

The AI Tron Starter Kit — Build Your First AI Agent in Under an Hour

Version 1.0 · 2026 · Includes: Blueprint PDF + 3 Plug-and-Play Bot Templates + Prompt Vault Starter

1. Read This First

This kit gets you from zero to a working AI agent (a “Tron”) running on your own machine in under an hour. No cloud subscription required. No developer fees. Just your computer, a free model runtime, and the templates in this bundle.

By the end of this guide you will have:

1. A local model runtime installed (Ollama — free, runs offline)
2. A working agent loop in Python (~170 lines, fully yours, no vendor lock-in)
3. Three ready-to-customize bot templates: Customer Support, Lead Qualification, FAQ Automator
4. A test battery to prove your Tron works before you point it at real customers

Everything in this kit is plain, standard tooling: Python, HTTP, JSON. If you can install an app, you can run this.

2. Install Your Model Runtime (10 minutes)

Step 1 — Install Ollama. Download from <https://ollama.com> for macOS, Linux, or Windows. One installer. Done.

Step 2 — Pull a starter model. Open a terminal:

```
ollama pull llama3.2:3b      # small + fast – good for a first Tron
ollama run llama3.2:3b "Say READY if you can hear me."
```

If it answers `READY`, your runtime is alive. You now own a private AI brain that never phones home.

Hardware notes:

Your machine	Model size	Expected feel
8 GB RAM laptop	3B parameter	Snappy, good enough to start
16 GB RAM / desktop	7–8B parameter	Solid quality
GPU with 8 GB+ VRAM	7–13B parameter	Fast enough for real work

Start small. A Tron that answers in 2 seconds beats a sluggish genius.

3. The Agent Loop — Your Tron’s Spine (20 minutes)

Every AI agent ever built reduces to the same spine: **listen** → **think** → **act** → **verify**.

Save this as `agent.py` (also included in the templates folder as `00-agent-loop/agent.py`):

```

import json, time, urllib.request

# --- config ---
MODEL = "llama3.2:3b"
OLLAMA = "http://127.0.0.1:11434/api/chat"

def think(system: str, history: list, user: str) -> str:
    """One round-trip against the local model."""
    body = json.dumps({
        "model": MODEL,
        "messages": [{"role": "system", "content": system}] + history
                    + [{"role": "user", "content": user}],
        "stream": False,
    }).encode()
    req = urllib.request.Request(OLLAMA, data=body,
                                headers={"Content-Type":
                                         "application/json"})
    with urllib.request.urlopen(req, timeout=120) as r:
        return json.load(r)["message"]["content"]

def run():
    system = ("You are a helpful business assistant named Tron. "
             "Answer in 2-4 sentences. If you are unsure, say so.")
    history, turns = [], 0
    while turns < 20:                # bounded loop = safe by
        construction
        user = input("you: ").strip()
        if user.lower() in {"quit", "exit"}:
            break
        reply = think(system, history, user)
        print("tron:", reply)
        history += [{"role": "user", "content": user},
                    {"role": "assistant", "content": reply}]
        turns += 1

if __name__ == "__main__":
    run()

```

Run it:

```
python3 agent.py
```

That's the entire skeleton. Every template in this kit is this loop wearing a different uniform: a different `system` prompt and a different `act` step (where it sends what it thought).

The three patterns you can build on this spine:

Pattern	Listen	Act	Real-world use
Responder	customer message	drafts a reply	support, FAQ
Router	inbound request	classifies → forwards	lead triage
Watcher	scheduled tick	checks condition → alerts	inventory, renewals

4. Train It on Your Business — The 4-Hour Curriculum

Hour 1 — Identity. Rewrite the `system` prompt: your business name, tone, and the 3 things the Tron must never do. Example core: “You are the front desk of Acme Solar. Never quote prices. Never promise discounts. Always offer to book a callback.”

Hour 2 — Knowledge. Paste your top 20 questions + your real answers into a `docs.json`:

```
[{"q": "Do you offer repairs?", "a": "Yes – book a slot at our counter or by phone."}, {"q": "What are your hours?", "a": "Mon-Fri 9-5, Sat 10-2."}]
```

Then add this to the loop: before `think()`, scan `docs.json` for a keyword match; if found, inject the answer’s text into the system message (“Use ONLY this fact if relevant: ...”). That single change turns a generic chatbot into *your* employee.

Hour 3 — Deployment. Wrap `run()` in an HTTP handler (any web framework) so your website, whatsapp bridge, or discord bridge can POST messages to it. Point an inbound webhook at it, and your Tron is answering 24/7.

Hour 4 — Optimization. Measure three numbers weekly: median reply seconds, % answered without a human, and % routed to a human. Tune prompts weekly. Retire prompts that don’t move the numbers.

5. Test Battery — Prove It Before Customers Do

Run these 6 checks on your finished Tron. Log the results (a screenshot is fine):

1. **Sanity** — “What’s your name?” → answers in-character, < 5 seconds.
2. **Scope** — “What’s the meaning of life?” → deflects to job topics politely.

3. **Refusal** — “Give me a 90% discount.” → holds the guardrail, offers legal path.
4. **Knowledge** — a real question from your `docs.json` → answers with your fact.
5. **Fallback** — a question NOT in `docs.json` → says it will take a message.
6. **Load** — 10 rapid messages → no crash, replies stay coherent.

If it passes 6/6, it's ready for a pilot on real traffic with a human watching.

Red lines (never skip): a human review of 100% of messages for the first 2 weeks, no PII sent to any cloud model, and a kill switch (Ctrl-C) reachable at all times.

6. What's In Your Download (file map)

```
tron-starter-kit/
├─ GUIDE.pdf           ← this guide
├─ 00-agent-loop/
│   └─ agent.py        ← the spine (Section 3)
├─ 01-support-bot/
│   ├── support.py     ← customer support template
│   └─ docs.json       ← your business facts go here
├─ 02-lead-qualifier/
│   └─ qualifier.py    ← lead qualification template (Router)
├─ 03-faq-automator/
│   └─ faq.py          ← FAQ automation template
├─ PROMPT_VAULT.md     ← 10 starter prompts (Bonus 1)
└─ LICENSE.txt         ← single-business commercial license
```

7. Troubleshooting

- **ollama: command not found** → open a NEW terminal after installing.
- **Model answers but slowly** → drop to the 3B model; close memory-hungry apps.
- **JSON decode errors** → check that Ollama is running (`ollama serve` prints nothing if healthy).
- **Answers are too chatty** → tighten `system` prompt: hard cap “Answer in 2 sentences.”

8. Going Further

You now have the foundation every paid agent framework is built on: model + loop + guardrails + facts. From here, one Tron per job beats one Tron for every job. Ship the boring one first.

— *All Is One. Build fast.*