GSDC

Global Skill Development Council

FREE TOOLKIT PACK · 2026 EDITION

The Agentic AI Developer Toolkit Pack

A free bundle: a starter boilerplate & template checklist, recommended ai agent development tools, and a curated list of code examples & GitHub repositories — everything to start building, in one place.

What's inside

- A starter agent boilerplate + project-structure checklist
- Recommended tools across models, memory, orchestration & eval
- Framework picker + debugging-tools checklist
- A copy-ready ai agent code-examples index
- Curated GitHub repositories to learn from
- The 34 Learn-by-Doing builds, categorised

Published by the Global Skill Development Council (GSDC) — a vendor-neutral certification body trusted by 2,50,000+ professionals across 100+ countries. Tool and repository names are current at publication; APIs and repos evolve.

00 · ORIENTATION

How to use this pack

This is a working toolkit, not a reading list. The fastest path:

- ✔ **Clone the boilerplate structure** (next pages) into a new repo
- ✔ **Pick your tools** from the recommended stack
- ✔ **Choose a framework** with the picker
- ✔ **Copy a code example** and adapt it to your task
- ✔ **Work the 34 builds** to fill out a portfolio

Everything here maps to the certification’s hands-on builds — so the toolkit and the program reinforce each other.

01 · STARTER BOILERPLATE

Project structure

A clean layout that scales from one agent to a multi-agent app.

```
agent-app/
  app/
    agents/        # agent definitions
    tools/         # tool functions (typed, documented)
    memory/        # vector store + state
    prompts/       # prompt templates
    eval/          # eval sets + scoring
    api.py         # FastAPI entrypoint
  tests/           # unit + conversation tests
  .env             # API keys (never commit)
  .gitignore
  Dockerfile
  requirements.txt
  README.md
```

Keep tools small and testable; keep prompts in files, not buried in code.

01 · STARTER BOILERPLATE

Template checklist

- ✔ **Pinned dependencies** in requirements.txt (or a lockfile)
- ✔ **.env + .gitignore** — keys out of source control from day one
- ✔ **Config loader** — one place for model, keys and settings
- ✔ **Structured logging** of prompts, tool calls and decisions
- ✔ **Error handling** — timeouts and retries around model/tool calls
- ✔ **Eval folder** — a small scored test set from the start
- ✔ **Dockerfile** — reproducible runs anywhere
- ✔ **README + demo** — so the repo is portfolio-ready

02 · RECOMMENDED TOOLS

Models & frameworks

LLM backends

A hosted model (GPT-4o-class, Claude, Gemini) for quality; a local/open model (Llama-class) for cost and privacy. Abstract the provider so you can switch.

Agent frameworks

LangChain & LangGraph for chains and stateful graphs; AutoGen for multi-agent conversations; CrewAI for role-based crews; Semantic Kernel for plugin/planner workflows.

Data & RAG

LlamaIndex for data-heavy retrieval; an embeddings model for vectors.

02 · RECOMMENDED TOOLS

Memory, orchestration & eval

Vector stores

Chroma for local dev; Pinecone for managed scale; Redis when you already run it.

Orchestration & serving

FastAPI to expose agents; Docker to package; a queue (Redis/Celery) for async work.

Observability & eval

Tracing (LangSmith or OpenTelemetry) to see every step, token and cost; an eval set to measure quality; pytest for tools.

Reliability

Retry/back-off (e.g. tenacity), rate limiting and circuit breakers around every external call.

03 · FRAMEWORK PICKER

Pick the right framework

If you need…	Reach for
Quick RAG + tool-calling agents	LangChain
Explicit control: loops, retries, human-in-the-loop	LangGraph
Agents that converse to solve a task	AutoGen
Clear roles & tasks, low boilerplate	CrewAI
Plugin/planner workflows, MS / .NET	Semantic Kernel
Data-heavy retrieval & indexing	LlamaIndex

Prototype with the simplest fit, then graduate to an explicit graph as control and reliability needs grow.

03 · DEBUGGING

Debugging-tools checklist

- ✔ **Tracing** (LangSmith / OpenTelemetry) — inspect each step, tool call and token
- ✔ **Structured logs** of the decision path, not just final output
- ✔ **Token & cost counters** per run to catch runaway loops
- ✔ **pytest** for tools in isolation; scripted conversation tests
- ✔ **Deterministic replay** from saved logs to reproduce failures
- ✔ **Output validators / guardrails** to catch malformed or unsafe results
- ✔ **A small eval set** so every change is measurable, not vibes

04 · GITHUB REPOSITORIES

Official framework repos

github.com/langchain-ai/langchain
Chains, tools, RAG and prebuilt agents

github.com/langchain-ai/langgraph
Stateful, cyclic agent graphs

github.com/microsoft/autogen
Multi-agent, conversable agents with code execution

github.com/crewAIInc/crewAI
Role-based agent crews and tasks

github.com/microsoft/semantic-kernel
Skills, plugins and planners (Python / .NET)

Read the examples folders first — they are the fastest way to learn each framework's idioms.

04 · GITHUB REPOSITORIES

Ecosystem & learning repos

github.com/openai/openai-cookbook
Copy-ready API recipes, including tool calling

github.com/run-llama/llama_index
Data framework for RAG and indexing

github.com/chroma-core/chroma
Embedded vector store for local development

github.com/langfuse/langfuse
Open-source LLM observability and tracing

github.com/microsoft/generative-ai-for-beginners
Structured beginner course with runnable notebooks

Repository paths are current at publication; if one has moved, search GitHub for the project name.

05 · CODE EXAMPLES INDEX

Copy-ready examples — part 1

Function calling: let the model choose a tool.

```
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI

@tool
def get_weather(city: str) -> str:
    "Return current weather for a city."
    return fetch_weather(city)

llm = ChatOpenAI(model="gpt-4o").bind_tools([get_weather])
print(llm.invoke("Weather in Pune?").tool_calls)
```

ReAct agent: reason → act → observe.

```
from langgraph.prebuilt import create_react_agent
agent = create_react_agent(model=llm, tools=[search, calc])
print(agent.invoke({"messages": [("user", "Find X, compute Y")]}))
```

05 · CODE EXAMPLES INDEX

Copy-ready examples — part 2

Minimal RAG: retrieve, then answer with context.

```
docs = retriever.invoke(question)
context = "\n\n".join(d.page_content for d in docs)
prompt = f"Use the context to answer.\n{context}\n\nQ: {question}"
print(llm.invoke(prompt).content)
```

Role-based crew (CrewAI).

```
from crewai import Agent, Task, Crew
r = Agent(role="Researcher", goal="Find facts")
w = Agent(role="Writer", goal="Draft copy")
crew = Crew(agents=[r, w],
            tasks=[Task(description="Research X", agent=r),
                   Task(description="Write X", agent=w)])
print(crew.kickoff())
```

06 · THE 34 BUILDS

Track 1 — Creating AI Agents (9)

1 Introduction to Building Agents with Microsoft Foundry

2 Building Agentic AI and Autonomous Systems with Hands-on Use Cases

3 Transforming Leadership and Development with AI Co-Pilots

4 Implementing Agents for Microsoft 365: From Traditional Software to Autonomous AI Systems

5 AI Agents: Components, Tools and Practical Implementation

6 Agentic DevOps in the Autonomous Cloud: Architecting Application Modernization

7 AI That Works Together: Designing Multi-Agent Systems for the 2026 Innovation Era

8 Getting Started with Zapier AI Agents

9 Simulating Software Development Teams with Coordinated AI Agent Roles

06 · THE 34 BUILDS

Track 2 — Prompt Engineering (16)

1 Prompt Engineering: From Fundamentals to Advanced Techniques

2 Safe and Ethical AI: Context, Common Scenarios, and Guidance

3 Prompt Engineering and AI Agents: Evolution, Importance, and Security

4 Create Interactive Learning Quizzes with AI in Minutes

5 My Pen.ai Complete Platform Demo and Feature Overview

6 Transform Teams Meetings into Clear Summaries and Action Items with Copilot

7 AI for Technologists: Perplexity, LangChain and Knowledge Graphs

8 How to Create High Quality Documents Instantly

9 Building Data-Aware AI Agents with Prompt Engineering for Smarter Decisions

10 Prompt Engineering: Unlocking the Next Level of Productivity with Your Words

11 Gamma App: Transform Presentations with AI-Powered Creativity

12 Creating Professional Podcasts with AI: From Script to Polished Audio

13 Mastering ChatGPT and Unlocking the Full Potential

14 AI in the Workplace: Document Automation and Financial Data Capture

15 Mastering Generative AI: From ChatGPT to Real-World Application

16 Prompt Engineering for LLM-Infused Applications

06 · THE 34 BUILDS

Track 3 — Enterprise AI Copilots (9)

1 Creating AI Agents using Microsoft Copilot and SharePoint for Enterprise Search
2 Data Governance as the Bedrock for AI: A Practical Enterprise Approach
3 AI Agent for Automated Test Case Generation
4 AI Agents for Data Privacy and PII Detection
5 AI Copilot for Developers: Enterprise Coding
6 Building Enterprise Chat Solutions with Microsoft Copilot Studio and Azure OpenAI
7 Autonomous Systems and AI Agents for Proactive Monitoring and Incident Resolution
8 CRM Data Quality Enhancement: AI Agent Architecture for Enterprise Scale
9 AI Agent for Architecture Review

07 · PRODUCTION

Deployment toolchain

- ✔ **Package** with Docker; orchestrate with Kubernetes if you need scale
- ✔ **Serve** via FastAPI behind Uvicorn/Gunicorn
- ✔ **Protect calls** with rate limits, retries and circuit breakers
- ✔ **Async work** through a queue (Redis / Celery)
- ✔ **Secrets** in a vault or managed env, never in code
- ✔ **Monitor** latency, cost and errors (Prometheus / Grafana, Sentry)
- ✔ **Ship safely** with CI/CD and staged rollouts

08 · SAFETY & GOVERNANCE

Security & governance toolkit

- ✔ **Prompt-injection defence** — treat tool output & web content as untrusted
- ✔ **Tool allow-lists & RBAC** for sensitive actions
- ✔ **PII detection & redaction** in inputs and logs
- ✔ **Audit logging & traceability** for every action an agent takes
- ✔ **Human-in-the-loop** approval for irreversible steps

Related roles on the rise: AI Safety / Guardrails Engineer, AI Governance Specialist, Responsible AI Engineer, AI Compliance / Risk Analyst.

09 · CAREER & PAY

What this toolkit is worth

Employers increasingly screen for exactly these tools and builds — fluency is both a hiring filter and a salary lever.

$147K	$115–191K	$239K
median agentic AI pay (US, Glassdoor 2026)	typical range	top earners

2026 hiring trends

- ✔ Production & deployment skills push offers up a band
- ✔ Verifiable builds and repos beat buzzwords
- ✔ Governance and safety skills are increasingly paid for

10 · PUT IT TO WORK

Turn the pack into a portfolio

Combine the pieces in this toolkit into three demoable projects:

- ✔ **Project 1 — a single tool-using agent** on the boilerplate, with tests
- ✔ **Project 2 — a RAG or multi-agent app** using a framework from the picker
- ✔ **Project 3 — a deployed agent** with tracing, guardrails and a live URL

Document each with a README and a short demo, name the frameworks explicitly, and pin them to the top of your GitHub. That is the portfolio recruiters actually open.

YOUR NEXT STEP

The toolkit + the credential

This pack gives you the tools and the templates; the certification gives you the reps and the proof — all 34 builds above, two capstones, and a recognised credential.

- ✔ 12 modules · 34 hands-on builds · capstones
- ✔ Daily live sessions & 1-on-1 mentoring
- ✔ Job support · globally recognised · 7-day money-back guarantee

Offer is time-limited. Visit the program page for the current intake.

48 HOURS LEFT **Get the toolkit and the credential — Claim your 50% offer now →**