

Desenvolvendo soluções com **IA** de forma rápida na GCP com **Serverless**

Gabriel Prando

Quem é Gabriel Prando?


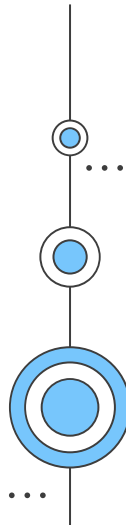


Engenheiro de Computação por formação (UTFPR-PB) e mestrando de Engenharia Elétrica e Computação no PPGEEC, com mais de 5 anos de experiência em desenvolvimento de software, entre eles como Full stack, DevOps, backend engineer, plataforma e atualmente Engenheiro de Plataforma/SRE no iFood, em qualquer rede com @prandogabriel.

...



Agenda

- Alinhamento de conhecimento;
 - Intro serverless;
 - Como utilizar cloud functions;
 - Vertx AI;
 - Exemplo de como criar algo com IA.
- 
- 

...



Antes de falar sobre serverless...

Vamos alinhar um pouco as coisas

— Qual o foco de hoje?

Function as a Service da GCP (FaaS)



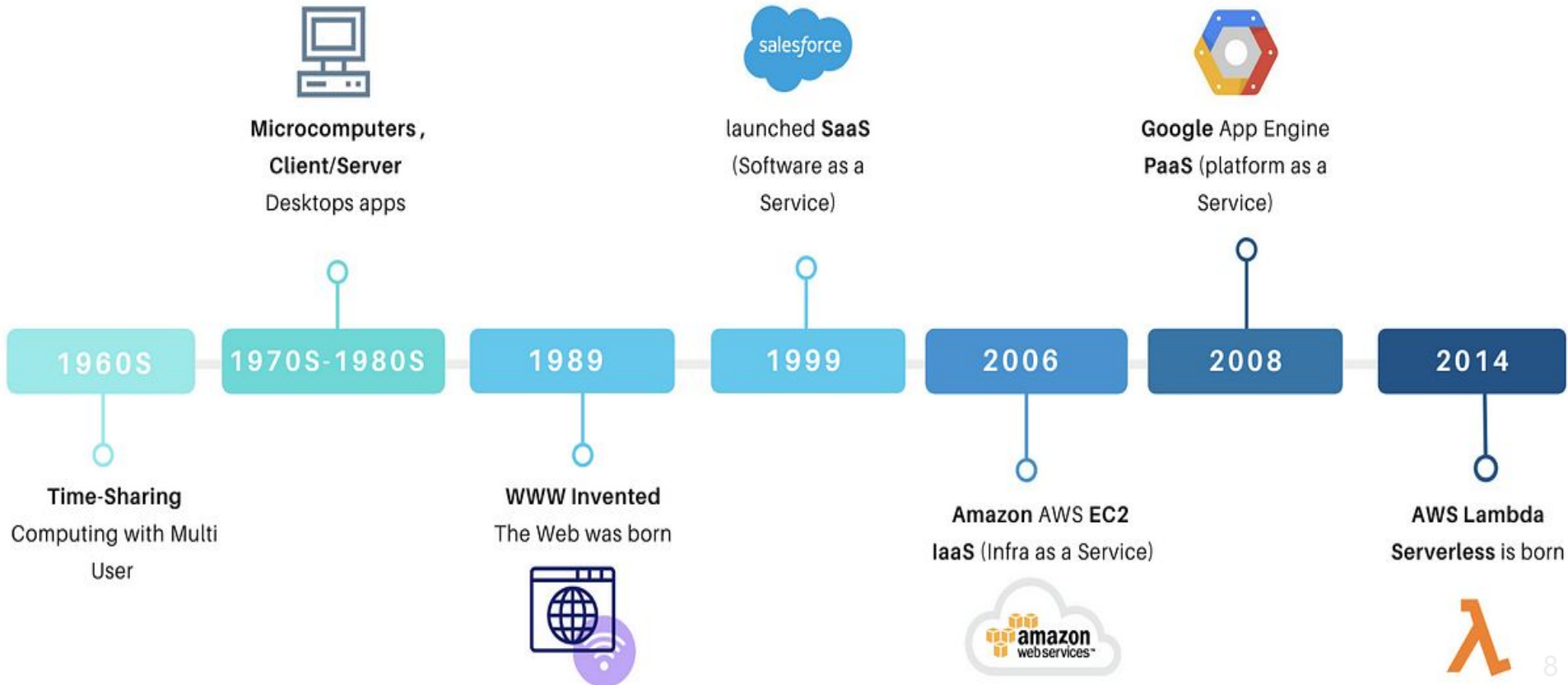
CLOUD FUNCTIONS

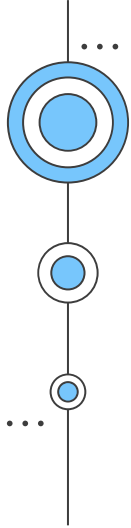
- Quem aqui já mexeu com serverless?
GCP/AWS/Azure...?

- **Evolução da computação até os dias atuais**

HISTORY OF COMPUTING

journey towards serverless ...

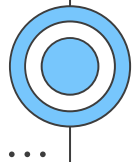


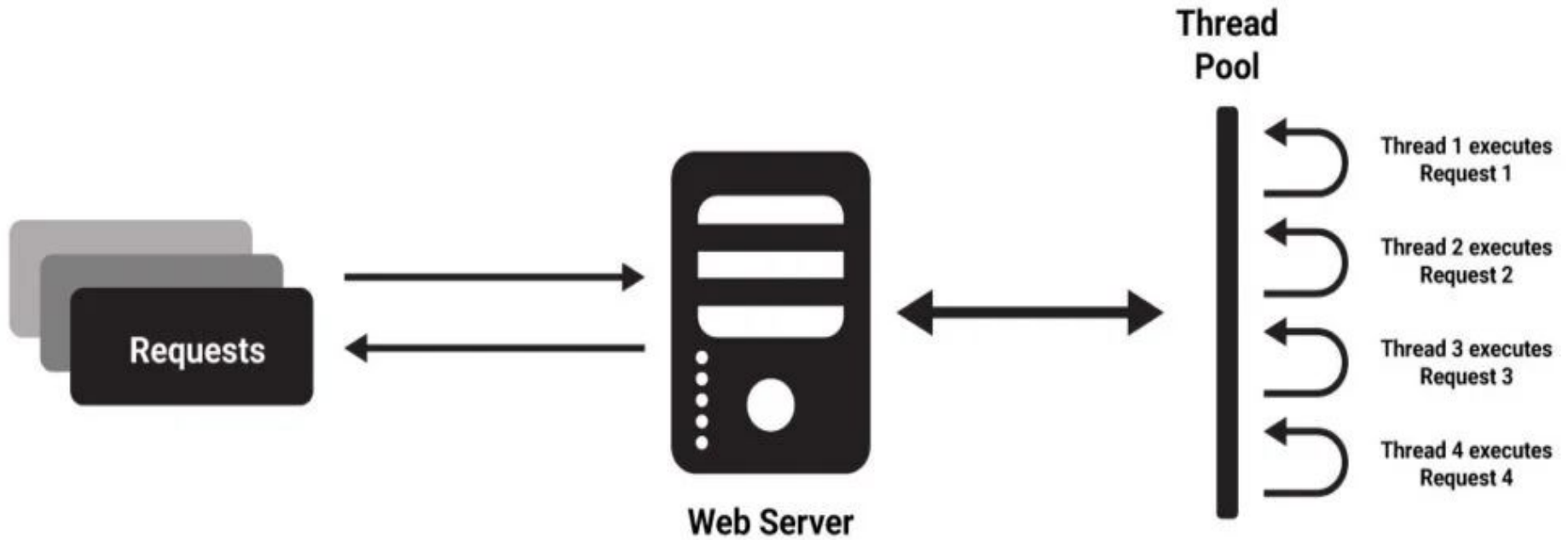


WEB server tradicional

Como funciona e processa requisições?

...







Serverless

Sem servidor, porém depende

**O que é serverless
computing? (computação
sem servidor)**



— Ora ora...

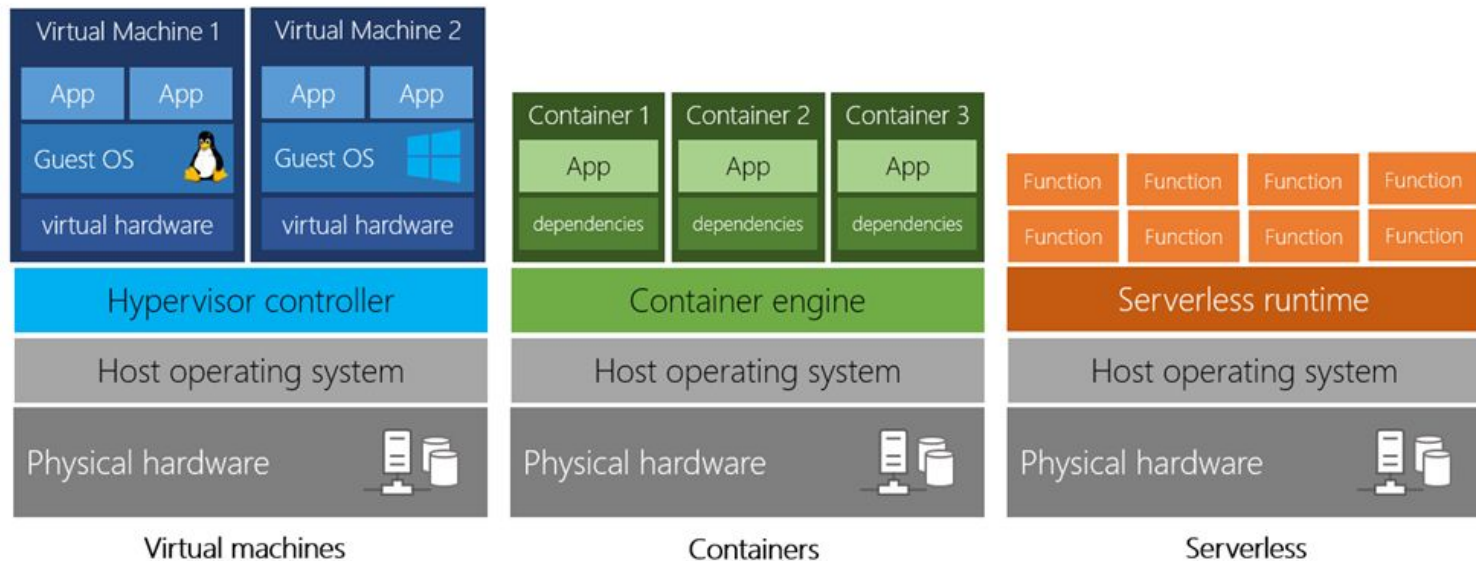
'Sem servidor' descreve a experiência. Os servidores são invisíveis **para o desenvolvedor**, que não os vê, gerencia ou interage, mas eles **existem sim**.



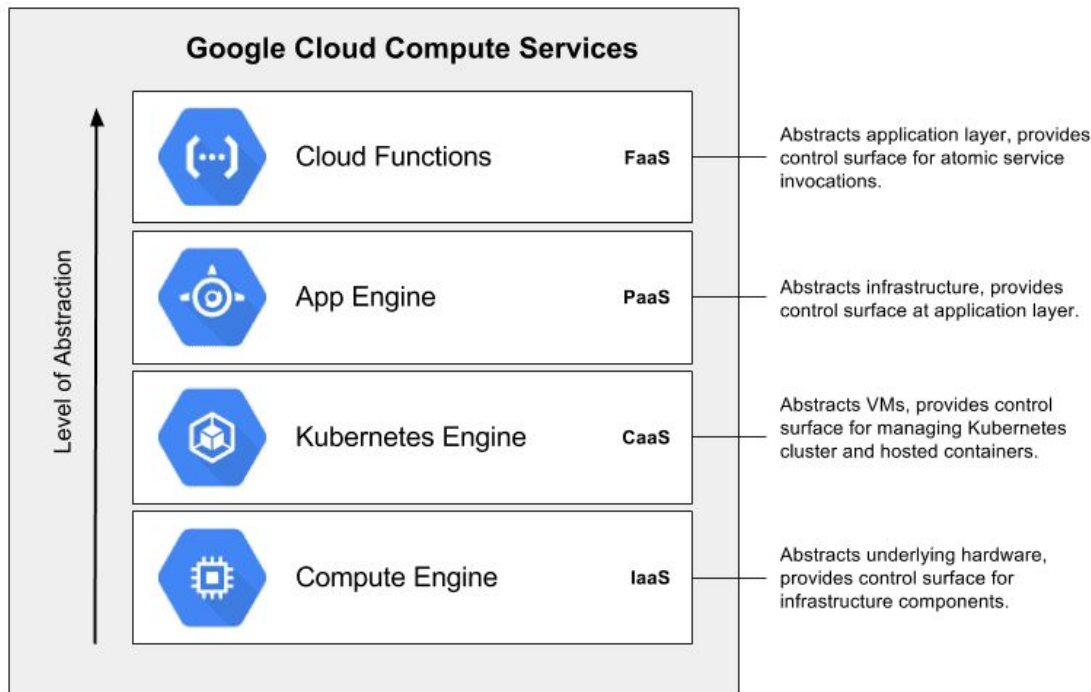
SERVERLESS IS NOT REALLY SERVERLESS!



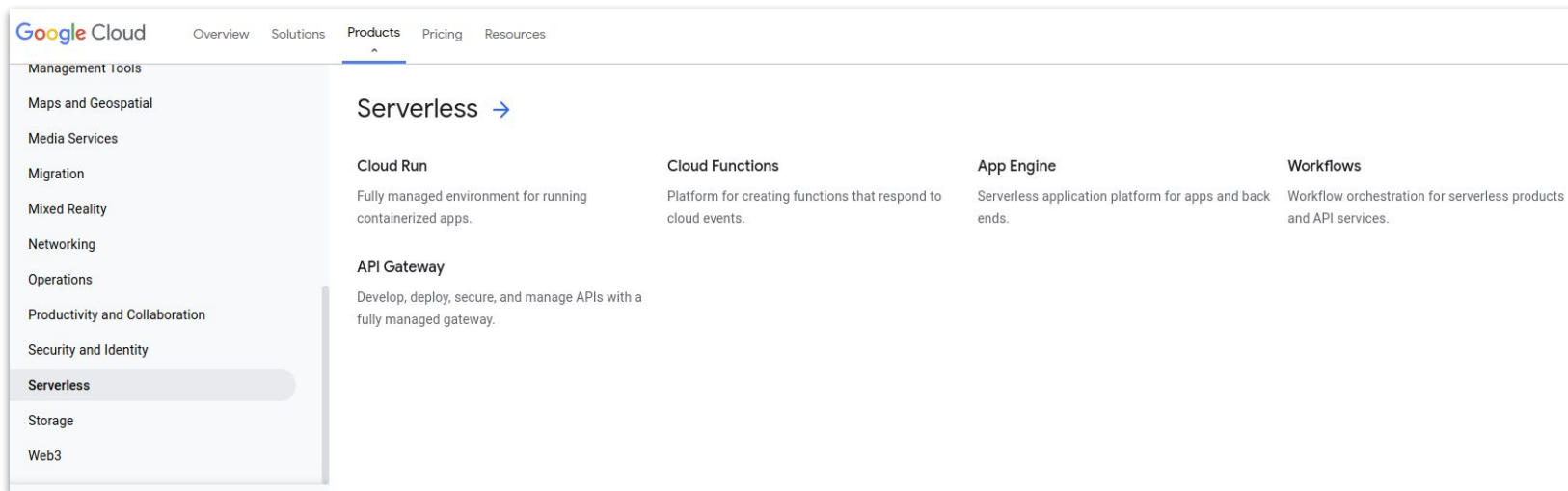
— VMs vs. Containers vs. Serverless



— VMs vs. Containers vs. Serverless na GCP



Alguns serviços da GCP que rodam no modelo serverless



The screenshot displays the Google Cloud website's 'Products' page, specifically the 'Serverless' section. The left sidebar lists various Google Cloud categories, with 'Serverless' highlighted. The main content area features a 'Serverless' heading with a right-pointing arrow, followed by four service cards: Cloud Run, Cloud Functions, App Engine, and Workflows. Each card provides a brief description of the service.

Google Cloud Overview Solutions **Products** Pricing Resources

Management Tools
Maps and Geospatial
Media Services
Migration
Mixed Reality
Networking
Operations
Productivity and Collaboration
Security and Identity
Serverless
Storage
Web3

Serverless →

Cloud Run Fully managed environment for running containerized apps.	Cloud Functions Platform for creating functions that respond to cloud events.	App Engine Serverless application platform for apps and back ends.	Workflows Workflow orchestration for serverless products and API services.
---	---	--	--

API Gateway
Develop, deploy, secure, and manage APIs with a fully managed gateway.

Voltando para cloud functions




— Premissas

- Sem servidor para gerenciar
- Escale com o uso
- Orientado a eventos / triggers
- Disponibilidade
- Execução do código não gerenciado pela gente
- Agnóstico a tecnologia
- Mais fácil de desenvolver = mais tempo para pensar
- Maior agilidade



Casos de uso

- Rest APIs
 - Processamento assíncrono
 - Processamento de dados (ETL)
 - Startups (economia, foco dev, fácil desenvolvimento e manutenção)
 - Integrações e soluções de IA
- 

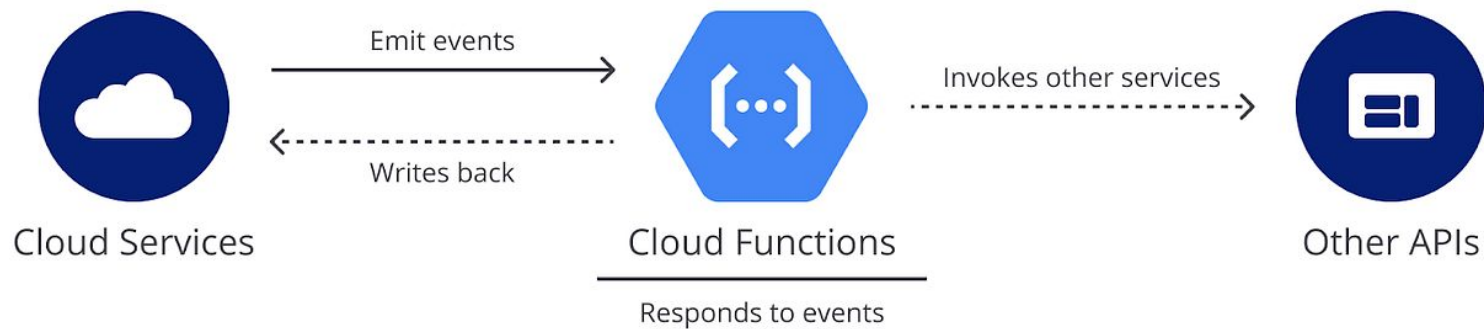


**Quando estamos falando
de cloud functions**

FUNCTIONS, FUNCTIONS

EVERYWHERE

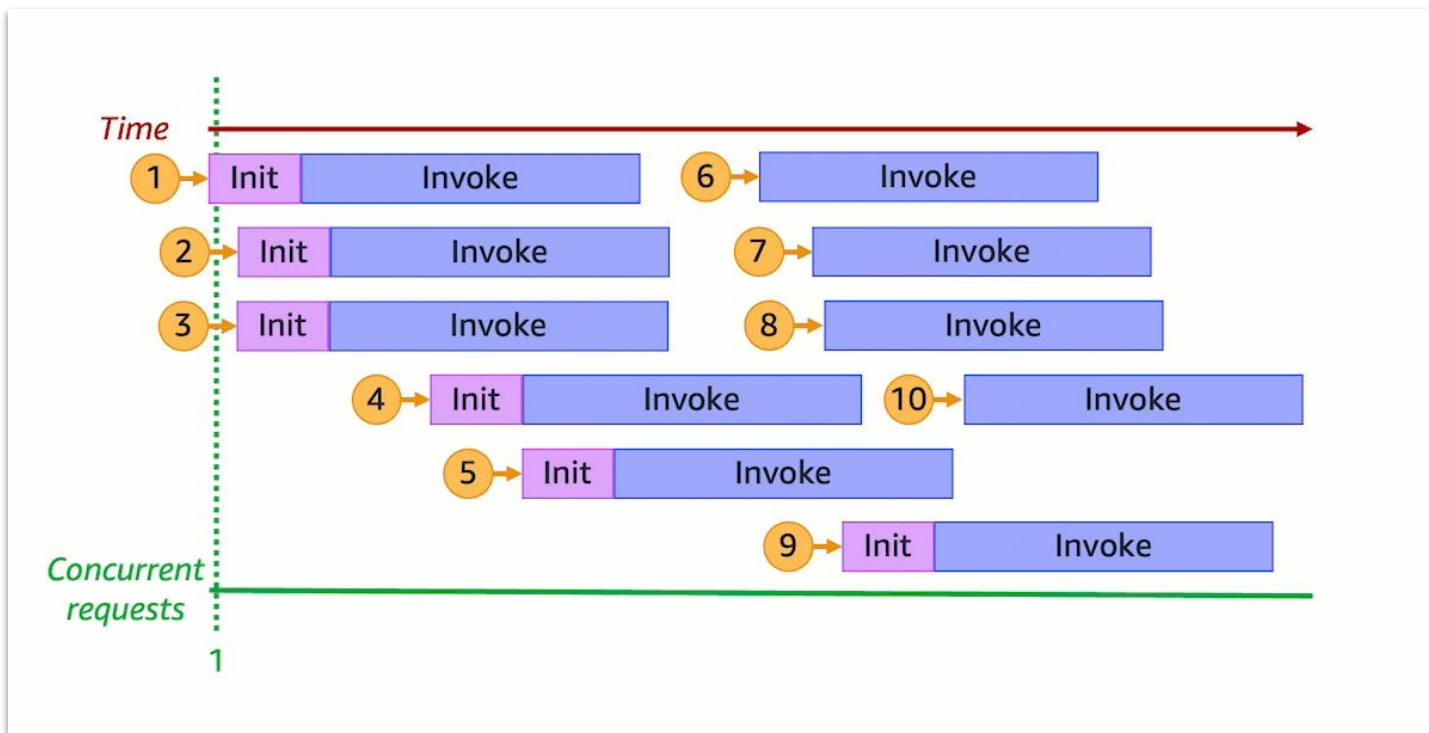
— Como uma function é chamada?



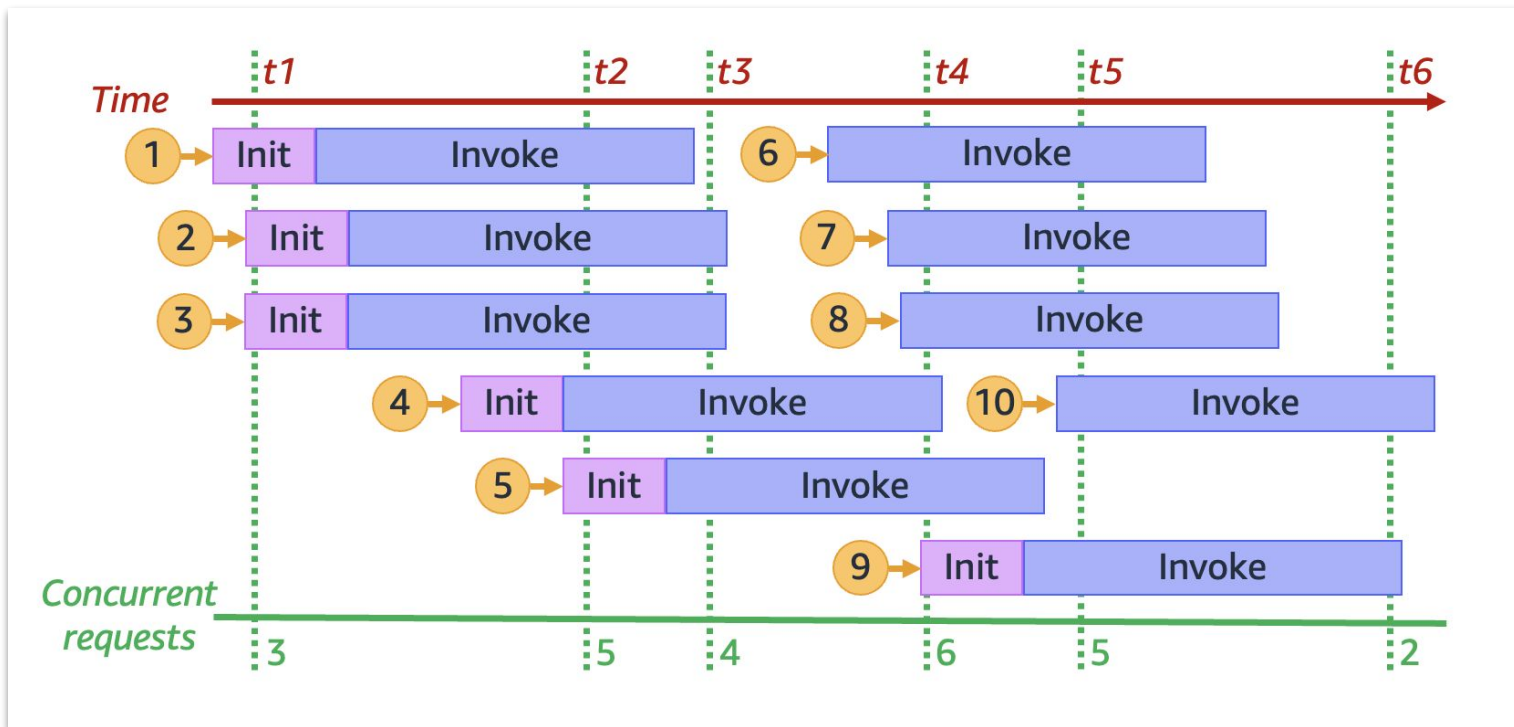
— Como functions escalam?



— Como functions escalam?



— Como functions escalam?



Anatomia de uma cloud function

```
package function

import (
    "fmt"
    "net/http"

    "github.com/GoogleCloudPlatform/functions-framework-go/functions"
)

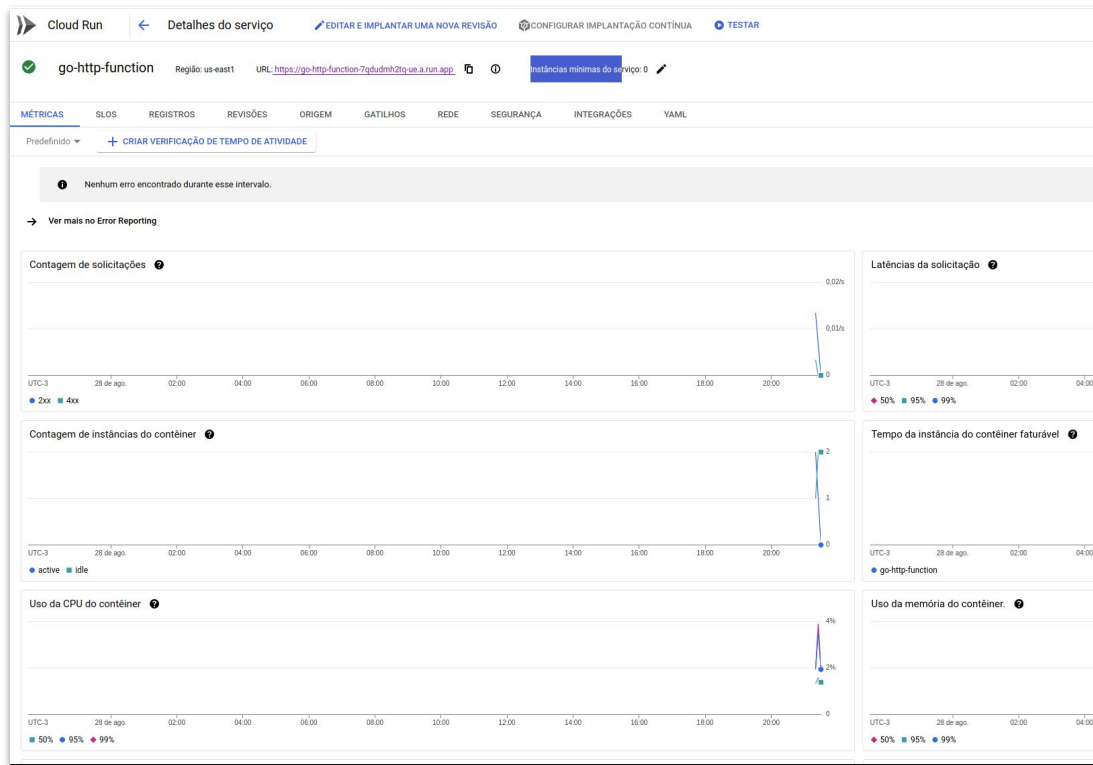
func init() {
    functions.HTTP("HelloAI", HelloAI)
}

// HelloAI writes "Hello, AI!" to the HTTP response.
func HelloAI(w http.ResponseWriter, r *http.Request) {
    fmt.Fprintln(w, "Hello, AI!")
}
```

Deploy de uma function

```
gcloud functions deploy $FUNCTION_NAME \  
  --gen2 \  
  --project $YOUR_PROJECT_ID \  
  --runtime=go121 \  
  --region=us-east1 \  
  --source=. \  
  --entry-point=HelloAI \  
  --trigger-http \  
  --allow-unauthenticated
```

Como fica na GCP

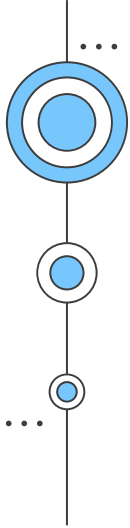


Resource de definição da function

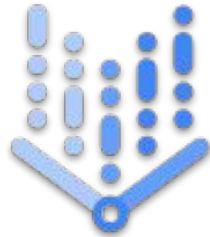
```
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
  name: go-http-function
  namespace: '961825277982'
  uid: 815970e4-612e-445d-b982
  resourceVersion: AAYgx19LT
  generation: 1
  creationTimestamp: '2024-08-29T00:15:16.410024Z'
  labels:
    goog-managed-by: cloudfunctions
    goog-cloudfunctions-runtime: go121
    cloud.googleapis.com/location: us-east1
  annotations:
    run.googleapis.com/launch-stage: BETA
    run.googleapis.com/ingress: all
    run.googleapis.com/ingress-status: all
    cloudfunctions.googleapis.com/function-id: go-http-function
spec:
  template:
    metadata:
      name: go-http-function-00001-fal
      labels:
```

Chamada da function

```
curl -m 70 -X POST URI \  
  -H "Authorization: Bearer $(gcloud auth print-identity-token)" \  
  -H "Content-Type: application/json" \  
  -d '{}'
```



Como começo a utilizar IA nesse contexto?

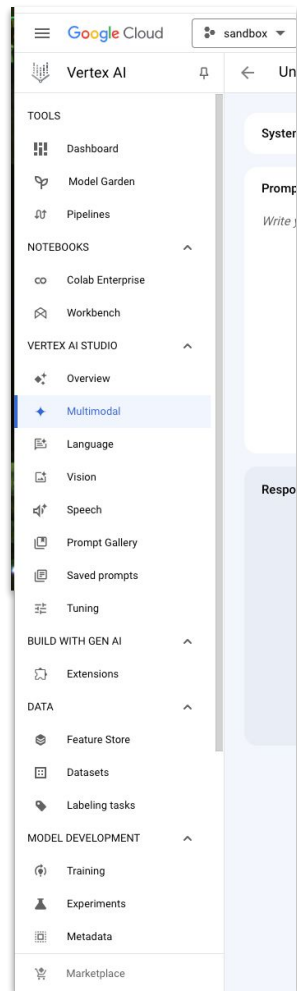


Vertex AI



O que é a Vertex AI?

- Plataforma de machine learning;
- Ferramentas para construir, treinar, implantar e escalar modelos de machine learning (ML);
- Suporte modelos pré-treinados e personalizados;
- Facilita o treinamento de modelos com menos código;
- Amigável para iniciantes e eficiente para especialistas.

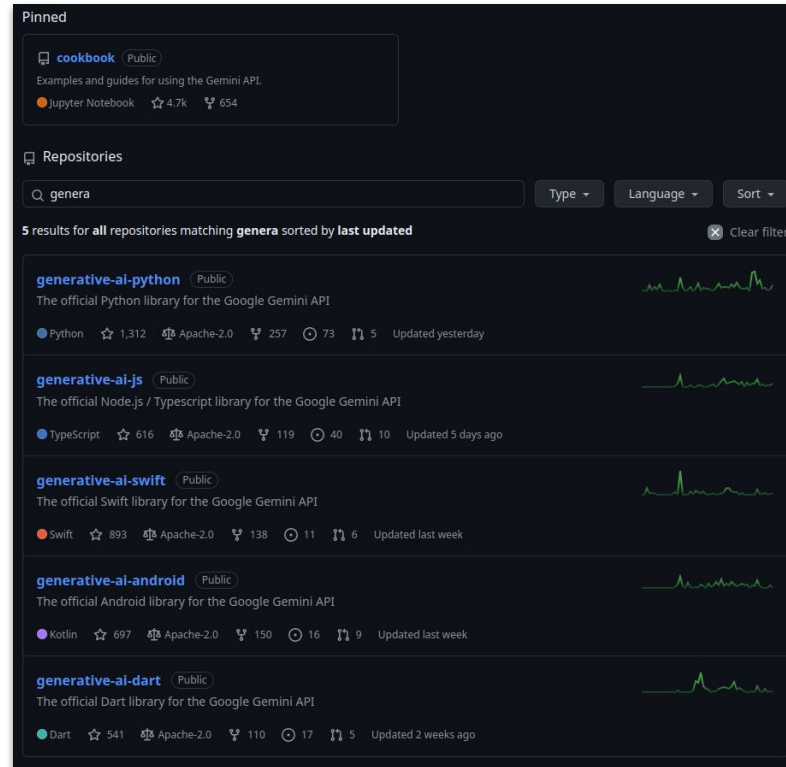


— Como podemos utilizar?

- Console;
- API;
- SDK.

Como podemos utilizar?

- SDKs (org /google-gemini):
 - Python;
 - JS;
 - Swift;
 - Android;
 - Dart;
- Go (org /google).



Poucas linhas e temos algo funcional



```
client, _ := genai.NewClient(ctx, option.WithAPIKey("your-api-key"))
defer client.Close()

// gemini promodel
model := client.GenerativeModel("gemini-pro")

resp, err := model.GenerateContent(ctx, genai.Text(inputText))
```

— API respondendo nossas perguntas

```
curl -X POST https://go-http-function-7qdudmh2tq-ue.a.run.app/oi \
-H "Content-Type: application/json" \
-d '{"input": "o que é gen ai"}'
```

```
# Gen ai é...
```

Poderíamos até fazer uma CLI para perguntas

```
#!/bin/bash

# Nome da função
function gemini() {
  # Verifica se o parâmetro foi passado
  if [ -z "$1" ]; then
    echo "Uso: gemini <input>"
    return 1
  fi

  # Define a URL do endpoint
  local url="https://go-http-function-7qdudmh2tq-ue.a.run.app/oi"

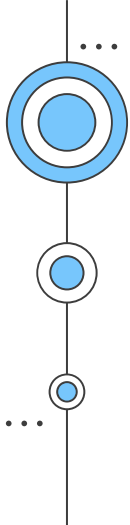
  # Define o payload JSON
  local payload=$(jq -n --arg input "$1" '{"input": $input}')

  # Faz a chamada ao endpoint usando curl
  curl -X POST "$url" \
    -H "Content-Type: application/json" \
    -d "$payload"
}

# Exemplo de uso da função:
gemini "o que é gen ai"
```



**E se a gente fizesse o
nosso chat Gemini?**



Com layout inovador

Digite sua pergunta...

Enviar



Exemplo de pergunta

o que é serverless computing?

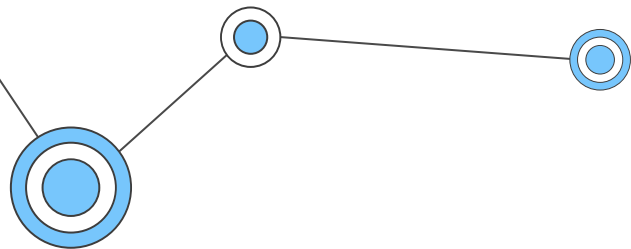
Computação sem Servidor Computação sem servidor é um modelo de computação em nuvem que permite que os desenvolvedores criem e executem aplicativos sem gerenciar ou provisionar servidores. **Como funciona:** **Fornecedores de Nuvem:** Os provedores de nuvem, como AWS, Azure e GCP, fornecem a infraestrutura de servidor subjacente. **Eventos:** Os aplicativos sem servidor são acionados por eventos, como solicitações HTTP, alterações no banco de dados ou mensagens de fila. **Funções:** Os desenvolvedores escrevem funções que processam esses eventos. **Escalabilidade Automática:** Os provedores de nuvem escalam automaticamente as funções com base na carga, eliminando a necessidade de gerenciamento

Digite sua pergunta...

Enviar

Ficou com vontade de testar?



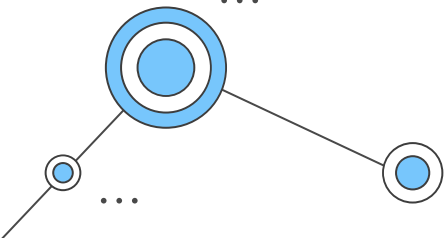


...

Perguntas?

Obrigado!

...



Aponte o celular e conecte-se :)



SCAN ME!