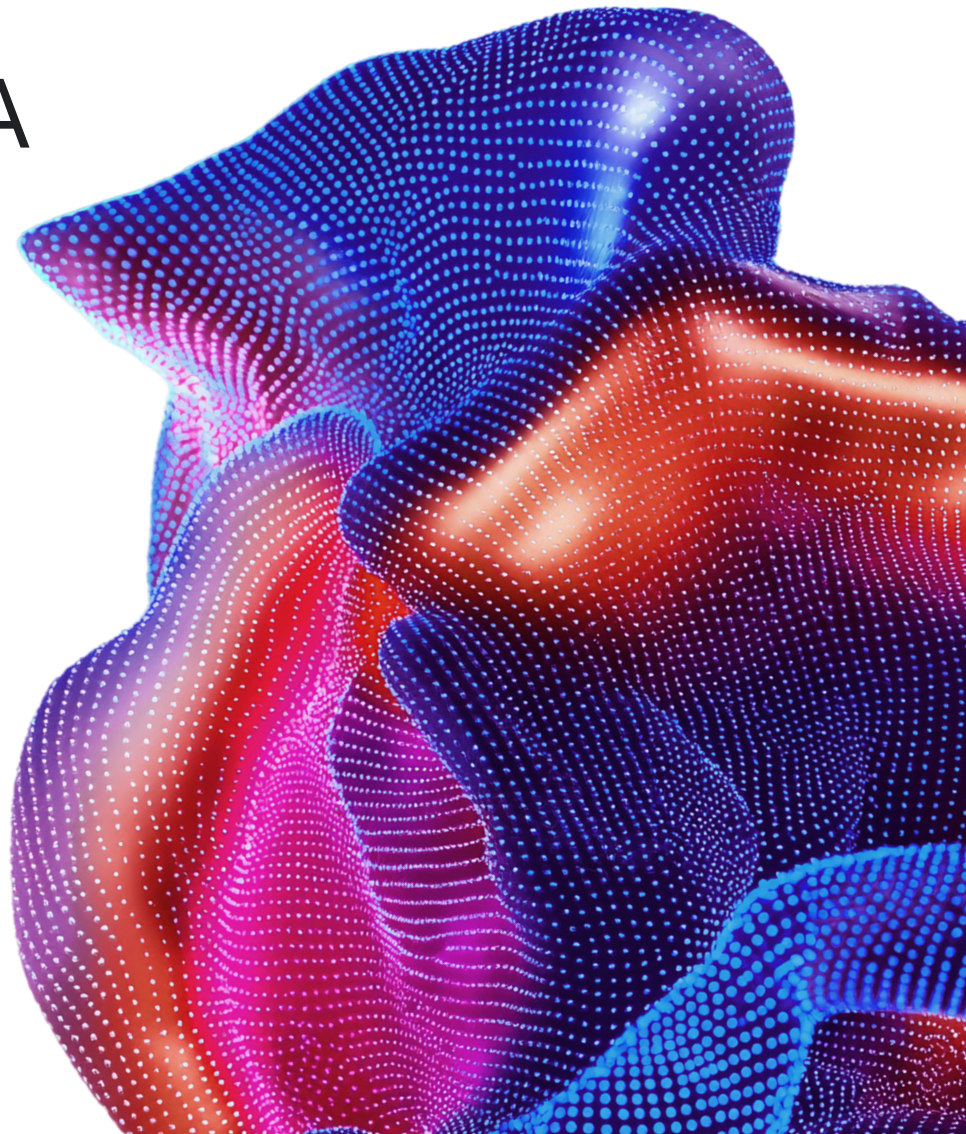
Google Cloud

Guia técnico para Startups

Agentes de IA



Índice

Introdução

O desenvolvimento dos agentes de IA representa uma mudança de paradigma na engenharia de software e permite que as startups automatizem fluxos de trabalho complexos, criem novas experiências para os usuários e resolvam problemas de negócios que antes eram tecnicamente impossíveis.

Mas, para transformar um protótipo promissor em um agente pronto para produção, é preciso enfrentar outra série de desafios. Como lidar com o comportamento não determinístico desses agentes? Como verificar suas complexas linhas de raciocínio? E, principalmente, por onde começar?

Este guia técnico ajuda a responder perguntas como essas. Ele fornece um roteiro estruturado, focado na prática, para orientar quem está entrando nesse novo universo, e foi pensado para ajudar as startups e os desenvolvedores que querem explorar rapidamente o potencial dos sistemas agênticos.

Você vai aprender os conceitos básicos sobre os sistemas agênticos, desde os principais componentes arquiteturais até os princípios que garantem uma operação confiável e responsável no ambiente de produção. Também vai conhecer o conjunto completo de ferramentas que tornam mais eficiente a criação e o uso de agentes no Google Cloud, como o desenvolvimento com código usando o Agent Development Kit (ADK) e a automação operacional com o Agent Starter Pack, além da criação de agentes sem programação de código com o Gemini Enterprise.

Este guia oferece assistência em todas as etapas do seu projeto, seja para validar uma ideia, construir um MVP ou acompanhar um produto em produção.

Como usar este guia

Não está familiarizado com agentes de IA?

Comece pela Seção 1 para entender os conceitos básicos.

Pronto para desenvolver?

Vá direto para a Seção 2 e crie seu primeiro agente com o ADK.

Já criou um agente?

Explore a Seção 3 para garantir a segurança, a estabilidade e a escalabilidade dele.

Precisa de uma força a mais?

Use o Gemini Kit para acelerar seus protótipos e inscreva-se no programa Google for Startups Cloud para receber orientação especializada e até 350 mil dólares em créditos de nuvem.

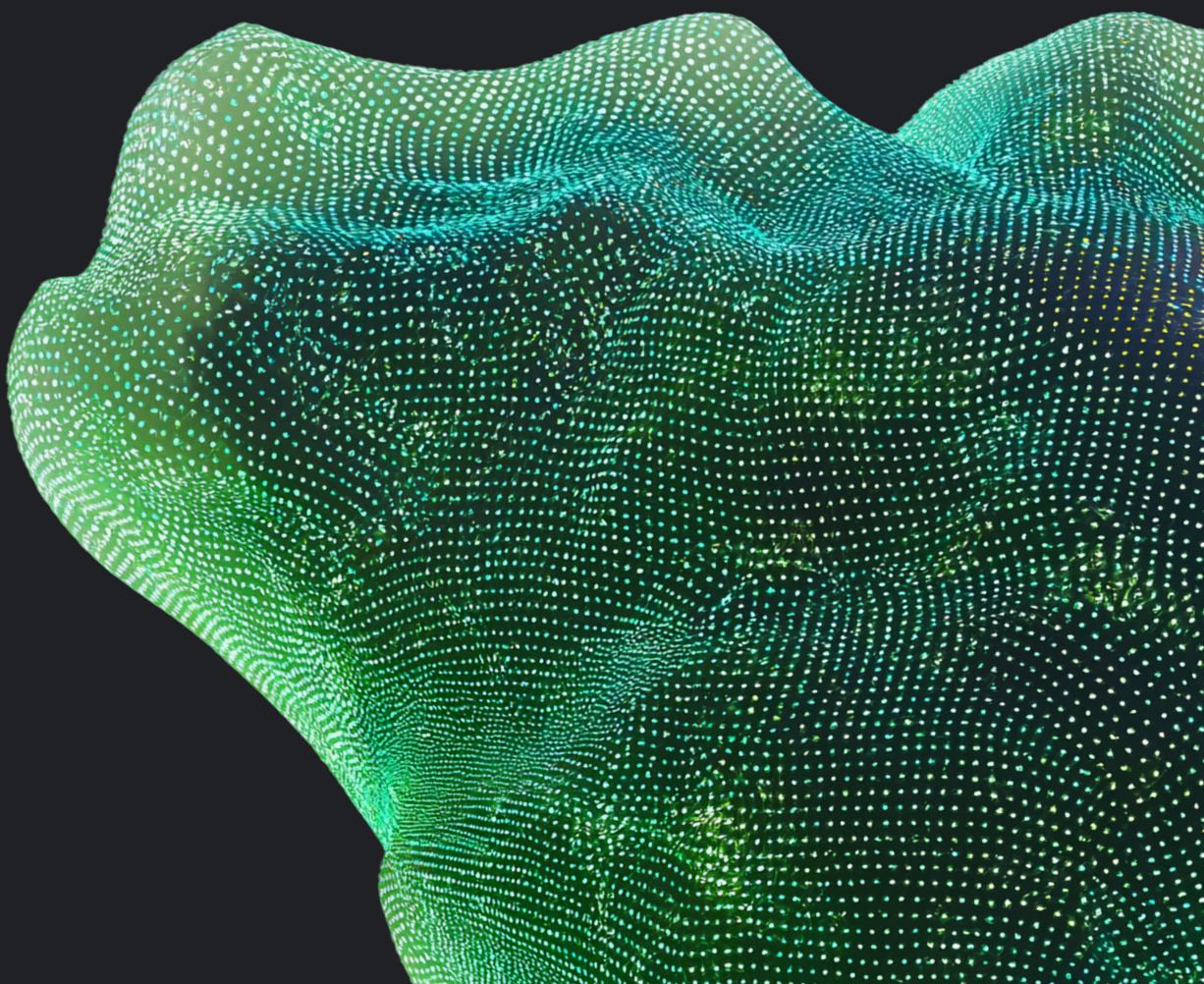
O foco deste guia

O ecossistema da IA baseada em agentes oferece uma série de ferramentas, bibliotecas e abordagens para desenvolver arquiteturas cognitivas. Existem frameworks de código aberto criados pelo Google, como o Genkit, além das soluções de IA conversacional do Google Cloud, e também bibliotecas amplamente utilizadas como LangChain e CrewAI.

O foco principal deste guia é apresentar o ADK, incluindo os conceitos e padrões arquiteturais que permitem construir agentes robustos e escaláveis no Google Cloud, sem abrir mão da flexibilidade para integrar outras ferramentas e bibliotecas de sua preferência.

Seção 1

Conceitos básicos dos agentes de IA



O campo da IA baseada em agentes está evoluindo rapidamente. Esta seção apresenta os fundamentos dos agentes de IA, explicando seus principais conceitos, objetivos e funcionamento. Ela também traz informações sobre as ferramentas e os serviços relevantes disponíveis no Google Cloud.

Prefere ouvir? Confira a versão em podcast desta seção, criada com o NotebookLM.

Este podcast foi criado com o NotebookLM a partir do seguinte comando: “Como apresentador, crie um podcast educativo e em tom de conversa sobre o ‘Guia técnico para startups: agentes de IA’, voltado para um público técnico de fundadores e desenvolvedores de startups. O episódio deve explorar as três principais formas de trabalhar com agentes de IA (desenvolver, aplicar e integrar), apresentando em detalhes ferramentas como o Agent Development Kit (ADK) e os agentes Gemini pré-configurados.

“Em seguida, explique os componentes centrais de um agente, incluindo modelos, ferramentas, orquestração e tempo de execução. Também aborde como garantir a confiança e o desempenho por meio de técnicas como grounding (embasamento) com Retrieval-Augmented Generation (RAG) e uso de multimodalidade. Finalize com um resumo dos principais insights e um convite direto para que os ouvintes explorem os recursos do Google”.

1.1 Uma visão geral do ecossistema de agentes do Google Cloud

> **O fluxo de trabalho baseado em agentes é o novo desafio. Não se trata apenas de fazer uma pergunta e receber uma resposta, mas de dar à IA um objetivo complexo — como 'planeje este lançamento de produto' ou 'resolva esta falha na cadeia de fornecimento' — e permitir que ela orquestre as etapas necessárias para alcançar esse resultado. Isso vai transformar a produtividade de forma profunda."**
>
> **Thomas Kurian**
> CEO do Google Cloud

Criar agentes de IA prontos para produção exige mais do que escolher um modelo de linguagem. Uma solução completa requer uma infraestrutura escalável, ferramentas robustas de integração de dados e padrões arquiteturais que atendam a diferentes exigências técnicas.

O Google Cloud oferece suporte ao desenvolvimento completo de sistemas agênticos, seja para criar seus próprios agentes, usar agentes prontos do Google Cloud ou integrar agentes de parceiros. Com base no Model Context Protocol (MCP) e no protocolo Agent2Agent (A2A), esse framework comum foi projetado para garantir a interoperabilidade. Assim, independentemente da origem ou da arquitetura, seus agentes podem colaborar dentro do ecossistema do Google Cloud.[1]

1. Os protocolos MCP e A2A são abordados em detalhes na seção 2 deste guia.

Desenvolva seus próprios agentes

Se você deseja criar agentes personalizados voltados para tarefas específicas, este é o caminho ideal. Há duas abordagens possíveis: uma orientada por código, para controle total, e outra baseada em aplicativo, para acelerar o desenvolvimento.

Agent Development Kit para um desenvolvimento personalizado orientado por código

Essa abordagem é indicada para desenvolvedores, startups técnicas e equipes que precisam de alto controle sobre o comportamento dos agentes. O Agent Development Kit (ADK) do Google Cloud foi criado justamente para esse tipo de desenvolvimento customizado.

O ADK oferece aos desenvolvedores recursos para construir, gerenciar, avaliar e implantar agentes com inteligência artificial. Ele proporciona um ambiente robusto e flexível para criar agentes conversacionais e não conversacionais, capazes de executar tarefas e fluxos de trabalho complexos.

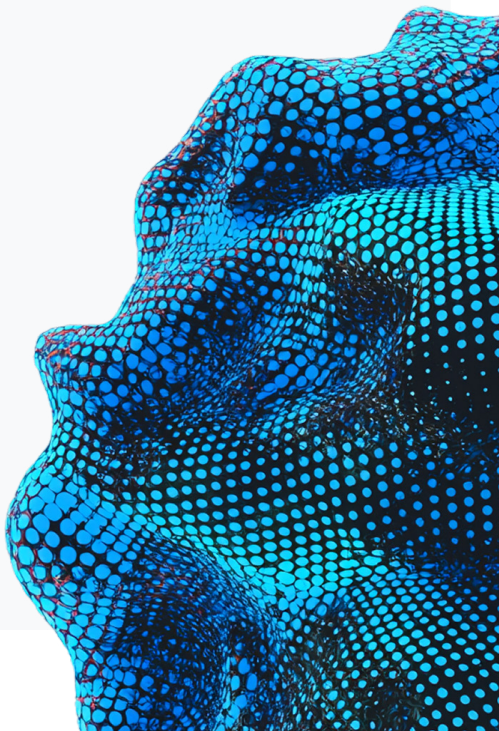
Os agentes desenvolvidos com o ADK podem ser facilmente implantados no Vertex AI Agent Engine, um ambiente gerenciado e escalável projetado especificamente para isso. Como são conteinerizados, esses agentes também podem ser executados em qualquer ambiente compatível com contêineres, como o Cloud Run e o Google Kubernetes Engine (GKE).

Principais recursos

- **Lógica de orquestração:** o processo central de raciocínio do agente, como o framework ReAct (veja a seção 1.2), permite que ele planeje e execute uma sequência de chamadas de ferramentas e ações para atingir um objetivo complexo.
- **Definição e registro de ferramentas:** uma interface para definir funções e APIs personalizadas, permitindo que o agente interaja com dados, APIs e sistemas externos.
- **Gerenciamento de contexto:** um sistema que fornece memória ao agente, permitindo que ele recorde as preferências do usuário e o histórico de conversas ao longo de múltiplas interações, oferecendo uma experiência mais coerente.
- Avaliação e observabilidade: um conjunto de ferramentas integradas para testar rigorosamente a qualidade do agente, depurar seu raciocínio passo a passo e monitorar seu desempenho em ambiente de produção.
- **Conteinerização**: a capacidade de empacotar o agente em um contêiner padrão e portátil, pronto para ser implantado em qualquer ambiente de nuvem compatível.
- **Composição multiagente:** a possibilidade de construir sistemas em que múltiplos agentes especializados colaboram, delegam tarefas e trabalham juntos para resolver um problema.

Por que isso é relevante para startups

- **Automatize, não apenas conversas, mas também fluxos de trabalho:** implemente a lógica de orquestração em várias etapas para resolver problemas complexos de negócios, criando a eficiência operacional que uma equipe reduzida precisa para crescer.
- **Construa um produto com vantagem competitiva:** conecte agentes diretamente às suas APIs proprietárias e dados internos para criar uma solução com diferenciais difíceis de replicar.
- **Lembre-se dos seus clientes para oferecer uma experiência realmente personalizada**: integre de forma fluida o contexto das conversas recentes com o conhecimento acumulado ao longo do tempo, permitindo que o agente recorde as interações passadas e construa um relacionamento genuíno com o cliente.
- **Faça o lançamento com confiança:** use ferramentas integradas de avaliação e observabilidade para testar e depurar seu agente com rigor, garantindo um produto confiável e pronto para produção.
- **Foque no produto, não na infraestrutura:** empacote seu agente em um contêiner padrão para tornar o processo até a produção mais rápido e estável, seguindo as práticas consolidadas de DevOps.



Gemini Enterprise para um desenvolvimento orientado por aplicativo

A segunda principal maneira de construir agentes é usando o Gemini Enterprise. Diferente do ADK, orientado por código, o Gemini Enterprise permite orquestrar toda sua força de trabalho de IA e capacitar profissionais não técnicos da equipe a criar agentes personalizados com um designer sem código.

Essa abordagem por plataforma é ideal para gerenciar vários agentes e escalar seu uso conforme a startup cresce e seu portfólio de aplicativos SaaS se expande.

Principais recursos

- **Busca unificada em toda a empresa:** conecta-se e realiza buscas em diversos aplicativos SaaS.
- **Síntese multimodal de dados:** compreende e integra informações de texto, imagens, gráficos e vídeos, respeitando as permissões de dados.
- **Biblioteca de agentes pré-configurados:** oferece um conjunto de agentes prontos para uso em tarefas complexas, como pesquisas aprofundadas ou geração de ideias.
- **Construtor de agentes personalizado sem código:** inclui o Agent Designer, que permite que usuários não técnicos criem agentes por meio de uma interface guiada por comandos.

Por que isso é relevante para startups

- **Elimine silos de dados:** equipes sem desenvolvedores podem criar e implantar agentes capazes de acessar e agir sobre fontes e aplicativos de dados fragmentados.
- **Automatize fluxos de trabalho:** crie processos entre plataformas sem sobrecarregar os recursos escassos de engenharia, liberando sua equipe técnica para focar no desenvolvimento do produto principal.

> “Tornar o Gemini um modelo mundial é um passo crítico no desenvolvimento de um novo tipo de IA, mais geral e mais útil, um assistente universal de IA. Trata-se de uma IA inteligente, que entende o contexto em que você está e que pode planejar e agir em seu nome, em qualquer dispositivo”.
>
> **Demis Hassabis**
> CEO do Google DeepMind

Use agentes do Google Cloud

Com prototipagem rápida e formas fáceis de integrar IA aos seus aplicativos existentes, os agentes gerenciados permitem que você se concentre na lógica principal do negócio, em vez de gerenciar infraestrutura. Eles também são ideais se seus recursos de engenharia forem limitados.

Gemini Code Assist

O Gemini Code Assist é um assistente com tecnologia de IA para desenvolvedores. Ele se integra a vários pontos do ciclo de vida do desenvolvimento de software, oferecendo suporte por meio de extensões de IDE, interface de linha de comando, integração com o GitHub e dentro de diversos serviços do Google Cloud.

Principais recursos

- **Integração com IDEs:** nos ambientes de desenvolvimento integrados populares (VS Code, IDEs da JetBrains, Android Studio), ele oferece preenchimento automático de código, geração de funções sob demanda e uma interface de chat. Ele usa a ampla janela de contexto do Gemini para fornecer respostas relevantes para a base de código aberto. As edições empresariais podem ser conectadas a repositórios privados de código-fonte para fornecer sugestões mais personalizadas.
- **Interface de linha de comando:** o Gemini CLI é um agente de IA de código aberto que leva as capacidades do Gemini direto para o terminal para realizar tarefas como compreensão de código, manipulação de arquivos e solução de problemas dinâmica.
- **Integração com GitHub:** no GitHub, o Gemini Code Assist pode revisar automaticamente pull requests para identificar bugs e problemas de estilo, sugerindo alterações específicas no código.
- **Desenvolvimento orientado por agentes:** ele implanta agentes de IA capazes de realizar edições complexas em vários arquivos dentro do contexto completo de um projeto. Esses fluxos de trabalho agênticos incorporam a supervisão humana (HITL) e podem se integrar com ferramentas do ecossistema que seguem o MCP.
- **Integração com serviços do Google Cloud:** ele oferece assistência de IA diretamente em serviços como Firebase (análise de erros de aplicativos, insights de desempenho), Colab Enterprise (geração de código Python), BigQuery (de linguagem natural para SQL, otimização de consultas), Cloud Run e Apigee.optimization), Cloud Run, and Apigee.

Por que isso é relevante para startups

O Gemini Code Assist atua como um multiplicador de força. Ele pode lidar com tarefas de desenvolvimento de software ao longo de todo o ciclo de vida, desde tarefas rotineiras como escrever código padrão até operações mais complexas como refatoração envolvendo vários arquivos.

Você pode delegar uma ampla variedade de tarefas ao Gemini Code Assist. A seguir, estão alguns exemplos que demonstram seus recursos.

- **Para automatizar código padrão:** gere uma Cloud Function em Python que seja disparada por uma requisição HTTP. Ela deve analisar um payload JSON com userId e documentId, usar a biblioteca cliente google-cloud-firestore para buscar um documento específico na coleção “users” e retorná-lo como resposta JSON.

- **Para testes abrangentes:** forneça uma de suas funções existentes e peça ao Code Assist para gerar uma suíte de testes completa, incluindo as maquetes necessárias para serviços do Google Cloud como Cloud Storage ou Firestore.

- **Para refatoração em larga escala com Gemini:** solicite que ele analise vários serviços no seu código e proponha um plano estratégico. Por exemplo: “Dado nosso ‘user-service’ e ‘auth-service’, proponha um plano passo a passo para refatorar a lógica de autenticação em uma única biblioteca compartilhada, detalhando os prós e os contras dessa abordagem”.

Gemini Cloud Assist

O Gemini Cloud Assist é um especialista em IA para seu ambiente do Google Cloud, oferecendo assistência contextual para gerenciamento de infraestrutura e operações de aplicativos. Ele usa as informações do seu projeto, como IDs de projeto do Google Cloud e a página de produto visualizada no console, para adaptar seu suporte.[2]

Principais recursos

- **Design e implantação:** no Application Design Center, você pode descrever os resultados desejados da infraestrutura em linguagem natural. O Gemini Cloud Assist gera diagramas de arquitetura e modelos de aplicativos, que podem ser exportados como Terraform para integração com fluxos de trabalho existentes de infraestrutura como código (IaC).

- **Diagnóstico e resolução:** integra-se ao Cloud Observability para resumir entradas de log complexas e explicar mensagens de erro. Para problemas mais profundos, é possível iniciar investigações, nas quais o Gemini analisa logs e métricas para identificar a causa raiz.

- **Configuração e otimização:** oferece recomendações personalizadas de custo e uso dentro do FinOps Hub e do painel de Otimização de Custos.

- **Segurança e análise:** permite a investigação em linguagem natural sobre fluxos de rede e logs. Fornece orientações sobre tarefas de segurança como criptografia de dados, gerenciamento de segredos e criação ou teste de políticas organizacionais personalizadas. Também pode recomendar funções IAM e diagnosticar erros de permissão.

Por que isso é relevante para startups

- **Ganhe tempo:** o gerenciamento da nuvem pode consumir muito tempo da equipe de engenharia. O Gemini Cloud Assist libera você para se concentrar na criação do seu produto.

Experimente estes prompts no Gemini Cloud Assist

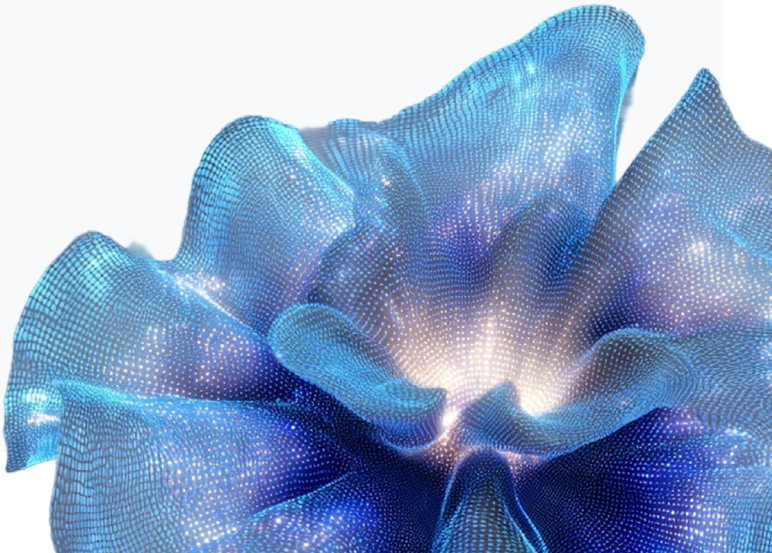
> Como uso a Vertex AI para implantar um modelo?

> Crie um plano de alto nível para projetar, construir e implantar um aplicativo web no Google Cloud.

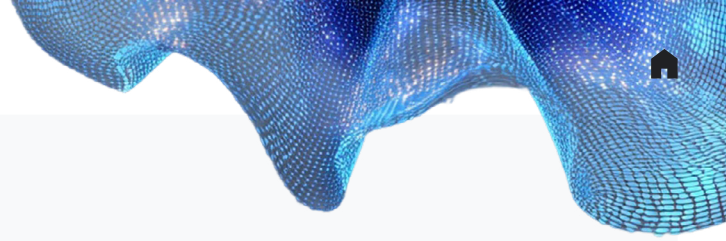
> Liste todos os buckets do Cloud Storage no projeto prod-v1 que não estão habilitados para controle de versão de objetos.

> Quais são as regras de firewall públicas aplicadas às instâncias com a tag external-web-server?

> Mostre todas as funções IAM concedidas à conta de serviço data-pipeline@my-project.iam.gserviceaccount.com



2. Para detalhes sobre o grounding do Gemini Cloud Assist, consulte a documentação oficial.



Gemini no Colab Enterprise

Se a sua startup atua em ciência de dados, machine learning ou análise de dados, o Gemini no Colab Enterprise transforma cada notebook em um espaço colaborativo com IA. Ele foi desenvolvido para gerar, explicar e depurar código Python com base no contexto.

Principais recursos

- Completar e gerar código Python automaticamente dentro do Colab
- Explicar a lógica do código e erros em linguagem simples
- Filtrar, transformar e visualizar dados
- Recomendar conjuntos de dados públicos e recursos de pesquisa
- Resumir notebooks inteiros ou células de código

Experimente estes prompts no Gemini usando o Colab Enterprise:

> Como faço para filtrar um DataFrame do Pandas?
>
> Faça um gráfico da receita média por região.
>
> Forneça uma lista de conjuntos de dados disponíveis publicamente para tecnologia climática.
>
> Resuma o objetivo deste notebook.

Por que isso é relevante para startups

- **Acelere pesquisa e desenvolvimento:** automatize os aspectos mais repetitivos da preparação, análise e visualização de dados, permitindo que os desenvolvedores iterem novos modelos e ideias muito mais rapidamente.
- **Reduza a barreira de entrada:** engenheiros iniciantes em ciência de dados podem começar com mais facilidade, enquanto profissionais experientes podem focar mais na experimentação de modelos e menos na manipulação de dados.

Integre agentes de parceiros

Se o seu caso de uso for mais especializado, você pode integrar facilmente agentes de terceiros ou de código aberto ao seu stack usando o ecossistema aberto do Google Cloud e o Google Cloud Marketplace.

Explore o Agent Garden para implantar agentes do ADK pré-construídos que já oferecem suporte a raciocínio sobre dados e colaboração entre agentes. Você pode combiná-los com os agentes que desenvolver, acelerando o tempo de impacto.

1.2 Principais componentes de todo agente

Modelos: seleção e ajuste

Pense no modelo como o cérebro do seu agente. Ele é usado para interpretar solicitações do usuário, entender o que precisa ser feito e gerar respostas inteligentes.

Como escolher o modelo certo

Escolher o modelo ideal não significa optar pelo mais poderoso disponível, mas sim encontrar o equilíbrio ideal entre capacidade, velocidade e custo para o seu caso de uso. Todo modelo pode ser avaliado com base nessas três características conflitantes, e o objetivo é identificar a opção mais eficiente para uma tarefa específica.

À medida que a capacidade de um modelo aumenta, seu custo e latência geralmente também aumentam. O erro mais comum é investir demais em capacidade quando o caso de uso não exige isso, resultando em gastos desnecessários e desempenho mais lento. A estratégia ideal é selecionar o modelo mais eficiente para cada tarefa.

Esse princípio é ainda mais importante quando aplicado no nível do sistema. Arquiteturas cognitivas robustas utilizam vários agentes especializados, cada um selecionando dinamicamente o modelo mais enxuto para sua subtarefa específica. Isso garante, por exemplo, que um modelo mais pesado seja reservado para raciocínios complexos, enquanto um modelo leve lida com consultas rotineiras. Essa abordagem multiagente oferece flexibilidade arquitetural para otimizar o custo e o desempenho de todo o sistema, e não apenas de um componente isolado.

> “Agentes de IA são sistemas que combinam a inteligência de modelos avançados com acesso a ferramentas, para que possam agir em seu nome, sob seu controle”.
>
> **Sundar Pichai**
> CEO da Google e Alphabet

Casos de uso

Prototipagem inicial e tarefas em larga escala

- **Perfil do modelo:** um modelo leve e de baixo custo como o Gemini 2.5 Flash-Lite.
- **Justificativa:** é o modelo 2.5 mais rápido e econômico, ideal para tarefas de grande volume e sensíveis à latência, como tradução e classificação.

Aplicativos de alto volume e alta qualidade

- **Perfil do modelo:** um modelo equilibrado de médio porte como o Gemini 2.5 Flash.
- **Justificativa:** foi projetado para equilibrar qualidade, custo e velocidade. Oferece desempenho sólido em tarefas complexas por um custo menor que o Pro, sendo perfeito para aplicativos em produção que exigem inteligência e economia.

Raciocínio complexo, com múltiplas etapas, e geração de código avançada

- **Perfil do modelo:** um modelo de raciocínio avançado como o Gemini 2.5 Pro.
- **Justificativa:** é o modelo mais capacitado para tarefas difíceis, onde o desempenho é essencial e não pode ser comprometido.[3]

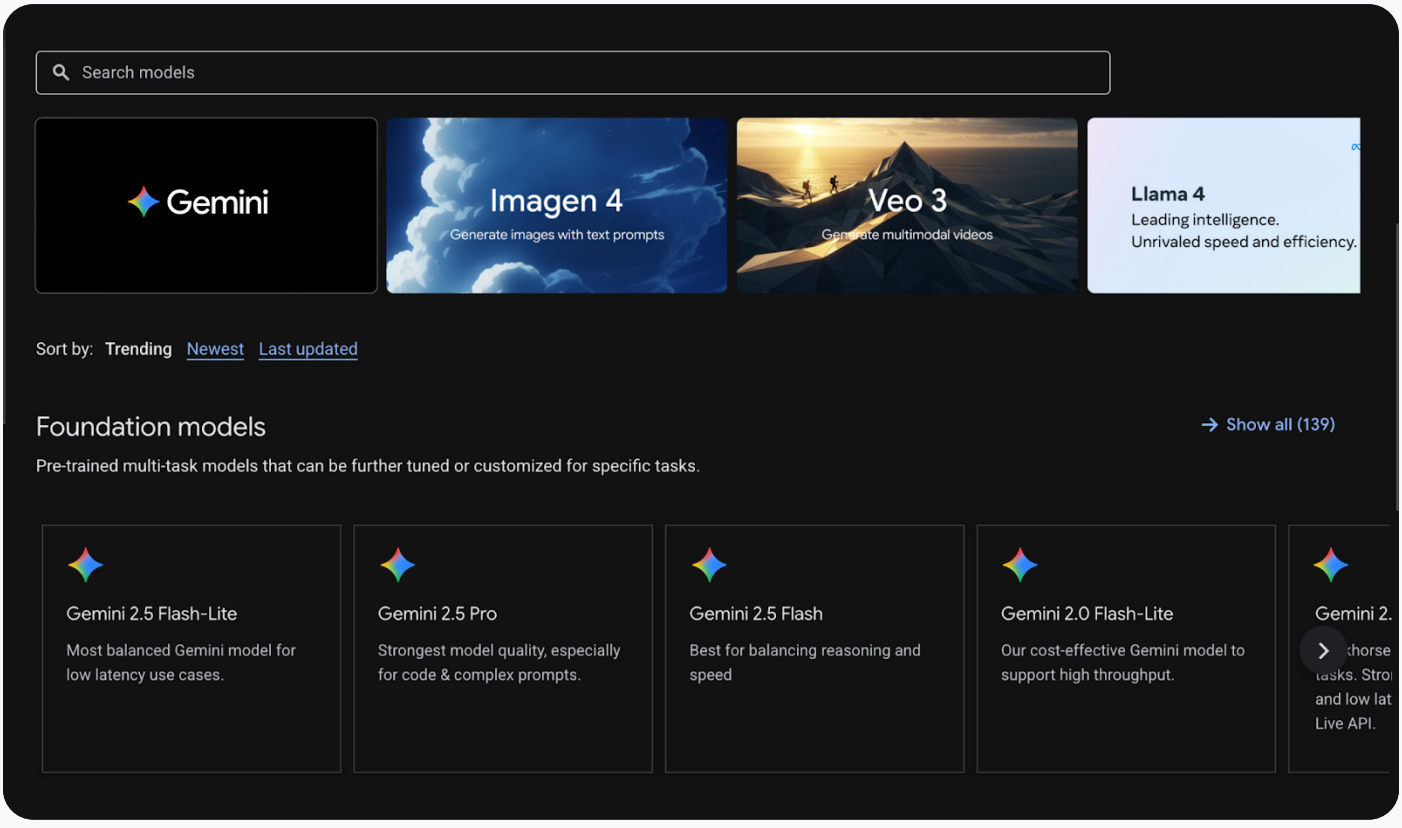
3. O Gemini 2.5 Pro alcançou resultados de última geração em benchmarks avançados: 82,2% no Aider Polyglot (programação de código) e 86,4% no GPQA diamond (raciocínio). Os dados são de junho de 2025.

Você pode usar a família de modelos Gemini 2.5 para decompor problemas, formular planos e utilizar ferramentas. Esse processo de raciocínio é configurável. Ao alocar mais tokens de raciocínio em uma chamada específica, o desenvolvedor orienta o modelo a aplicar mais esforço computacional, trocando deliberadamente um aumento previsível de latência e custo por uma possível melhoria na precisão.

Esse controle por token, combinado com a seleção de modelos e modos de raciocínio configuráveis, oferece aos desenvolvedores um conjunto dinâmico de ferramentas para otimização avançada. O custo e o desempenho de um sistema multiagente completo podem ser ajustados para atender a requisitos técnicos e comerciais específicos.

Dica

Use o Model Garden no Vertex AI para descobrir, personalizar e implantar modelos fundacionais a partir de uma plataforma centralizada. Ele oferece uma seleção personalizada de mais de 200 modelos do Google, de parceiros como Anthropic e diversos modelos abertos de provedores como a Meta (família Llama) e Mistral. Em vez de gerenciar a infraestrutura manualmente, você pode implantar os modelos em aplicativos com um clique e escalar esses modelos usando recursos integrados de MLOps de ponta a ponta.



Ajuste do modelo

Depois de escolher um modelo que atenda às suas necessidades de custo, latência e qualidade, é possível ajustá-lo. Você especializa o conhecimento e o estilo do modelo para os objetivos específicos do seu negócio, usando um conjunto de dados selecionado com exemplos de alta qualidade.

A disponibilidade do ajuste depende de cada modelo. No portfólio do Google, essa funcionalidade é compatível com a família Gemma (modelos com parâmetros abertos) e com versões específicas do Gemini. É essencial revisar a documentação e o contrato de licença de cada modelo para confirmar se o ajuste é permitido e tecnicamente viável.

Dica

Para saber quais modelos podem ser ajustados na Vertex AI, consulte a documentação oficial.

Caso de uso

Ajuste fino de um agente de suporte

Imagine que você esteja criando um agente de atendimento ao cliente para seu produto SaaS. Você pode fazer o ajuste fino com um conjunto de dados contendo milhares de tickets de suporte anteriores e suas soluções ideais, ajudando o modelo a aprender sobre problemas comuns e a responder com um estilo alinhado ao tom da sua equipe de suporte.

Observação

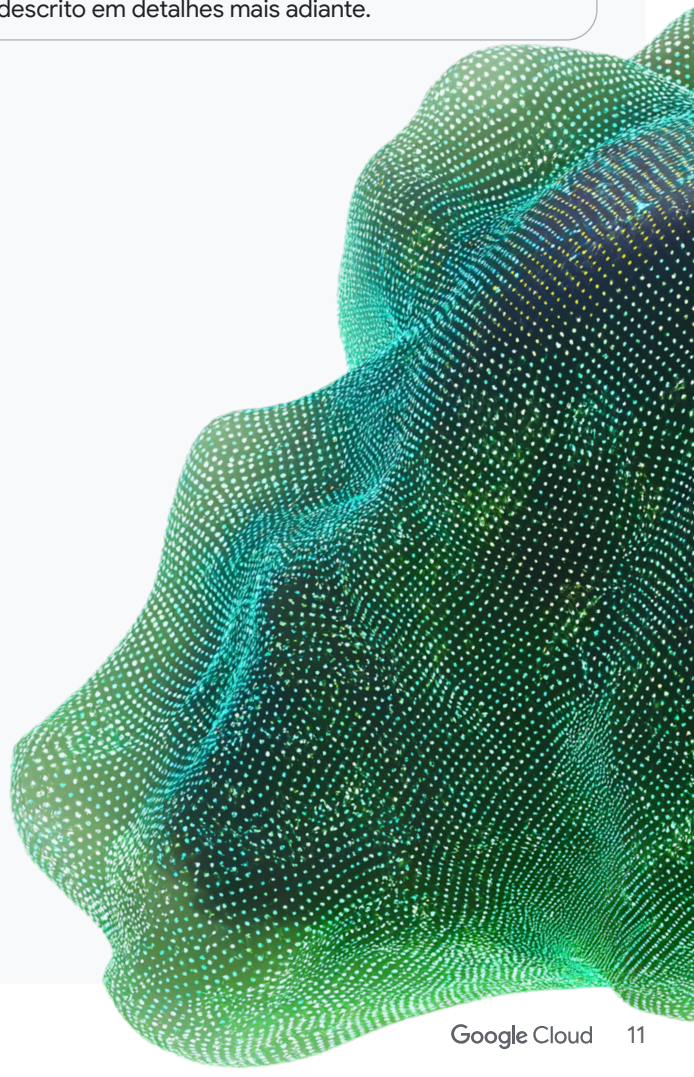
Ajuste fino não é grounding.
O ajuste fino adapta o estilo do modelo e refina seu conhecimento para uma tarefa específica. Já o grounding (ou embasamento) conecta o modelo a fontes de dados verificáveis e em tempo real, garantindo que suas respostas sejam factualmente corretas. O grounding será descrito em detalhes mais adiante.

Ferramentas: permitir as ações dos agentes

As ferramentas são capacidades definidas que permitem ao agente ir além das funções nativas do seu modelo de raciocínio. Elas variam desde cálculos internos simples até interações com sistemas externos via chamadas de API. As ferramentas fazem a ponte entre o raciocínio do agente e sua capacidade de obter novas informações ou executar operações com estado.

As ferramentas podem incluir uma ampla variedade de componentes:

- **Funções e serviços internos:** lógica proprietária desenvolvida pela sua própria equipe.
- **APIs:** conexões com serviços internos e também com serviços externos de terceiros.
- **Fontes de dados:** capacidade de consultar bancos de dados, repositórios vetoriais ou outros depósitos de informação.
- **Outros agentes:** em sistemas multiagentes, um agente pode utilizar outro agente especializado como ferramenta.



Arquitetura de dados para sistemas com agentes

Os dados são a base da memória de curto e longo prazo de um agente. Uma arquitetura de dados robusta precisa atender a três necessidades distintas: armazenamento persistente para recuperação de conhecimento de longo prazo, acesso de baixa latência para contexto conversacional de curto prazo e registro durável para auditoria transacional. Ao mapear serviços específicos do Google Cloud para cada uma dessas necessidades, você garante que cada decisão arquitetônica seja escalável e econômica, atendendo à demanda imediata dos negócios e ao prazo de entrega.

1. Base de conhecimento de longo prazo (grounding e recuperação)

A memória de longo prazo de um agente é a base para sua inteligência, seu grounding e sua personalização. Ela é distinta do contexto rápido e de curto prazo de uma conversa ao vivo. Uma arquitetura robusta de memória de longo prazo deve incluir três componentes principais: uma base de conhecimento estruturada para o grounding baseada em fatos por meio de geração aumentada por recuperação (RAG), um armazenamento persistente para o histórico de interações com o usuário, permitindo uma experiência contínua e personalizada, e um data lake operacional para materiais brutos como transcrição de conversas e estados de fluxo de trabalho, voltado para processos cognitivos mais complexos e análises futuras.

Serviço de dados	Visão geral	Casos de uso para startups
Vertex AI Search	Um serviço gerenciado para criação de aplicativos de busca vetorial de alto desempenho. É a principal ferramenta para permitir compreensão semântica e recuperação de informações em grandes conjuntos de dados não estruturados.	Encontrar instantaneamente respostas dentro da documentação interna do produto, registros de atendimento ao cliente e postagens em fóruns da comunidade, permitindo que o agente ofereça suporte preciso e contextualizado a novos usuários. Isso reduz a sobrecarga da pequena equipe de suporte.
Firestore	Um banco de dados de documentos NoSQL serverless com capacidade de sincronização em tempo real. Seu modelo de dados hierárquico e flexível é ideal para armazenar contexto estruturado e o estado dinâmico de tarefas ativas de longa duração ou persistentes de um agente.	Manter o estado em tempo real de um fluxo de integração de usuário guiado por agente e com múltiplas etapas. À medida que o usuário conclui cada etapa (por exemplo, "criar perfil", "conectar API", "convidar membro da equipe"), o agente atualiza um documento no Firestore. Os desenvolvedores podem então acompanhar o progresso das tarefas do agente em tempo real, e o usuário pode retomar o processo de forma contínua entre sessões.
Vertex AI Memory Bank (prévia)	Um serviço gerenciado do Vertex AI Agent Engine, projetado especificamente para gerar, armazenar e recuperar dinamicamente memórias de longo prazo a partir de conversas com usuários.	Em vez de construir manualmente uma lógica para extrair preferências do usuário, o agente pode chamar automaticamente a função GenerateMemories sobre o histórico da conversa. Isso extrai de forma assíncrona fatos-chave (por exemplo, "o usuário prefere voos diretos", "o cachorro do usuário se chama Fido") e os armazena. Em sessões futuras, o agente pode recuperar essas memórias por meio de busca por similaridade, oferecendo uma experiência bem personalizada e contínua — com o mínimo de código personalizado.
Cloud Storage	Um armazenamento de objetos altamente escalável e durável para dados brutos e não estruturados (como PDFs, imagens e vídeos), que alimenta outros serviços para indexação e processamento.	Ele funciona como um repositório inicial econômico e persistente para todos os documentos enviados pelos usuários, imagens de relatórios de erros ou gravações de chamadas com feedback de clientes. Esses dados brutos são então processados e indexados por serviços como o Vertex AI Search, enriquecendo o conhecimento do seu agente.
BigQuery	Um data warehouse totalmente gerenciado e serverless para armazenar e analisar grandes volumes de dados estruturados e semiestruturados, permitindo que agentes sejam equipados com ferramentas capazes de executar consultas analíticas complexas.	Um agente pode requisitar coisas como: "Resuma os padrões de engajamento dos usuários com o novo recurso que lançamos na semana passada" ou "Quais grupos de clientes apresentam maior risco de cancelamento com base na atividade recente?" O BigQuery oferece business intelligence instantâneo.

2. Memória de trabalho (contexto conversacional e estado de curto prazo)

Essa camada gerencia as informações transitórias necessárias para uma tarefa ou conversa em andamento.
Ela precisa oferecer acesso com latência extremamente baixa para garantir uma experiência responsiva ao usuário.

Serviço de dados	Visão geral	Casos de uso para startups
Memorystore	Um armazenamento de dados totalmente gerenciado e na memória, com latência inferior a um milissegundo. É ideal para cache de dados acessados com frequência e para gerenciamento de estado de sessão.	Sua principal função é fazer o cache em alta velocidade, armazenando os resultados de operações computacionais de alto custo ou com alta latência. Em vez de executar repetidamente tarefas onerosas (como chamadas de API de modelos de linguagem, consultas complexas a bancos de dados ou chamadas a serviços de terceiros), o agente verifica primeiro o Memorystore em busca de um resultado armazenado. Isso reduz drasticamente tanto a latência de resposta quanto os custos operacionais recorrentes, fatores essenciais para o sistema agêntico de uma startup.

3. Memória transacional (gerenciamento de estado e auditoria de ações)

Essa camada é responsável por registrar ações e mudanças de estado com muita consistência e integridade.
Ela funciona como o sistema oficial de registro, frequentemente exigindo garantias ACID para assegurar a confiabilidade.

Serviço de dados	Visão geral	Casos de uso para startups
Cloud SQL	Um serviço totalmente gerenciado para bancos de dados relacionais tradicionais, que oferece forte consistência para cargas de trabalho transacionais em uma única região. É a escolha padrão para um gerenciamento de estado confiável.	Quando um agente executa com sucesso uma ação crítica de negócio (como processar um pagamento de assinatura ou provisionar um novo serviço para um usuário via chamada de API), ele registra essa ação em um banco de dados Cloud SQL. Isso cria um log de auditoria permanente e compatível com ACID, garantindo que cada ação importante realizada pelo agente seja rastreável e verificável com confiabilidade.
Cloud Spanner	Um banco de dados relacional distribuído globalmente, com forte consistência e escalabilidade horizontal. Ele foi projetado para aplicativos críticos que exigem alta disponibilidade e integridade transacional em múltiplas regiões geográficas.	Uma startup normalmente migra do Cloud SQL para o Spanner apenas depois que seu produto já está bem estabelecido no mercado e começa a atrair usuários de diferentes partes do mundo. Por exemplo, um aplicativo de viagens ou de comércio eletrônico que inicialmente usava Cloud SQL agora precisa processar reservas ou pedidos de usuários na América do Norte, Europa e Ásia simultaneamente, sem conflitos de dados. A consistência transacional global do Spanner oferece suporte a essa escala.

Orquestração de agentes: a função executiva

A orquestração é o núcleo operacional que orienta um agente na execução de tarefas com diversas etapas. Para qualquer processo que exija mais de uma ação, ela determina quais ferramentas são necessárias, em que sequência devem ser usadas e como os dados que elas geram devem ser combinados para alcançar o objetivo final.

Como função executiva do agente, a orquestração pode ser aplicada para assumir a responsabilidade de planejamento e de tomada de decisão. Ao automatizar processos de negócios complexos, ela potencializa bastante a eficiência das equipes enxutas de startups.

Conceitos de orquestração e arquitetura cognitiva

Um padrão comum e eficaz de orquestração é o **ReAct (Raciocínio + Ação)**, uma estrutura que combina as capacidades de raciocínio e ação dos modelos de linguagem de grande escala.[4]

O ReAct estabelece um ciclo dinâmico de múltiplas etapas, em que o modelo gera tanto rastros de raciocínio (pensamentos) quanto ações específicas para a tarefa, de forma intercalada. Isso cria uma sinergia maior — o raciocínio ajuda o modelo a acompanhar e atualizar os planos de ação, enquanto as ações coletam informações de ferramentas externas para alimentar o processo de raciocínio.

Veja como funciona:

1. **Raciocínio:** o agente avalia o objetivo e o estado atual, formando uma hipótese sobre o melhor passo a seguir e se é necessário usar alguma ferramenta.
2. **Ação:** o agente escolhe e aciona a ferramenta apropriada.
3. **Observação:** o agente recebe os dados gerados pela ferramenta. Essa nova informação é integrada ao contexto do agente e alimenta a próxima etapa de raciocínio do ciclo.

4. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. Apresentado como artigo na conferência ICLR 2023.

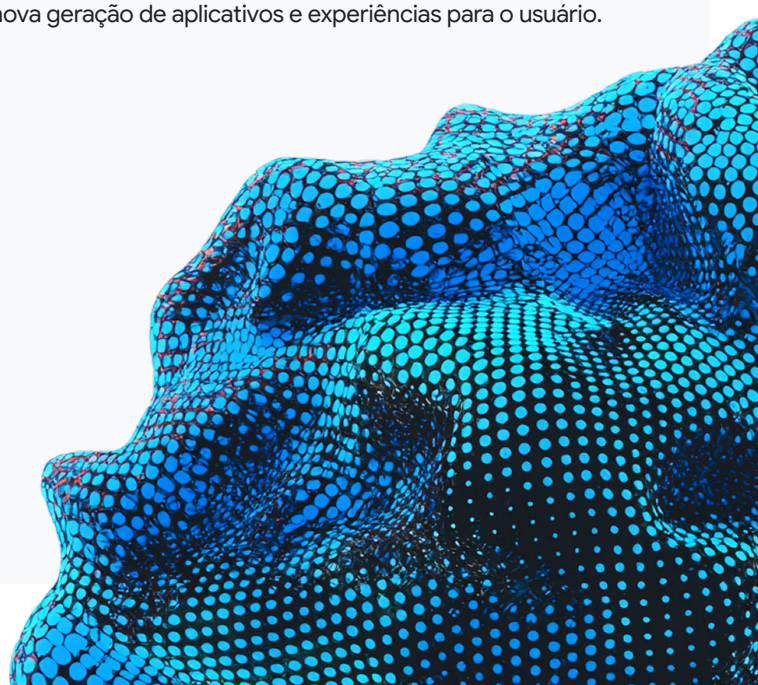
Exemplo: Processar um reembolso com orquestração ReAct

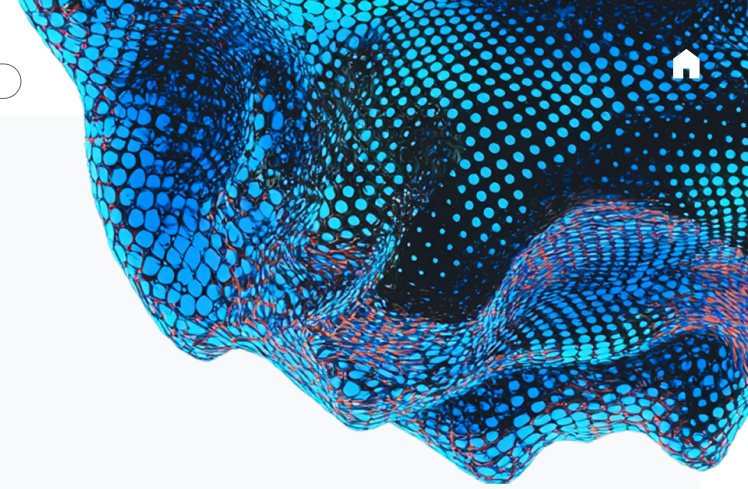
- **Raciocínio:** o usuário quer um reembolso. O primeiro passo é entender as regras da empresa para reembolsos.
- **Ação:** use a ferramenta semantic_search para consultar a base de conhecimento interna com o termo “política de reembolso”.
- **Observação:** a ferramenta retorna: “Reembolsos integrais estão disponíveis para todos os produtos dentro de 30 dias após a data da compra”.
- **Raciocínio:** a política exige a data da compra. Essa informação, referente ao pedido específico do usuário, deve vir do sistema de CRM.
- **Ação:** Chame a função get_order_details da ferramenta CRM com o ID do usuário.
- **Observação**: a ferramenta retorna um objeto de pedido, incluindo purchase_date: ‘2025-07-20’.
- **Raciocínio:** a data atual é 29 de julho de 2025. A compra foi feita há 9 dias, dentro do prazo de 30 dias. Os critérios foram atendidos e o reembolso pode ser iniciado.
- **Ação:** chame a ferramenta process_refund com o ID do pedido e o valor do reembolso.
- **Observação:** a ferramenta retorna status: ‘success’.
- **Resposta final:** “Seu reembolso foi processado com sucesso. O valor deve aparecer na sua conta dentro de 3 a 5 dias úteis”.

Casos de uso

- **Onboarding automatizado de clientes:** um agente pode ser orquestrado para guiar um novo usuário na configuração inicial. Ele pode começar criando uma nova conta via API, depois usar uma ferramenta send_email para enviar uma mensagem de boas-vindas por e-mail e, por fim, verificar no banco de dados se o usuário realizou sua primeira ação, disparando um lembrete caso não o tenha feito.
- **Monitoramento proativo e correção de sistemas:** a orquestração pode ser ativada por um alerta de monitoramento. Primeiro, o agente busca mais contexto consultando os logs do Cloud Logging. Com base nos registros, pode decidir reiniciar um pod específico no GKE usando a ferramenta kubectl para reiniciar um pod específico no GKE e, por fim, usar uma ferramenta slack_notification para enviar uma notificação via Slack para o canal de plantão.
- **Qualificação complexa de leads:** um agente de vendas pode ser orquestrado para aprimorar o e-mail de um novo lead com dados da empresa via API. Ele consulta o CRM interno para verificar se o lead já é cliente. Com essas informações, ele decide se deve encaminhar o lead para um representante sênior ou adicioná-lo a uma campanha de e-mails de engajamento.

É essencial dominar a orquestração para ir além de agentes simples que executam uma única tarefa. Quando isso é feito corretamente, é possível criar sistemas sofisticados e autônomos capazes de resolver problemas que antes não eram tecnicamente viáveis. Isso abre caminho para uma nova geração de aplicativos e experiências para o usuário.





Tempo de execução: implantar agentes em larga escala

A implantação de um protótipo funcional de agente em um ambiente de produção exige uma infraestrutura robusta de tempo de execução. Essa infraestrutura permite que o agente seja implantado em larga escala, transformando o protótipo em um produto confiável que lida com exigências operacionais complexas, como segurança, balanceamento de carga e tratamento de erros, especialmente quando há um aumento imprevisível de usuários.

Conceitos e arquitetura de tempo de execução

Um ambiente de tempo de execução de nível profissional para agentes de IA precisa oferecer alguns recursos fundamentais:

- **Escalabilidade:** a infraestrutura deve ser escalada automaticamente para lidar com cargas variáveis, desde zero até milhões de requisições. Isso inclui balanceamento de carga baseado em requisições e escalada automática de recursos para gerenciar a demanda computacional com eficiência.
- **Segurança:** o tempo de execução deve garantir um ambiente seguro, gerenciando a identidade, o controle de acesso à rede e os canais de comunicação incluídos (como TLS), para proteger tanto o agente quanto os dados acessados.
- **Confiabilidade e visibilidade:** o sistema precisa ter mecanismos para lidar com erros, fazer novas tentativas automaticamente e oferecer monitoramento completo. Isso significa registrar as ações do agente e os dados gerados pelas ferramentas, além de coletar métricas de desempenho e uso de recursos para diagnosticar e resolver problemas.

Casos de uso

A escolha do tempo de execução afeta diretamente os custos operacionais e a capacidade de escalar.

- **Vertex AI Agent Engine:** uma startup em fase inicial, com uma equipe de engenharia pequena, implanta seu primeiro agente de atendimento ao cliente, passando de protótipo funcional para um ponto de produção escalável e seguro em poucos dias, em vez de semanas.
- **Cloud Run:** uma startup que enfrenta crescimento rápido e imprevisível para uma nova funcionalidade com IA implanta seu agente ADK nessa arquitetura sem servidor, pagando apenas pelo processamento quando o agente está ativo. É uma forma econômica de lidar com picos de tráfego sem precisar superdimensionar a infraestrutura.
- **Google Kubernetes Engine (GKE):** uma startup de série B, com uma equipe de engenharia de plataforma consolidada e dezenas de microsserviços, decide hospedar seu novo agente interno de automação no cluster GKE já existente. Assim, ela pode aproveitar os processos de CI/CD, políticas de segurança e painéis de monitoramento já estabelecidos, garantindo que o agente siga os mesmos padrões operacionais dos demais serviços em produção.

1.3 O papel do grounding em sistemas agênticos

A credibilidade e a utilidade de um agente dependem da sua capacidade de fornecer respostas precisas e confiáveis com base em fatos verificáveis — um processo conhecido como grounding. Esta seção aborda a evolução das técnicas de grounding, oferecendo um caminho para construir agentes cada vez mais sofisticados e confiáveis.

Começamos com o padrão básico do RAG, que fornece grounding para o agente ao recuperar textos por similaridade semântica. Depois, exploramos o GraphRAG, que aprofunda o grounding ao compreender as relações explícitas entre os dados em um grafo de conhecimento. Por fim, abordamos o Agentic RAG, quando o agente deixa de ser um receptor passivo de informações e passa a atuar como participante ativo e reflexivo no processo de busca, capaz de executar estratégias em várias etapas para encontrar a melhor resposta possível.

RAG: essencial no primeiro passo

O primeiro passo para criar um grounding mais sofisticado é o padrão arquitetural RAG (geração aumentada por recuperação). Essa abordagem melhora as respostas de um modelo de linguagem grande (LLM) ao recuperar informações relevantes de uma base de conhecimento externa antes de gerar a resposta. Em vez de depender apenas do que foi aprendido durante o treinamento, o agente realiza uma busca semântica para encontrar dados verificáveis, que são então usados como contexto para o modelo. Isso garante uma base mínima de respostas embasadas e confiáveis.

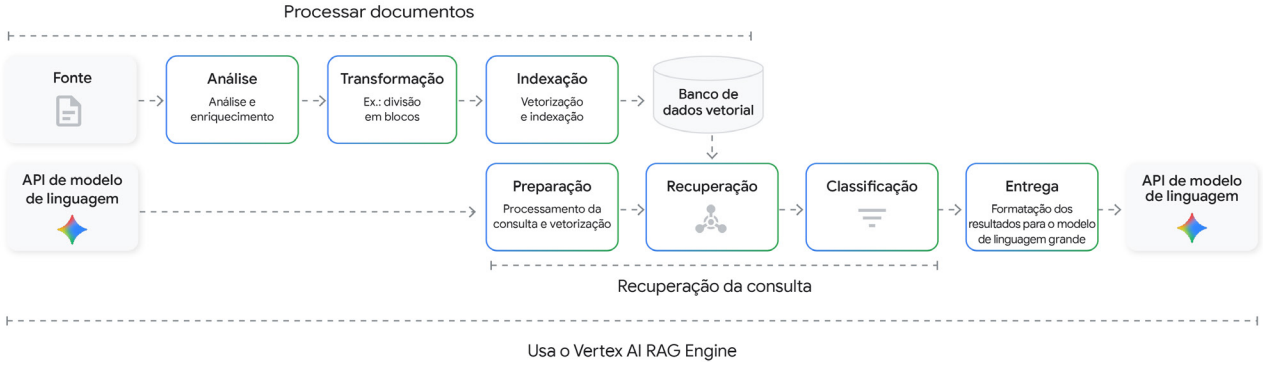
Apesar de ser um ponto de partida importante, esse processo simples de “buscar e depois gerar” trata o conhecimento como uma coleção plana de fatos desconectados. É eficaz para perguntas diretas, mas não atende bem a consultas mais complexas que exigem compreensão das relações entre os dados.

Benefícios do RAG para sistemas agênticos

- **Acesso à informação atualizada:** os dados recuperados são mais recentes do que a última data de treinamento do modelo, permitindo comportamentos mais relevantes e oportunos.
- **Maior precisão:** o RAG reduz significativamente o risco de haver respostas que possam causar ações do agente incorretas ou inadequadas.
- **Respostas mais rápidas:** embeddings vetoriais e bancos de dados especializados permitem buscas semânticas ultrarrápidas em grandes volumes de dados, tornando os agentes mais ágeis nas decisões.
- **Consciência maior do agente:** o fluxo de trabalho do RAG — que inclui ingestão, análise, divisão, vetorização, armazenamento e recuperação — pode ser aplicado a textos, imagens e outros tipos de dados. Com esse entendimento mais profundo, os agentes conseguem realizar tarefas de raciocínio mais complexas e em várias etapas.

A solução gerenciada e pronta para uso do Google Cloud baseada em RAG se chama Vertex AI Search. Ela simplifica a integração de fontes de dados e também pode utilizar ferramentas de código aberto ou de terceiros. O Vertex AI RAG Engine oferece uma estrutura de dados para desenvolver aplicativos com modelos de linguagem aumentados por contexto, capazes de fornecer respostas precisas e controladas, alinhadas a conhecimentos e políticas específicas. Isso é ideal para aplicativos críticos de startups, como atendimento ao cliente, gestão de conhecimento interno e tarefas relacionadas à conformidade.

Vertex AI RAG Engine em ação



Ferramenta

Use o Vertex AI Search e o Vertex AI RAG Engine para embasar as respostas incluindo seu conteúdo proprietário.

Dica

Use a API de verificação de grounding para conferir se as respostas da IA estão baseadas em informações atualizadas e confiáveis.

Bancos de dados vetoriais: busca por significado

A capacidade de buscar por significado, e não apenas por palavras-chave, é possível graças aos embeddings vetoriais. Essas representações numéricas capturam a essência conceitual dos dados (como textos e imagens), permitindo que o sistema encontre informações relevantes independentemente da forma como a pergunta foi feita. Os bancos de dados vetoriais são a infraestrutura que torna isso viável em larga escala. São sistemas altamente especializados, projetados para armazenar, indexar e consultar milhões desses embeddings com a baixa latência necessária para um sistema de agentes responsivo.

Como funciona:

1. **Os dados são transformados em embeddings vetoriais:** o modelo de machine learning posiciona itens semanticamente semelhantes próximos uns dos outros em um espaço vetorial multidimensional.
2. **Armazenamento e indexação:** o banco vetorial armazena esses embeddings e constrói índices especializados para permitir buscas rápidas e eficientes por similaridade.
3. **Consulta:** a pergunta do usuário é convertida em um embedding usando o mesmo modelo. O banco então localiza os embeddings mais próximos no índice, recuperando as informações mais relevantes em termos semânticos para embasar a resposta do modelo.

Caso de uso

Melhoria do atendimento ao cliente

Uma empresa de calçados usa um banco vetorial com busca semântica para alimentar seu chatbot de suporte ao cliente:

- Descrições de produtos, informações sobre garantia e perguntas frequentes são convertidas em embeddings e armazenadas.
- O banco vetorial entende que “bom para pessoas com pés largos” está semanticamente relacionado a conceitos como “forma larga”, “extra largo” ou “confortável para pés largos”.
- Ele recupera as recomendações de produto relevantes e oferece uma experiência muito melhor ao cliente.

Compare isso com o uso de um banco de dados tradicional. Uma consulta usando LIKE ‘%good for people with wide feet%’ não retornaria nenhum dado de saída, pois essa frase exata não existe no banco.

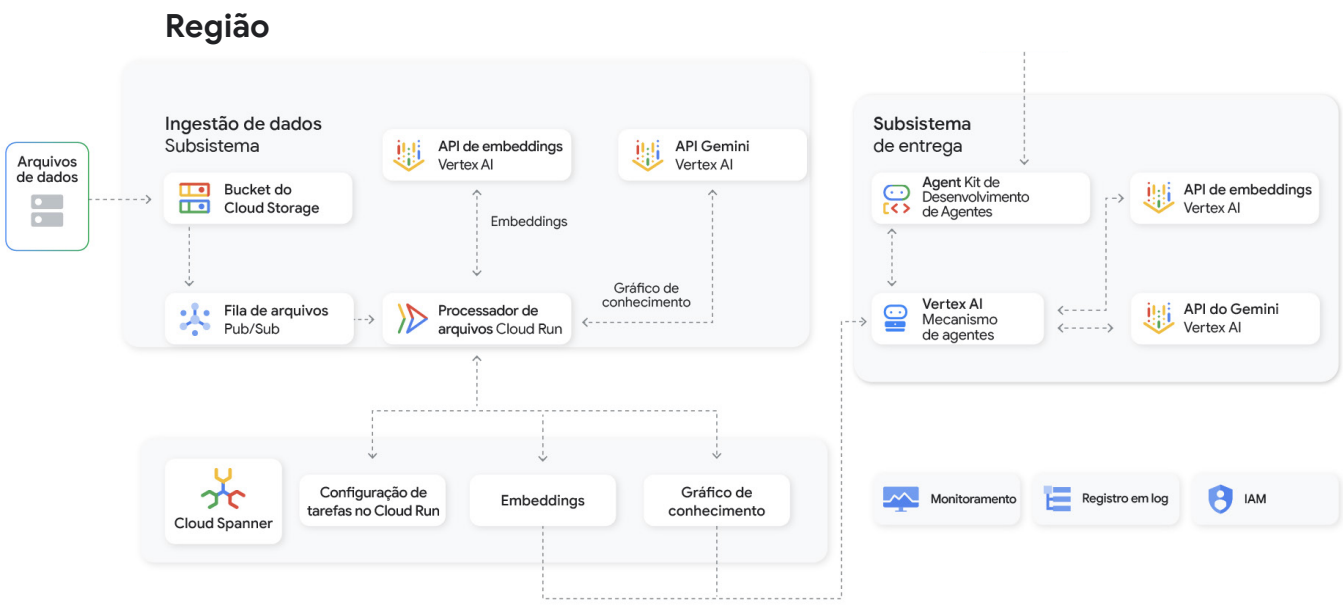
GraphRAG: um grounding mais inteligente

O GraphRAG constrói um grafo de conhecimento, então, em vez de apenas combinar frases semelhantes, o agente entende como os conceitos se relacionam.

Caso de uso

Um assistente médico com IA que precisa compreender **"sintomas → causas → tratamentos"**, e não apenas recuperar trechos relacionados.

A hierarquia do conhecimento no GraphRAG



Agentic RAG: raciocínio dinâmico e busca inteligente

A abordagem mais poderosa para grounding é o Agentic RAG — uma técnica que transforma o agente de um receptor passivo de dados recuperados em um participante ativo e reflexivo na busca por conhecimento. Seguindo estruturas como o ReAct, o agente pode analisar uma consulta complexa, formular um plano com várias etapas e executar chamadas de ferramentas em sequência para encontrar as melhores informações possíveis.

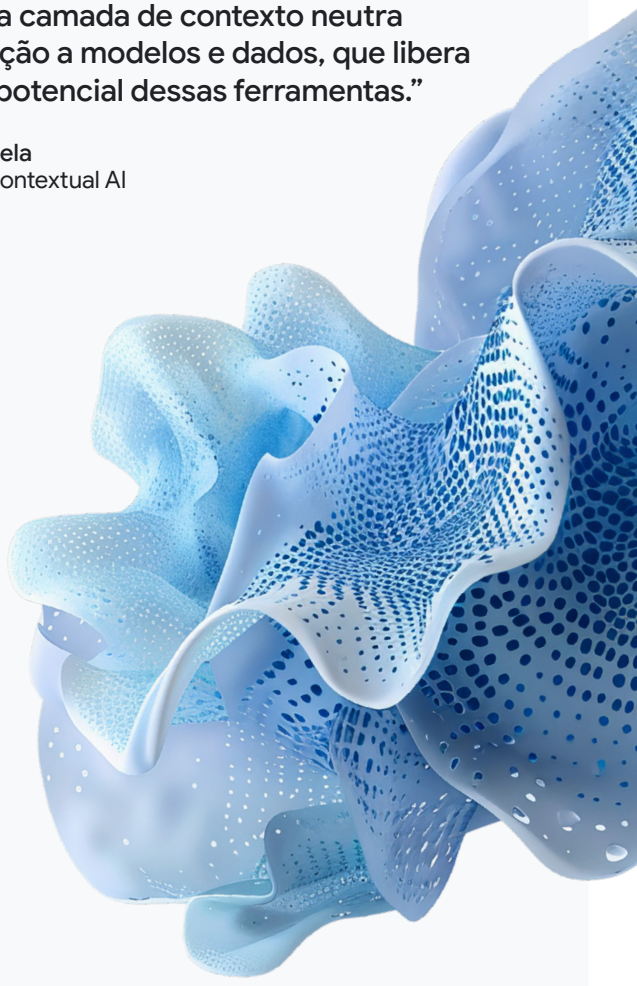
Um exemplo claro desse padrão é o grounding com a Pesquisa do Google. É possível usar os modelos da família Gemini 2.5 para integrar raciocínio avançado, permitindo que eles combinem capacidades de busca com processos internos de pensamento para responder a perguntas complexas, com múltiplos saltos de lógica, e executar tarefas de longo prazo. O agente pode cuidar de todo o fluxo de trabalho automaticamente: analisar o pedido, formular e fazer pesquisas precisas, e sintetizar uma resposta final embasada com fontes.

Um agente construído com os modelos da família Gemini vai além do simples reconhecimento de conteúdo — ele resolve problemas de forma ativa e em várias etapas. Por exemplo, um agente pode:

- Analisar uma foto para identificar uma espécie específica de planta e, em seguida, buscar de forma autônoma instruções detalhadas de cuidados.
- Processar um áudio de uma ligação de suporte para não apenas transcrever as palavras, mas também detectar o sentimento do cliente (como frustração) e escalar o atendimento de forma adequada.

Essa capacidade de perceber e raciocinar com diferentes tipos de dados transforma o agente de um processador de informações em uma ferramenta de solução de problemas que entende e interage com o mundo de forma mais completa.

> “A ideia que se tinha era que o desempenho dos modelos fundacionais melhoraria exponencialmente, mas estamos chegando a um ponto de inflexão em que essa curva começa a se estabilizar, e o verdadeiro diferencial está na especialização e na engenharia de contexto. O Agentic RAG é um dos pilares centrais dessa camada de contexto que permite que agentes de IA encontrem, recuperem e raciocinem sobre dados reais antes de gerar uma resposta final.
>
> O futuro é multi-LLM: diferentes modelos para diferentes tarefas, conectados por uma camada de contexto neutra em relação a modelos e dados, que libera todo o potencial dessas ferramentas.”
>
> **Douwe Kiela**
> CEO da Contextual AI



Exemplo: verificação de estoque em tempo real

Defina uma função chamada check_inventory que recebe um product_ID e retorna os níveis atuais de estoque do seu sistema de inventário em tempo real. Da mesma forma, outra função chamada check_warranty_status, pode receber um a product_ID e retornar as informações de garantia diretamente do seu sistema de gestão de garantias.

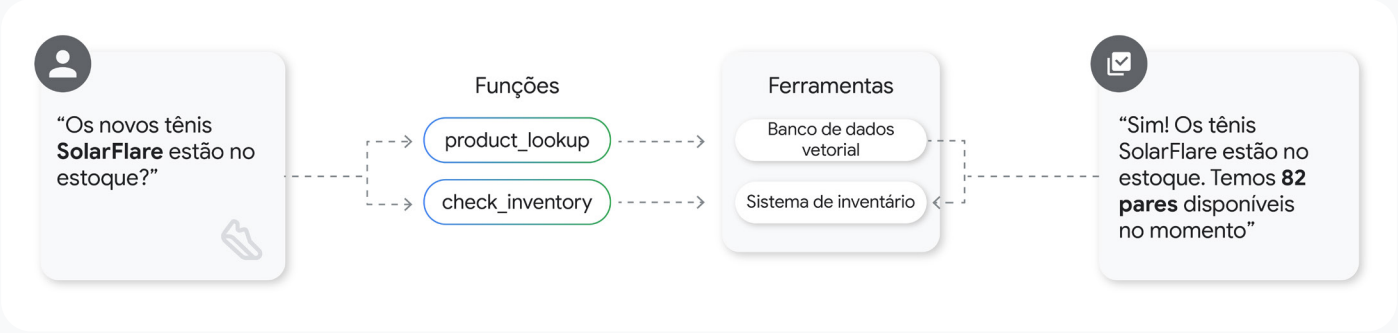
Então, quando um cliente pergunta sobre a disponibilidade de um produto específico, o agente de IA:

1. **Identifica o produto:** usa a busca semântica (alimentada pelo banco vetorial) para identificar com precisão o modelo de calçado mencionado, mesmo que a descrição seja vaga.
1. **Dispara a ação:** reconhece que é necessário obter informações de estoque em tempo real e chama a função check_inventory.
2. **Fornece resposta imediata:** a função check_inventory é executada, busca os dados atualizados no sistema de inventário e retorna ao agente, que então fornece ao cliente uma resposta precisa e imediata sobre a disponibilidade.

Essa combinação de recuperação (saber o que é relevante) e ação (executar operações em tempo real) torna os agentes de IA mais inteligentes, rápidos e úteis.

Dica

Use a Vertex AI e a pesquisa vetorial do Google Cloud para adicionar essa funcionalidade ao seu agente.



O fluxo de trabalho do Agentic RAG no Google Cloud

O Google Cloud oferece serviços gerenciados que cuidam de todo o fluxo de trabalho do Agentic RAG:

- **Geração de embeddings e indexação:** o primeiro passo é converter seus dados em embeddings vetoriais. Uma opção é o modelo Gemini Embedding, disponível tanto no Vertex AI quanto na API Gemini, com suporte para mais de 100 idiomas. Esse e outros modelos fazem parte do conjunto mais amplo de APIs de embeddings do Vertex AI.
- **Armazenamento e indexação:** os embeddings vetoriais são armazenados e indexados no Vertex AI Vector Search. Trata-se de um banco vetorial de alto desempenho e totalmente gerenciado, que constrói automaticamente os índices especializados necessários para buscas por similaridade rápidas e eficientes em larga escala.
- **Recuperação e raciocínio:** quando um usuário envia uma consulta, ela é convertida em um embedding e usada pelo Vertex AI Vector Search para encontrar as informações mais relevantes. Esse contexto direcionado é então passado para o modelo de linguagem para gerar uma resposta final embasada.

Dica

Use a abordagem de "recuperar e reordenar". Para equilibrar abrangência (encontrar todos os documentos relevantes) e precisão (garantir que cada documento recuperado seja realmente útil), use a abordagem em duas etapas de "recuperar e reordenar" (como mostrado no exemplo agentic_rag). Primeiro, amplie o alcance da busca configurando a Vertex AI Vector Search para recuperar um conjunto maior de documentos candidatos. Depois, esse conjunto é enviado ao modelo de linguagem ou a um serviço especializado de reordenação, que identifica os documentos mais relevantes e descarta os irrelevantes ou semanticamente opostos.

Da recuperação ao raciocínio: uma vantagem estratégica

O Agentic RAG representa um salto fundamental, indo além da simples recuperação de informações para uma verdadeira solução de problemas. Ao permitir que o agente atue como participante ativo e reflexivo, os desenvolvedores podem criar sistemas capazes de executar consultas complexas, em várias etapas, e tarefas de longo prazo — características centrais das capacidades agênticas de nova geração.

Para uma startup, dominar essa abordagem é essencial para construir um produto inteligente e competitivo, capaz de desbloquear novos fluxos de trabalho e experiências para o usuário.

Outros métodos de grounding

Embora o RAG seja uma técnica fundamental, a Vertex AI oferece outras formas de garantir que seus agentes forneçam respostas precisas e confiáveis. Por exemplo:

- **Grounding com a Pesquisa do Google:** conecte seu modelo ao conhecimento global e a uma ampla variedade de tópicos.
- **Grounding com o Google Maps:** use os dados do Google Maps com seu modelo para fornecer respostas mais precisas e contextualizadas.
- **Grounding do Gemini com seus dados:** use RAG para conectar seu modelo aos dados do seu site ou aos seus conjuntos de documentos.
- **Grounding do Gemini com Elasticsearch:** use RAG com seus índices existentes do Elasticsearch e o modelo Gemini.

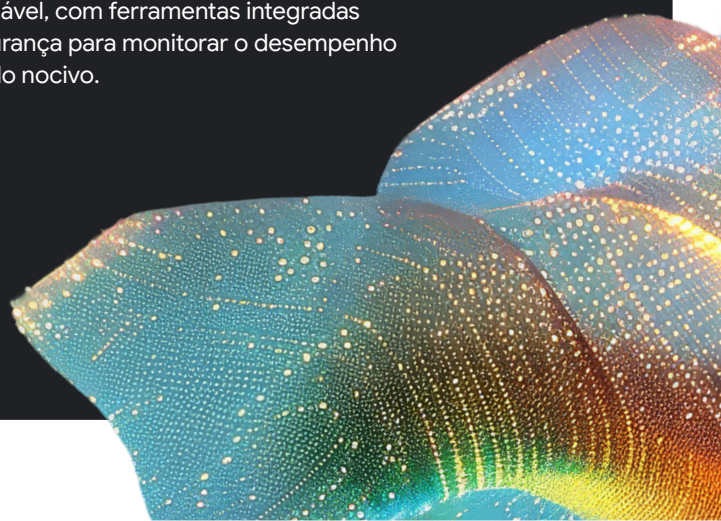
Exemplo: grounding com a Pesquisa do Google

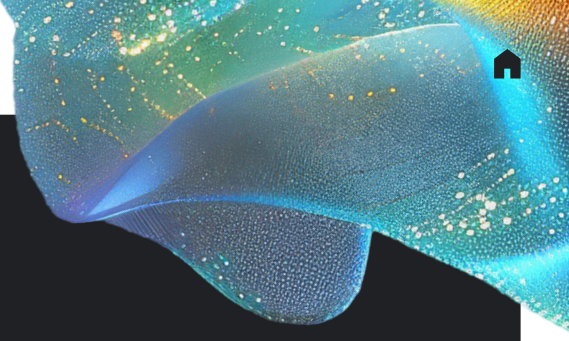
Usando a ferramenta de Pesquisa do Google, o modelo lida automaticamente com todo o fluxo de pesquisa, processamento e citação das informações.

1. **Prompt do usuário:** o aplicativo envia o prompt para a API Gemini junto com a ferramenta Pesquisa do Google.
2. **Análise do prompt:** o modelo analisa o prompt e decide se uma pesquisa no Google pode melhorar a resposta.
3. **Pesquisa do Google:** se necessário, o modelo gera automaticamente uma ou mais consultas de busca e as executa.
4. **Processamento dos resultados:** o modelo processa os resultados da pesquisa, sintetiza as informações e formula uma resposta.
5. **Resposta embasada:** a API retorna uma resposta final, clara e fácil de entender, embasada nos resultados da pesquisa. A resposta inclui o texto gerado pelo modelo e os metadados de grounding, contendo as consultas feitas, os resultados encontrados na web e as citações.

Principais aprendizados: escolher os componentes do seu agente de IA

Seu objetivo	Melhor opção
Escolher a inteligência central do agente.	Selecione um modelo (ex.: Gemini) com base no seu caso de uso e ajuste com seus dados específicos.
Tornar seu agente confiável e baseado em fatos.	Use técnicas de grounding como RAG com banco vetorial para que ele verifique os fatos, e não apenas faça suposições.
Gerenciar uma tarefa complexa com várias etapas.	Use orquestração para criar um plano que defina qual ferramenta usar, em que ordem e como combinar os resultados.
Conectar-se a dados e serviços públicos em tempo real.	Use extensões prontas para se integrar facilmente a APIs de terceiros.
Conectar-se com segurança às suas ferramentas internas.	Escreva funções personalizadas para dar ao agente acesso controlado aos seus bancos de dados, CRMs ou outros sistemas internos.
Implantar um produto confiável e seguro em larga escala.	Use um tempo de execução gerenciado para uma infraestrutura escalável, com ferramentas integradas de avaliação e segurança para monitorar o desempenho e bloquear conteúdo nocivo.





Pronto para transformar sua visão de IA em realidade? Estamos aqui para ajudar.

Aprenda a criar mais aplicativos de IA generativa com as sessões sob demanda da Startup.

Comece agora

Receba até US$ 350 mil em créditos do Google Cloud com o programa Google Cloud for Startups.

Inscreva-se já

Entre em contato com a nossa Equipe de Startups.

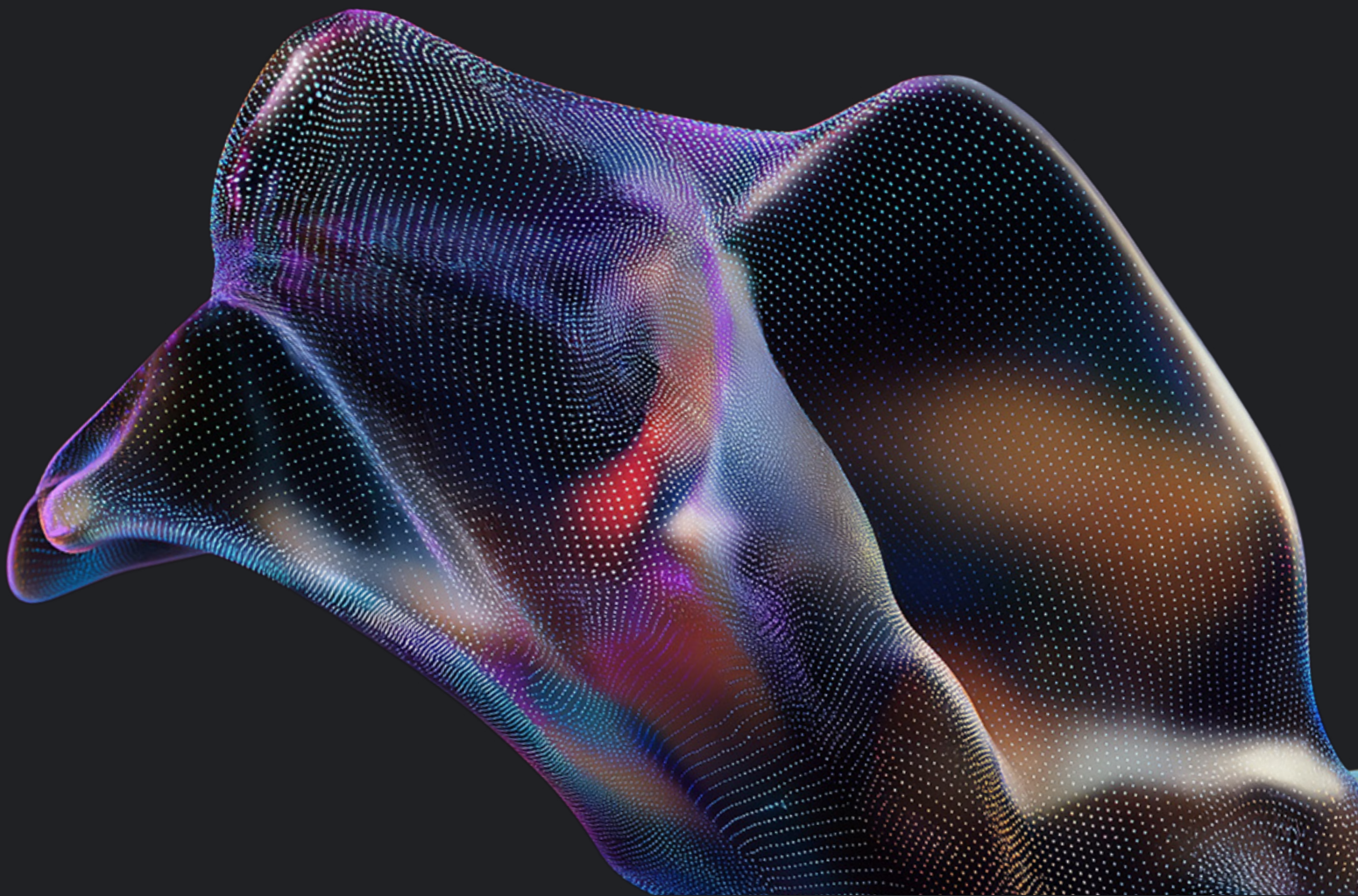
Fale com a gente

Fique atualizado e receba todas as nossas novidades assinando a newsletter do Google Cloud para Startups.

Assinar

Seção 2

Como construir agentes de IA



A seção anterior definiu os componentes fundamentais de um sistema agêntico moderno: um modelo central de raciocínio, um conjunto de ferramentas para permitir ações, opções de arquitetura de dados para manter a memória de curto e longo prazo do agente, um mecanismo de grounding para garantir precisão factual, e opções de implantação.

Com esse framework conceitual estabelecido, agora podemos focar em como construir um agente. Esta seção oferece um guia prático e com recomendações claras sobre as decisões arquiteturais envolvidas na criação de um agente pronto para produção.

Aberto e flexível, o ecossistema do Google Cloud reconhece que o mercado oferece muitos frameworks excelentes. Aqui, no entanto, a intenção é dar foco ao ADK, com uma implementação completa e robusta que se encaixa perfeitamente no ecossistema do Google Cloud. Também há recomendações específicas baseadas em padrões abertos do setor, como o Model Context Protocol (MCP) e o protocolo Agent2Agent (A2A).

Dica

Antes de construir seu agente, explore 601 casos reais de uso de agentes adotados por organizações líderes no mundo.

Prefere ouvir? Escute a versão em podcast desta seção, criada com o NotebookLM.

Este podcast foi criado usando o NotebookLM com o seguinte prompt: “Como apresentador de podcast, crie um episódio prático voltado para desenvolvedores e fundadores técnicos. O podcast deve apresentar o Agent Development Kit (ADK) como solução para os desafios comuns na construção de agentes. É necessário detalhar os principais benefícios do ADK e explicar seus tipos principais de agentes, como o inteligente LlmAgent e os agentes estruturados de fluxo de trabalho (por exemplo, Sequencial, Paralelo e em Loop).

O podcast deve então abordar o ecossistema mais amplo, incluindo o Model Context Protocol (MCP), o Vertex AI Agent Engine para implantação e o protocolo Agent2Agent (A2A) para comunicação. Mencione brevemente ferramentas alternativas, como Gemini Enterprise, Firebase Studio e Gemini CLI, e conclua com um resumo e um chamado à ação direcionando os ouvintes aos recursos para startups do Google”.

2.1 Um kit completo para construir agentes de IA

Ao construir um agente de IA personalizado para sua startup, os fundadores enfrentam um dilema importante: rapidez no desenvolvimento versus flexibilidade.

De um lado, há soluções fáceis de usar, como plataformas low-code ou produtos prontos. Elas são rápidas de implementar, mas oferecem menos controle, sendo mais adequadas para problemas padronizados. Do outro lado, existem frameworks altamente flexíveis ou a opção de construir tudo do zero. Embora ofereçam a máxima personalização, exigem mais recursos de desenvolvimento e conhecimento técnico profundo. O ADK se posiciona no meio desse cenário de desenvolvimento.

Explore o ADK

Componentes essenciais para construir agentes de IA

Agent Development Kit	Ferramenta de código aberto, voltada para desenvolvedores, para construir, avaliar e implantar agentes de IA.
Model Context Protocol	Protocolo aberto que padroniza como os aplicativos fornecem conteúdo para modelos de linguagem grandes (LLMs).
Vertex AI Agent Engine	Plataforma gerenciada para implantar, administrar e escalar agentes de IA em produção.
Protocolo Agent2Agent (A2A)	Padrão aberto projetado para permitir comunicação e colaboração entre agentes de IA.

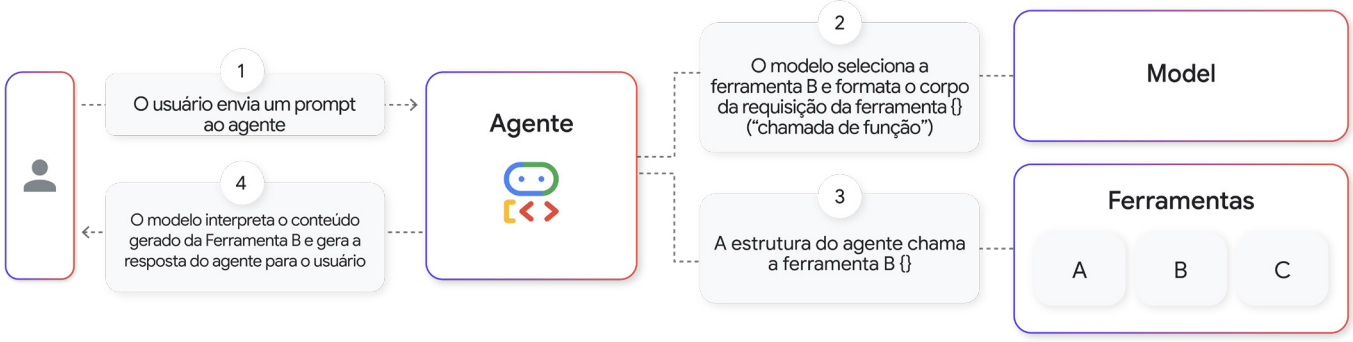
Desenvolva com o ADK

Se você precisa de mais poder do que uma ferramenta low-code simples, mas quer acelerar o processo de desenvolvimento, o ADK oferece controle para construir um sistema único de agentes colaborativos, ao mesmo tempo que simplifica desafios técnicos complexos. Por exemplo, protocolos específicos como MCP e A2A facilitam a expansão das capacidades dos agentes (como veremos a seguir).

Além disso, o ADK complementa as ferramentas que a sua equipe já pode estar usando e se integra ao ecossistema mais amplo do Google Cloud, então você não fica preso a uma única abordagem. Essa flexibilidade também se aplica à forma como você executa seus agentes depois de construí-los, com opções para implantar em um serviço totalmente gerenciado como o Vertex AI Agent Engine ou em uma plataforma serverless versátil como o Cloud Run. Tudo depende de escolher a base certa para suas necessidades operacionais específicas.

> “Estamos focados em construir um ecossistema sofisticado de agentes de IA. O Agent Development Kit de código aberto já teve mais de um milhão de downloads em menos de quatro meses”.
>
> **Sundar Pichai**
> CEO do Google e da Alphabet

Construir fluxos de trabalho complexos fica mais fácil com o ADK



O que você pode fazer com o ADK

1. Construir sistemas de IA complexos e colaborativos

O ADK é projetado para ser multiagente. É fácil construir soluções de IA altamente especializadas que automatizam fluxos de trabalho complexos e com várias etapas. E, com orquestração flexível (sequencial, paralela ou dinâmica), você pode começar com automações simples e evoluir para sistemas altamente adaptativos. Por exemplo, é possível criar um sistema inteligente de gerenciamento de projetos com um “Agente de Divisão de Tarefas” que delega subtarefas para agentes especializados como “Agentes de Geração de Código”, “Agentes de Design” e “Agentes de Documentação”.

2. Integrar IA a ferramentas, agentes e fluxos de trabalho existentes

O ADK possui um ecossistema rico de ferramentas que permite que seus agentes interajam com todas as ferramentas e dados que você já utiliza. Você pode conectar seus agentes a ferramentas de produtividade como Notion, Slack ou um CRM, além de frameworks de ferramentas, como LangChain e LlamaIndex, ou frameworks de agentes como LangGraph ou CrewAI. As ferramentas podem ser compartilhadas via MCP e os agentes que você cria podem ser compartilhados via A2A. Dessa forma, você insere inteligência artificial em todas as áreas da sua operação, aprimorando os sistemas que já estão em uso sem precisar reformular toda a arquitetura.

3. Garantir a qualidade e a confiabilidade desde o início

Para ganhar a confiança dos usuários, é essencial testar e avaliar cuidadosamente seus agentes antes de colocá-los em produção. As ferramentas integradas de observabilidade e avaliação do ADK ajudam você a:

- **Iterar rapidamente:** antes de implantar, você pode testar sistematicamente como seus agentes respondem a diferentes cenários e como executam tarefas complexas.
- **Depurar o comportamento do agente:** inspecione todo o rastreamento de execução do agente, incluindo seu raciocínio (pensamentos), chamadas de ferramentas e observações, para entender seu processo de decisão e depurar fluxos de trabalho complexos.
- **Comparar desempenho:** avalie diferentes designs de agentes ou atualizações de modelos com base em métricas predefinidas, usando uma abordagem baseada em dados para melhorar continuamente o desempenho.

Isso leva seu processo para além de um simples "vibe testing" (teste de impressão), permitindo lançar agentes com qualidade profissional rapidamente, ganhar a confiança com uma confiabilidade comprovada e iterar com segurança com base em dados.

4. Escalar a IA com confiança

À medida que você cresce, as soluções de IA precisam escalar de forma fluida, sem se tornar gargalos. O ADK acelera o caminho até a produção usando o AgentOps (descrito mais abaixo) para conectar o desenvolvimento local à implantação. O framework expõe os agentes como serviços web padrão usando FastAPI, que podem ser conteinerizados. Isso libera os desenvolvedores da necessidade de construir infraestrutura personalizada de implantação. Eles podem implantar em qualquer lugar, desde testes locais até tempo de execução completamente gerenciado e automaticamente escalável, como o Vertex AI Agent Engine ou o Cloud Run.

Núcleo do ADK: arquiteturas de agentes

Um passo fundamental ao construir com o ADK é escolher a arquitetura de agente adequada. Classes distintas de agentes são projetadas para diferentes padrões de execução, e sua escolha determinará como o agente raciocina e opera. Normalmente, trata-se de equilibrar o poder flexível e não determinístico de um LLM com o controle previsível e determinístico da lógica embutida no código. Compreender a interação entre essas classes de agentes é essencial para construir sistemas de IA robustos e eficazes.

Os tipos de agentes do ADK são organizados em três categorias

Principais tipos de agentes

1 Agente LLM (LlmAgent)

Determinismo: não-determinístico (flexível)

Esse é o tipo de agente mais comum e, em geral, é chamado simplesmente de “Agent”. Ele usa um modelo de linguagem como o Gemini para raciocínio complexo, tomada de decisões dinâmica e compreensão de linguagem natural, e constitui o núcleo da maioria dos agentes conversacionais e de solução de problemas.

2 Agente de fluxo de trabalho (SequentialAgent, ParallelAgent, LoopAgent)

Mecanismo principal: lógica predefinida

Determinismo: determinístico (previsível)

Esses orquestradores controlam de forma determinística como outros agentes são executados em padrões predefinidos. Eles são usados para processos estruturados.

Sequential agents (SequentialAgent)

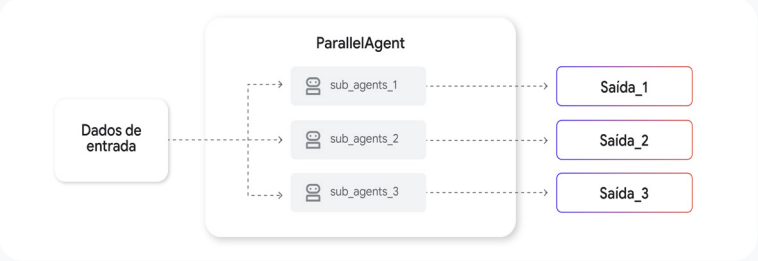
Executam subagentes em uma ordem fixa, passando os dados de saída de um como entrada para o próximo. O Agent A conclui a Tarefa 1, e, em seguida, passa os dados de saída para o Agent B realizar a Tarefa 2.

Exemplo: você quer construir um agente que possa resumir qualquer página da web, usando duas ferramentas: Get Page Contents e Summarize Page. Como não é possível resumir sem conteúdo, o agente deve sempre chamar Get Page Contents antes de chamar Summarize Page.

Obtenha o código completo

Agentes paralelos (ParallelAgent)

Executam múltiplos subagentes simultaneamente. São usados para otimizar o desempenho quando as tarefas são independentes. O Agent A e o Agent B trabalham em subtarefas independentes ao mesmo tempo, e seus resultados são combinados.

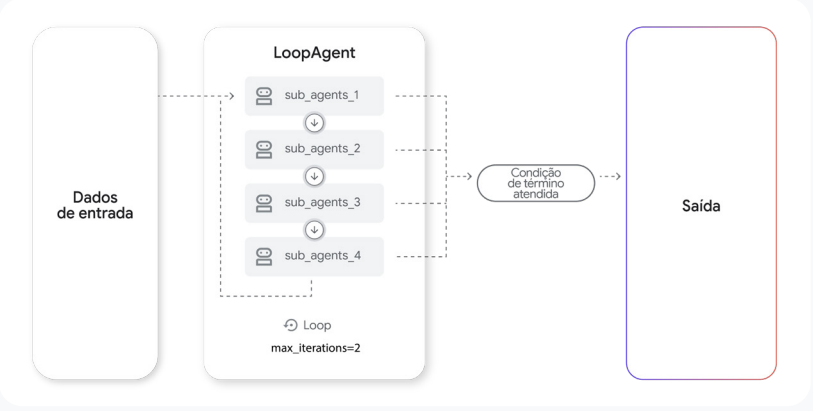


Exemplo: para operações como recuperação de dados de múltiplas fontes ou cálculos pesados, a paralelização oferece um ganho significativo de desempenho. Importante: essa estratégia parte do princípio de que não há necessidade inerente de compartilhamento de estado ou troca direta de informações entre os agentes que executam simultaneamente.

Obtenha o código completo

Agentes em loop (LoopAgent)

Um agente de fluxo de trabalho executa seus subagentes em loop (de forma iterativa). Ele executa repetidamente uma sequência de agentes por um determinado número de iterações ou até que uma condição de término seja atendida. Use o LoopAgent quando seu fluxo de trabalho envolver repetição ou refinamento iterativo, como revisão de código.



Exemplo: você quer construir um agente que possa gerar imagens contendo quantidades específicas de alimentos (por exemplo, cinco bananas), usando duas ferramentas: Generate Image, Count Food Items. Como você deseja continuar gerando imagens até que o número especificado de itens seja corretamente gerado, ou até atingir um número máximo de iterações, o agente deve ser construído usando um LoopAgent.

Obtenha o código completo

3 Agente personalizado (subclasse de BaseAgent)

Mecanismo principal: código Python personalizado

Determinismo: pode ser determinístico ou não, dependendo da implementação

Para requisitos únicos e fluxos de trabalho sob medida que vão além de um ciclo padrão de raciocínio, você pode criar um agente personalizado herdado diretamente de BaseAgent e escrever uma lógica Python personalizada para controlar seu comportamento. Essa abordagem é necessária quando as ações do agente não são determinadas por um LLM, mas por regras específicas embutidas no código.

Como é feito: um desenvolvedor cria uma nova classe Python herdada da classe BaseAgent. Em seguida, ele implementa o método _run_async_impl, que contém a lógica exclusiva que o agente vai executar. Esse método tem acesso completo ao estado da sessão e pode gerar eventos para se comunicar com outros agentes ou encerrar um fluxo de trabalho.
Consulte o guia rápido do Gemini Fullstack Agent Development Kit (ADK) para obter um exemplo de como implementar um agente personalizado.

Orquestração com ADK: implementar o ciclo ReAct

Como descrito na seção 1, o paradigma ReAct é um padrão fundamental para sistemas baseados em agentes. O ADK fornece as abstrações e classes essenciais para implementar esse processo dinâmico e cíclico de forma estruturada. Seu LlmAgent foi projetado especificamente para executar esse ciclo e gerenciar as transições entre os estágios principais:

- **Raciocínio (pensamento):** a classe LlmAgent gerencia esse estágio internamente. Ela recebe o prompt do usuário e seu estado interno atual, então chama o modelo de linguagem subjacente para formar uma hipótese e determinar a próxima melhor ação.
- **Ação (uso de ferramentas e delegação para agentes):** o ADK viabiliza esse estágio por meio de seu sistema flexível de ferramentas. Quando o LlmAgent decide agir, ele pode invocar uma função simples em Python ou, para tarefas mais complexas, delegar o trabalho a outro subagente especializado usando o padrão Agent-as-a-Tool.
- **Observação: o ADK captura automaticamente o dicionário retornado pela ferramenta ou subagente e o repassa ao** LlmAgent. Essa saída se torna a nova informação que o agente integra ao seu contexto, alimentando o próximo passo de Raciocinar no ciclo.

Ao oferecer uma implementação nativa desse padrão essencial, o ADK elimina o código repetitivo, permitindo que você transforme rapidamente o poderoso conceito do ciclo ReAct em um agente funcional de múltiplas etapas.

Ferramentas do ADK: um framework para a ação agêntica

No ADK, um agente pode usar ferramentas para realizar ações além das capacidades nativas do seu modelo de raciocínio central. Essas funcionalidades definidas permitem que o agente execute um código, interaja com sistemas externos e atue fora de seu contexto imediato de execução. Uma ferramenta é uma função Python (ou um método Java) que pode conter lógica independente ou servir como um wrapper para operações mais complexas, como chamadas de API, uso do MCP para acessar diversos sistemas externos, ou delegação de tarefas a outro agente especializado localmente ou remotamente via A2A.

Esta seção descreve como projetar ferramentas eficazes e apresenta a taxonomia dos tipos de ferramentas disponíveis.

Dica

Para uma discussão completa, incluindo exemplos de código e padrões avançados de uso, consulte a documentação do ADK.

Projetando ferramentas eficazes: o contrato de API para o modelo

Para que um modelo utilize corretamente uma ferramenta, sua definição deve funcionar como um **contrato de API** claro e sem ambiguidades, composto por:

- **Assinatura da função:** use nomes descritivos para as ferramentas e seus parâmetros. As anotações de tipo em Python são obrigatórias, pois fornecem o esquema estrutural que o modelo usa para gerar argumentos válidos.
- **Docstring (o núcleo semântico):** é a principal fonte de informação semântica para o modelo. Uma docstring bem escrita deve definir com precisão o propósito da ferramenta, critérios de uso, parâmetros e o formato esperado de retorno.
- **Esquema de retorno:** a ferramenta deve retornar um dicionário. Embora não seja uma exigência sintática rígida, é uma boa prática incluir uma chave de status (por exemplo, success ou error) nesse dicionário. Essa estrutura é essencial para que o agente consiga distinguir de forma confiável entre resultados bem-sucedidos e falhas na etapa de Observação e raciocinar sobre como prosseguir.
- **Ferramentas com estado e ToolContext:** para ferramentas que precisam ler ou escrever em um estado de sessão persistente, pode-se adicionar o parâmetro opcional tool_context: ToolContext à assinatura da função. O agente injeta automaticamente esse objeto, dando à ferramenta acesso ao dicionário de estado da sessão.

Dica

Para obter práticas recomendadas e exemplos de como definir parâmetros e esquemas de ferramentas, estruturar prompts eficazes e implementar fluxos de trabalho complexos com múltiplos agentes, consulte o repositório de exemplos do ADK.

Uma taxonomia das ferramentas do ADK

O ADK oferece uma arquitetura flexível para implementação de ferramentas, que vão desde funções simples até sistemas multiagentes interoperáveis.

Conjuntos de ferramentas: agrupando capacidades relacionadas

Um padrão central no ADK é o Toolset, uma classe que agrupa um conjunto de ferramentas relacionadas em um único objeto configurável (por exemplo, BigQueryToolset, MCPToolset).

Ferramentas de função personalizadas

É o método mais direto para estender um agente com lógica proprietária.

- FunctionTool: o wrapper padrão para funções Python síncronas.
- LongRunningFunctionTool: uma ferramenta especializada para tarefas assíncronas ou fluxos com intervenção humana.

Ferramentas hierárquicas e remotas

O ADK permite a criação de sistemas complexos por meio da composição de agentes.

- **Agent-as-a-tool:** padrão de delegação onde um agente pai utiliza outro agente especializado. Isso permite que o agente pai invoque outro agente, receba uma resposta e mantenha o controle para lidar com entradas futuras. (Isso é diferente do modelo de subagente, onde o controle total da conversa é passado ao subagente e todas as entradas subsequentes são tratadas por ele.)
- RemoteA2aAgent: para a comunicação entre agentes em processos diferentes, o ADK fornece a classe RemoteA2aAgent, que usa o protocolo **Agent2Agent** (A2A) para integrar sistemas distribuídos de forma transparente.

Ferramentas pré-construídas e integradas

O ADK inclui um conjunto de ferramentas e wrappers para acelerar o desenvolvimento.

- **Ferramentas integradas:** ferramentas prontas para uso como Google Search e Code Execution.
- **Conjuntos de ferramentas do Google Cloud:** integrações avançadas com serviços como Vertex AI Search e BigQuery.
- **Interoperabilidade com terceiros:** wrappers como LangchainTool e CrewaiTool permitem o reaproveitamento direto de ferramentas de ecossistemas open-source populares.

Padronize com o Model Context Protocol

O Model Context Protocol (MCP) é um padrão aberto emergente para conectar IA e LLMs a fontes de dados e ferramentas externas. Ele permite que você conecte seus aplicativos de IA a diversas fontes de dados e ferramentas sem precisar criar integrações personalizadas ponto a ponto para cada uma.

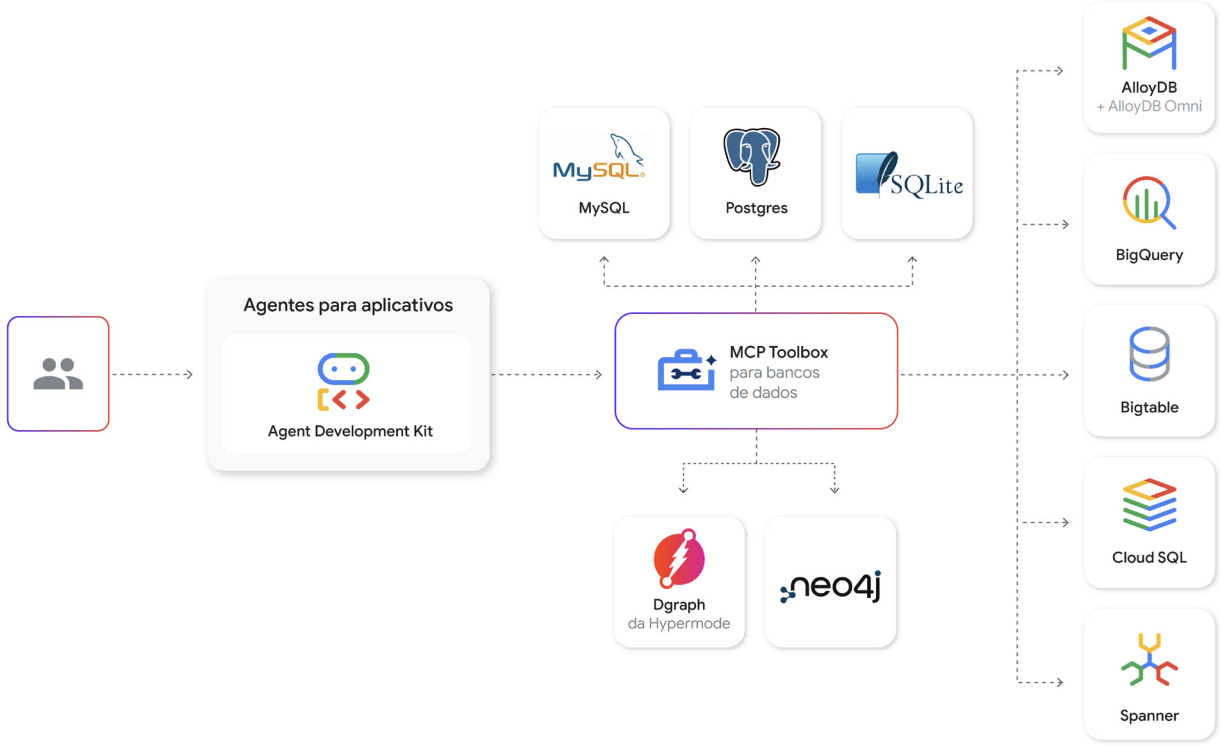
Com o ADK, seus agentes podem participar desse ecossistema de duas formas:

- **Consumir ferramentas externas:** Um agente ADK pode atuar como cliente MCP, permitindo que ele use qualquer ferramenta exposta por um servidor MCP de terceiros.
- **Expor ferramentas nativas:** Os desenvolvedores podem empacotar suas ferramentas ADK em um servidor MCP, tornando-as disponíveis de forma segura para qualquer outro agente ou aplicativo compatível com MCP.

Dica

Use a MCP Toolbox de código aberto para bancos de dados e conecte seus agentes de forma fácil e segura a uma ampla variedade de fontes de dados populares.

O MCP funciona como um adaptador universal para as fontes de dados e ferramentas de um agente.



Gerencie seus dados com os serviços do Google Cloud

Como descrito na seção anterior, a memória transacional, de longo prazo e de trabalho desempenham papéis distintos na arquitetura de dados de um agente. Aqui explicamos como construí-la. O ADK fornece os padrões e integrações necessários para mapear essa arquitetura conceitual diretamente para determinados serviços de dados escaláveis do Google Cloud.

1. Base de conhecimento de longo prazo (grounding, contexto e análise)

Essa é a memória permanente do agente, que combina uma biblioteca de conhecimento pesquisável, um registro de interações com o usuário e um repositório para análise.

- **Vertex AI Search:** serve como biblioteca de conhecimento pesquisável do agente para informações não estruturadas. No ADK, a VertexAISearchToolset permite que o agente embase suas respostas recuperando informações relevantes de um conjunto específico de documentos.
- **Firestore:** funciona como a memória persistente do usuário. No ADK, é usado para armazenar e recuperar o histórico de conversas e o estado de tarefas de longa duração, permitindo uma experiência contínua e personalizada que pode ser retomada entre sessões.
- **Cloud Storage:** atua como o sistema de arquivos durável do agente. O ADK o utiliza como fonte da verdade para documentos brutos (como PDFs e imagens), que são indexados por serviços como a Vertex AI Search.
- **BigQuery:** funciona como o banco de dados analítico do agente. O BigQueryToolset no ADK permite que os agentes respondam perguntas executando consultas analíticas complexas em grandes conjuntos de dados estruturados.

2. Memória de trabalho (cache e estado de sessão)

É a memória transitória e de alta velocidade do agente para gerenciar o contexto imediato de uma conversa em tempo real.

- **Memorystore:** fornece um cache de alta velocidade para o agente. No ADK, sua principal função é armazenar os resultados de chamadas de ferramentas frequentes ou de alto custo, reduzindo drasticamente a latência e os custos operacionais.

3. Memória transacional (auditoria e execução confiável)

É o livro-razão durável do agente para registrar ações críticas e mudanças de estado com alta integridade.

- **Cloud SQL:** serve como sistema confiável de registro do agente. O ADK permite padrões onde as ferramentas registram suas ações no Cloud SQL, criando um rastro de auditoria permanente e compatível com ACID para cada ação importante realizada pelo agente.
- **Cloud Spanner:** atua como backend globalmente consistente para ações críticas do agente. Em uma implementação avançada do ADK, uma ferramenta que representa um processo de negócio central (como process_global_order) dispararia uma transação em um sistema baseado no Spanner para garantir a integridade global.

4. A próxima fronteira: memória conversacional destilada

Conforme o histórico de interações de um agente com um usuário aumenta no decorrer de semanas ou meses, fornecer todo o contexto bruto ao modelo para cada consulta se torna ineficiente e caro. Além disso, os modelos podem se confundir.

A destilação de memória é a nova fronteira. Ela usa um LLM para destilar, de forma dinâmica e contínua, longos históricos de conversa em um conjunto compacto e estruturado de fatos e preferências essenciais. A memória de longo prazo resultante, filtrada, é muito mais eficiente para se recuperar e utilizar.

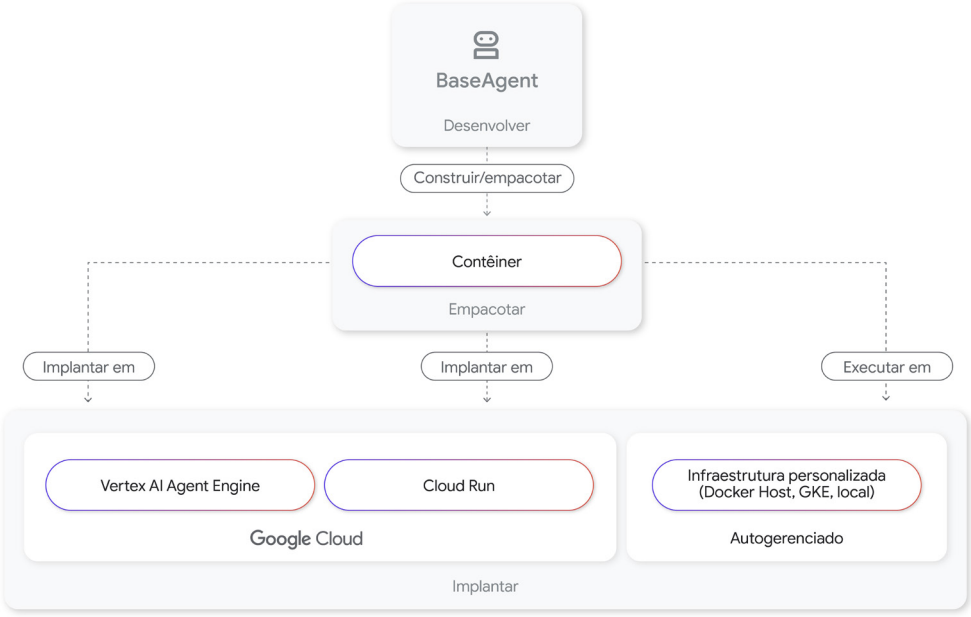
Trata-se de uma área ativa de pesquisa, mas os padrões iniciais já estão surgindo. Um exemplo é o Vertex AI Memory Bank, um serviço gerenciado no Vertex AI Agent Engine. Ele oferece mecanismos para implementar a destilação de memória:

- **Destilação automatizada:** pode processar historicamente conversas de forma assíncrona para extrair e gerar automaticamente uma lista de fatos relevantes sobre o usuário (GenerateMemories).
- **Destilação dirigida pelo agente:** para maior controle, o agente pode usar a memória como ferramenta para decidir quais informações são importantes o suficiente para serem explicitamente escritas no banco de memória (CreateMemory).

Focar em um conjunto destilado de memórias, em vez do histórico bruto, é mais escalável, eficiente e “humano”; é ideal para a nova geração de sistemas baseados em agentes.

Implante no ambiente gerenciado com o Vertex AI Agent Engine

Por design, o ADK é agnóstico quanto ao ambiente de implantação. A lógica central do agente que você define em Python é desacoplada da infraestrutura de execução, o que permite que você desenvolva e teste localmente, e depois implante o mesmo agente em diversos ambientes de produção.



Os agentes do ADK são disponibilizados para implantação como serviços web padrão usando FastAPI. O comando adk api_server empacota automaticamente seu agente em um servidor de API pronto para produção, que pode então ser transformado em contêiner.

Embora esse contêiner possa ser implantado em diversos serviços do Google Cloud, os três principais destinos gerenciados para implantação de agentes do ADK são:

- **Cloud Run:** uma plataforma de computação gerenciada para executar seu agente como um aplicativo baseado em contêiner. É uma excelente escolha para integrar seu agente a uma arquitetura de microsserviços existente ou para casos que exigem configurações personalizadas de contêiner.
- **Vertex AI Agent Engine:** serviço totalmente gerenciado e com escalabilidade automática no Google Cloud, projetado especificamente para implantar, gerenciar e escalar agentes de IA construídos com frameworks como o ADK. Oferece integração profunda com o ecossistema Vertex AI para MLOps, monitoramento e segurança.
- **Google Kubernetes Engine (GKE):** serviço Kubernetes gerenciado, ideal para quem já possui uma infraestrutura baseada em Kubernetes ou está tomando uma decisão estratégica desde o início para priorizar a portabilidade de longo prazo, o controle arquitetônico profundo e o ecossistema de código aberto do Kubernetes. Fornece um controle detalhado sobre a rede, cargas de trabalho com estado e hardware especializado como GPUs e TPUs, o que o torna ideal para equipes com expertise em engenharia de plataforma ou que estão construindo aplicativos complexos e escaláveis com múltiplos serviços.

Observação

É importante entender a relação entre as ferramentas de criação de agentes do Google Cloud. O Vertex AI Agent **Builder** é o conjunto completo de recursos para todo o ciclo de vida do agente, desde a descoberta até a implantação. Um componente central desse conjunto é o Vertex AI Agent Engine, o serviço gerenciado projetado especificamente para implantar, gerenciar e escalar seus agentes em produção.

Neste guia, quando falamos sobre o ambiente de execução em produção, estamos nos referindo ao Vertex AI Agent Engine.

Para as startups que estão usando o ADK, o Vertex AI Agent Engine é o destino de implantação recomendado. Ele é otimizado para ser uma solução econômica com escalabilidade automática, oferecendo o caminho mais fácil e direto para um agente escalável e pronto para produção. Como serviço totalmente gerenciado, ele abstrai a infraestrutura subjacente, permitindo que seus engenheiros se concentrem na lógica central do agente, sem sobrecarga operacional.

Como serviço projetado para cargas de trabalho de agentes, o Vertex AI Agent Engine oferece diversos benefícios importantes:

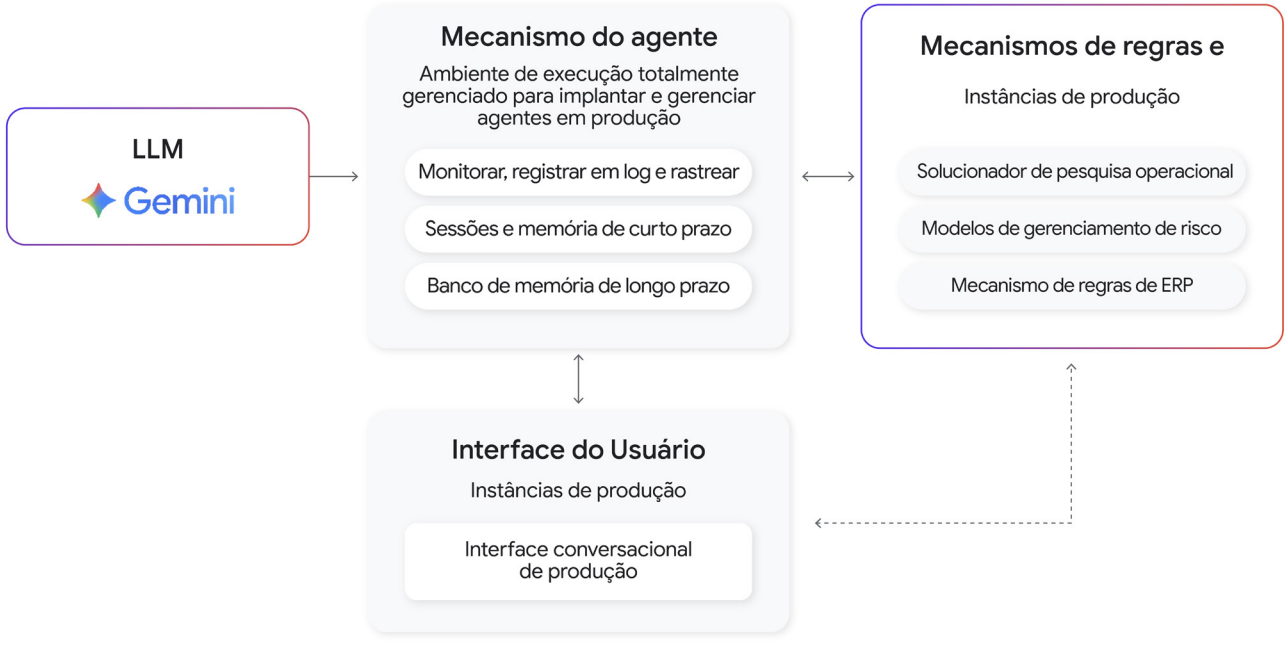
Principais recursos

- **Escalabilidade automatizada:** lida automaticamente com a variação de carga dos usuários.
- **Segurança e autenticação:** fornece gerenciamento integrado de identidade e acesso.
- **Independente de framework:** aceita agentes construídos com diversos frameworks, e não apenas o ADK.
- **Gerenciamento do ciclo de vida do agente:** fornece APIs para criar, ler, atualizar e excluir agentes implantados.

Recursos especializados para agentes

- Memory Bank: um serviço gerenciado para gerar e recuperar memórias personalizadas de longo prazo com base nas conversas dos usuários.
- Example Store: permite aos desenvolvedores fornecer e gerenciar exemplos few-shot para melhorar e direcionar o desempenho do agente em tarefas específicas.

Arquitetura do sistema para um mecanismo de agente alimentado pelo Gemini



Colabore com a comunicação Agent2Agent

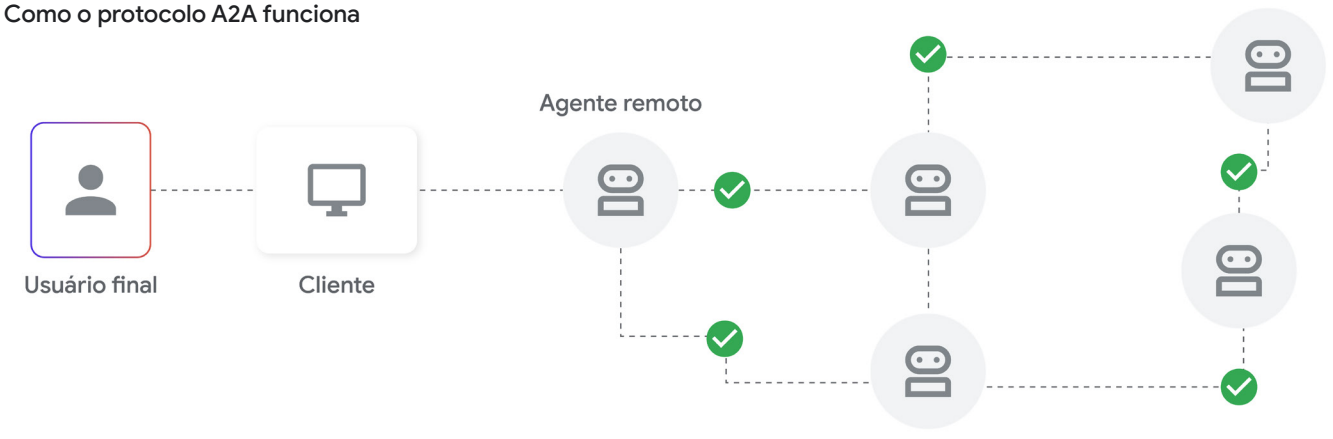
O verdadeiro poder dos agentes especializados é revelado quando eles conseguem colaborar. Para tornar isso possível, o Google promove o protocolo Agent2Agent (A2A), um padrão aberto que garante que os agentes que você constrói hoje possam descobrir, se comunicar e coordenar ações com segurança com outros agentes, independentemente de quem os criou ou qual framework utilizam. Esse compromisso com um ecossistema aberto e interoperável é central na estratégia de agentes do Google Cloud.

Os conceitos-chave do protocolo A2A são:

- **Cartão de agente:** um "cartão de visita" digital (geralmente um arquivo JSON hospedado em um endpoint conhecido) que o agente usa para divulgar suas capacidades, URL de acesso e requisitos de autenticação, permitindo que outros agentes o descubram.
- **Arquitetura baseada em tarefas:** as interações são estruturadas como "tarefas". Um agente cliente envia uma solicitação de tarefa a um agente servidor, que a processa e retorna uma resposta. Um agente pode atuar como cliente e servidor.
- **Independente de modalidade:** o A2A suporta comunicação por texto, áudio e vídeo, refletindo a natureza multimodal e em constante evolução das interações entre agentes.

Ecossistema de parceiros robusto do protocolo A2A

Como o protocolo A2A funciona



Os agentes do ADK podem participar nativamente desse ecossistema. Eles expõem um endpoint HTTP padrão e um arquivo agent.json, permitindo que sejam descobertos e se comuniquem com qualquer outro agente compatível com o protocolo A2A.

Dica

Explore estes recursos do A2A para começar:

- Organização do projeto A2A no GitHub
- Documentação do protocolo A2A
- Especificação do protocolo A2A

Caso de cliente

Como a BioCorteX usa o protocolo A2A para acelerar a descoberta de medicamentos.

A BioCorteX usa grafos de conhecimento e simulações in silico para modelar as interações complexas entre bactérias, medicamentos e o hospedeiro, revelando interações ocultas e reduzindo riscos no processo de desenvolvimento de fármacos para seus parceiros da indústria farmacêutica.

Situação

Na biociência, conectar e transformar conjuntos de dados díspares em conhecimento comercialmente relevante é um processo lento e incerto. O processo de testar hipóteses pode levar anos e é dificultado por modelos ruins, teorias conflitantes e pela complexidade da biologia.

Solução

A BioCorteX construiu um sistema multiagente no GCP que investiga hipóteses sob três dimensões: plausibilidade biológica, relevância clínica no mundo real e importância comercial. Ela usa agentes com a tecnologia Gemini, o ADK e um graphRAG para lidar com o nosso gráfico de conhecimento com 44 bilhões de conexões de amostras globais — tudo orquestrado via A2A.

Impacto

O que antes levava anos agora leva dias. Os agentes de grafo da BioCorteX oferecem um planejamento de cenários totalmente transparente para tomadores de decisão em todo o portfólio, sustentando decisões comerciais de alto nível com conhecimento científico profundo — o que acelera os testes de novos mecanismos e áreas terapêuticas e reduz desperdícios ao longo do pipeline.

> “Na BioCorteX, nossos agentes Carbon Graph são diferentes. Ao contrário de outros agentes, eles trabalham com fatos, e não opiniões ou associações. Ao navegar pelo maior grafo de conhecimento biológico mecanístico do mundo [em vez de usar um LLM], os agentes Carbon Graph não sugerem hipóteses, eles testam sua plausibilidade, relevância clínica e aspectos comerciais, permitindo o alinhamento total entre as equipes de P&D, Regulatória e Comercial da organização.”

2.2 Guia passo a passo: como definir um agente LLM

Construir um agente de IA é um processo iterativo de definição, testes e implantação. Esta seção se dedica à primeira e mais crítica fase: definir a identidade central, as instruções e os recursos do agente.

Embora o repositório de código aberto ADK Samples ofereça uma biblioteca abrangente de agentes prontos para uso, seu objetivo é mostrar como fica um agente finalizado. Esta seção, por sua vez, foi criada para ensinar como pensar na construção de um agente. Ela explica os princípios arquiteturais e o "porquê" estratégico por trás de cada componente central, oferecendo o conhecimento fundamental necessário para você usar os exemplos e criar suas próprias soluções personalizadas.

Para tornar isso prático, vamos seguir o processo de construção de um Software Bug Assistant, um agente LlmAgent projetado para ajudar uma equipe de suporte a filtrar novos relatórios de bugs.

1. Defina a identidade do agente

Primeiro, você estabelece o que o agente é e para que serve. Isso é feito com três parâmetros principais:

- **name (obrigatório):** um identificador único em formato de string, essencial para operações internas e delegação entre agentes. Exemplo: software_bug_triage_agent.
- **description (recomendado):** um resumo conciso das capacidades, usado por outros agentes para decidir quando encaminhar tarefas. Exemplo: "Analisa novos relatórios de bugs de software, categoriza sua prioridade e os direciona para a equipe de engenharia apropriada".
- **model (obrigatório):** o LLM subjacente que alimenta o raciocínio do agente, como gemini-2.5-flash.

2. Guie o agente com instruções

O parâmetro instruction é o componente mais crítico para moldar o comportamento do agente. Ele define a tarefa principal, a persona, as restrições e como usar suas ferramentas. Para o Software Bug Assistant, por exemplo, você o orientaria a agir como um gerente de engenharia experiente, explicaria como usar suas ferramentas para buscar dados de usuários e especificaria que seu output final deve ser um objeto JSON para o sistema de tickets.

Uma instrução eficaz deve:

- **Ser clara e específica** quanto aos resultados desejados.
- **Fornecer exemplos** (few-shot prompting) para tarefas complexas.
- **Orientar o uso de ferramentas** explicando quando e por que usá-las.
- **Injetar dados dinâmicos** do estado do agente usando a sintaxe {variável}.

Dica

Seja preciso, porque sua definição inteira é um prompt.

Um LLM usa cada parte da definição do agente — do nome e descrição até os nomes e descrições das ferramentas — para raciocinar. Ele interpreta essas informações com alto grau de literalidade. Evite nomes e descrições ambíguos, confusos ou contraditórios, que possam causar "envenenamento de contexto", levando o agente a se confundir, buscar objetivos errados ou usar ferramentas de forma incorreta. Trate cada string de configuração como uma instrução cuidadosamente elaborada para o modelo.

Observação

Os agentes de IA e seus frameworks estão evoluindo em ritmo acelerado. Embora este guia apresente os princípios e padrões de arquitetura sólidos para construir sistemas agênticos, os trechos de código e detalhes de API apresentados a seguir fornecem um retrato momentâneo. Nosso objetivo é ensinar o "porquê" e o "como" do design de agentes, e não fornecer código-fonte para ser copiado e colado diretamente em uma solução de produção.

Para detalhes atualizados de implementação, assinaturas de API e práticas recomendadas, consulte sempre a documentação oficial do ADK e o repositório ADK Samples.

3. Equipe o agente com ferramentas

Os recursos que as ferramentas fornecem vão além do raciocínio interno e permitem que ele interaja com o mundo externo. Nosso Software Bug Assistant precisaria de várias ferramentas para cumprir sua função, como:

- Uma função para obter informações sobre o usuário que relatou o bug (get_user_details(user_id)).
- Uma função para buscar arquivos relevantes no código-fonte (search_codebase(file_name)).
- Uma função para criar um ticket em um sistema de gerenciamento de projetos (create_jira_ticket(...)).

O LLM usa o nome da ferramenta, sua docstring e o esquema de parâmetros para decidir qual ferramenta chamar.

Dica

Seja breve e direto.

Cada ferramenta definida adiciona uma nova opção para o modelo considerar. Especialmente quando um agente possui muitas ferramentas, qualquer ambiguidade ou sobreposição nas descrições pode confundir o modelo, levando a comportamentos repetitivos ou seleção incorreta de ferramentas. Para garantir que o modelo escolha corretamente, o nome e a descrição de cada ferramenta devem refletir sua finalidade de forma clara e inequívoca.

4. Complete o ciclo de desenvolvimento

Agora você está pronto para testar e avaliar o desempenho do agente. Essa tarefa consiste em analisar a qualidade dos dados de saída do agente examinando sua execução passo a passo, ou "trajetória". A próxima seção aborda em detalhes o importante tema da avaliação de agentes.

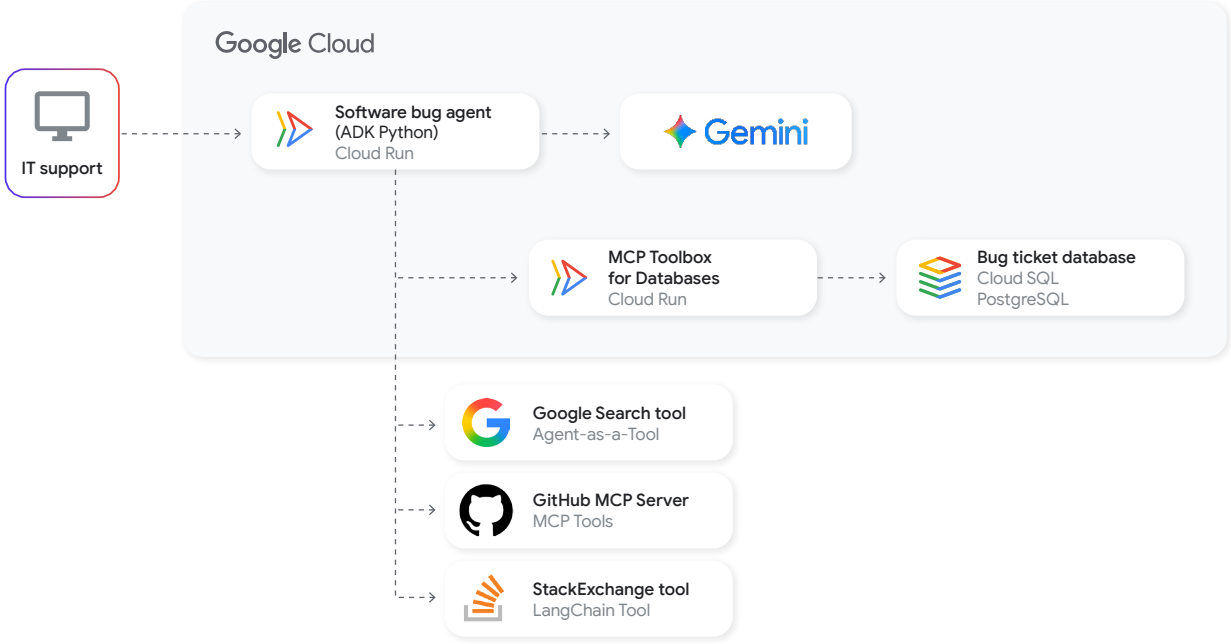
Quando tiver certeza de que ele está funcionando bem, você precisará de uma forma eficiente de implantá-lo. É nesse momento que seu protótipo se transforma em uma aplicação pronta para produção para sua equipe ou clientes, tornando seu agente uma ferramenta confiável de negócios.

Dica

Teste, teste e teste de novo

Os sistemas agênticos não são determinísticos e podem apresentar comportamentos emergentes. Os testes unitários padrão não bastam. A única forma de garantir a qualidade e confiabilidade do agente é fazer uma avaliação rigorosa. Faça os testes pensando em duas áreas principais: a trajetória de raciocínio (a lógica passo a passo e o uso de ferramentas) e a qualidade do output final (precisão, utilidade e fundamentação). Testes extensivos mostram que até mesmo modelos de última geração podem gerar alucinações ou se perder em loops de raciocínio, por isso a avaliação contínua é essencial durante o ciclo de desenvolvimento.

Arquitetura de um assistente de bug de software com ADK Python



Caso de cliente

Como a Box usa o ADK e o protocolo A2A para acelerar o desenvolvimento de conteúdo.

Box é uma plataforma de Gerenciamento Inteligente de Conteúdo que permite às organizações impulsionar a colaboração, gerenciar todo o ciclo de vida do conteúdo, proteger informações críticas e transformar fluxos de trabalho empresariais.

Situação

Processos empresariais críticos, como verificação de conformidade, gestão de contratos e aprovações de empréstimos, são atrasados porque os funcionários precisam buscar e interpretar grandes volumes de informações armazenadas em documentos na Box. Isso gera ineficiência e atrasa decisões importantes.

Solução

A Box está lançando um agente compatível com A2A, construído com o ADK do Google e baseado no Gemini. Esse agente se conecta diretamente à Box Intelligent Content Cloud, permitindo que os usuários façam perguntas complexas em linguagem natural e recebam respostas resumidas, contextualizadas e com insights extraídos instantaneamente de seus documentos.

Impacto

Ele acelera drasticamente os fluxos de trabalho centrados no conteúdo e melhora a qualidade das decisões. Além disso, ele cria uma base para agentes transacionais mais avançados, capazes de governar, gerenciar e iniciar processos, como assinaturas eletrônicas e aprovações, transformando fundamentalmente a forma como o trabalho é realizado na empresa.

> “Estamos entrando em uma nova era em que os agentes de IA transformarão a forma como o trabalho é feito — e o conteúdo está no centro de tudo. Com a Box como uma camada segura de conteúdo e o protocolo A2A do Google Cloud que permite uma colaboração fluida em todo o ecossistema, estamos viabilizando uma nova forma sofisticada de automatizar processos empresariais, acelerar decisões e gerar resultados concretos para nossos clientes”.

box

2.3 Gerencie e escale sua força de trabalho de agentes

Ao passar da construção de um único agente para a implantação de um portfólio de agentes especializados, sua startup se depara com novos desafios. Como gerenciá-los? Como membros não técnicos da equipe podem aproveitá-los? Como controlar o acesso a dados e ferramentas?

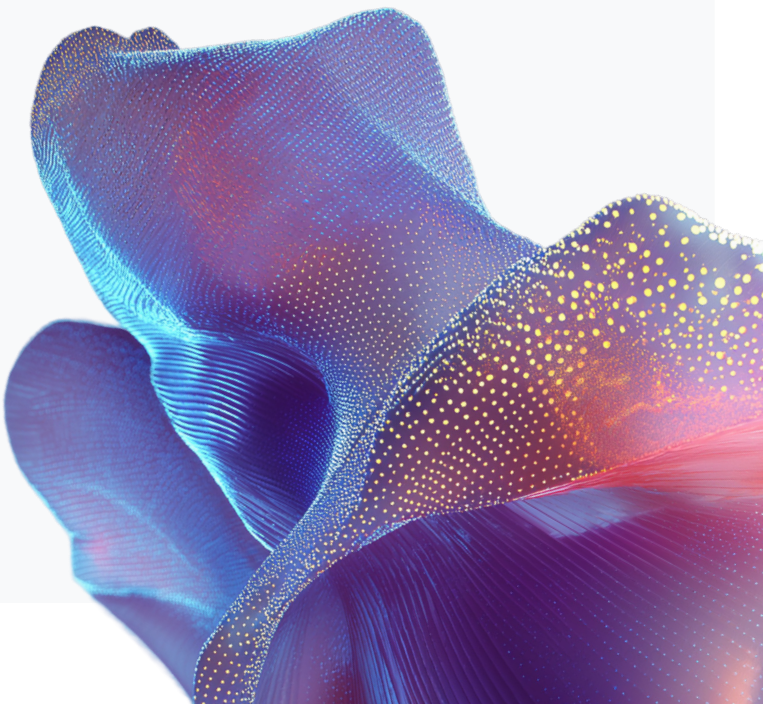
O Gemini Enterprise resolve essas questões de escalabilidade. Essa plataforma segura sozinha permite criar, governar e orquestrar toda a sua força de trabalho de agentes de IA, unificando aplicativos e fontes de dados distintas. Ela complementa o desenvolvimento baseado em código do ADK ao fornecer a estrutura necessária para escalar o uso de agentes em toda a organização e gerenciá-los de forma eficaz.

Com o Gemini Enterprise, você pode:

- **Unificar e acessar dados da empresa:** o Gemini Enterprise elimina silos de dados usando conectores prontos para seus aplicativos corporativos existentes (como Microsoft SharePoint, Google Workspace, Jira). Ele aplica a tecnologia de pesquisa multimodal do Google sobre esses dados conectados, permitindo que qualquer funcionário obtenha respostas instantâneas e sintetize insights a partir de uma fonte central da verdade, respeitando todos os controles de acesso existentes.
- **Habilitar a automação em toda a equipe:** enquanto o ADK é ideal para o desenvolvimento complexo de agentes com base em código, o Gemini Enterprise capacita toda a organização a automatizar fluxos de trabalho. Especialistas de domínio em produto, marketing ou operações podem usar o Agent Designer, sem programação de código, para criar seus próprios agentes personalizados com uma interface baseada em prompts. Isso transforma seu conhecimento específico em soluções automatizadas sem depender de recursos de engenharia.
- **Governar e orquestrar uma frota de agentes:** o Gemini Enterprise oferece uma plataforma única para gerenciar e governar agentes construídos com ADK, com o designer sem código ou por parceiros. A Agent Gallery funciona como um portal central para sua equipe descobrir, gerenciar e implantar todos os agentes, incluindo soluções personalizadas e agentes prontos do Google para tarefas complexas como pesquisa profunda ou geração de ideias.

Experimente estes prompts no Gemini Enterprise:

> Agende a nossa reunião semanal de equipe para quinta-feira às 10h.
>
> Resuma as atualizações da semana para o canal #produto no Slack.
>
> Crie uma pauta de reunião para discutir a preparação com investidores.



Caso de cliente

Como o agente de IA do Zoom agenda reuniões automaticamente a partir do Gmail usando o Google Agentspace [agora Gemini Enterprise]

A Zoom é uma empresa de tecnologia de comunicação que oferece uma plataforma aberta e baseada em IA para reuniões virtuais, webinars, chat, colaboração online, experiência do cliente e muito mais.

Situação

A estratégia de IA da Zoom busca transformar o AI Companion em uma estrutura totalmente agêntica, não apenas capaz de raciocínio avançado e orquestração de tarefas, mas também integrada de forma fluida aos principais sistemas de terceiros dos clientes. Ao permitir a colaboração com outros agentes de IA, a Zoom promove resultados de trabalho mais significativos por meio de um ecossistema aberto e interoperável

Solution

O Zoom AI Companion está sendo integrado ao Gemini Enterprise para simplificar o agendamento de reuniões. Com lançamento previsto para meados deste ano, essa colaboração permitirá que agentes de IA compatíveis com A2A agendem automaticamente reuniões do Zoom a partir do contexto do Gmail, atualizem o Google Agenda e mantenham os participantes informados, eliminando o vai e vem do agendamento manual

Impacto

- Redução de barreiras técnicas na integração de IA entre plataformas.
- Interação fluida entre o Zoom AI Companion e agentes externos compatíveis com A2A, sem necessidade de código personalizado.
- Automação de fluxos de trabalho aprimorada e maior eficiência para clientes corporativos.
- Suporte futuro para interações multiagente mais sofisticadas.

> “Nossa contribuição para o protocolo A2A permite uma integração mais profunda com o Google Cloud e outras plataformas de terceiros, oferecendo aos clientes flexibilidade e liberdade de escolha”.
>
> zoom

2.4 Outras opções para construir agentes

Experimente o Gemini CLI

Para startups que precisam de uma forma imediata e econômica de experimentar IA, o Gemini CLI é um agente de código aberto que leva o Gemini direto para o terminal. Ele oferece:

- **Economia significativa:** acesso gratuito ao Gemini 2.5 com limites generosos de uso (contexto de 1 milhão de tokens, 60 consultas por minuto)
- **Aumento de produtividade:** ao se integrar aos fluxos de trabalho existentes dos desenvolvedores, ele acelera a programação de código, a depuração e a documentação.
- **Flexibilidade total:** como é uma ferramenta de código aberto (Apache 2.0), ele pode ser auditado, modificado e incorporado à sua cadeia de ferramentas, evitando a dependência de fornecedores e permitindo uma personalização profunda.

Dica

Confira o Gemini CLI configurado para desenvolvimento com o ADK.

Acelere o desenvolvimento com o Firebase Studio

Um agente backend, mesmo sendo robusto e construído com o ADK, representa apenas parte de um produto completo. Para que ele funcione, é necessário construir um aplicativo de stack completo como base, com interface do usuário, bancos de dados e hospedagem. O Firebase Studio é um ambiente integrado baseado em nuvem que usa IA agêntica para acelerar todo o ciclo de desenvolvimento. As equipes podem usá-lo para lidar com tudo, desde prototipagem de UI e geração de código até implantação segura na infraestrutura do Google Cloud.

Combinando o ADK para a lógica de backend do agente, o Agent Starter Pack para infraestrutura de produção (abordado na seção 3) e o Firebase Studio para o aplicativo de stack completo, você tem o conjunto de ferramentas necessário para que uma startup construa e implante um sistema agêntico robusto e de última geração.

O Firebase Studio acelera todo o ciclo de desenvolvimento com IA:

- **Configuração rápida:** use o App Prototyping Agent para criar um novo projeto usando linguagem natural, maquetes ou capturas de tela. Comece explorando o amplo catálogo de modelos para frameworks e linguagens populares, ou importe um projeto existente.
- **Gemini no Firebase:** use a assistência de IA para tarefas como programação de código, depuração, testes, refatoração, explicação e documentação de código.
- **Colaboração:** compartilhe ambientes de trabalho com membros da equipe e forneça uma URL para que os primeiros testadores possam visualizar os aplicativos.
- **Otimização:** visualize os apps como os usuários os veem com prévias para web integradas e emuladores Android, e otimize com acesso a milhares de extensões no repositório Open VSX.
- **Implantação:** publique no Firebase App Hosting com poucos cliques ou implante apps de produção no Cloud Run, Firebase Hosting ou em sua própria infraestrutura personalizada.

Experimente estes prompts com o App Prototyping Agent:

> Gere um painel de atendimento ao cliente que consuma dados do Zendesk e exiba métricas importantes como volume de tickets e tempo de resolução.

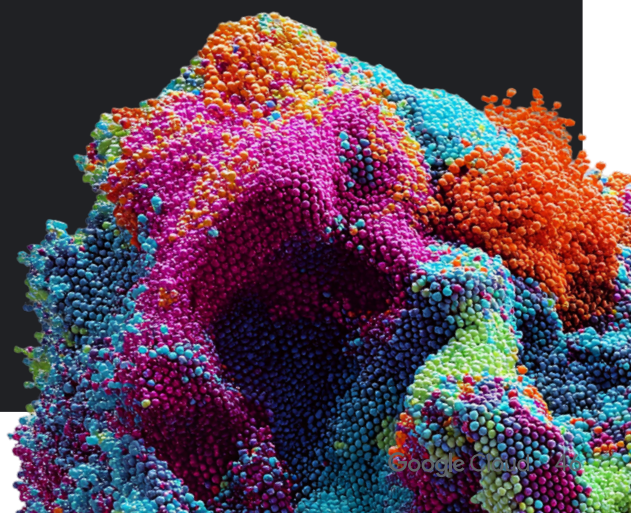
> Crie um aplicativo B2B SaaS com autenticação de usuário, banco de dados PostgreSQL e uma página de cobrança por assinatura.

> Construa um aplicativo completo para rastreamento interno de bugs, com formulário de envio e um quadro kanban para visualizar o status dos tickets.

Você também pode começar explorando um catálogo amplo de modelos para frameworks e linguagens populares, ou importar um projeto existente.

Principais aprendizados: da construção à escala

Seu objetivo	Melhor opção
Construir um sistema personalizado com múltiplos agentes a partir de código.	Use o kit de desenvolvimento de código aberto Agent Development Kit (ADK).
Implantar, escalar e gerenciar seu agente em produção.	Implante no Vertex AI Agent Engine.
Dar ao seu agente a capacidade de memória de longo prazo.	Use o recurso Memory Bank dentro do Vertex AI Agent Engine.
Permitir que seu agente descubra e converse com outros agentes.	Permita a comunicação entre agentes no Vertex AI Agent Engine.
Criar um aplicativo completo com IA a partir de um prompt.	Use o Firebase Studio como ambiente de desenvolvimento assistido por IA.
Experimentar rapidamente o Gemini no seu terminal.	Use o Gemini CLI para uma interface simples via linha de comando.



Pronto para transformar sua visão de IA em realidade? Estamos aqui para ajudar.

Aprenda a criar mais aplicativos de IA generativa com as sessões sob demanda da Startup School.

Comece agora

Receba até US$ 350 mil em créditos do Google Cloud com o programa Google Cloud for Startups.

Inscreva-se já

Entre em contato com a nossa Equipe de Startups.

Fale com a gente

Fique atualizado e receba todas as nossas novidades assinando a newsletter do Google Cloud para Startups.

Assinar

Seção 3

Como garantir que os agentes de IA sejam confiáveis e responsáveis

Devido à natureza não determinística dos sistemas baseados em LLM, pode ser difícil alcançar confiabilidade em nível de produção. Para não se restringir a testes superficiais, é preciso adotar uma abordagem de engenharia rigorosa que garanta que o

Esta seção apresenta as metodologias e ferramentas necessárias para enfrentar esses desafios, com foco em três áreas principais:

- **Exatidão e confiabilidade:** avaliar a precisão do resultado final e a validade dos passos intermediários de raciocínio.
- **Desempenho e escalabilidade:** medir e otimizar a latência e a capacidade de resposta do agente sob alta demanda.
- **Segurança e responsabilidade:** implementar mecanismos de proteção, monitorar comportamentos indesejados e garantir que o agente opere dentro de limites definidos.

Essas práticas representam a aplicação concreta do compromisso do Google com a IA responsável, permitindo que as startups construam agentes robustos e confiáveis, alinhados aos princípios líderes do setor em segurança.

> **Os agentes são a chave para um novo nível de produtividade, mas seu sucesso depende da nossa orientação".**
>
> **Harrison Chase**
> CEO e cofundador da LangChain

Dica

A observabilidade em nível de produção vai além das métricas de aplicativo. É preciso monitorar também as métricas operacionais de baixo nível, como uso de CPU e memória. É fundamental acompanhar o consumo de recursos com precisão para diagnosticar gargalos de desempenho, otimizar o tempo de execução e reduzir diretamente os custos operacionais. O ADK e o Agent Starter Pack oferecem suporte nativo ao OpenTelemetry, permitindo enviar dados operacionais críticos diretamente para suas ferramentas de monitoramento.

Prefere ouvir? Confira a versão em podcast desta seção, criada com o NotebookLM.

Este podcast foi criado usando o NotebookLM com o seguinte prompt: "Como apresentador de podcast, gere um episódio para desenvolvedores e fundadores técnicos. Apresente o AgentOps como o framework para construir uma IA confiável, indo além dos testes informais para um processo rigoroso e automatizado.

Explique como o AgentOps avalia o raciocínio, a precisão e a segurança de um agente, e como ele mitiga riscos como desinformação e vulnerabilidades de segurança. Descreva como o Agent Starter Pack, trabalhando com o Agent Development Kit (ADK), implementa rapidamente isso com ferramentas pré-configuradas para infraestrutura, CI/CD e avaliação contínua. Conclua destacando que essa abordagem disciplinada é uma vantagem competitiva e indique aos ouvintes recursos do Google para startups."

3.1 AgentOps: um framework para agentes prontos para produção

Agent Operations (AgentOps) é uma metodologia operacional que aborda os desafios de confiabilidade e responsabilidade em produção. Ela adapta os princípios de DevOps, MLOps e DataOps aos desafios únicos de construir, implantar e gerenciar agentes de IA ao longo de seu ciclo de vida. Ela também oferece uma estrutura sistemática, automatizada e reproduzível para lidar com as complexidades dos sistemas baseados em LLM não determinísticos em ambientes de produção.

Uma estratégia robusta de AgentOps sistematiza o processo de desenvolvimento, fornecendo ciclos de feedback contínuos para melhorar a confiabilidade, a segurança e o desempenho de um agente em suas ferramentas, capacidade de raciocínio e modelos subjacentes.

Um framework sistemático para avaliação de agentes

Avaliar sistemas agentes não determinísticos é um dos desafios mais complexos da engenharia de software moderna. Os testes tradicionais geralmente se concentram na correção lexical, mas a avaliação de agentes deve abordar dois problemas mais difíceis: correção semântica (o agente entendeu e respondeu de forma útil à intenção do usuário?) e correção de raciocínio (o agente seguiu um caminho lógico e eficiente até sua conclusão?).

Como mostramos na seção 1, a arquitetura cognitiva que governa esse raciocínio é frequentemente um framework como o ReAct, que estabelece um ciclo dinâmico onde o agente intercala pensamento e ação. Uma falha em qualquer ponto desse ciclo pode produzir um resultado incorreto. Portanto, é necessária uma estrutura de avaliação rigorosa e em várias camadas. Essa estrutura é implementada usando uma combinação do ADK para a lógica central do agente e a instrumentação, e do Agent Starter Pack para a infraestrutura de produção, automação e observabilidade.

Camada 1: avaliação no nível de componente (testes unitários determinísticos)

Essa camada foca nos componentes previsíveis e não baseados em LLM do sistema do agente.

- **Objetivo:** verificar a correção lexical de cada bloco de construção, garantindo que bugs simples nos componentes do agente não causem falhas.
- **O que testar:**
  - **Ferramentas:** comportamento esperado com entradas válidas, inválidas e de casos extremos.
  - **Processamento de dados:** robustez das funções de parsing e serialização.
  - **Integrações de API:** tratamento de condições de sucesso, erro e timeout.
- **Implementação:**
  - O ADK define as ferramentas do agente como funções Python (ou métodos Java). Essas funções são os alvos diretos dos testes em nível de componente.
  - O Agent Starter Pack fornece a infraestrutura de testes. Ele gera um projeto com um ambiente padrão de pytest configurado no diretório tests/unit/. Os desenvolvedores podem imediatamente escrever testes unitários para suas ferramentas definidas no ADK e executá-los com o comando make test.

Camada 2: avaliação de trajetória (correção procedimental)

É a camada mais crítica para avaliar o processo de raciocínio do agente. Uma "trajetória" é a sequência completa de etapas de Raciocinar, Agir e Observar que o agente realiza para concluir uma tarefa.

- **Objetivo:** verificar a correção do raciocínio do agente dentro do ciclo ReAct.
- **O que testar:**
  - **Etapa de raciocínio:** o agente avalia corretamente o objetivo e o estado atual para formar uma hipótese lógica para o próximo passo?
  - **Etapa de ação:** ele seleciona corretamente a ferramenta (Seleção de Ferramenta) e extrai e formata corretamente os argumentos para essa ferramenta (Geração de Parâmetros)?
  - **Etapa de observação:** ele integra corretamente os dados de saída da ferramenta para informar a próxima etapa de Raciocinar do ciclo?
- **Implementação:**
  - O runtime central do ADK executa o ciclo ReAct do agente, integrando-se diretamente ao Google Cloud Trace para instrumentar cada etapa de Raciocinar, Agir e Observar. Isso permite aos desenvolvedores visualizar toda a trajetória, inspecionar os dados de entrada e de saída das ferramentas, e examinar a "cadeia de pensamentos" do modelo para depurar seu raciocínio.
  - O Agent Starter Pack automatiza e escala a avaliação de trajetória. Um diretório tests/integration/ cria um "conjunto de ouro" de prompts com trajetórias ReAct esperadas. O pipeline automatizado de CI/CD (configurado com agent-starter-pack setup-cicd) executa esses testes em cada pull request para evitar regressões. A infraestrutura de observabilidade do starter pack é o que captura os dados de rastreamento emitidos pelo agente ADK.

Camada 3: Avaliação de resultado (correção semântica)

Essa camada avalia a resposta final, voltada para o usuário, gerada após a conclusão do ciclo ReAct.

- **Objetivo:** verificar a correção semântica, precisão factual e qualidade geral da resposta final.
- **O que testar:**
  - **Precisão factual e grounding:** a resposta está correta e baseada de forma verificável nas informações coletadas durante as etapas de Observar?
  - **Utilidade e tom:** a resposta atende plenamente à necessidade do usuário com o estilo apropriado?
  - **Abrangência:** a resposta contém todas as informações necessárias?
- **Implementação:**
  - O conjunto de ferramentas do ADK é essencial para verificar a precisão factual. Os desenvolvedores podem criar ferramentas especializadas ou usar APIs para a verificação de grounding. Essas ferramentas, chamadas durante a etapa de Agir, verificam programaticamente se a resposta final do agente é sustentada pelo contexto recuperado, fornecendo uma medida quantitativa contra alucinações.
  - O Agent Starter Pack oferece a base necessária para realizar essas avaliações em larga escala. Ele se conecta ao serviço de avaliação Gen AI da Vertex AI, que utiliza modelos de linguagem como juízes para atribuir pontuações. Além disso, sua interface interativa conta com recursos de feedback que permitem registrar avaliações feitas por humanos diretamente no BigQuery, viabilizando uma análise com validação humana (HITL) de alta precisão.

Camada 4: monitoramento em nível de sistema (em produção)

A avaliação não termina com a implantação. É fundamental monitorar continuamente o desempenho ao vivo do agente.

- **Objetivo:** acompanhar o desempenho real e detectar falhas operacionais ou desvios de comportamento.
- **O que testar:** taxas de falha das ferramentas, pontuações de feedback dos usuários, métricas de trajetória (por exemplo, número de ciclos ReAct por tarefa) e latência de ponta a ponta.

Implementação:

- O agente ADK, executado em produção, é a fonte dos dados operacionais, que emite eventos e rastreamentos para cada interação ao vivo com o usuário.
- O Agent Starter Pack fornece um stack de observabilidade de nível de produção pronto para uso. Ele configura automaticamente o OpenTelemetry, um roteador de logs para o BigQuery e fornece modelos para dashboards no Looker Studio. Isso permite que as equipes acompanhem imediatamente o desempenho do agente, analisem tendências e depurem problemas usando dados de uso real sem configuração adicional.

Essa metodologia abrangente e prática para avaliação de agentes é a implementação tangível de uma estratégia robusta de AgentOps, levando as equipes além dos testes informais baseados em "vibe testing" para um processo sistemático, automatizado e reproduzível. Ao dividir a avaliação em componentes, trajetória, resultado e monitoramento em nível de sistema, ela aborda diretamente os domínios centrais do AgentOps.

Adotar uma estrutura sistemática de avaliação não é apenas uma boa prática, mas uma vantagem competitiva. Ela estabelece um processo rigoroso, baseado em dados e automatizado que ajuda as equipes a inovar mais rápido, implantar com confiança e construir agentes comprovadamente mais seguros e eficazes.

Kit de ferramentas de AgentOps: o ADK e Agent Starter Pack

Pipelines automatizados de CI/CD implementam os princípios do AgentOps, de modo que qualquer alteração no código, ferramentas ou prompts do agente acione um processo padronizado de construção, testes unitários e avaliação quantitativa contra um conjunto de dados predefinido. Essa etapa de avaliação automatizada é essencial para evitar regressões e fornece feedback contínuo e objetivo sobre o desempenho e a segurança do agente antes da implantação.

Para acelerar a adoção dos princípios do AgentOps, o Agent Starter Pack fornece uma implementação de referência pronta para produção. Seus modelos holísticos abordam desafios comuns (por exemplo, implantação e operações, avaliação, personalização e observabilidade) ao construir e implantar agentes de IA. Simplificando, ele inicializa um novo projeto de agente com a infraestrutura e os pipelines necessários, para que os desenvolvedores possam focar na lógica central.

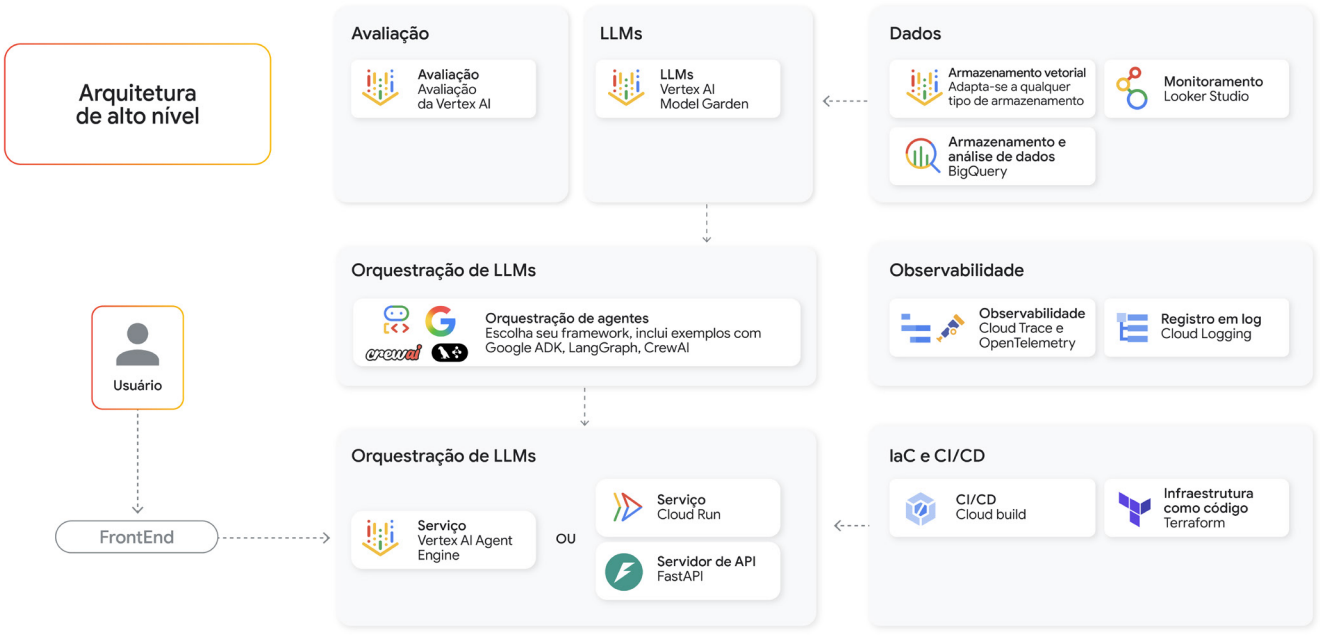
Dica

Você pode criar um novo projeto de agente pronto para produção com um único comando: uvx agent-starter-pack create my-agent -a adk@gemini-fullstack.

O Agent Starter Pack inclui estes componentes principais:

- **Infraestrutura como Código (Terraform):** fornece modelos reproduzíveis para provisionar e gerenciar o ambiente em nuvem do agente, incluindo serviços como Cloud Run, permissões IAM e rede.
- **Pipelines de CI/CD (Cloud Build):** um arquivo cloudbuild.yaml pré-configurado automatiza o processo de build, testes unitários, avaliação quantitativa e implantação, implementando diretamente o fluxo de trabalho CI/CD do AgentOps CI/CD.
- **Observabilidade e registro em log (Cloud Trace e Cloud Logging):** estabelece a base para monitoramento e depuração ao integrar o Cloud Trace para uma análise aprofundada de rastreamentos de execução do agente e o Cloud Logging para o gerenciamento centralizado de logs.
- **Integração de dados (BigQuery):** inclui componentes fundamentais para agentes que precisam se conectar e analisar dados estruturados empresariais usando o BigQuery.
- **Avaliação contínua (avaliação da Vertex AI):** integra-se à Vertex AI para executar conjuntos de dados de avaliação em relação às alterações do agente, medindo continuamente o desempenho nos principais domínios discutidos anteriormente.

Arquitetura de alto nível do Agent Starter Pack



Melhor juntos: ADK e Agent Starter Pack

O ADK e o Agent Starter Pack foram projetados para oferecer uma separação clara entre a lógica de aplicativo de um agente e seu ciclo de vida operacional, permitindo um processo de desenvolvimento robusto e escalável.

O ADK é usado basicamente para escrever o código do aplicativo do agente, enquanto o Agent Starter Pack fornece a base operacional pronta para produção para executar e gerenciar esse código em larga escala.

- **O ADK gerencia o comportamento em tempo de execução do agente:** como um SDK em Python/Java, o ADK oferece APIs e abstrações centrais para definir a lógica da aplicação do agente. Os desenvolvedores o utilizam para implementar fluxos de orquestração, definir ferramentas e configurar interações com os LLMs.
- **O Agent Starter Pack gerencia o ambiente operacional:** como uma ferramenta de estruturação, ele gera a infraestrutura como código (Terraform) para provisionar os alvos de implantação (por exemplo, Cloud Run) e as configurações de pipeline CI/CD (Cloud Build) para automatizar todo o ciclo de vida.

Essa separação se manifesta em um fluxo de trabalho de cinco etapas:

1. **Inicialização com o Agent Starter Pack:** o desenvolvedor executa um único comando para gerar um novo projeto contendo todos os componentes operacionais necessários, incluindo arquivos Terraform para infraestrutura, configurações do Cloud Build para CI/CD e arquivos modelo para conjuntos de dados de avaliação.
2. **Desenvolvimento com o ADK:** dentro dessa estrutura, o desenvolvedor usa o ADK para escrever a lógica do aplicativo do agente, implementando ferramentas personalizadas, compondo agentes e escrevendo as instruções principais.
3. **Commit e automação:** quando o código é enviado ao repositório de origem, o pipeline CI/CD pré-configurado gerenciado pelo Cloud Build é acionado automaticamente.
4. **Avaliação contínua:** o pipeline constrói o agente ADK em um contêiner e executa uma avaliação quantitativa com base em um conjunto de testes predefinido, validando programaticamente o desempenho e a segurança do agente.
5. **Implantação com confiança:** após uma avaliação bem-sucedida, o pipeline implanta automaticamente a nova versão validada do agente em seu ambiente de produção.

Ao integrar o framework de desenvolvimento do ADK com a automação operacional do Agent Starter Pack, você estabelece um processo completo de MLOps/DevOps, especificamente voltado para construir e gerenciar agentes de IA prontos para produção. É o AgentOps em larga escala.

3.2 Construa agentes de IA responsáveis e seguros com o AgentOps

Ao criar agentes poderosos, é preciso garantir, sem exceções, que eles sejam seguros, protegidos e alinhados com princípios éticos. Isso significa projetá-los desde o início com salvaguardas para evitar resultados prejudiciais ou indesejados, incluindo preconceitos injustos, violações de privacidade e vulnerabilidades de segurança.

Para lidar com essa questão, é essencial adotar uma abordagem estruturada. O diagrama abaixo oferece uma visão geral de alto nível dos principais riscos envolvidos e das salvaguardas técnicas e procedimentais usadas para mitigá-los. Embora sirva como um excelente ponto de partida, se quiser consultar um guia abrangente de padrões e práticas recomendadas, recomendamos fortemente o Secure AI Framework (SAIF) do Google.

> **À medida que os agentes de IA se integramà nossa vida, é essencial enfrentarmos os novos desafios relacionados à confiança, privacidade e segurança. É importante refletirmos sobre segurança e privacidade, e nos perguntarmos: como construímos produtos confiáveis?**
>
> **Jia Li**
> Co-Founder, President
> and Chief AI Officer of LiveX AI

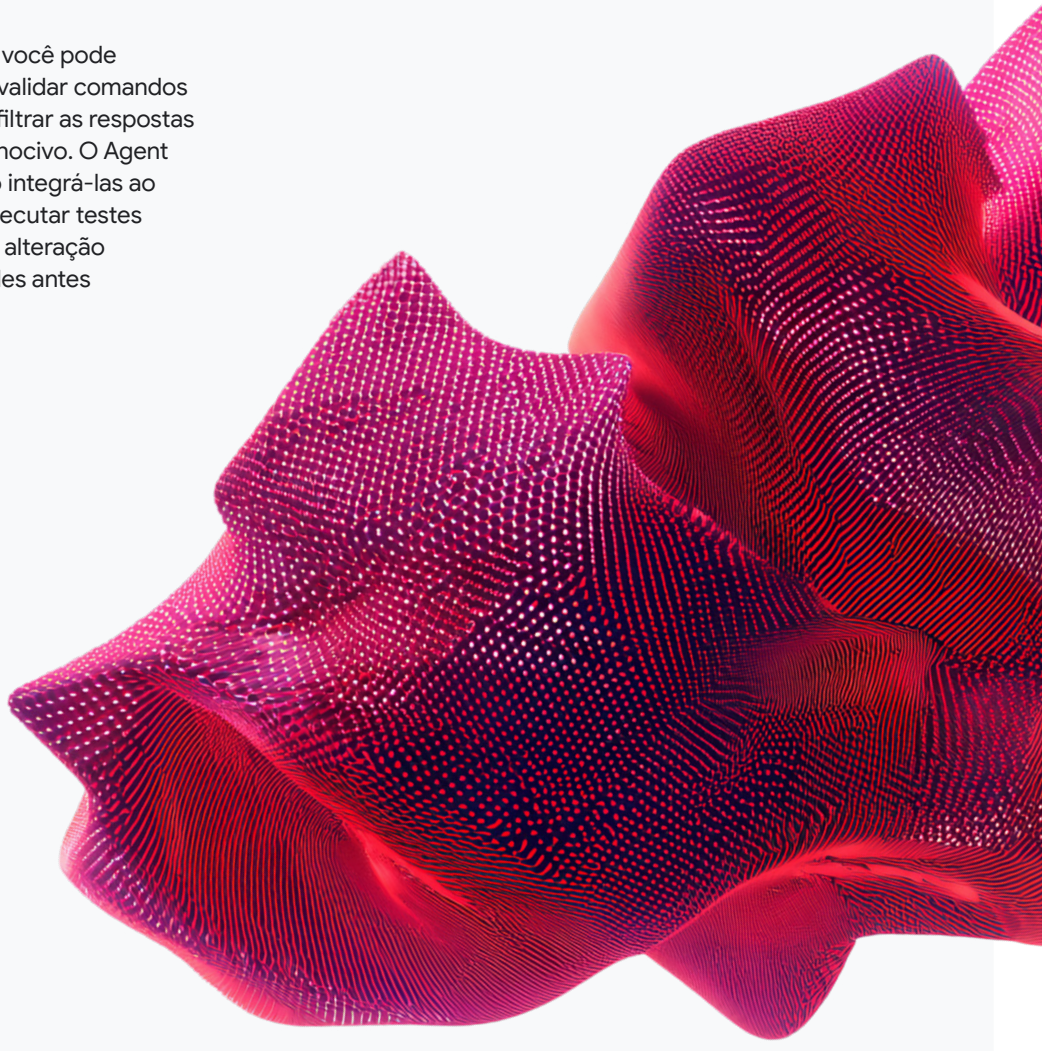
.

ADK e o Agent Starter Pack oferecem uma estratégia de defesa em profundidade para essa área crítica. Primeiro, você pode implementar controles de segurança granulares no nível do aplicativo com o ADK. Em segundo lugar, o Agent Starter Pack automatiza a implantação de uma infraestrutura de nuvem reforçada que aplica esses controles em larga escala.

Essa abordagem combinada aborda aspectos essenciais de segurança e conformidade regulatória:

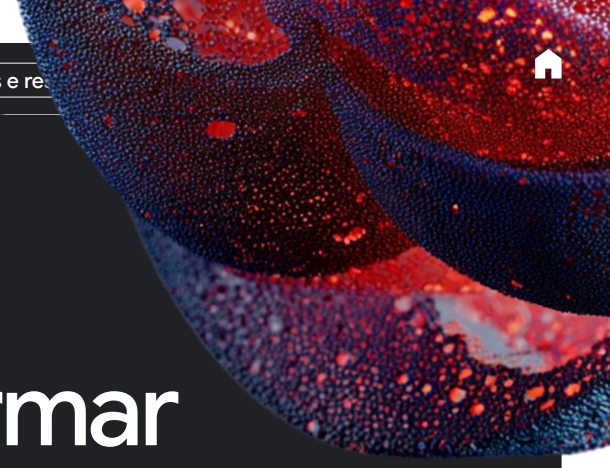
- **Infraestrutura segura e controle de acesso:** o Agent Starter Pack usa o Terraform para provisionar uma base segura, implantando seu agente em ambientes como o Cloud Run e configurando funções específicas de IAM para aplicar o princípio do menor privilégio. As ferramentas que você define no ADK operam dentro dessas permissões rígidas no nível da nuvem, garantindo que o agente não possa acessar recursos não autorizados, mesmo que sua lógica interna seja comprometida.
- **Barreiras de entrada e saída:** no ADK, você pode implementar lógica de aplicativo para validar comandos contra possíveis ataques de injeção e filtrar as respostas finais do agente para evitar conteúdo nocivo. O Agent Starter Pack reforça essas barreiras ao integrá-las ao pipeline de CI/CD. Assim, é possível executar testes automatizados de segurança em cada alteração de código para verificar vulnerabilidades antes que cheguem à produção.
- **Auditoria e monitoramento:** a observabilidade detalhada no ADK cria um rastreamento granular de cada pensamento e a chamada de ferramenta feita pelo agente. O Agent Starter Pack operacionaliza isso ao configurar automaticamente destinos de logs que enviam esses dados para o BigQuery, garantindo um armazenamento seguro e de longo prazo. Isso cria um histórico de auditoria durável, essencial para revisões de conformidade regulatória e resposta a incidentes.

A segurança é um trabalho conjunto. Enquanto o ADK fornece a estrutura para a arquitetura cognitiva do agente e o Agent Starter Pack oferece os componentes para sua implantação, os dois operam dentro do grande ecossistema do Google Cloud. Tudo isso proporciona uma postura de segurança robusta, construída sobre uma base segura por design, com controles integrados projetados para proteger qualquer carga de trabalho.



Principais aprendizados: Como construir agentes confiáveis

Seu objetivo	Melhor opção
Gerenciar profissionalmente o ciclo de vida do seu agente.	Adote o AgentOps para automatizar processos, desde o desenvolvimento até a implantação e o monitoramento.
Garantir que seu agente seja preciso e seguro antes de entrar em produção.	Implemente avaliações automatizadas no seu pipeline de CI/CD para testar rigorosamente a qualidade, o grounding e a segurança.
Acompanhar o desempenho real, o custo e os erros do seu agente.	Configure o monitoramento com ferramentas de observabilidade para obter dados em tempo real sobre latência, uso de tokens e taxa de sucesso nas chamadas de ferramentas.
Entender por que seu agente tomou uma determinada decisão.	Inspecione a trajetória do agente (sua "cadeia de pensamentos") usando ferramentas de registro e rastreamento para depurar o processo de raciocínio.
Proteger seu agente, seus dados e o acesso às ferramentas.	Aplique os princípios de segurança do AgentOps, que incluem segurança da infraestrutura, governança de dados e controles de conformidade.
Começar rapidamente a usar o AgentOps.	Use o Agent Starter Pack com modelos pré-configurados para CI/CD, avaliação e infraestrutura.



Pronto para transformar sua visão de IA em realidade? Estamos aqui para ajudar.

Aprenda a criar mais aplicativos de IA generativa com as sessões sob demanda da Startup School.

Comece agora

Receba até US$ 350 mil em créditos do Google Cloud com o Programa Google Cloud for Startups

Inscreva-se já

Entre em contato com a nossa Equipe de Startups.

Fale com a gente

Fique atualizado e receba todas as nossas novidades assinando a newsletter do Google Cloud para Startups.

Assinar

Mais do stack completo de IA do Google

Crie rapidamente com o Gemini no Google AI Studio.

Veja todos os modelos Gemini disponíveis.

Explorar agora

Destaque

Gemini 2.5 Flash Image (também conhecido como Nano Banana)

Esse modelo de geração e edição de imagens permite combinar várias imagens em uma só, manter a consistência dos personagens para narrativas envolventes, realizar transformações específicas usando linguagem natural e aproveitar o conhecimento global do Gemini para gerar e editar imagens.

Destaque

Veo e Imagen

Esses modelos de ponta permitem gerar vídeos e imagens de alta qualidade a partir de comandos de texto, editar elementos visuais existentes com linguagem natural e criar experiências narrativas envolventes com recursos avançados de síntese visual.

PROMPT: O homem olha para cima e sorri para a câmera.

PROMPT: O cachorro se levanta, balançando o rabo, feliz e olhando para a câmera.

Conclusão

A jornada de um protótipo a um sistema pronto para produção é uma questão de engenharia disciplinada. Ao usar um framework code-first como o ADK e os princípios operacionais contidos neste guia, você pode ir além do teste de feeling informal para um processo rigoroso e confiável para construir e gerenciar todo o ciclo de vida do seu agente.

Para sua startup, essa abordagem disciplinada se torna uma grande vantagem competitiva. Sua equipe pode iterar e inovar mais rápido, automatizando avaliações de alto consumo de recursos ao longo do caminho. Além disso, é possível escalar com confiança, sem comprometer a segurança ou a proteção.

Como este guia demonstrou, o Google Cloud apoia essa inovação, com o hardware de IA desenvolvido sob medida e uma plataforma de dados unificada, até os modelos, serviços e ferramentas necessários para transformar seu conceito em um sistema de IA sofisticado. A plataforma é a fundação; sua visão única e os princípios descritos neste guia são o plano. Juntos, eles formam a base para criar sistemas inteligentes de última geração, capazes de levar sua startup a novos patamares.

Recursos

- AdkApp: desenvolva e implemente agentes no Vertex AI Agent Engine.
- Agent Development Kit (ADK): ADK é uma estrutura flexível e modular para desenvolver e implantar agentes de IA.
- Agent2Agent (A2A): protocolo aberto que permite comunicação e interoperabilidade entre aplicativos agênticos não transparentes.
- Agent Starter Pack: obtenha agentes prontos para produção no Google Cloud com mais rapidez. Vá da ideia à implantação com modelos e ferramentas pré-configurados.
- BigQuery: data warehouse de análise de dados em escala de petabytes, totalmente gerenciado e econômico do Google Cloud, que permite executar análises em grandes volumes de dados em tempo quase real.
- Check grounding API: incluída na experiência RAG (geração aumentada por recuperação) em aplicativos de IA, permite verificar o grau de grounding de um texto (chamado de "candidato a resposta") em um conjunto de textos de referência (chamados de "fatos").
- Cloud Functions API: gerencia funções leves fornecidas pelo usuário, executadas em resposta a eventos.
- Cloud Run: execute serviços de frontend e backend, tarefas em lote, hospede LLMs e processe filas de carga de trabalho sem precisar gerenciar a infraestrutura.
- Cloud Storage bucket: buckets são os contêineres básicos que armazenam seus dados. Tudo o que você armazena no Cloud Storage deve estar dentro de um bucket.
- Colab Enterprise: ambiente colaborativo de notebooks gerenciados com recursos de segurança e conformidade do Google Cloud.
- Example Store: permite armazenar e recuperar dinamicamente exemplos few-shot.
- Firestore: banco de dados NoSQL altamente escalável para aplicativos web e móveis.
- Gemini 2.5 Flash: projetado para equilibrar qualidade, custo e velocidade.
- Gemini 2.5 Flash Image (também conhecido como Nano Banana): gera e processa imagens de forma conversacional. Você pode usar texto, imagens ou ambos para criar, editar e iterar elementos visuais com controle sem precedentes.
- Gemini 2.5 Pro: nosso modelo Gemini mais avançado em raciocínio, capaz de resolver problemas complexos.
- Gemini CLI: gratuito e de código aberto, leva o Gemini 2.5 direto para o terminal dos desenvolvedores, com acesso excepcional para o usuário final.
- Gemma: conjunto de modelos abertos, leves e de última geração, construídos com a mesma tecnologia que alimenta os modelos Gemini.
- Gen AI evaluation service: serviço de avaliação de IA generativa na Vertex AI que permite avaliar qualquer modelo ou aplicativo generativo e comparar os resultados com seus próprios critérios.
- Google AI Studio: a maneira mais rápida de começar a criar com o Gemini, nossa nova geração de modelos multimodais de IA generativa.
- Observabilidade do Google Cloud: conjunto de serviços que ajudam a entender o comportamento, a saúde e o desempenho dos seus aplicativos.
- Google Kubernetes Engine (GKE): serviço Kubernetes totalmente automatizado e escalável. Coloque seus contêineres no piloto automático e execute cargas de trabalho empresariais com segurança, sem precisar de conhecimento avançado em Kubernetes.
- GraphRAG: combina grafos de conhecimento com RAG para melhorar a precisão, o contexto e a explicabilidade de modelos de linguagem grandes (LLMs).
- Imagen: fornece recursos de geração de imagens de ponta do Google para desenvolvedores de aplicativos na Vertex AI.
- MCP Toolbox for Databases: servidor MCP de código aberto que ajuda a construir ferramentas de IA generativa para que seus agentes acessem os dados no banco de dados.

- Model Context Protocol (MCP): protocolo aberto que padroniza a forma como os aplicativos fornecem contexto para LLMs.
- Avaliação de modelos da Vertex AI: serviço de avaliação preditiva que permite avaliar o desempenho de modelos em casos de uso específicos.
- Model Garden da Vertex AI: inicie seu projeto de ML com um único lugar para descobrir, personalizar e implantar uma ampla variedade de modelos do Google e parceiros.
- Ajuste de modelos: processo essencial para adaptar o Gemini a tarefas específicas com maior precisão.
- ReAct: orquestração com um agente ReAct (raciocínio + ação) envolve interações de múltiplas etapas entre um aplicativo e um ou mais modelos, onde o agente gerencia conversas, transações e instruções para LLMs.
- AI responsável: o Vertex AI Studio possui filtragem de conteúdo integrada, e nossas APIs de IA generativa têm pontuação de atributos de segurança para ajudar os clientes a testar os filtros do Google e definir limites de confiança adequados ao seu caso de uso.
- Geração aumentada por recuperação (RAG): estrutura de IA que combina os pontos fortes dos sistemas tradicionais de recuperação de informação (como busca e bancos de dados) com as capacidades dos LLMs generativos.
- Banco de dados vector: banco de dados que permite armazenar, indexar e consultar embeddings vetoriais, ou representações numéricas de dados não estruturados, como texto, imagens ou áudio.
- Veo: use o Veo na Vertex AI para gerar novos vídeos a partir de comandos de texto ou imagem.
- Vertex AI Agent Engine: conjunto de serviços que permite aos desenvolvedores implantar, gerenciar e escalar agentes de IA em produção.
- Vertex AI notebooks: acesse todos os recursos da Vertex AI Platform para trabalhar no ciclo completo da ciência de dados, da exploração de dados ao protótipo e à produção.
- Vertex AI Platform: plataforma de desenvolvimento de IA totalmente gerenciada e unificada para construir e usar IA generativa.
- Vertex AI RAG Engine: estrutura de dados para desenvolver aplicativos LLM com contexto aumentado.
- Vertex AI Search: une o poder da recuperação profunda de informações ao processamento de linguagem natural de ponta e aos recursos mais recentes de LLM para entender a intenção do usuário e retornar os resultados mais relevantes.
- Vertex AI Studio: simplifique seus fluxos de trabalho com modelos fundacionais no Vertex AI Studio. crie protótipos rapidamente, refine e implante modelos em seus aplicativos com facilidade.

Google Cloud

Dúvidas?

Entre em contato com a nossa Equipe de Startups.

Fale com a gente