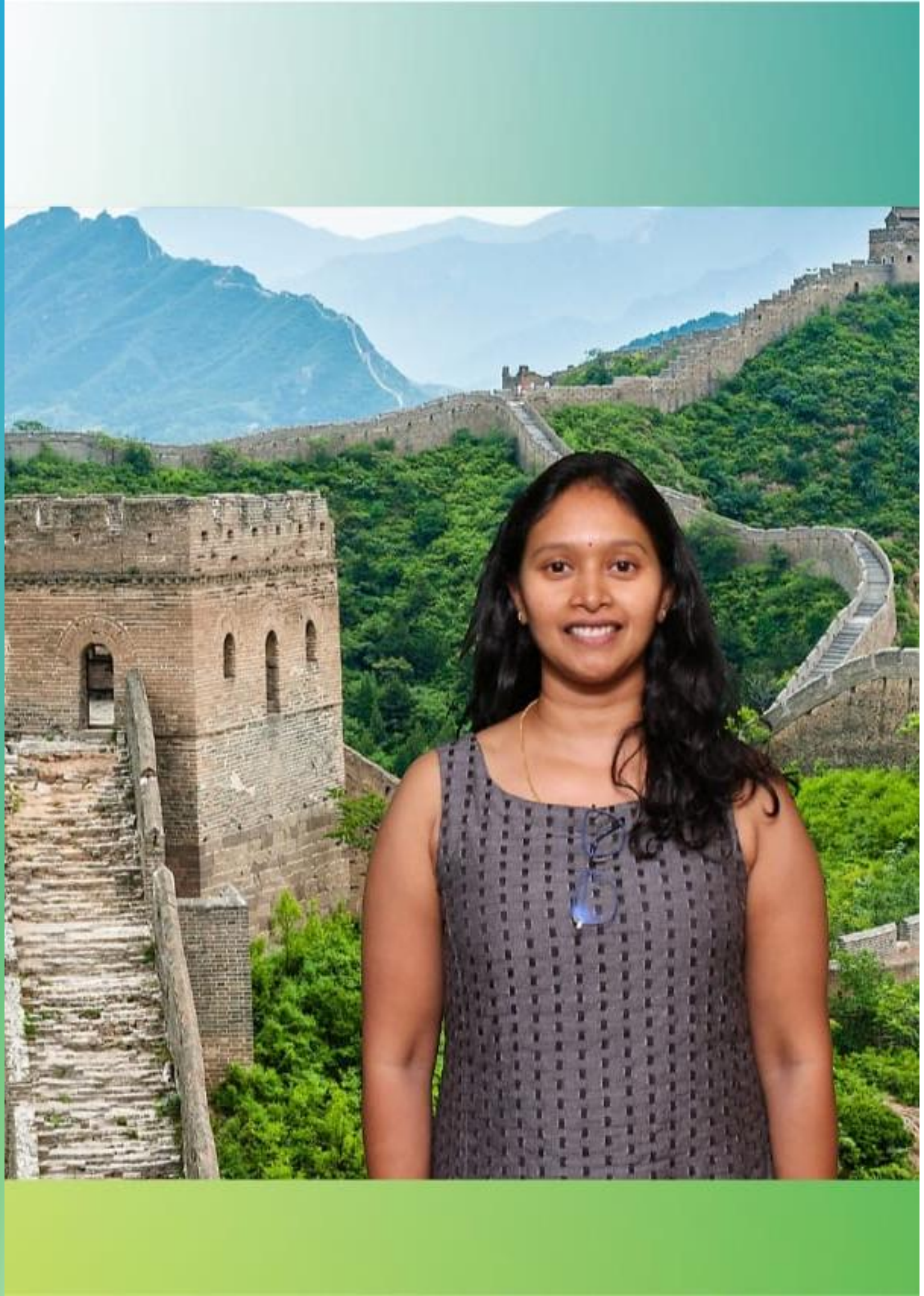
Azure AI Foundry

Introduction and Hands-on Workshop

Bhavana Konchada

Principal Software Engineer, Microsoft



- Engineer at Microsoft in the past 10 years
- Worked in Azure DevOps, GitHub, Engineering Systems, Azure Cloud Shell, Azure Impact Reporting
- Currently working in Azure Resiliency
- Worked in India and US
- Windows standalone application -> Using Cloud services -> Building Azure Services -> Making Azure Resilient
- Mom of a 12-year-old
- Bharatanatyam dancer
- Voracious reader

https://www.linkedin.com/in/bhavana-konchada/

Link to all workshop material

GH link: ai-projects/AI-Foundry-Intro-Workshop-Project at main · bhavanakonchada/ai-projects

Welcome & Agenda

Today's Journey:

- What is Azure AI Foundry?
- Core Technical Components & Architecture
- Advanced AI Features & Capabilities
- Live Demo: Smart Customer Support
- Hands-On Workshop
- Development Best Practices
- Key Takeaways
- Next Steps

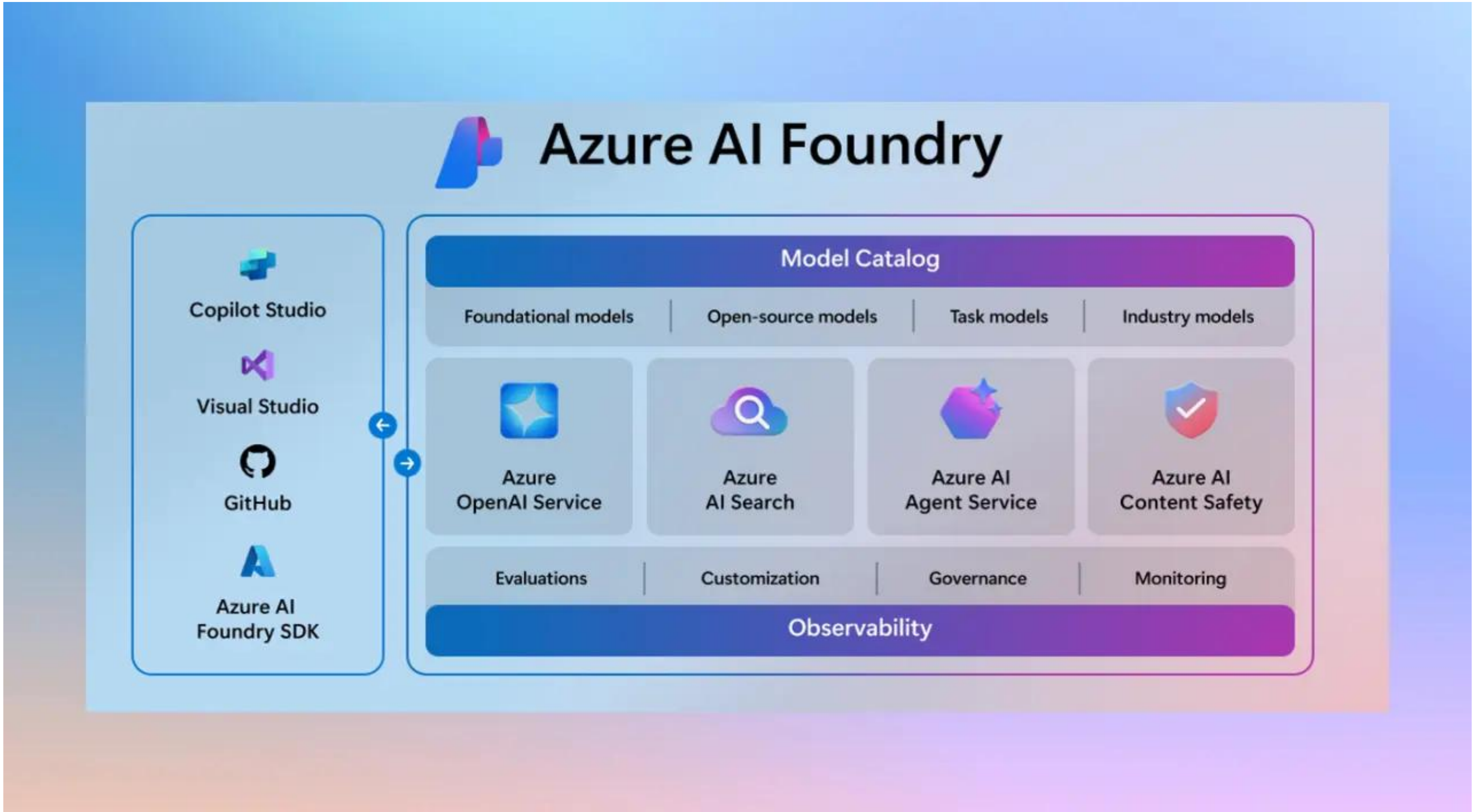
Duration: 90 minutes

An AI agent is an entity that observes its environment, reasons about the observations using algorithms, and takes actions to maximize its chances of achieving a goal.

Feature	**Traditional Program**	**AI Agent**
Works on rules	Yes	Yes + learning
Adapts over time	No	Yes
Handles uncertainty	No	Yes
Goal-driven	Often procedural	Explicitly optimized

What is Azure AI Foundry

Technical Architecture Overview



Azure AI Foundry

Copilot Studio

Visual Studio

GitHub

Azure AI Foundry SDK

Model Catalog

Foundational models | Open-source models | Task models | Industry models

Azure OpenAI Service

Azure AI Search

Azure AI Agent Service

Azure AI Content Safety

Evaluations | Customization | Governance | Monitoring

Observability

What is Azure AI Foundry

Microsoft's Unified AI Development Platform

Technical Architecture:

- **Unified AI Hub**: Single interface for all AI workloads
- **Multi-Model Support**: GPT, Claude, Llama, custom models
- **Enterprise Infrastructure**: Managed compute, storage, networking
- **Developer Tools**: SDKs, APIs, visual designers

Key Differentiators:

- **Model Agnostic**: Switch between AI models seamlessly
- **RAG-Native**: Built-in retrieval augmented generation
- **Vector-First**: Integrated embedding and similarity search
- **Production-Ready**: Auto-scaling, monitoring, deployment

Foundation Models & Capabilities

Microsoft Models

GPT-4o: 128K context, function calling, vision

GPT-4o mini: Cost-optimized, 128K context

Ada-002: Embeddings, semantic search

Partner Models

Claude 3.5 Sonnet: 200K context, advanced reasoning

Llama 3.1: 8B, 70B, 405B parameter variants

Mistral Large: European AI, multilingual

Specialized Capabilities

Vision Models: GPT-4V, Florence, CLIP

Speech Models: Whisper, Azure Speech

Code Models: GitHub Copilot, CodeT5

Embedding Models: text-embedding-ada-002, multilingual.

Model Selection Framework

Task Complexity: Simple → Complex (mini → GPT-4o)

Context Length: 4K → 200K tokens

Cost Optimization: $0.0001 → $0.03 per 1K tokens

Latency Requirements: 100ms → 5s response times

Vector Databases & RAG Integration

Built-in Knowledge Retrieval Architecture

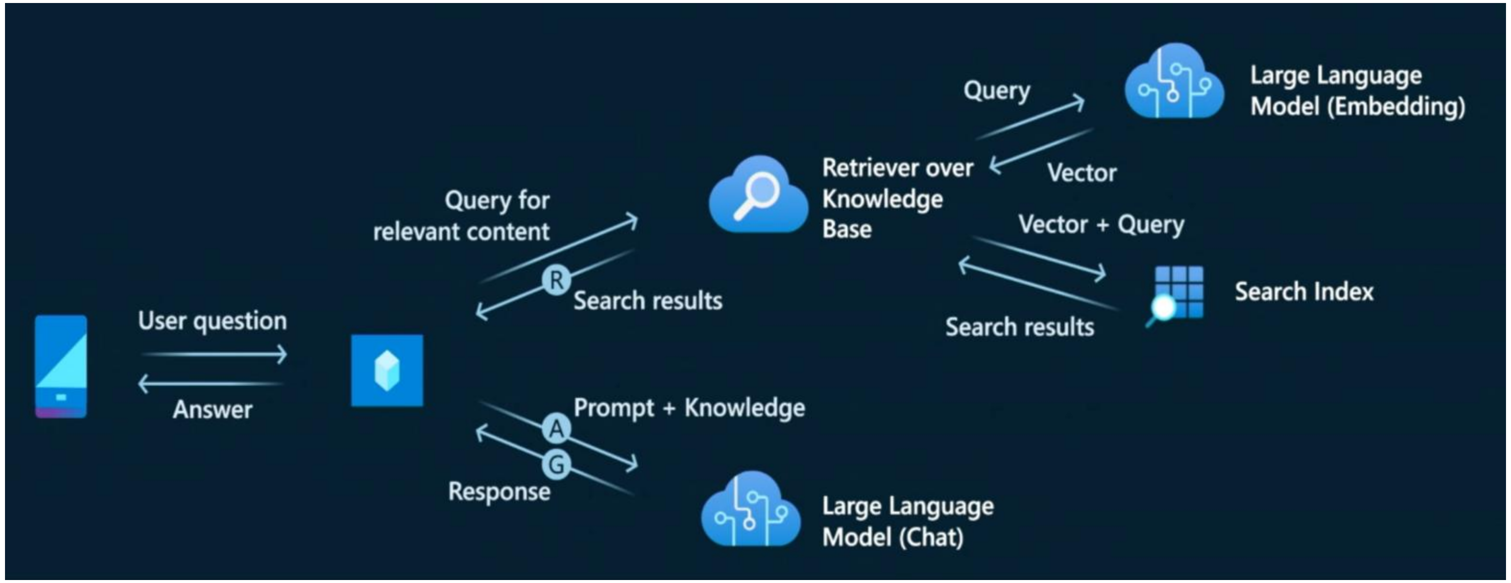
Vector Database Options:
Azure AI Search: Managed, enterprise-grade
Azure Cosmos DB: MongoDB vCore with vector search
Pinecone Integration: Specialized vector database
Custom Connectors: Weaviate, Chroma, Qdrant

Chunking Strategies:

Fixed Size: 512, 1024, 2048 tokens
Semantic Splitting: Paragraph/sentence boundaries
Hierarchical: Document → Section → Paragraph
Overlapping Windows: Maintain context continuity

Advanced RAG Features:

Hybrid Search: Keyword + semantic search
Reranking: Improve retrieval relevance
Metadata Filtering: Scope search by attributes
Citation Tracking: Source attribution


Query
Large Language Model (Embedding)
Retriever over Knowledge Base
Vector
Query for relevant content
Vector + Query
R
Search results
Search Index
Search results
User question
Answer
Prompt + Knowledge
A
G
Response
Large Language Model (Chat)

Advanced Agent Capabilities

Function Calling & Tool Integration:

- Native Functions: Database queries, API calls, calculations
- External Tools: REST APIs, Graph QL, webhooks
- Multi-Step Workflows: Chain function calls
- Parallel Execution: Simultaneous tool usage

Memory & State Management:

- Conversation Memory: Multi-turn context preservation
- Working Memory: Temporary data storage
- Long-term Memory: Persistent knowledge base
- Shared Memory: Cross-agent information sharing

Advanced Reasoning Patterns:

- Chain-of-Thought: Step-by-step problem solving
- Tree-of-Thoughts: Explore multiple solution paths
- Self-Reflection: Agent validates its own outputs
- Meta-Reasoning: Reasoning about reasoning

Code Execution Environment:

- Sandboxed Python: Safe code execution
- Data Analysis: Pandas, NumPy, visualization
- File Processing: Excel, CSV, PDF parsing
- API Integration: Real-time data access

Multi-Agent Orchestration Patterns

Sophisticated Agent Coordination

- **Communication Patterns:**
- **Direct Messaging**: Agent-to-agent communication
- **Broadcast**: One-to-many information sharing
- **Event-Driven**: Trigger-based activation
- **Hierarchical**: Manager-worker relationships

Error Handling & Resilience:

- **Retry Logic**: Automatic failure recovery
- **Fallback Agents**: Alternative execution paths
- **Circuit Breakers**: Prevent cascade failures
- **Graceful Degradation**: Partial functionality maintenance

Performance Optimization:

- **Caching**: Reuse expensive computations
- **Load Balancing**: Distribute across agent instances
- **Resource Pooling**: Efficient compute utilization
- **Streaming**: Real-time response delivery

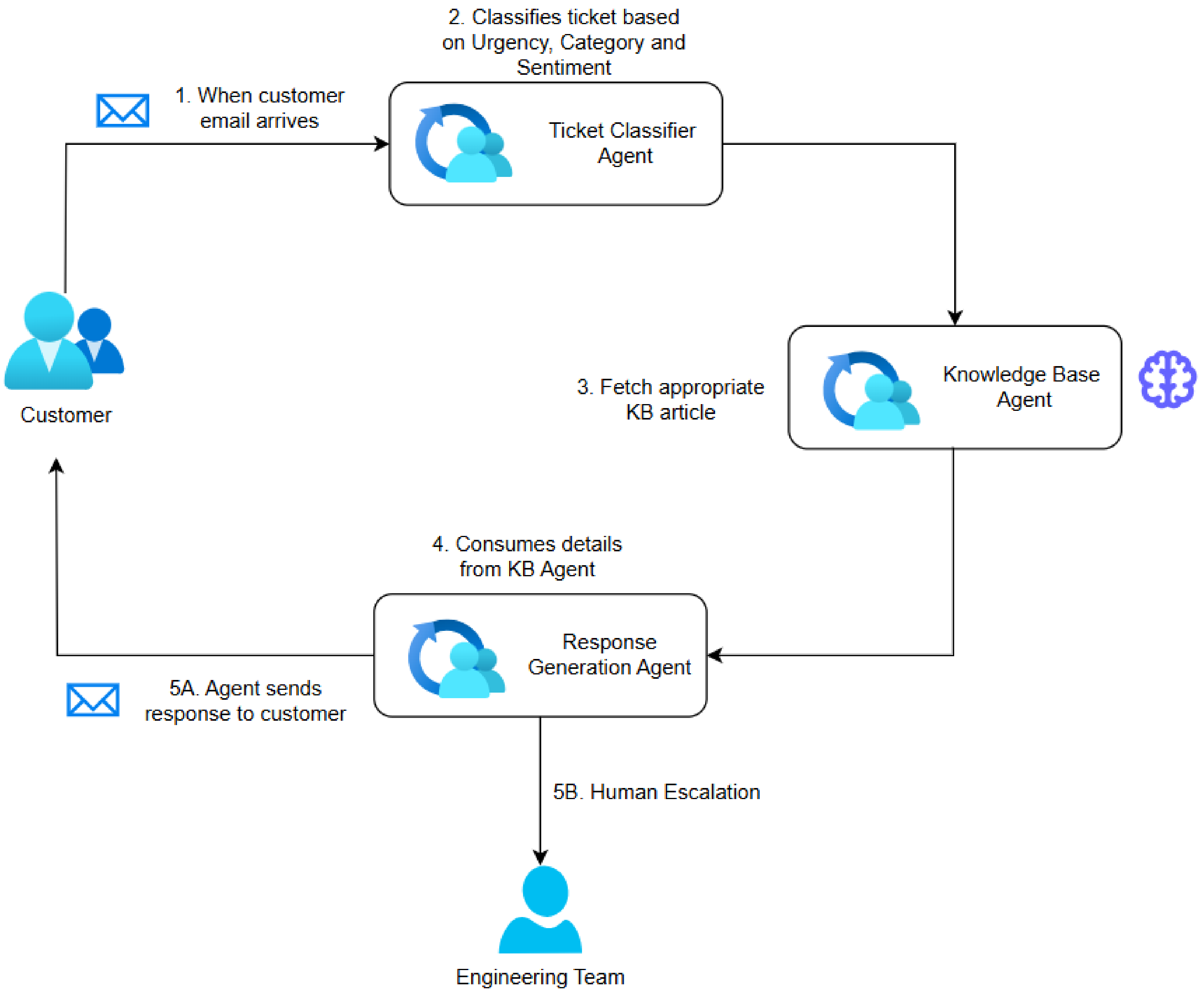
Azure AI Foundry

Tool Walkthrough – https://ai.azure.com

Demo: Multi-Agent Scenario

Scenario: Smart Customer Support using AI Agents

Real-time System Architecture


1. When customer email arrives
2. Classifies ticket based on Urgency, Category and Sentiment
Ticket Classifier Agent
Customer
3. Fetch appropriate KB article
Knowledge Base Agent
4. Consumes details from KB Agent
Response Generation Agent
5A. Agent sends response to customer
5B. Human Escalation
Engineering Team

Live Demo Technical Walkthrough

Pre-Demo Setup:

- 3 specialized agents deployed
- Vector database with 500+ KB articles
- Real customer data (anonymized)
- Monitoring dashboard active

What you’ll observe

- **Real-time Metrics**: Token usage, latency, costs
- **Decision Trees**: How agents choose their actions
- **Error Handling**: Graceful failure recovery

Interactive Elements:

- Modify prompts and see immediate impact
- Switch models and compare results
- Adjust parameters and observe changes
- Test edge cases and error scenarios

What you’ll *NOT* observe (out of scope)

- Advanced agent orchestration using Semantic Kernal or Azure AI Foundry SDK
- Fine tuning of models
- Bench marking and perf evaluation
- Multi model inferencing

Workshop: Agents Building and Orchestration

Hands-on Workshop Structure : Prerequisites and Tools

- An active Azure Subscription – Most commonly named Visual Studio Enterprise.
- Fundamental knowledge of software
- *Lots of enthusiasm and curiosity (I mean it ..)*

Hands-on Workshop Structure

Phase 1: Agent Creation (5-8 minutes)

- Create Ticket Classifier with GPT-4o mini
- Configure system prompts and parameters
- Test with sample inputs
- Validate structured outputs

Phase 2: Knowledge Integration (10-15 minutes)

- Set up vector database connection
- Upload knowledge base documents
- Configure embedding and search
- Test retrieval accuracy

Phase 3: Response Generation (5-8 minutes)

- Implement function calling
- Add content safety filters

Phase 4: Orchestration (10-12 minutes)

- Connect agents in sequence
- Test end-to-end workflow

Development Best Practices

Agent Design Principles:

- **Single Responsibility**: One clear purpose per agent
- **Stateless Design**: Minimize dependencies
- **Error Resilience**: Graceful failure handling
- **Performance Optimized**: Efficient prompt and model usage
- **Testing Strategies**: Unit tests, Integration tests, Perf tests, Quality tests
- **Monitoring & Maintenance:** Perf metrics, error analysis, model drift

Thank you !

And remember folks, building AI is like telling jokes – timing is everything, and if it doesn't work the first time, you probably need better training data!