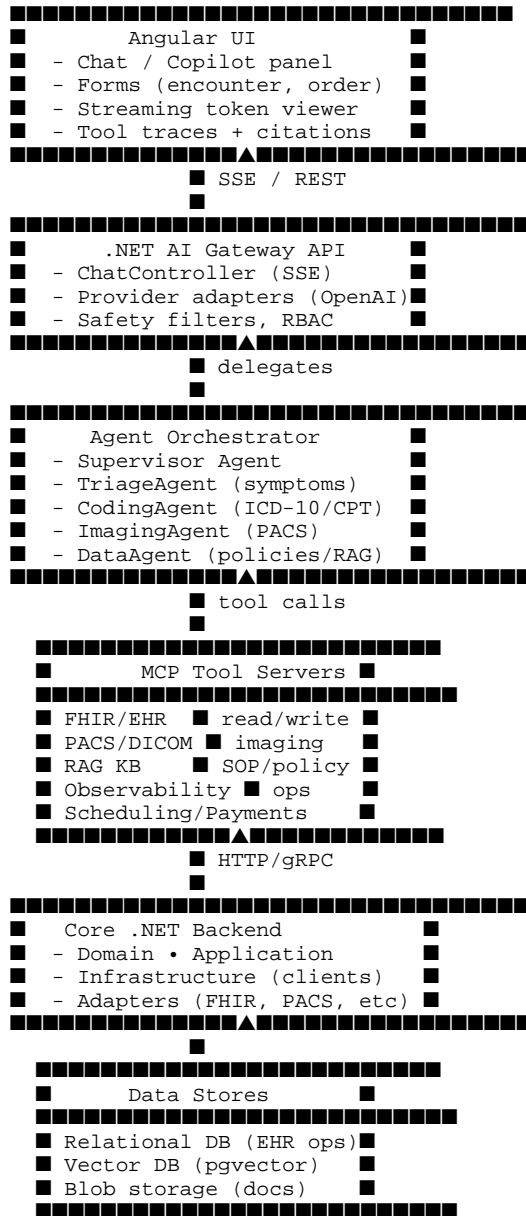# Smart Healthcare Assistant — Full-Stack Architecture

This document outlines the architecture of the Smart Healthcare Assistant, leveraging .NET + Angular with AI (LLM → MCP → Agents). It is designed for use in README files, project documentation, or as a shareable artifact for platforms like LinkedIn.

```
Angular UI
 - Chat / Copilot panel
 - Forms (encounter, order)
 - Streaming token viewer
 - Tool traces + citations
              ▲
          SSE / REST

        .NET AI Gateway API
 - ChatController (SSE)
 - Provider adapters (OpenAI)
 - Safety filters, RBAC
              ▲
          delegates

        Agent Orchestrator
 - Supervisor Agent
 - TriageAgent (symptoms)
 - CodingAgent (ICD-10/CPT)
 - ImagingAgent (PACS)
 - DataAgent (policies/RAG)
              ▲
          tool calls

        MCP Tool Servers
 FHIR/EHR    read/write
 PACS/DICOM  imaging
 RAG KB      SOP/policy
 Observability  ops
 Scheduling/Payments
              ▲
          HTTP/gRPC

       Core .NET Backend
 - Domain • Application
 - Infrastructure (clients)
 - Adapters (FHIR, PACS, etc)
              ▲

          Data Stores
 Relational DB (EHR ops)
 Vector DB (pgvector)
 Blob storage (docs)
```

Key Business Cases

• Intelligent Triage – Symptom input, policy lookup, suggest next steps.
• Coding Copilot – FHIR read, ICD-10/CPT suggestions, optional billing draft.
• Imaging Assistant – PACS queries, dose policy checks, study links.
• Policy & SOP Q&A; – RAG knowledge base, citation-backed answers.
• Ops/Dev Copilot – Observability MCP, error summaries, on-call aid.

This architecture is built to be modular, PHI-safe, and auditable. It supports incremental rollout: start with RAG (no PHI), then FHIR read-only, and gradually add imaging, scheduling, and observability integrations.