Bu sayfada

Maistro Docs (Public)

Maistro, **agent**, **RAG**, **automation** ve **tool entegrasyonlarını** tek bir platformda birleştirerek
AI tabanlı çözümlerin **tasarlanmasını, yönetilmesini ve güvenle üretime alınmasını** sağlar.

Amaç; tek tek çalışan yapılar yerine, **uçtan uca yönetilebilir bir AI operasyon katmanı** sunmaktır.

Karşılaşılan problemler

Bugün birçok ekip yapay zekâyı ürünlerine entegre etmeye çalışıyor.
Ancak çoğu zaman benzer problemlerle karşılaşılıyor:

- Prompt'lar farklı ortamlarda dağınık halde tutuluyor
- Agent davranışları standart bir yapıya oturmuyor
- RAG sistemleri uygulamadan kopuk çalışıyor
- Tool entegrasyonları kırılgan ve yönetilemez hale geliyor
- Üretime çıkış süreci karmaşık ve riskli oluyor

Bu problemler ilk başta küçük görünür.
Ancak sistem büyüdükçe şu sonuçlar ortaya çıkar:

- Geliştirme süresi uzar
- Aynı problemler tekrar tekrar çözülür
- Güvenlik ve yetkilendirme zorlaşır
- AI projeleri gerçek ürüne dönüşemez

Birçok ekip bu noktada AI projelerini **PoC seviyesinde bırakmak zorunda kalır**.

Maistro neyi farklı yapar?

Maistro, yapay zekâ projelerini parçalı yapılardan kurtararak
tek bir platform üzerinden yönetilebilir hale getirir.

Platform sayesinde:

- Agent'lar standart bir yapıda oluşturulur
- Konuşma ve iş akışları bağlamını kaybetmeden ilerler
- Tool entegrasyonları kontrollü ve güvenli şekilde yönetilir
- Kurumsal dokümanlar RAG ile sisteme entegre edilir
- Yetki, limit ve onay mekanizmaları merkezi olarak uygulanır
- AI çözümleri üretime hazır şekilde deploy edilebilir

Sonuç olarak Maistro:

> AI projelerini deneme ortamından çıkarıp
> **gerçek iş süreçlerinin parçası haline getirir.**

Kimler için?

Aynı platform, farklı ekiplerin aynı dili konuşmasını sağlar.

- **Ürün ekipleri**: AI destekli akışları ürüne entegre etmek isteyenler
- **Mühendisler**: Agent, RAG ve tool yapılarını sürdürülebilir biçimde yönetenler
- **Operasyon/Support**: Otomasyon ve self-service senaryolarını merkezi olarak kurgulayanlar

Platformu hızlıca anlamak için

Maistro, yapay zekâ tabanlı iş akışlarını uçtan uca tasarlamak, yönetmek ve üretime almak için geliştirilmiş bir platformdur.

Aşağıda Maistro platformunun ana ekranını görebilirsiniz. 

Tek bir arayüz üzerinden:

- Agent'lar oluşturabilir
- Konuşma ve iş akışlarını yönetebilir
- Tool'lar ile gerçek sistemlere bağlanabilir
- Kurumsal dokümanlar üzerinden bilgiye erişebilir
- AI destekli otomasyon senaryoları kurgulayabilirsiniz

Maistro, yalnızca cevap üreten bir sistem değil;
AI'ı gerçek iş süreçlerine entegre eden bir orkestrasyon katmanıdır.

Platform, bu yapıyı dört temel bileşen etrafında kurgular:

- **Agent**
  Belirli bir hedef doğrultusunda karar veren ve süreci yöneten yapı
- **Session**
  Konuşmaların ve iş akışlarının bağlamının korunduğu alan
- **Tool**
  Agent'ın dış sistemlerle güvenli şekilde etkileşime geçmesini sağlayan bileşenler
- **RAG**
  Kurumsal bilgi kaynaklarından bağlam çekerek doğru cevap üretme yaklaşımı

Bu bileşenler birlikte çalışarak uçtan uca AI deneyimi oluşturur.

Bu dokümanda neler bulacaksınız?

Bu dokümantasyon, Maistro'yu adım adım deneyimleyebilmeniz için tasarlanmıştır.

- **Hızlı Başlangıç**
  İlk agent'ınızı oluşturma, deploy etme ve konuşma başlatma adımları
- **Uçtan Uca Örnekler**
  Gerçek senaryolar üzerinden agent + tool + RAG kullanım örnekleri
- **Platform Kullanımı**
  Arayüz üzerinden agent yönetimi, konfigürasyon ve akış tasarımı
- **Temel Mimari Kavramlar**
  Agent, session, tool ve RAG mimarisinin sade açıklamaları
- **Terimler Sözlüğü**
  Platformda geçen kavramların kısa ve net tanımları

Bu yapı sayesinde Maistro'yu **bizzat deneyimleyerek öğrenirsiniz.**

Başlayalım

- İlk Agent'ını Oluştur
- İlk Konuşma (Session)
- Tool Kullanımı

Bu sayfada

1. İlk Agent'ını Oluşturmak

Bu bölümde Maistro üzerinde **ilk agent'ınızı** oluşturacağız.

Amaç; platformun temel çalışma mantığını anlamak için **en basit, çalışır ve üretime hazır bir agent'ı** birkaç dakika içinde ayağa kaldırmaktır.

Bu adımda karmaşık senaryolara girilmez. Hedef yalnızca:

- Agent oluşturmak
- Modelini belirlemek
- Gerekli minimum ayarları tanımlamak
- Agent'ı aktif hale getirmek

En küçük çalışır agent (MVP)

Bu ilk agent:

- Kullanıcıdan mesaj alır
- LLM üzerinden cevap üretir
- Gerekirse tanımlı tool'ları kullanabilir
- Session başlatmaya hazırdır

Bu yapı, ilerleyen adımlarda:

- Tool entegrasyonu
- Memory yönetimi
- Guardrails politikaları
- Workflow ve Automation Flow'lar

için temel oluşturur.

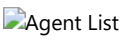
Adım 1 — Agents ekranına git

Platformda tüm agent'lar **Agents** ekranı üzerinden yönetilir.

Bu ekranda:

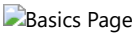
- Mevcut agent'ları görebilir
- Aktif / pasif durumlarını takip edebilir
- Yeni agent oluşturabilirsiniz

👉 Sağ üstten New Agent butonuna tıklayın.



Adım 2 — Agent Basics

Bu adımda agent'ın kimliği tanımlanır.



Agent Name

Agent'ın kullanıcı arayüzünde görünen adıdır.

Agent Code

Sistemde agent'ı temsil eden benzersiz teknik koddur.

Description

Agent'ın ne yaptığına dair kısa ve açıklayıcı bilgi.

Agent Type

Bu aşamada iki tip bulunur:

- Standard Agent: Tek başına görev yapan agent.
- Supervisor: Alt agent'lara yönlendirme yapan yönetici agent.

👉 İlk agent için Standard Agent seçiyoruz.

Version & Status

Version: Agent versiyonlaması içindir (örn: 1.0.0)

Status: Aktif / Pasif durumu

Model Seçimi

Bu adımda agent'ın hangi LLM ile çalışacağı belirlenir.

Örnek:

- Azure GPT-4o
- Groq Llama
- Kurum içi fine-tuned modeller

Bu model, agent'ın tüm cevap üretiminden sorumludur.

Agent artık hazır

Gerekli bilgileri doldurup agent'ı kaydettikten sonra,
bir sonraki adım olan prompts adımına geçiyoruz

Sonraki adım

2. Prompt Yapılandırması

Bu sayfada

2. Prompt Yapılandırması

Bu adımda agent'ın **nasıl düşüneceği**,
nasıl konuşacağı ve
hangi kurallar altında hareket edeceği tanımlanır.

Prompt'lar, agent'ın karakterini ve davranış biçimini belirleyen en kritik yapılandırmadır.

Bu ekranda yapılan tanımlar:

- Agent'ın kullanıcıyı nasıl karşılayacağını
- Hangi sorulara nasıl cevap vereceğini
- Tool'ları hangi koşullarda kullanacağını
- Belirsiz durumlarda nasıl davranacağını

doğrudan etkiler.

Prompt & Runbook yapısı

Prompt ekranı üç ana bölümden oluşur:

1. **System Prompt**
2. **In-Context Examples**
3. **Humanize / Tone**

Her alanın sistemdeki rolü farklıdır.

System Prompt

System Prompt, agent'ın **temel kimliğini** tanımlar.

System Prompt doğru tanımlandığında agent:

- Ne yapacağını bilir
- Ne yapmaması gerektiğini bilir
- Tutarlı cevap üretir

Agent bu metni, tüm konuşmalar boyunca **değişmez sistem talimatı** olarak kabul eder.

System Prompt içerisinde tool davranışları da belirlenebilir.

Bu sayede agent'ın tool'ları:

- Gereksiz yere çağırması engellenir
- Eksik bilgiyle işlem yapması önlenir
- Daha kontrollü çalışması sağlanır

In-Context Examples

Bu alan agent'a nasıl davranması gerektiğini örneklerle öğretir.

Amaç:

- Belirsiz durumlarda doğru refleksi kazandırmak
- Yanlış veya hayali cevapları engellemek
- Davranışı stabilize etmektir

Humanize / Tone

Bu alan, agent'ın cevabı nasıl sunduğunu belirler.

- Dil tonu
- Cevap uzunluğu
- Bilgi sıralaması

buradan yönetilir.

Özet

Bu adımda amaç:

- Agent'a karakter kazandırmak
- Davranışı netleştirmek
- Üretim ortamı için güvenli bir temel oluşturmaktır

Doğru yazılmış prompt'lar:

- Daha kaliteli cevaplar üretir
- Tool kullanımını dengeler
- Kullanıcı deneyimini güçlendirir

Sonraki adım

Prompt yapılandırması tamamlandıktan sonra agent'ın çalışacağı model seçilir.

3. Model Yapılandırması

Bu sayfada

3. Model Configuration

Bu adımda, **Basics ekranında seçilen modelin** cevap üretim davranışı yapılandırılır.

Burada model değiştirilmez.
Sadece seçili modelin **nasıl yanıt üreteceği** kontrol edilir.

LLM Configuration

Bu alan, modelin yanıt üretme şeklini belirler.

Amaç:

- Daha tutarlı cevaplar almak
- Gereksiz token kullanımını önlemek
- Agent davranışını stabilize etmektir

Temperature

Modelin yanıt üretme stilini kontrol eder.

- **0 – Precise**
  Daha net, tutarlı ve kurumsal yanıtlar üretir.
- **1 – Balanced**
  Doğal ve dengeli cevaplar verir.
- **2 – Creative**
  Daha serbest ve yaratıcı çıktılar üretir.

Max Tokens

Modelin tek bir yanıtta üretebileceği maksimum içerik miktarıdır.

- Düşük değer → kısa cevaplar
- Yüksek değer → daha detaylı açıklamalar

  İlk agent için önerilen kullanım: 1000 – 2000

Tool Choice

Modelin tool kullanma davranışını belirler.

- **Auto**: Model ihtiyaca göre tool kullanır. *(önerilen)*
- **Required**: Model her yanıtta tool çağırmak zorundadır.
- **None**: Model tool kullanamaz.

  İlk agent için önerilen ayar: Auto

Advanced Parameters

Bu alan, ileri seviye model ayarları içindir.

Varsayılan değerler çoğu senaryo için yeterlidir.
İhtiyaç olmadıkça değiştirilmesi önerilmez.

Özet

Bu adımda seçili modelin davranışı ayarlanır.

Doğru konfigürasyon sayesinde agent:

- Daha öngörülebilir çalışır
- Daha stabil çıktılar üretir
- Üretim ortamına hazır hale gelir

Sonraki adım

Model davranışı yapılandırıldıktan sonra
agent'a hangi **tool'ların** yetkili olacağı belirlenir.

4. Tool Kullanımı

Bu sayfada

4. Tool Kullanımı

Agent oluşturma sürecinde tool'lar **Tools** adımında yapılandırılır.

Bu ekranda agent'ın hangi tool'ları kullanabileceği belirlenir.

Ekran yapısı

Ekran iki ana bölümden oluşur.

Selected Tools

Bu alanda, agent'a özel olarak aktif edilen tool'lar yer alır.

- Bu tool'lar yalnızca ilgili agent tarafından kullanılabilir
- Agent, ihtiyaç duyduğunda bu tool'ları otomatik olarak çağırır
- Kullanıcı tool'ları manuel olarak tetiklemez

Her agent için farklı bir tool seti tanımlanabilir.

Tool Catalog

Bu alanda sistemde tanımlı tüm tool'lar listelenir.

- Kurumda yüklenmiş hazır tool'lar burada görünür
- Arama ve filtreleme yapılabilir
- Seçilen tool'lar agent'a eklenebilir

Tool'lar merkezi olarak yönetilir;
hangi agent'ın hangi tool'u kullanacağı bu ekrandan kontrol edilir.

Tool ekleme yöntemleri

Maistro'da agent'a tool eklemek için iki farklı yöntem bulunur.

Add Tool

Add Tool butonuna tıklandığında sağ tarafta Tool Catalog paneli açılır.

Bu yöntemle:

- Sistemde hazır bulunan tool'lar görüntülenir
- İlgili tool seçilerek agent'a eklenir

Hazır tool'ları kullanmak isteyen senaryolar için hızlı ve pratiktir.

Browse Tool Catalog

Browse Tool Catalog seçeneği ile sisteme yeni bir tool yüklenir.

Bu işlem:

- YAML manifest dosyası üzerinden yapılır
- Yeni geliştirilen tool'ların platforma tanıtılmasını sağlar
- Tool registry yapısına kayıt oluşturur

Yüklenen tool'lar otomatik olarak Tool Catalog içinde görünür.

Tool yapılandırma seçenekleri

Bir tool agent'a eklendiğinde aşağıdaki ayarlar aktif hale gelir.

Enable

- Tool'un agent tarafından kullanılabilir olup olmadığını belirler
- Pasif tool'lar çağrılmaz

Human-in-the-Loop (HITL)

- Tool çalıştırılmadan önce insan onayı alınmasını sağlar
- Kritik aksiyonlar için kontrol noktası oluşturur

Max calls / turn

- Tek bir konuşma adımı içinde tool'un kaç kez çağrılabileceğini sınırlar
- Döngüsel veya gereksiz çağrıları engeller

Timeout (ms)

- Tool çalışması için maksimum süreyi belirler
- Süre aşımı durumunda çağrı sonlandırılır

Fail open (continue on error)

- Tool hata alsa bile agent akışının devam etmesini sağlar
- Destekleyici tool'lar için tercih edilir

Özetle

Bu adımda amaç:

- Agent'a gereksiz yetki vermemek
- Yalnızca gerçekten ihtiyaç duyulan tool'ları tanımlamak
- Agent davranışını kontrollü tutmaktır

Tool seçimi tamamlandıktan sonra bir sonraki adımda Memory yapılandırmasına geçilir.

Sonraki adım

Memory Yapılandırılması

Bu sayfada

5. Memory Yapılandırması

Bu adımda agent'ın **konuşma bağlamını nasıl hatırlayacağı** belirlenir.

Memory ayarları sayesinde agent:

- Önceki mesajları dikkate alır
- Daha tutarlı yanıtlar üretir

Short-Term Memory

Short-Term Memory, agent'ın **aktif konuşma sırasında**
önceki mesajları hatırlamasını sağlar.

Bu yapı, kısa süreli bağlam yönetimi için kullanılır.

Mode

- **Mix**
  Mesaj sayısı ve zaman aralığını birlikte kullanır. *(önerilen)*

- **Last N**
  Sadece son N mesajı dikkate alır.

Window

Agent'ın hatırlayacağı maksimum mesaj sayısını belirler.

Örnek: 10 mesaj

Days

Mesajların geçerlilik süresini tanımlar.

Belirtilen süreden eski mesajlar bağlama dahil edilmez.

Örnek: 7 gün

Customer Memory

Customer Memory, kullanıcıya özel bilgilerin
uzun süreli olarak saklanmasını sağlar.

Bu yapı sayesinde agent:

- Kullanıcı tercihlerini
- Daha önce paylaşılan bilgileri
- Kişiselleştirilmiş verileri

hatırlayabilir.

İlk kullanım için **kapalı bırakılması önerilir**.

External Memory

External Memory, üçüncü parti memory sağlayıcıları ile
entegrasyon kurmak için kullanılır.

Kurumsal senaryolarda:

- Harici vektör veritabanları
- Özel memory servisleri

bu alan üzerinden bağlanabilir.

Özet

Bu adımda:

- Agent'ın konuşma hafızası tanımlanır
- Bağlam yönetimi kontrol altına alınır
- Daha doğal diyaloglar sağlanır

Doğru memory yapılandırması sayesinde agent:

- Aynı soruları tekrar sormaz
- Konuşma kopukluğu yaşamaz
- Kullanıcı deneyimini güçlendirir

Sonraki adım

Memory yapılandırması tamamlandıktan sonra
agent için güvenlik ve kontrol kuralları tanımlanır.

Guardrails Yapılandırılması

Bu sayfada

6. Guardrails Yapılandırması

Bu adımda agent için **güvenlik ve kontrol kuralları** tanımlanır.

Guardrails, hem kullanıcı girdilerini hem de agent çıktısını denetlemek için kullanılır.

Enable Guardrails

Bu alan aktif edildiğinde güvenlik kontrolleri devreye alınır.

Guardrails kapalıysa:

- Giriş kontrolleri çalışmaz
- Çıkış filtreleri uygulanmaz

Üretim ortamı için açık olması önerilir.

Engine Mode

Guardrails kontrollerinin nerede çalışacağını belirler.

- **Local Only**
  Kontroller yalnızca sistem içinde çalışır. Daha hızlıdır ancak tespit edebileceği ihlal türleri sınırlıdır.

- **External Only**
  Bankanın guardrails servisleri kullanılır. Daha geniş bir ihlal yelpazesini tespit edebilir.
- **Hybrid**
  Local ve external kontroller birlikte çalışır.

Input Checks

Kullanıcıdan gelen mesajlar bu aşamada kontrol edilir.

Prompt Injection

Kullanıcının agent davranışını değiştirmeye yönelik
zararlı yönlendirmeleri engeller.

- **Block** seçildiğinde istek doğrudan durdurulur.
- **Allow** seçildiğinde isteğin geçmesine izin verilir ve bu durum kayıt altına alınır.

Output Checks

Agent tarafından üretilen cevaplar kontrol edilir.

PII

Kişisel veri içeren çıktılar tespit edilir.

- **Block** seçildiğinde istek doğrudan durdurulur.
- **Sanitize** seçildiğinde hassas bilgiler maskeleme ile temizlenir.
- **Allow** seçildiğinde isteğin geçmesine izin verilir ve bu durum kayıt altına alınır.

External Category

External Category, bankanın guardrails servisinde tespit edilmeye çalışılan ihlalleri ifade eder. Yapılandırmak istediğiniz external category'i seçerek bu kategori için block, sanitize veya allow seçeneklerinden birini belirleyebilirsiniz.

Özet

Bu adımda:

- Kullanıcı girdileri korunur
- Agent çıktıları denetlenir
- Güvenli kullanım sağlanır

Guardrails, agent'ın kurumsal ortamlarda
kontrollü ve güvenli şekilde çalışmasını sağlar.

Sonraki adım

Tüm ayarlar tamamlandıktan sonra
agent yapılandırması gözden geçirilir.

Review

Bu sayfada

7. Review

Bu adımda agent oluşturulmadan önce
tüm yapılandırmalar **tek ekranda özetlenir**.

Görüntülenen bilgiler

Review ekranında aşağıdaki bilgiler yer alır:

- Agent temel bilgileri
- Seçilen model ve model ayarları
- Tanımlı tool'lar
- Memory yapılandırması
- Guardrails kuralları

Bu ekran, son kontrol noktasıdır.

Amaç

Review adımının amacı:

- Yapılandırma hatalarını önlemek
- Eksik veya yanlış ayarları fark etmek

- Agent'ı oluşturmadan önce son kontrolü sağlamaktır

Agent oluşturma

Tüm bilgiler doğrulandıktan sonra
sağ alt köşedeki **Complete** butonuna basılarak agent oluşturulur.

Agent oluşturulduktan sonra:

- Agent aktif hale gelir
- Agent listesinde görünür
- Session başlatmaya hazır olur

Bu adımla birlikte agent oluşturma süreci tamamlanır.

> Agent'ınız artık Maistro'nun uçtan uca AI orchestration altyapısı üzerinde aktif olarak çalışmaya hazırdır.

Bu sayfada

Agent Nedir?

Maistro'da **Agent**, belirli bir iş hedefi için yapılandırılmış, konuşma yürütürken gerektiğinde **tool çağırabilen** ve çıktıyı kontrollü şekilde üreten çalışma birimidir. Agent; tek başına "model" değildir. **Model + prompt + araçlar + bellek + güvenlik politikaları + (opsiyonel) RAG** birleşimidir.

Bu sayede agent'lar:

- Sadece cevap üretmez
- Karar alabilir
- Dış sistemlerle etkileşime girebilir
- Süreçleri uçtan uca yönetebilir

Agent ne zaman gerekir?

- **Çok adımlı işler:** Ara kararlar, koşullar ve yönlendirme (routing) gerektiren süreçler
- **Dış sistemlerle etkileşim:** API, veri tabanı veya workflow sistemleriyle işlem yapılması
- **Bilgi toplama + cevap üretme:** Dokümanlardan bağlam çekilerek güvenilir ve kaynaklı yanıt oluşturulması (RAG)

Agent hangi bileşenlerden oluşur?

Bir agent aşağıdaki yapıların birleşimidir:

Prompt

Agent'ın nasıl konuşacağını, nasıl düşüneceğini ve hangi kurallar altında hareket edeceğini belirler.

Model

Agent'ın cevap üretirken kullanacağı dil modeli tanımlanır.

Tool'lar

Tool'lar, agent'ı pasif bir asistan olmaktan çıkararak aksiyon alabilen bir yapıya dönüştürür.
(API çağrıları, kayıt oluşturma, veri okuma/yazma, workflow tetikleme gibi)

Memory

Konuşma bağlamının ve geçmiş mesajların nasıl yönetileceğini belirler.

Guardrails

Güvenli kullanım için sınırlar, kurallar ve kısıtlar tanımlanır.

RAG (opsiyonel)

Agent'ın dokümanlardan bilgi çekerek, kaynaklı ve güncel yanıt üretmesini sağlar.

Agent tasarım prensipleri

Etkili ve sürdürülebilir agent'lar için aşağıdaki prensipler önerilir:

1. **Hedefi tek cümleyle tanımlayın**
2. **Tool setini minimal tutun**
3. **Güvenlik kurallarını prompt'un parçası yapın**
4. **Çıktı formatını standardize edin** (ör. JSON)

Agent tipleri

Maistro'da iki farklı agent tipi bulunur.
Seçilecek agent tipi, kullanım senaryosuna göre belirlenir.

Standard Agent

Standard Agent, tek bir iş hedefi için yapılandırılan agent tipidir.

- Belirli bir göreve odaklanır
- Kendi prompt, model, tool ve kurallarıyla çalışır
- Kullanıcıdan gelen isteği doğrudan işler

Tipik kullanım senaryoları:

- Doküman soru-cevap agent'ı
- Hava durumu, bilgi sorgulama, veri çekme agent'ı
- Ticket açma veya kayıt oluşturma agent'ı

Tek başına çalışan, net görev tanımı olan senaryolar için önerilir.

Supervisor Agent

Supervisor Agent, gelen istekleri analiz edip uygun alt agent'a yönlendiren agent tipidir.

Supervisor agent:

- Kullanıcı isteğini yorumlar
- Hangi agent'ın bu isteği karşılayabileceğine karar verir
- İlgili agent'a yönlendirme yapar

Bu yapı özellikle:

- Birden fazla uzman agent bulunan senaryolarda
- Farklı iş alanlarının tek giriş noktasıyla yönetilmesi gerektiğinde
- Karmaşık yönlendirme ihtiyaçlarında

kullanılır.

Örnek:

- Kullanıcı "hava durumu" sorar → hava agent'ı çalışır
- Kullanıcı "ticket aç" der → operasyon agent'ı çalışır

Supervisor agent, bu yönlendirme kararını merkezi olarak verir.

Hangi agent tipi ne zaman tercih edilmeli?

Senaryo	Önerilen agent
Tek amaçlı görev	Standard Agent
Belirli bir işi yapan agent	Standard Agent
Birden fazla agent yönetimi	Supervisor Agent
Akıllı yönlendirme ihtiyacı	Supervisor Agent

> **Not:**
> Supervisor agent'lar da standart agent'lar gibi prompt, model, tool ve güvenlik kurallarına sahiptir.
> Farkı, doğrudan işlem yapmak yerine **yönlendirme rolü** üstlenmesidir.

Agent + RAG ilişkisi

Maistro'da RAG, agent'ın içinde **bir adım** olarak çalışır.

Tipik akış şu şekildedir:

- Kullanıcı sorusu alınır
- İlgili dokümanlardan **retrieve** işlemi yapılır
- Elde edilen bağlam modele aktarılır
- Yanıt **kaynaklarıyla birlikte** üretilir

Bu yaklaşım, tahmine dayalı cevaplar yerine **dayanaklı bilgi** üretmeyi sağlar.

Agent + Automation

Agent'lar yalnızca sohbet etmek için kullanılmaz.

Gerektiğinde:

- Ticket oluşturabilir
- Onay süreci başlatabilir
- Job veya workflow tetikleyebilir
- Sonuçları toparlayıp kullanıcıya raporlayabilir

Bu sayede agent, pasif bir asistan değil; **operasyonel bir yürütücü** haline gelir.

Özet

Maistro'da agent:

- Modeli üretime hazır hale getirir
- Karar alabilen ve aksiyon üretebilen bir yapı sunar
- RAG ve tool'lar ile gerçek iş değeri üretir
- Güvenli, izlenebilir ve yönetilebilir şekilde çalışır

Bu yapı sayesinde GenAI çözümleri, deneme seviyesinden çıkarak **kurumsal kullanım** için uygun hale gelir.

İlgili bölümler

- İlk Agent'ını Oluştur
- Tool Kullanımı

Bu sayfada

Tool nedir?

Bu sayfa, Maistro'da tool kavramının ne olduğunu ve agent'lar üzerinde nasıl yapılandırıldığını açıklar.

Tool; agent'ın ihtiyaç duyduğunda çağırabildiği, belirli bir işi gerçekleştiren fonksiyonel yeteneklerdir.

Tool'lar sayesinde agent yalnızca cevap üreten bir yapı olmaktan çıkar;
aksiyon alabilen, sistemlerle entegre çalışan bir yapıya dönüşür.

Kullanıcı açısından bakıldığında tool'lar:

- Arka planda çalışır
- Manuel olarak tetiklenmez
- Agent tarafından otomatik olarak kullanılır

Kullanıcı yalnızca ortaya çıkan sonucu görür.

Tool'lar nasıl çalışır?

Bir konuşma sırasında agent:

1. Kullanıcının isteğini analiz eder
2. Bu isteğin bir işlem gerektirip gerektirmediğini değerlendirir
3. Gerekli görürse uygun tool'u çağırır
4. Ortaya çıkan sonucu cevap olarak kullanıcıya sunar

Bu süreç tamamen agent tarafından yönetilir.

Platformda kullanılabilecek tool türleri

Kuruluma ve yetkilere bağlı olarak Maistro platformunda farklı tool türleri kullanılabilir.

Örnek kullanım alanları:

- Dosya okuma veya yazma
- CSV / Excel gibi veri dosyalarıyla işlem
- Doküman içeriklerinin incelenmesi
- Müşteri veya kayıt aramaları
- Harici kaynaklardan bilgi alma
- E-posta gönderimi

Kullanılabilir tool seti, kurumun ihtiyaçlarına göre değişebilir.

Maistro'da tool yaklaşımı

Maistro'da tool'lar:

- Registry üzerinden merkezi olarak yönetilir
- YAML manifest ile tanımlanır
- Versiyonlanır
- Runtime sırasında dinamik olarak yüklenir

Bu yapı sayesinde:

- Agent konfigürasyonu bozulmaz
- Tool güncellemeleri kontrollü ilerler
- Kurumsal ortamlarda güvenli kullanım sağlanır

Tool manifest yapısı

Her tool, sisteme tanıtılırken belirli bir tanım yapısına sahip manifest dosyası ile yapılandırılır.

Bu yapı içerisinde:

- Tool adı ve açıklaması

- Parametre yapısı
- Runtime ve package bilgisi
- Versiyon bilgisi

yer alır.

Bu sayede tool'lar standart, taşınabilir ve tekrar kullanılabilir hale gelir.

Versiyonlama ve latest yönetimi

Tool'lar semantik versiyonlama yaklaşımıyla yönetilir.

Örnek:

- `1.0.0`
- `1.1.0`
- `2.0.0`

Her tool için:

- Birden fazla versiyon aynı anda sistemde bulunabilir
- Aktif kullanılan sürüm `latest` olarak işaretlenir

İhtiyaç halinde:

- Belirli bir versiyon pinlenebilir
- Eski versiyonlar kaldırılabilir
- Runtime güncellemeleri kontrollü şekilde yapılır

Tool güvenliği ve kontrol

Tool kullanımı platform tarafından kontrol altındadır.

- Parametreler manifest şemasına göre doğrulanır
- Yetkisiz veya hatalı çağrılar engellenir
- Tool çalışmaları izlenebilir
- Gerekli senaryolarda insan onayı (HITL) devreye alınabilir

Bu sayede üretim ortamında agent davranışı kontrol altında tutulur.

Hazır tool'lar (örnek)

Kuruluma bağlı olarak platformda aşağıdaki türde tool'lar kullanılabilir:

- **csv_tool**
  CSV dosyalarını okuma veya yazma işlemleri için kullanılır.
- **excel_reader_tool / excel_writer_tool**
  Excel dosyalarından veri okumak veya çıktı üretmek için kullanılır.
- **docx_reader_tool**
  Word (.docx) dosyalarının içeriğini okumak için kullanılır.
- **customer_search_tool**
  İsim veya müşteri numarası üzerinden kayıt araması yapmak için kullanılır.
- **ddg_search_tool**
  Harici kaynaklardan (web) bilgi araması yapmak için kullanılır.
- **email_sender_tool**
  Belirli senaryolarda e-posta gönderimi için kullanılır.

  Kullanılabilir tool seti, kurumun ihtiyaçlarına ve kurulumuna göre değişebilir.

Tool kullanımına dair iyi pratikler

Agent davranışını netleştirmek için tool kullanımına dair açık kurallar tanımlanmalıdır.

Etkili ve sürdürülebilir bir agent davranışı için:

- Tool'ları **az ve amaç odaklı** kullanın
- Açıklamaya **hangi durumda kullanılacağını** net şekilde yazın
- Agent'a gereksiz yetkiler vermekten kaçının
- Tool güncellemelerini versiyonlayarak yayınlayın

Örnek policy yaklaşımları:

- "Veri dosyasıyla işlem yapılacaksa önce **csv_tool** veya **excel_reader_tool** kullan."
- "Doküman içeriği gerekiyorsa **docx_reader_tool** ile metni oku."
- "Kullanıcı veya müşteri bilgisi gerekiyorsa **customer_search_tool** çağır."
- "Dış kaynaklı bilgi gerekiyorsa **ddg_search_tool** ile arama yap."
- "Kullanıcıya aksiyon bildirilmesi gerekiyorsa **email_sender_tool** kullan."

Bu prensipler sayesinde agent:

- Ne zaman hangi tool'u kullanacağını bilir
- Gereksiz çağrılardan kaçınır
- Daha öngörülebilir ve güvenli davranır

Özetle

Tool yapısı sayesinde:

- Agent yalnızca cevap veren bir yapı olmaktan çıkar
- Otomasyon senaryoları mümkün hale gelir
- AI çıktıları somut aksiyonlara dönüşür

Maistro'da tool'lar,
AI'ı sadece konuşan değil, çalışan bir sisteme dönüştürür.

Bu sayfada

Guardrails Nedir?

Guardrails, bir agent'ın çalışması sırasında ortaya çıkabilecek
güvenlik, uyum, veri gizliliği ve kurumsal riskleri yöneten bir kontrol katmanıdır.

Bu katman:

- Kullanıcıdan gelen girdileri (**input**)
- Yapay zekâ tarafından üretilen çıktıları (**output**)

ayrı ayrı değerlendirir ve her iki yönde de
kontrollü, politika uyumlu ve izlenebilir bir davranış sağlar.

İşlem Aşamaları

Guardrails, agent çalışma akışına iki farklı noktada entegre olur.

Input Değerlendirme Aşaması

Kullanıcı girdisi modele iletilmeden önce analiz edilir.

Bu aşamada Guardrails:

- Model davranışını manipüle etmeye yönelik girişimleri tespit eder
- Kurumsal politikalara aykırı talepleri filtreler
- Riskli içeriğin modele hiç ulaşmamasını sağlar

Bu yaklaşım, model seviyesinde oluşabilecek riskleri daha en başta engeller.

Output Değerlendirme Aşaması

Model yanıtı üretildikten sonra, kullanıcıya gösterilmeden önce kontrol edilir.

Bu aşamada Guardrails:

- Hassas veya kişisel verilerin sızmasını önler
- Modelin ürettiği içeriğin bağlamsal olarak güvenli olup olmadığını değerlendirir
- Kurumsal dil, etik ve yasal uyumu garanti altına alır

Bu katman, özellikle **veri gizliliği ve regülasyon uyumu** açısından kritiktir.

Kontrol Motorları (Validation Engines)

Guardrails, doğrulama işlemlerini iki farklı motor üzerinden yürütür.
Bu motorlar birlikte veya ayrı ayrı çalışabilir.

Local Engine (Yerel Doğrulamalar)

Local engine, düşük gecikmeli ve deterministik kontroller için tasarlanmıştır.

Özellikleri:

- Sistem içinde çalışır
- Hızlıdır
- Anında karar üretir

Ele aldığı başlıca konular:

- Prompt injection ve benzeri manipülasyon girişimleri
- Kişisel veya hassas veri kalıpları (ör. kimlik, finansal bilgi)

External Engine (Harici Doğrulamalar)

External engine, daha kapsamlı ve bağlamsal analiz gerektiren durumlar için kullanılır.
Bankanın harici guardrails servisleri üzerinden çalışır ve kategori bazlı, politika odaklı kontroller uygular.

Özellikleri:

- Bankanın harici guardrails servislerini kullanır
- Geniş kapsamlı kategori bazlı analiz yapar
- Kurumsal, etik ve regülasyon odaklı kontroller sağlar
- Local engine'e kıyasla daha yüksek doğruluk ve kapsama alanı sunar
  (ek gecikme pahasına)

Aşağıda external engine tarafından tespit edilebilen tüm ihlal kategorileri yer almaktadır.
Her başlıkta **önce İngilizce kategori adı**, parantez içinde **Türkçe karşılığı** verilmiştir.

External Engine İhlal Kategorileri

- **Explicit Content** *(Açık / Müstehcen İçerik)*
  Cinsel içeriklerin tespiti.
- **Profanity** *(Küfür ve Argo Dil)*
  Küfür, hakaret veya argo kullanımının tespiti.
- **Hate Speech and Discrimination** *(Nefret Söylemi ve Ayrımcılık)*
  Kişi veya gruplara yönelik nefret, dışlama veya ayrımcı dilin tespiti.
- **Violence and Harm** *(Şiddet ve Fiziksel Zarar)*
  Şiddeti teşvik eden, betimleyen veya fiziksel zarara yol açabilecek içerikler.
- **Sensitive Topics** *(Hassas Konular)*
  Politik, kültürel veya sosyal açıdan hassas ve rahatsız edici olabilecek konular.
- **Illegal Activities** *(Yasa Dışı Faaliyetler)*
  Yasa dışı eylemlerin teşviki, yönlendirilmesi veya tartışılması.
- **Harmful Advice** *(Zararlı Tavsiyeler)*
  Kişisel, finansal veya hukuki zarara yol açabilecek yönlendirmeler.
- **Privacy Violations** *(Gizlilik İhlalleri)*
  Kişisel, gizli veya özel bilgilerin izinsiz paylaşımı.
- **Sensitive Information Disclosure** *(Hassas Bilgi İfşası)*
  Müşteriler, çalışanlar veya banka ile ilgili kritik ve korunması gereken veriler.
- **Prompt Injection** *(Prompt Manipülasyonu)*
  Model davranışını değiştirmeye veya güvenlik sınırlarını aşmaya yönelik girişimler.
- **Insecure Output Handling** *(Güvensiz Çıktı Üretimi)*
  Güvenlik açığına veya yanlış kullanıma yol açabilecek model çıktıları.
- **Model Theft** *(Model Yeteneği Sömürüsü / Model Hırsızlığı)*
  Modelin iç işleyişini, kurallarını veya yeteneklerini açığa çıkarmaya yönelik girişimler.
- **Misinformation** *(Yanlış / Yanıltıcı Bilgi)*
  Gerçeğe aykırı, doğrulanmamış veya yanıltıcı bilgi üretimi.
- **Overreliance** *(Aşırı Güven ve Bağımlılık)*
  Kullanıcının yalnızca yapay zekâ çıktısına güvenmeye yönlendirilmesi.
- **Competitor Mention** *(Rakip Firma Referansı)*
  Rakip firmalara yönelik uygunsuz, politika dışı veya riskli atıflar.
- **Tone Control** *(Üslup ve Ton Kontrolü)*
  Kurumsal, profesyonel ve etik iletişim standartlarına aykırı dil kullanımı.

External engine, bu kategoriler üzerinden
bağlam farkındalığı yüksek, **politika uyumlu** ve **regülasyon odaklı** bir güvenlik katmanı sağlar.

Bu kapsamlı analiz, local engine'e kıyasla daha yüksek gecikme süresi yaratabilir;
ancak üretim ortamlarında kritik risklerin tespit edilmesini mümkün kılar.

Çalışma Sırası ve Orkestrasyon

Guardrails, local ve external doğrulama motorlarını farklı çalışma stratejileri ile orkestre edebilir.
Bu stratejiler, güvenlik kapsamı ile performans gereksinimleri arasında denge kurulmasını sağlar.

- **Local-first yaklaşımı**
  Öncelikle local kontroller çalıştırılır.
  Gerekli görülmesi durumunda harici guardrails devreye alınır.

- **External-first yaklaşımı**
  Önce bankanın harici guardrails servisi çalıştırılır.
  Ardından local doğrulamalar uygulanır.

- **Parallel yaklaşım**
  Local ve external doğrulama motorları eş zamanlı çalışır.
  Sonuçlar birleştirilerek nihai karar oluşturulur.

İhlal Tespit Edildiğinde Uygulanan Davranışlar

Guardrails, bir ihlal tespit ettiğinde duruma göre farklı aksiyonlar alabilir.

Engelleme (Block)

- İçerik tamamen durdurulur
- Agent akışı devam etmez

Maskeleme (Sanitize)

- Riskli veya hassas kısımlar gizlenir
- İçeriğin güvenli bölümü kullanıcıya sunulur

İzin Verme (Allow)

- İçerik kullanıcıya iletilir
- İhlal durumu loglanır ve izlenebilir hâle getirilir

Bu aksiyonlar, güvenlik politikalarının **ince ayarlarla uygulanmasını** mümkün kılar.

Azure Modelleri ile Etkileşim

Azure tabanlı modeller, kendi **yerleşik güvenlik ve içerik filtreleme mekanizmalarına** sahiptir.

Bu nedenle:

- Guardrails yapılandırması kapalı olsa bile
- Azure modeli, kendi iç politikalarına dayanarak bir isteği veya yanıtı engelleyebilir

Bu durum, Guardrails sistemine
ek ve bağımsız bir güvenlik katmanı kazandırır.

Sağlanan Teknik Kazanımlar

Guardrails kullanımıyla:

- Agent davranışı deterministik ve öngörülebilir hâle gelir
- Hassas veri sızıntıları engellenir
- Kurumsal ve regülasyon uyumu güçlenir
- Model çıktıları denetlenebilir ve izlenebilir olur
- Güvenlik, performans ve kullanıcı deneyimi arasında kontrollü bir denge sağlanır

Bu sayfada

Arayüz Mimarisi

Bu bölümde, Maistro Automation ekranının genel yapısı ve temel bileşenleri açıklanmaktadır.
Amaç; kullanıcıların görsel akış tasarımına geçmeden önce platformun çalışma düzenini net biçimde kavramasını sağlamaktır.

Automation Flows Ekranı

Automation Flows alanı, Maistro'da oluşturulan tüm otomasyon akışlarının merkezi yönetim noktasıdır.

Bu ekran üzerinden:

- Mevcut automation flow'lar görüntülenir
- Yeni flow oluşturma işlemleri başlatılır
- Var olan akışlar düzenlenir ve yönetilir

Sağ üst köşede yer alan **New Automation Flow** butonu ile yeni bir otomasyon akışı oluşturulabilir.

Flow Canvas (Çalışma Alanı)

Bir automation flow açıldığında, kullanıcıyı merkezi bir çalışma alanı karşılar.

Bu alan:

- Akışın görsel olarak tasarlandığı
- Node'ların konumlandırıldığı

- Bağlantıların oluşturulduğu

ana tuvaldir.

Canvas yapısı, karmaşık otomasyon senaryolarının sade ve okunabilir biçimde tasarlanmasını amaçlar.

Sol Panel – Node Kataloğu

Canvas'ın sol tarafında, akışa eklenebilecek bileşenlerin yer aldığı node kataloğu bulunur.

Bu panel üzerinden aşağıdaki node türleri eklenebilir:

- **Input** – Akışın çalışması için gerekli verinin sisteme alınmasını sağlar
- **Output** – Agent tarafından üretilen çıktının görüntülenmesini ve dış sistemlere aktarılmasını sağlar
- **Model** – Agent'ın kullanacağı dil modelinin yapılandırılmasını sağlar
- **Agent** – Tanımlanan görevleri model ve tool'lar ile birlikte yürüten yürütücü bileşendir
- **Tool** – Agent'ın harici sistemlerle etkileşime geçmesini ve aksiyon almasını sağlar
- **Task** – Agent'ın hangi işi, hangi kapsamda gerçekleştireceğini tanımlar
- **Trigger** – Akışın hangi koşulda veya hangi olayla başlatılacağını belirler

Node'lar bu panelden seçilerek canvas üzerine eklenir.

Node Yapısı

Canvas üzerinde yer alan her node, belirli bir sorumluluğu temsil eder.

Node'lar:

- Girdi ve çıktı bağlantı noktalarına sahiptir
- Akış içerisindeki veri yönünü görsel olarak ifade eder
- Birbirleriyle bağlantı kurularak çalışır

Bu yapı sayesinde otomasyon akışı hem okunabilir hem de izlenebilir hale gelir.

Sağ Panel – Ayar ve Konfigürasyon Alanı

Canvas üzerinde herhangi bir node seçildiğinde, ekranın sağ tarafında ilgili node'a ait ayar paneli açılır.

Bu panel üzerinden:

- Node adı ve açıklaması düzenlenir
- Model parametreleri yapılandırılır
- Task içerikleri tanımlanır
- Output formatı belirlenir

Sağ panel, flow davranışının şekillendirildiği ana konfigürasyon alanıdır.

Flow Seviyesi Ayarlar

Automation flow'un tamamına ait ayarlar ayrı bir yapı üzerinden yönetilir.

Bu alanda:

- Flow adı
- Flow açıklaması
- Akışa ait genel bilgiler

tanımlanır.

Bu bilgiler, flow'ların yönetimi ve takibi açısından referans niteliği taşır.

Üst Araç Çubuğu

Canvas'ın üst bölümünde, akış yönetimine yönelik temel aksiyonlar yer alır:

- Yakınlaştırma ve uzaklaştırma
- Canvas görünümünü sıfırlama
- Flow'u kaydetme
- Akışı çalıştırma

Bu araçlar, tasarım ve test sürecinin kontrollü şekilde yürütülmesini sağlar.

Arayüzün Temel Yaklaşımı

Maistro Automation arayüzü aşağıdaki prensipler doğrultusunda tasarlanmıştır:

- Görsel okunabilirlik
- Kodsuz akış tasarımı
- Modüler yapı
- Net sorumluluk ayrımı

Bu yaklaşım sayesinde kullanıcılar, karmaşık otomasyon senaryolarını dahi sade ve yönetilebilir biçimde oluşturabilir.

Sonraki Adım

Arayüz mimarisi anlaşıldıktan sonra, automation flow'ların nasıl oluşturulduğu ve node'ların nasıl kullanıldığı
Görsel Akış Tasarımı bölümünde detaylı olarak ele alınacaktır.

Görsel Akış Tasarımı

Bu sayfada

Görsel Akış Tasarımı

Bu bölümde, Maistro'da bir automation flow'un görsel olarak nasıl oluşturulduğu ve node'ların akış içerisinde nasıl konumlandırıldığı açıklanmaktadır.

Görsel akış tasarımı, kod yazmadan otomasyon senaryolarının oluşturulmasını sağlar ve tüm sürecin okunabilir biçimde yönetilmesine olanak tanır.

Yeni Automation Flow Oluşturma

Automation Flows ekranında yer alan **New Automation Flow** butonu ile yeni bir akış oluşturma süreci başlatılır.

Bu adımda:

- Flow adı
- Flow açıklaması

tanımlanır.

Bu bilgiler, oluşturulan automation flow'un platform içerisinde ayırt edilmesini sağlar.

Flow Canvas Üzerinde Çalışma

Flow oluşturulduktan sonra kullanıcı, görsel tasarımın yapıldığı canvas alanına yönlendirilir.

Canvas üzerinde:

- Node'lar konumlandırılır
- Bağlantılar oluşturulur
- Akış sırası görsel olarak tanımlanır

Bu yapı sayesinde otomasyon senaryosu uçtan uca takip edilebilir hale gelir.

Node Ekleme

Canvas'ın sol tarafında yer alan node kataloğu üzerinden, akışa eklenecek bileşenler seçilerek canvas üzerine eklenir.

Akış içerisinde kullanılabilen temel node türleri şunlardır:

- **Input**
- **Task**
- **Model**
- **Agent**
- **Tool**
- **Output**
- **Trigger**

Her node, akış içerisinde belirli bir sorumluluğu temsil eder.

Akışın Temel Yapısı

Tipik bir automation flow aşağıdaki bileşenlerden oluşur:

- Akışa veri sağlayan **Input**
- Agent'ın gerçekleştireceği işi tanımlayan **Task**
- Kullanılacak dil modelini belirleyen **Model**
- Tüm süreci yürüten **Agent**
- Üretilen sonucun dışarı aktarıldığı **Output**

Bu yapı, flow'un okunabilir ve sürdürülebilir olmasını sağlar.

Node'lar Arası Bağlantı

Node'lar, üzerlerinde bulunan bağlantı noktaları aracılığıyla birbirine bağlanır.

Bağlantılar sayesinde:

- Veri akış yönü belirlenir
- Hangi bileşenin hangi bilgiyi kullandığı netleşir
- Akış sırası görsel olarak ifade edilir

Bağlantılar soldan sağa doğru ilerleyecek şekilde tasarlanmalıdır.

Agent Node Yapısı

Agent node, automation flow'un merkezinde yer alır.

Agent;

- Task'ten gelen görevi alır
- Model üzerinden dil yeteneklerini kullanır
- Gerekli durumlarda tool'lar ile etkileşime geçer
- Ortaya çıkan sonucu output'a iletir

Bu nedenle agent node, flow'un yürütücü bileşeni olarak konumlandırılır.

Task Yapılandırması

Task node içerisinde, agent'ın gerçekleştireceği iş tanımlanır.

Bu alanda:

- Görev adı
- Görev içeriği
- Beklenen çıktı türü

belirlenir.

Task, agent'ın ne yapacağını açık ve net şekilde ifade etmelidir.

Model Yapılandırması

Model node, agent'ın kullanacağı dil modelini temsil eder.

Bu alanda:

- Model sağlayıcısı
- Model adı
- Model parametresi (ör. temperature)

tanımlanır.

Model ayarları, agent'ın üretim davranışını doğrudan etkiler.

Output Yapılandırması

Output node, agent tarafından üretilen çıktının görünür hale gelmesini sağlar.

Bu node üzerinden:

- Çıktı tipi (text veya file)

belirlenir.

Output node üzerinde yer alan göz simgesi aracılığıyla, üretilen sonuçlar arayüz üzerinden doğrudan görüntülenebilir.

Flow’un Okunabilirliği

Görsel akış tasarımı sırasında amaç yalnızca çalışır bir flow oluşturmak değil, aynı zamanda okunabilir ve anlaşılabilir bir yapı kurmaktır.

Bu nedenle:

- Node’lar anlamlı şekilde konumlandırılmalı
- Gereksiz bağlantılardan kaçınılmalı
- Akış mantığı soldan sağa net biçimde ilerlemelidir

Bu yaklaşım, bakım ve genişletme süreçlerini kolaylaştırır.

Sonraki Adım

Görsel akış tasarımı tamamlandıktan sonra, oluşturulan automation flow’un nasıl çalıştırıldığı, izlendiği ve yönetildiği **Workflow Yönetimi** bölümünde ele alınacaktır.

Workflow Yönetimi

Bu sayfada

Workflow Yönetimi

Bu bölümde, oluşturulan automation flow'ların nasıl çalıştırıldığı, nasıl izlendiği ve çalışma sürecinin nasıl yönetildiği açıklanmaktadır.

Workflow yönetimi, tasarlanan akışların yalnızca oluşturulmasını değil, aynı zamanda kontrollü biçimde işletilmesini sağlar.

Workflow Çalıştırma

Görsel akış tasarımı tamamlandıktan sonra, workflow üst araç çubuğunda yer alan **Run** aksiyonu ile çalıştırılabilir.

Bu işlemle birlikte:

- Akış başlatılır
- Agent yürütme sürecine girer
- Tanımlanan adımlar sırayla işletilir

Workflow çalıştırıldığında sistem, akışı tanımlanan yapı doğrultusunda otomatik olarak yürütür.

Output Takibi

Workflow çalışması sonucunda üretilen çıktılar, output node üzerinden erişilebilir hale gelir.

Output node üzerinde yer alan göz simgesi aracılığıyla:

- Agent tarafından üretilen sonuçlar görüntülenebilir
- Oluşan çıktı kontrol edilebilir

Çıktı tipi file olarak tanımlanmışsa, üretilen dosya doğrudan indirilebilir veya dış sistemlere aktarılabilir.

Hata Durumları

Workflow çalışması sırasında herhangi bir adımda hata oluşması durumunda, akış otomatik olarak durdurulur.

Bu durumda hata output node üzerinden görülebilir.

Bu yaklaşım, hataların hızlı şekilde tespit edilmesini ve akışın güvenli biçimde yönetilmesini sağlar.

Workflow Güncelleme

Mevcut bir automation flow üzerinde değişiklik yapılmak istendiğinde:

- Node'lar düzenlenebilir
- Yeni bağlantılar eklenebilir
- Mevcut yapı güncellenebilir

Yapılan değişiklikler kaydedildikten sonra, workflow yeniden çalıştırılarak güncel yapı test edilebilir.

Bu sayede geliştirme süreci iteratif biçimde ilerler.

Yönetim Yaklaşımı

Workflow yönetiminde temel hedef:

- Kontrol edilebilirlik
- İzlenebilirlik
- Tekrarlanabilirlik

sağlamaktır.

Maistro, automation flow'ların tasarımından çalıştırılmasına kadar tüm süreci tek bir yapı altında yönetilebilir hale getirir.

Sonuç

Workflow yönetimi, oluşturulan otomasyonların sadece çalışmasını değil, sürdürülebilir biçimde işletilmesini sağlar.

Bu yapı sayesinde kullanıcılar:

- Akışları güvenle çalıştırabilir
- Çıktıları takip edebilir
- Süreci uçtan uca kontrol altında tutabilir

Bu noktada platform kullanımı tamamlanmış olur.

Bu sayfada

Sözlük

Bu bölümde Maistro platformunda kullanılan temel kavramların kısa ve net açıklamalarını bulabilirsiniz.

Agent

Bir hedefe ulaşmak için adım adım planlayan, karar veren, tool çağırabilen ve çıktı üreten orkestrasyon katmanıdır.

Agent; yalnızca cevap üretmez, süreci yönetir.

Session

Bir konuşmanın veya iş akışının bağlamını temsil eder.

Session içerisinde mesaj geçmişi, state bilgileri, ara sonuçlar ve tool çıktıları tutulur.

Tool

Agent'ın dış sistemlerle etkileşime geçmesini sağlayan fonksiyonel bileşendir.

API çağrıları, veri okuma/yazma işlemleri, workflow tetikleme gibi aksiyonlar tool'lar üzerinden gerçekleştirilir.

Automation (Flow)

Belirli bir iş sürecinin tanımlı adımlar halinde çalıştırıldığı yapıdır.

Agent'lar automation içinde görev alabilir veya automation tetikleyebilir.

RAG (Retrieval Augmented Generation)

Agent'ın cevap üretmeden önce dış bilgi kaynaklarından bağlam çekmesini sağlayan yaklaşımdır.

Bu sayede cevaplar daha güncel, doğrulanabilir ve kurumsal bilgiyle uyumlu hale gelir.

Model

Agent'ın cevap üretirken kullandığı büyük dil modelidir.

Model, yalnızca metin üretir; karar alma, yönlendirme ve aksiyon süreçleri agent tarafından yönetilir.

Prompt

Agent'ın nasıl konuşacağını, nasıl düşüneceğini ve hangi kurallar altında hareket edeceğini tanımlayan talimat setidir.

Prompt, agent davranışının temelini oluşturur.

Memory

Agent'ın konuşma geçmişini ve bağlam bilgisini nasıl saklayacağını belirleyen yapıdır.

Memory sayesinde agent önceki mesajları hatırlayabilir veya kontrollü şekilde unutabilir.

Tool Registry

Tool'ların merkezi olarak tanımlandığı, versiyonlandığı ve yönetildiği yapıdır.

Agent'lar ihtiyaç duydukları tool'ları runtime sırasında bu registry üzerinden çözümler.

System Prompt

Agent'ın tüm konuşma boyunca uyması gereken ana davranış kurallarıdır.

Kullanıcı mesajlarından bağımsız olarak her zaman aktif kalır.

Context

Agent'ın cevap üretirken kullandığı tüm bağlam bilgisidir.

Bu bağlam; kullanıcı mesajları, memory, tool çıktıları ve RAG sonuçlarından oluşabilir.

Manifest

Tool veya agent'a ait yapılandırma bilgisini tanımlayan dosyadır.

Parametre şemaları, versiyon bilgileri ve çalışma detayları manifest üzerinden belirlenir.

Versioning

Tool ve agent değişikliklerinin kontrollü şekilde yönetilmesini sağlayan sürümleme yaklaşımıdır.

Maistro, aynı bileşenin birden fazla versiyonunun birlikte çalışmasına olanak tanır.

HITL (Human in the Loop)

Kritik aksiyonlarda insan onayının sürece dahil edilmesini sağlayan mekanizmadır.

Özellikle veri güncelleme, ticket açma veya otomasyon tetikleme gibi işlemlerde kullanılır.

Workflow

Birden fazla adımın belirli bir sırayla çalıştığı otomasyon yapısıdır.

Agent, workflow tetikleyerek karmaşık süreçleri uçtan uca yürütebilir.

Policy

Agent davranışını sınırlayan veya yönlendiren kurallar bütünüdür.

Hangi tool'un ne zaman çağrılacağı, hangi durumda onay alınacağı policy ile belirlenir.

Runtime

Agent ve tool'ların gerçek çalıştığı yürütme ortamıdır.

Tool'lar ihtiyaç halinde runtime sırasında dinamik olarak yüklenir.

Orchestrator

Agent, tool, memory ve guardrails bileşenleri arasındaki akışı yöneten katmandır.

Tüm karar ve yönlendirme süreçleri bu katmanda gerçekleşir.

Latest

Bir tool için aktif olarak kullanılan versiyonu temsil eden işaretçidir.

Runtime, varsayılan olarak `latest` işaretli versiyonu kullanır.

Version Pinning

Tool'un belirli bir versiyona sabitlenmesini sağlayan mekanizmadır.

Bu sayede üretim ortamında beklenmeyen değişikliklerin önüne geçilir.

Fail Open / Fail Close

Tool veya kontrol mekanizması hata aldığında akışın devam edip etmeyeceğini belirleyen davranıştır.

Fail Open: Akış devam eder
Fail Close: Akış durdurulur
