Lesson 12 · Google AI Development Kit (ADK)

PDF: this lesson · **Install (Linux · macOS · Windows):** guide · PDF

> **Part of local-ai-lab** - a hands-on course for building local AI.

> **Course home:** https://nikolareljin.github.io/local-ai-lab/ **Source:** https://github.com/nikolareljin/local-ai-lab

> **Lessons:** 1 · RAG → 2 · MCP → 3 · Hybrid retrieval → 4 · RAG safety → 5 · RAG evaluation → 6 · Repo assistant → 7 · LangChain → 8 · LangGraph → 9 · Ollama tools → 10 · Semantic Kernel → 11 · Bedrock Agents → **12 · Google ADK (you are here)**

> **Status: planned.** Outline below; full published slideshow lesson + step-by-step coming later. ADK **runs locally** (`adk run` / `adk web`) against Gemini. Star the repo to follow along.

The idea

Google's Agent Development Kit (ADK) is an open-source framework for building, evaluating, and deploying agents, designed around **Gemini** but model-agnostic. In this final lesson you rebuild the document agent with ADK and run it locally - closing the loop on "the same agent, six different ways."

What you'll learn

- **ADK agents** - defining an `Agent` with instructions, a model, and tools
- **Tools** - exposing a Python function (your `search_docs`) as an ADK tool
- **Sessions & runners** - running the agent locally with `adk run` (CLI) or `adk web` (dev UI)
- **Gemini integration** - using `gemini-2.5-flash`, plus how to point ADK at other models
- **Evaluation** - ADK's built-in eval harness for checking agent behavior

The design we'll build

```
from google.adk.agents import Agent
from localrag.config import load_config
from localrag.engine import get_retriever

def search_docs(query: str) -> str:
   """Search the user's local documents and return cited passages."""
   hits = get_retriever(load_config()).search(query, 5)
   return "\n\n".join(f"[{h['source']}:{h['page_number']}] {h['text']}" for h in hits)

root_agent = Agent(
   name="docs_agent",
   model="gemini-2.5-flash",
   instruction="Answer from search_docs results and cite [source:page]. "
               "If it's not in the documents, say so.",
   tools=[search_docs],
)
# Run locally:  adk run .   (or)   adk web
```

> **The finale:** the `search_docs` function is, once again, Lesson 1's retriever. Across MCP, Ollama, Semantic Kernel, Bedrock, and ADK, the **capability** never changed - only the framework wrapping it. That's the whole point of the course: master the primitives, and every framework is just syntax.

Builds on

Concept	From
`search_docs` retriever tool	Lesson 1
function/tool calling	Lesson 9
agent frameworks compared	Lessons 7, 8, 10
ADK agents, tools, runners, eval	**Lesson 12 (this one)**

Prerequisites

Python, a Gemini API key (`GEMINI_API_KEY`), and `pip install google-adk`. Lesson 1 provides the retriever the agent calls.

Course complete

You've built the same local-first document agent six ways - from a hand-rolled RAG pipeline up through MCP, local function calling, Semantic Kernel, Bedrock Agents, and ADK - and you understand every layer. Back to the course home.
