Ollama Project Structure

The easiest way to get up and running with large language models locally. A Go application that wraps and manages the C++ inference engine.

Updated 2025-12-30

#ollama #go #cpp #llm #ai #inference #llama.cpp

PNG PDF Copy Prompt

Project Directory

```
ollama/
├── llama/  Inference Engine
│   ├── llama.cpp/  Upstream C++ co…
│   ├── patches/  Custom modifica…
│   └── llama.go  Go CGO bindings
├── server/  API Server
│   ├── server.go  Gin router setup
│   └── routes.go  API endpoints
├── api/  Go Client Libra…
│   └── client.go
├── ml/  Machine Learnin…
│   └── backend/ggml/  GGML backend
├── cmd/  CLI Entry Point
│   └── cmd.go
├── template/  Chat Templates
├── main.go  Application ent…
└── go.mod  Go dependencies
```

Repository Info

Repository - ollama/ollama

Stars - 55k+

License - MIT

Last Analyzed - December 2025

Tech Stack

Language - Go

Inference - C++ (llama.cpp)

Web Framework - Gin

Distribution - Static Binary

Architecture Notes

Ollama is a Go wrapper around the `llama.cpp` library. It uses CGO to call into the C++ code for model inference. The Go layer handles the API server (using Gin), model management (pulling from registry, verifying hashes), and the CLI interface. It essentially turns raw model weights into a usable REST API.

Key Directories

llama/ - Contains the C++ code for running LLMs. It embeds `llama.cpp` and applies custom patches to support specific hardware or features.

server/ - The HTTP server implementation. It accepts JSON requests from clients and translates them into calls to the inference engine.

ml/ - Abstracts the machine learning backend details, allowing Ollama to potentially support other backends in the future.

Why This Structure?

Ollama is the standard for local LLM inference. Its architecture prioritizes ease of use: a single binary that handles everything from downloading models to running them on your GPU.