RAG_System 代码目录结构（完整版）

说明：本页为可直接保存的 HTML 概览，包含完整目录树与每个文件的职责注释。你可将其另存为 `.html` 再转存为 Word。

目录树（不省略）

```
app/
├─ core/
│  ├─ dto.py                          # DTO 数据模型 (GraphBuildIn/Out,
RetrievalIn/Out, ReasoningIn/Out, VerifyIn/Out, Hit)
│  ├─ interfaces.py                   # 四大 Agent 接口协议
│  ├─ llm_router.py                   # LLMRouter：封装 complete/embed，路由到
provider
│  ├─ dataset_loader.py               # 数据集加载器 (HotpotQA JSON/JSONL，支持
index/count)
│  └─ providers/                      # ✅ LLM Provider 子目录
│     ├─ base.py                      # Provider 抽象类
(LLMProvider.complete/embed)
│     ├─ openai_provider.py           # OpenAI Provider (支持 from_settings
/ Mock 回退)
│     └─ ollama_provider.py           # Ollama Provider (本地 API，失败回退
Mock)
├─ adapters/
│  └─ graph_request_adapter.py        # Graph 请求适配器 (HotpotQA v1 →
GraphRequestV2)
├─ modules/
│  ├─ graph_construction/
│  │  ├─ flow.py                      # GraphConstructionFlow：Ingest →
BuildNodes → BuildEdges → AssembleSave → Summarize
│  │  ├─ node_builder.py              # NodeBuilder：构造 Question / Sentence
/ Document / Entity 节点
│  │  ├─ edge_builder.py              # EdgeBuilder：构造 next_in_doc /
q_match / in_doc / mentions / semantic_sim 边
│  │  └─ impl_networkx.py             # GraphConstructionNetworkX：networkx
组装图并落盘 (graph.json/gexf/manifest)
│  ├─ retrieval/
│  │  ├─ flow.py                      # RetrievalAgentFlow：Expand(LLM) →
BM25 → GraphExpand → RankSelect
│  │  ├─ text_index.py                # BM25LiteIndex：轻量 BM25 (docs.jsonl)
│  │  ├─ graph_utils.py               # Graph JSON 工具：build_index /
expand_qmatch_neighbors
│  │  ├─ retrieval_backend.py         # HybridRetrievalBackend：LLM
QueryExpand + BM25 + GraphNeighbor 融合
│  │  └─ retrieval_adapter.py         # RetrievalAdapter：统一输出 Hit(id,
score, meta)
│  ├─ reasoning/
│  │  ├─ flow.py                      # ReasoningAgentFlow：LangGraph (Plan
→ Synthesize)
```

```
│   │   └─ impl_planner_synth.py       # 简化 ReasoningAgent: plan + synth 两
次 LLM
│   └─ verification/
│       ├─ flow.py                     # VerifierAgentFlow: 规则检查 + LLM
consistency → 聚合
│       └─ impl_rules_llm.py           # VerifierAgentRulesLLM: LLM-only fact
check
├─ orchestrator/
│   ├─ nodes.py                        # NodeContext + 各节点
(Ingest/BuildGraph/ChooseRoute/Retrieval/Reasoning/Verify/PackResult)
│   ├─ state.py                        # WFState: 工作流全局状态 (TypedDict)
│   └─ workflow.py                     # build_workflow: 组装 LangGraph DAG
(含条件分支)
├─ schemas/
│   └─ graph_request_v2.py             # AssembleGraphRequestV2: Sentence /
Inputs / Provenance
├─ telemetry/
│   └─ sinks.py                        # TelemetrySink & LocalJsonlSink
(events.jsonl / run.json / flow.mmd)
├─ di/
│   └─ factory.py                      # ✅ 依赖注入工厂: 构建 providers /
router / modules
└─ system.py                           # 系统入口: init_system +
answer_question

my_code/
├─ ingest_hotpotqa.py                  # HotpotQA → Graph + docs.jsonl 的数据
预处理脚本
└─ run_system.py                       # CLI: 批量运行系统 (读取
config/settings.yaml 数据集配置)

config/
└─ settings.yaml                       # 系统配置 (providers, modules, dataset
等)

runs/                                  # Telemetry 输出 (自动生成)
└─ <trace_id>/
   ├─ events.jsonl                     # 节点执行事件
   ├─ run.json                         # 最终结果快照
   └─ assets/
      └─ flow.mmd                      # Mermaid 执行轨迹图

data/
├─ hotpotqa/
│   ├─ hotpot_dev_distractor_v1.json   # HotpotQA 原始数据
│   └─ docs.jsonl                       # ingest_hotpotqa 生成的句子索引
└─ graph/
   └─ <graph_id>/
      ├─ graph.json                    # 图 JSON
      ├─ graph.gexf                    # networkx GEXF
      └─ manifest.json                 # 图文件清单 (含路径信息)
```

说明 以上结构已按你的要求落位：`graph_request_adapter.py` → **app/adapters/**， Provider 文件 → **app/core/providers/**，`factory.py` → **app/di/**。

关键文件与职责

- `app/core/dto.py`：全系统 DTO（Graph/Retrieval/Reasoning/Verify 输入输出、`Hit` 等）。
- `app/core/interfaces.py`：`GraphConstruction` / `RetrievalAgent` / `ReasoningAgent` / `VerifierAgent` 接口。
- `app/core/llm_router.py`：统一的 `complete` / `embed` 路由，埋点可观测。
- `app/core/providers/`：Provider 抽象与具体实现（OpenAI / Ollama）。
- `app/adapters/graph_request_adapter.py`：HotpotQA v1 → GraphRequestV2 适配。
- `app/modules/graph_construction/*`：节点/边构建 + NetworkX 落盘。
- `app/modules/retrieval/*`：BM25Lite + 图邻域扩展 + 后端融合 + 适配层。
- `app/modules/reasoning/*`：两步式推理（Plan → Synthesize）。
- `app/modules/verification/*`：规则 + LLM 一致性检查。
- `app/orchestrator/*`：LangGraph 编排（条件路由、节点执行、结果打包）。
- `app/telemetry/sinks.py`：本地 JSONL 事件、Mermaid 轨迹与延迟指标。
- `app/di/factory.py`：从 `config/settings.yaml` 装配 providers/router/modules。
- `app/system.py`：系统入口：初始化、运行工作流、记录运行与产物。
- `my_code/ingest_hotpotqa.py`：将 HotpotQA 转存为 Graph + docs.jsonl。
- `my_code/run_system.py`：批量执行并保存结果。

使用提示

- 迁移后的 import 路径示例：
    - 原：`from app.core.graph_request_adapter import ...`
    - 现：`from app.adapters.graph_request_adapter import ...`
    - 原：`from app.core.base import LLMProvider`
    - 现：`from app.core.providers.base import LLMProvider`
- 运行入口：`my_code/run_system.py`（读取 `config/settings.yaml`）。
- 运行产物输出到 `runs/<trace_id>/`（事件 JSONL、结果快照、Mermaid 图）。