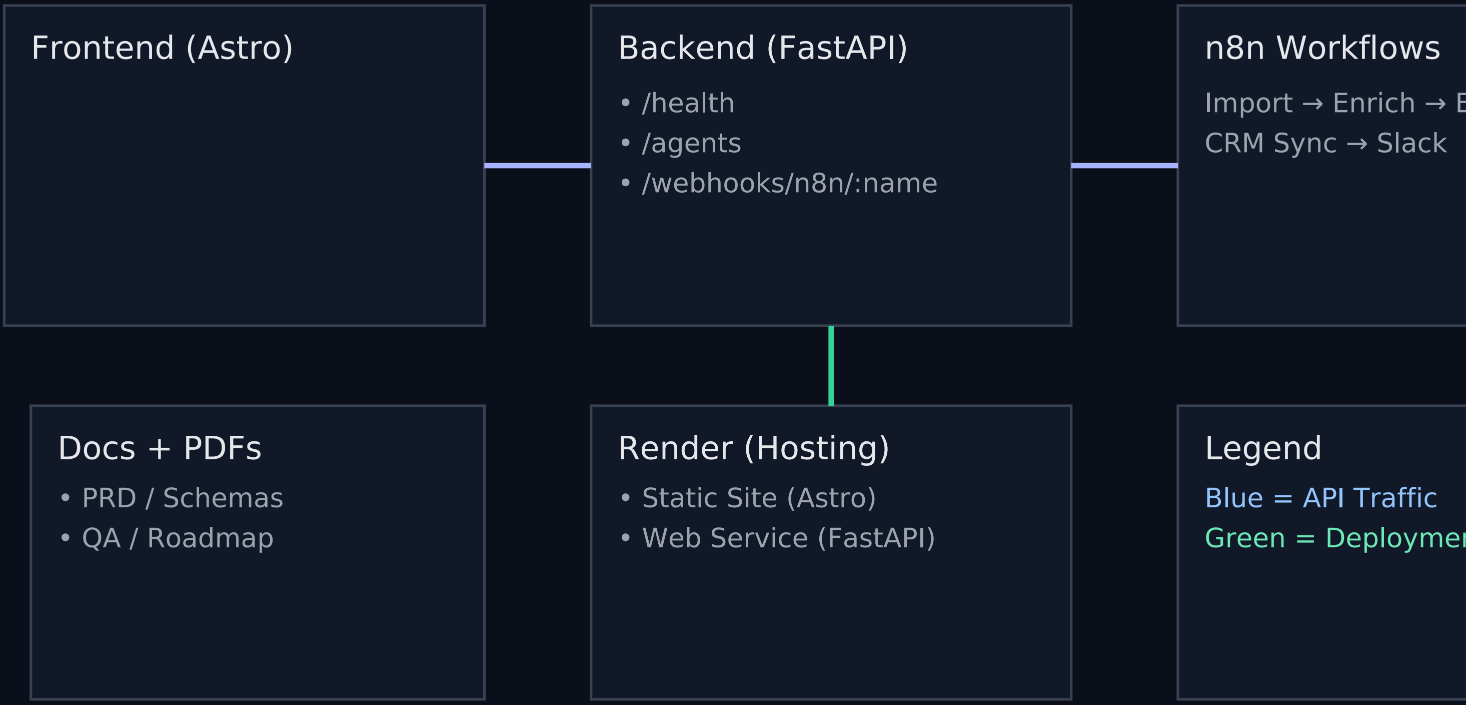
Users & Flows
Visitor → capabilities → contact
Operator → docs commit → PDF via CI
n8n → webhook → JSON ack

Success Metrics
Site live on Render; PDFs render in CI; Webhook 2xx.

4. Architecture & Repo Structure

Repo Structure
apps/backend — FastAPI app
apps/site — Astro static site
docs — PRD, Schemas, QA (CI → PDF)
workflows — n8n JSON + registry.json
render.yaml — Render blueprint

Tech
Astro, FastAPI, GitHub Actions (Pandoc), Render.


Frontend (Astro)
Backend (FastAPI)
• /health
• /agents
• /webhooks/n8n/:name
n8n Workflows
Import → Enrich → E
CRM Sync → Slack
Docs + PDFs
• PRD / Schemas
• QA / Roadmap
Render (Hosting)
• Static Site (Astro)
• Web Service (FastAPI)
Legend
Blue = API Traffic
Green = Deploymer

6. n8n Integration & Webhook Registry

Incoming Webhooks

• POST /webhooks/n8n/import_csv
• POST /webhooks/n8n/enrich_score
• POST /webhooks/n8n/send_campaign

Registry

workflows/registry.json maps friendly names to endpoints for consistent wiring.

7. Testing & QA Strategy

Acceptance Matrix

- /health returns 200
- Webhooks accept JSON & return 2xx
- Landing shows Agents grid

Validation

- Optional lint in CI
- Synthetic payload tests for webhooks

8. Deployment on Render (GitHub → Render)

Prereqs

• GitHub repo with render.yaml

• Services: static site + python web service

Steps

1) Push repo to GitHub

2) Render: New → Blueprint → select repo

3) Wait for builds

4) Point n8n webhooks to backend URL

9. Security, Privacy, and Compliance

Secrets: Use env vars (Render); do not commit secrets.
CORS: Restrict to known origins when domain stabilizes.
Privacy: Avoid PII in webhook payloads for v1.0.

10. Success Metrics (v1.0)

- Site live with Agents grid
- Backend /health and /agents stable
- CI renders PDFs from docs on push
- n8n POST to webhook returns 2xx with echo payload