A Deep Dive into Semantic Kernel Concepts

NAVEEN KUMAR M

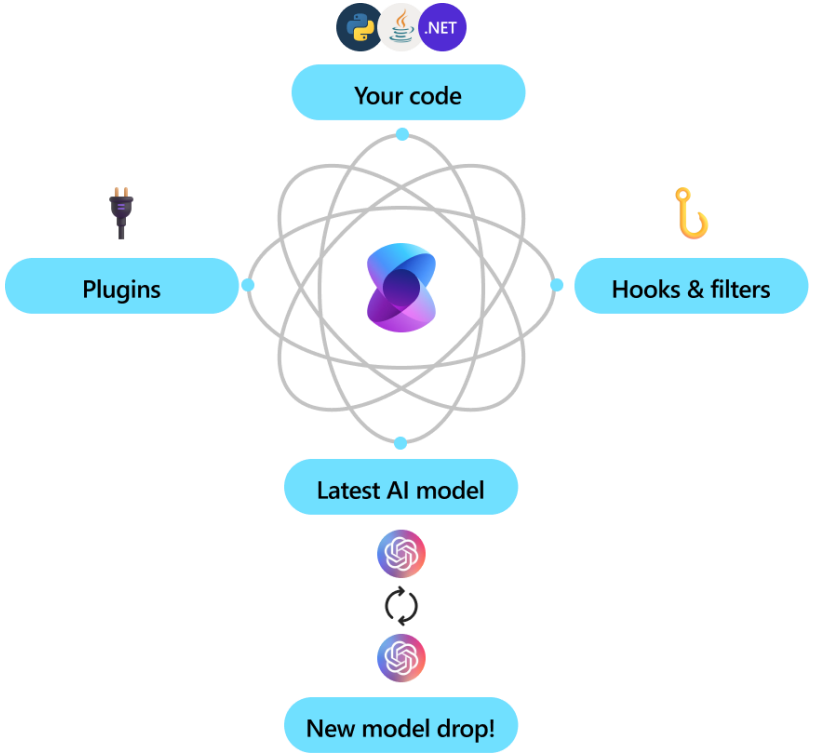
DATA ENGINEER | C# CORNER MVP | BLOGGER & SPEAKER

Agenda

- ❑ Understanding Semantic Kernel
- ❑ Core Concepts
  - ✓ What are Agent?
  - ✓ Plugin, Planning and Personas
  - ✓ What is Kernel?
  - ✓ Adding AI Services
- ❑ What makes it special?
- ❑ What it provides?
- ❑ Q & A

Understanding Sematic Kernel

- Lightweight open-source SDK.
- Integrates LLM's to create powerful AI solutions.
- C#, Python, Java.
- Built on the Copilot application.



Enterprise ready

- Already leveraged by Microsoft and Fortune 500 companies.
- It's flexible, modular, and observable.
- Security enhancing capabilities like telemetry support, and hooks and filters.

What are Agents?

- AI-driven software entities that perform tasks for you.
- Perform various tasks and are named according to their specific functions

- Semantic Kernel offers the infrastructure to create any type of agent you need, without requiring AI expertise.

Building Blocks of Agents

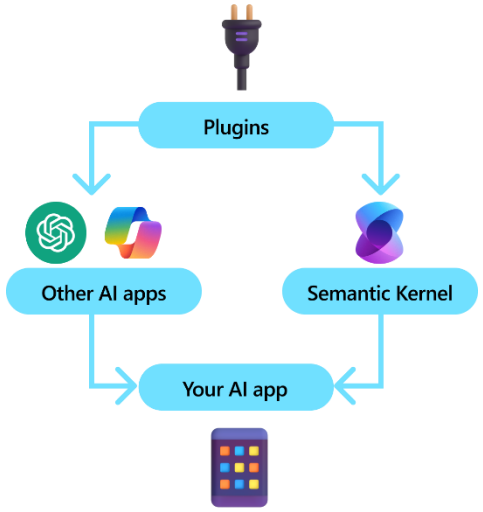
Example: A virtual assistant to schedule appointments

- ✓ Need to create a **plugin** that integrates with calendar services.
- ✓ A **planner** that generates a plan for scheduling appointments.
- ✓ A **persona** that interacts with you to gather the necessary details for setting up the appointments.

Plugins, Planner, and Personas

Plugins

- At a high level, a plugin is a collection of functions accessible to AI apps and services.
- Semantic Kernel uses function calling in modern LLMs to enable planning and API invocation.

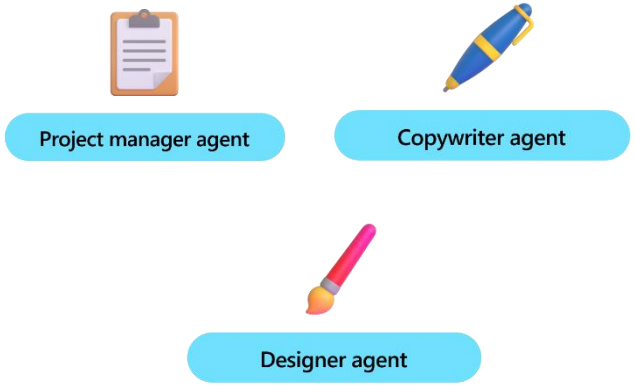


Planner

- With multiple plugins, you'll need a method for your AI agent to use them collectively to meet user needs. This is where planning comes in.

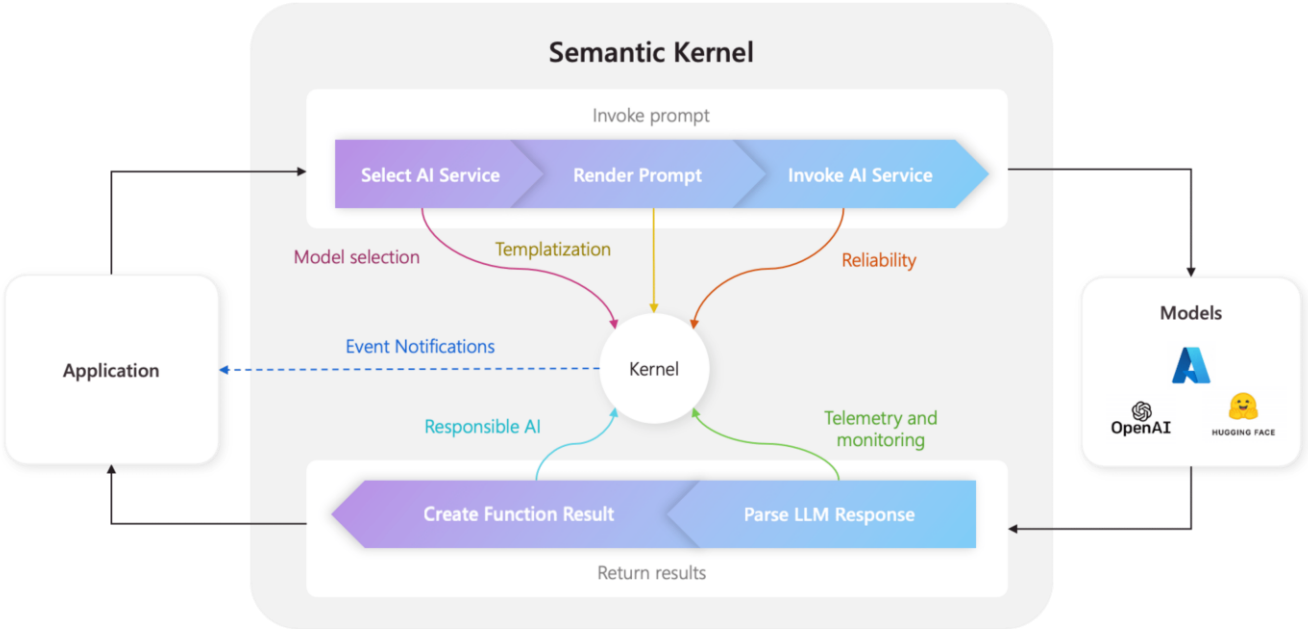
Persona

- Often referred to as a 'meta prompt' or 'instruction,' the persona is a prompt designed to shape how the agent responds to stimuli.



What is Kernel?

- It is the central component.
- The kernel acts as a Dependency Injection container, managing all the services and plugins required to operate your AI application.
- The kernel includes all the services and plugins needed for both native code and AI services.
- It is utilized by nearly every component within the Semantic Kernel SDK to drive your agents.
- As a developer have a single place where you can configure, and most importantly monitor, your AI agents



Adding AI Services to Semantic Kernel

Services	C#	Python	Java	Notes
Chat completion	✅	✅	✅	
Text generation	✅	✅	✅	
Embedding generation (Experimental)	✅	✅	✅	
Text-to-image (Experimental)	✅	❌	❌	
Image-to-text (Experimental)	✅	❌	❌	
Text-to-audio (Experimental)	✅	❌	❌	
Audio-to-text (Experimental)	✅	❌	❌	

What makes Semantic Kernel Special?

Automatic Plugin Orchestration

Semantic Kernel can automatically coordinate multiple plugins, allowing them to work together seamlessly.

Advanced Planning Capabilities

Semantic Kernel uses planners to enable LLMs (Large Language Models) to generate detailed plans based on specific user goals.

Execution of Plans

Once a plan is generated by an LLM, Semantic Kernel is responsible for executing it on behalf of the user.

Integration with AI and Native Code

The kernel manages both AI services and native code, integrating them into a cohesive system.

Unified Framework

Semantic Kernel serves as a central framework that unifies various components and services required for running AI applications.

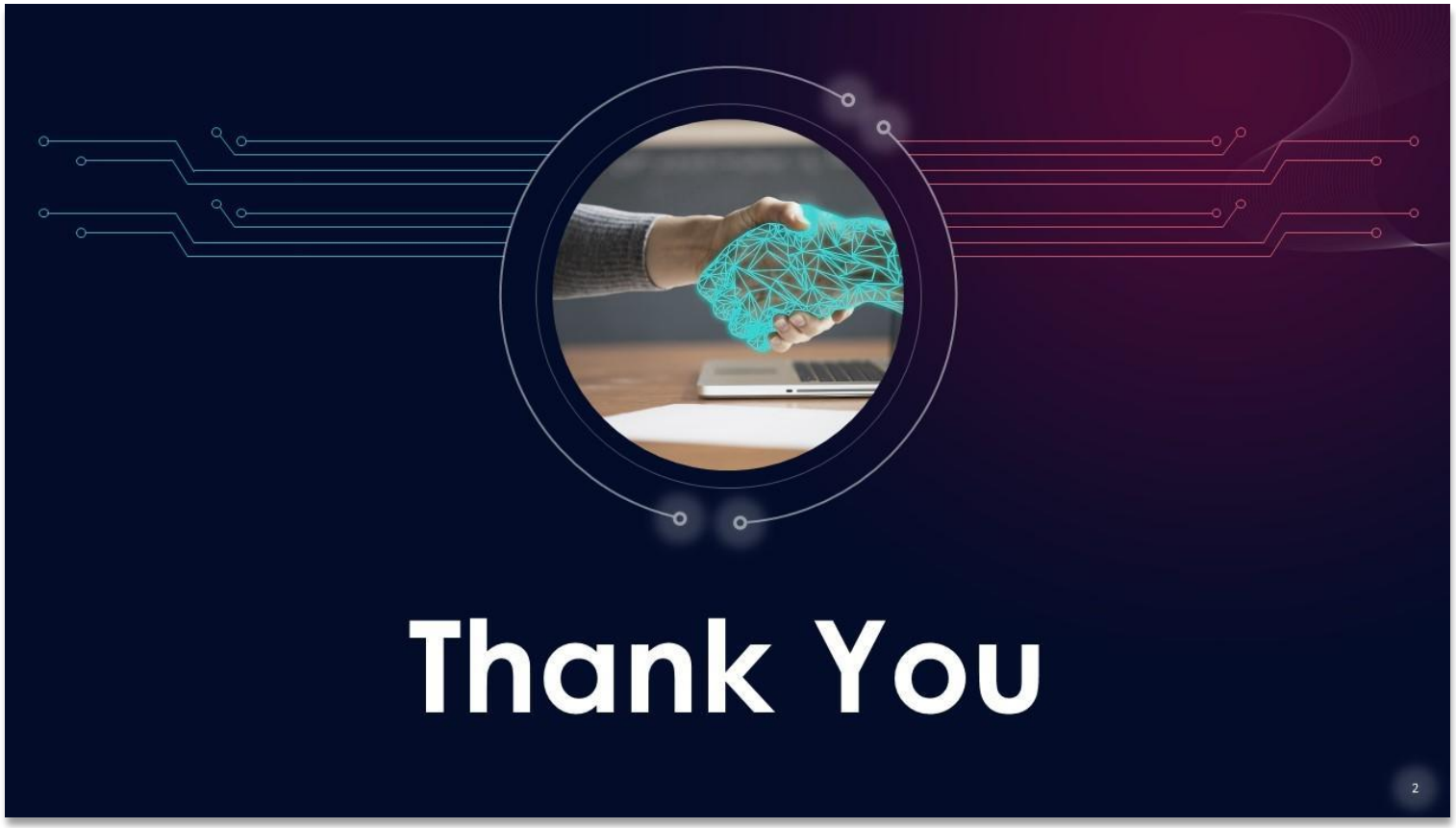
Enhanced Flexibility

By leveraging Semantic Kernel, developers can create more flexible and adaptive AI applications that can easily incorporate new plugins and services

What Semantic Kernel Provides?

- ✓ **Abstractions for AI Services:** Covers various functionalities such as chat, text-to-image, audio-to-text, and more, along with memory storage solutions.
- ✓ **Service Implementations:** Details the application of these abstractions across platforms like OpenAI, Azure OpenAI, Hugging Face, local models, and diverse vector databases including Chroma, Qdrant, Milvus, and Azure.
- ✓ **Unified Plugin Representation:** Describes a standardized format for plugins that enables automated orchestration by AI systems.
- ✓ **Plugin Creation Flexibility:** Highlights the capability to develop plugins from various sources, including OpenAPI specifications, prompts, and custom code in the target language.
- ✓ **Extensible Prompt Management:** Includes support for managing and rendering prompts with built-in handling of common formats such as Handlebars and Liquid.
- ✓ **Advanced Functionality:** Offers additional features built on these abstractions, including responsible AI filters, dependency injection integration, and more.


?
Q
&
&
A
!


Thank You
2