AI_O11y_Custos - Guia Completo (Passo a passo + Códigos) - v2

Objetivo: colocar o projeto AI_O11y_Custos de pØ do zero, com repositório GitHub, Jobs no Databricks,
ingestªo de custos da Azure e **infraestrutura 100% automatizada com Terraform**.

==========
0) PRÉ-REQUISITOS
- Conta no Azure (assinatura ativa).
- Conta no GitHub.
- Um Workspace do Azure Databricks (ou deixe o Terraform criar opcionalmente).
- (Opcional local) Git + Python 3.11.

==========
1) SUBIR O CÓDIGO PARA O GITHUB
OpçªO GUI:
  1. Baixe o pacote do projeto (ZIP) e descompacte.
  2. GitHub -> New repository -> nome: AI_O11y_Custos -> Create.
  3. Code -> Upload files -> suba as PASTAS/ARQUIVOS (branch main).

OpçªO Terminal:
  git clone https://github.com/<org>/AI_O11y_Custos.git
  cd AI_O11y_Custos
  git add . && git commit -m "init: AI_O11y_Custos skeleton" && git push

==========
2) GITHUB SECRETS (AZURE + DATABRICKS)

2.1 AZURE_CREDENTIALS (Service Principal)
  az login
  az account set --subscription <SUBSCRIPTION_ID>
  az ad sp create-for-rbac \
    --name "gh-ai-o11y-custos" \
    --role contributor \
    --scopes /subscriptions/<SUBSCRIPTION_ID> \
    --sdk-auth
  -> Copie o JSON de saída para o secret: AZURE_CREDENTIALS

2.2 DATABRICKS_HOST / DATABRICKS_TOKEN
  - Launch Workspace (Azure Databricks) -> copie a URL (HOST)
  - User Settings -> Developer -> Access tokens -> Generate (TOKEN)
  -> Crie secrets: DATABRICKS_HOST e DATABRICKS_TOKEN

==========
3) GITHUB ACTIONS (PIPELINES)
- Faça um commit simples (README, por exemplo).
- Pipelines:
  - ci   : checagens Python
  - deploy-databricks : cria/atualiza Repo/Job
  - deploy-infra   : (se vocŒ mexer em infra/**) aplica Terraform
  - deploy-functions : publica a Azure Function (Timer)

==========
4) PRIMEIRO JOB (DATABRICKS)
- Workflows -> Jobs -> AI_O11y_Custos-train-daily -> Run now.
- Ver mØtricas em MLflow (execuçªo).

==========

```
FASE 2 - DADOS REAIS (CUSTOS AZURE)
5) ADLS GEN2 (Storage) e permissıes
- Crie Storage Account com Hierarchical namespace = Enabled.
- Conceda ao Service Principal: Storage Blob Data Reader (IAM).

6) COST MANAGEMENT EXPORT (diÆrio -> ADLS)
- Cost Management -> Exports -> Add
  - Scope: Subscription | Recurrence: Daily | Format: CSV
  - Storage: o ADLS acima (container/pasta)
- Run now para gerar o primeiro CSV.

7) FUNCTION APP (timer) - opcional no iní cio
- Function App (Linux, Python 3.11, plano Consumption).
- App Settings: COSTS_ACCOUNT_NAME, COSTS_CONTAINER, COSTS_PATH_PREFIX.
- O workflow deploy-functions publica o código do timer.

========================================================================
FASE 3 - ALERTAS & PAINEL
- Log Analytics + KQL (exemplo):
  AI_O11y_Custos_CL
  | summarize avg(risk_score_d) by bin(TimeGenerated, 1d), scope_s
  | order by TimeGenerated desc
- Azure Monitor Alerts (condiçªo ex.: avg(risk_score_d) > 70).
- Grafana/Power BI lendo de Log Analytics/Storage.

========================================================================
NOVIDADE - INFRAESTRUTURA AUTOMATIZADA (TERRAFORM)
Como aplicar (exemplo simples):
  cd infra
  terraform init
  terraform apply -auto-approve -var="prefix=aio11y" -var="location=brazilsouth"

Opçıes:
  -var="create_databricks=true"          # cria workspace Databricks
  -var="create_cost_export=true"         # cria Export de Custos diÆrio
  -var="enable_function_diagnostics=true"# envia logs da Function para Log Analytics

Exemplo completo:
  terraform apply -auto-approve \
    -var="prefix=aio11y" \
    -var="location=brazilsouth" \
    -var="create_databricks=true" \
    -var="create_cost_export=true" \
    -var="enable_function_diagnostics=true"

O que o Terraform cria:
- Resource Group
- Storage (ADLS Gen2) + container cost-exports
- Log Analytics + Application Insights (workspace-based)
- Service Plan (Linux Y1) + Function App (Python 3.11) com identidade gerenciada
- RBAC: Function -> Storage Blob Data Reader
- Diagnostic Settings do Storage -> Log Analytics
- (Opcional) Workspace Databricks
- (Opcional) Export de Custos diÆrio para o ADLS

========================================================================
CHECKLIST R`PIDO
- Actions "deploy-functions" falhando? Crie o Function App antes ou ajuste o nome.
```

- Sem dados? Verifique export, permissıes IAM e caminho do container/prefixo.
- Databricks sem job? Confirme DATABRICKS_HOST/TOKEN e reexecute "deploy-databricks".

==
ANEXO - ARQUIVOS DO PROJETO (CÓDIGOS COMPLETOS)
A seguir, o conteœdo dos principais arquivos.

===== ARQUIVO: README.md =====
(arquivo nªo encontrado)

===== ARQUIVO: .gitignore =====
(arquivo nªo encontrado)

===== ARQUIVO: requirements.txt =====
(arquivo nªo encontrado)

===== ARQUIVO: pyproject.toml =====
(arquivo nªo encontrado)

===== ARQUIVO: infra/providers.tf =====
```
terraform {
  required_version = ">= 1.6.0"
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~> 3.113"
    }
    random = {
      source  = "hashicorp/random"
      version = "~> 3.6"
    }
  }
}

provider "azurerm" {
  features {}
}

data "azurerm_client_config" "current" {}
```

===== ARQUIVO: infra/variables.tf =====
```
variable "prefix" {
  description = "Prefixo curto para nomear recursos"
  type        = string
  default     = "aio11y"
}

variable "location" {
  description = "Regiªo Azure"
  type        = string
  default     = "brazilsouth"
}

variable "tags" {
  description = "Tags padrªo"
```

```
  type        = map(string)
  default = {
    project = "AI_O11y_Custos"
    owner   = "FinOps-SRE"
  }
}

variable "create_databricks" {
  description = "Cria workspace Databricks (opcional)"
  type        = bool
  default     = false
}

variable "create_cost_export" {
  description = "Cria Export diÆrio do Cost Management (requer permissıes)"
  type        = bool
  default     = false
}

variable "enable_function_diagnostics" {
  description = "Ativa diagnósticos do Function App para Log Analytics"
  type        = bool
  default     = false
}


===== ARQUIVO: infra/resource_group.tf =====
resource "azurerm_resource_group" "rg" {
  name     = "${var.prefix}-rg"
  location = var.location
  tags     = var.tags
}


===== ARQUIVO: infra/storage.tf =====
resource "random_string" "suf" {
  length  = 5
  upper   = false
  lower   = true
  number  = true
  special = false
}

resource "azurerm_storage_account" "sa" {
  name                            = "${replace(var.prefix, "-",
  "")}${random_string.suf.result}sa"
  resource_group_name             = azurerm_resource_group.rg.name
  location                        = azurerm_resource_group.rg.location
  account_tier                    = "Standard"
  account_replication_type        = "LRS"
  min_tls_version                 = "TLS1_2"
  allow_blob_public_access        = false
  enable_https_traffic_only       = true
  is_hns_enabled                  = true # ADLS Gen2
  tags                            = var.tags
}

resource "azurerm_storage_container" "cost" {
```

```
  name                  = "cost-exports"
  storage_account_name  = azurerm_storage_account.sa.name
  container_access_type = "private"
}


===== ARQUIVO: infra/log_analytics.tf =====
resource "azurerm_log_analytics_workspace" "law" {
  name                = "${var.prefix}-law"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  sku                 = "PerGB2018"
  retention_in_days   = 30
  tags                = var.tags
}

resource "azurerm_application_insights" "appi" {
  name                = "${var.prefix}-appi"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  application_type    = "web"
  workspace_id        = azurerm_log_analytics_workspace.law.id
  tags                = var.tags
}


===== ARQUIVO: infra/function.tf =====
resource "azurerm_service_plan" "plan" {
  name                = "${var.prefix}-plan"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  os_type             = "Linux"
  sku_name            = "Y1" # Consumption
  tags                = var.tags
}

resource "azurerm_linux_function_app" "func" {
  name                       = "${var.prefix}-func"
  location                   = azurerm_resource_group.rg.location
  resource_group_name        = azurerm_resource_group.rg.name
  service_plan_id            = azurerm_service_plan.plan.id

  # Armazenamento para conteœdos/funcionamento
  storage_account_name       = azurerm_storage_account.sa.name
  storage_account_access_key = azurerm_storage_account.sa.primary_access_key

  functions_extension_version = "~4"

  identity {
    type = "SystemAssigned"
  }

  site_config {
    application_stack {
      python_version = "3.11"
    }
    ftps_state = "Disabled"
  }
```

```
  app_settings = {
    "WEBSITE_RUN_FROM_PACKAGE"         = "1"
    "APPINSIGHTS_INSTRUMENTATIONKEY"   =
    azurerm_application_insights.appi.instrumentation_key
    "APPLICATIONINSIGHTS_CONNECTION_STRING" =
    azurerm_application_insights.appi.connection_string

    # Ajuste conforme sua leitura de custos
    "COSTS_ACCOUNT_NAME" = azurerm_storage_account.sa.name
    "COSTS_CONTAINER"    = azurerm_storage_container.cost.name
    "COSTS_PATH_PREFIX"  = "exports/"
  }

  tags = var.tags
}

# Permissªo para a Function ler blobs do Storage
resource "azurerm_role_assignment" "func_blob_reader" {
  scope                = azurerm_storage_account.sa.id
  role_definition_name = "Storage Blob Data Reader"
  principal_id         = azurerm_linux_function_app.func.identity[0].principal_id
}


===== ARQUIVO: infra/diagnostics.tf =====
# Logs do Storage -> Log Analytics (œtil para auditoria/monitoramento)
resource "azurerm_monitor_diagnostic_setting" "sa_to_law" {
  name                       = "${var.prefix}-sa-diag"
  target_resource_id         = azurerm_storage_account.sa.id
  log_analytics_workspace_id = azurerm_log_analytics_workspace.law.id

  enabled_log {
    category = "StorageRead"
  }
  enabled_log {
    category = "StorageWrite"
  }
  enabled_log {
    category = "StorageDelete"
  }

  metric {
    category = "AllMetrics"
    enabled  = true
  }
}

# (Opcional) Logs da Function App -> Log Analytics (algumas categorias podem variar;
deixe desativado se nªo tiver certeza)
resource "azurerm_monitor_diagnostic_setting" "func_to_law" {
  count                      = var.enable_function_diagnostics ? 1 : 0
  name                       = "${var.prefix}-func-diag"
  target_resource_id         = azurerm_linux_function_app.func.id
  log_analytics_workspace_id = azurerm_log_analytics_workspace.law.id

  enabled_log {
    category = "FunctionAppLogs"
```

```
  }
  metric {
    category = "AllMetrics"
    enabled  = true
  }
}


===== ARQUIVO: infra/cost_export.tf =====
# Export diÁrio do Azure Cost Management para o ADLS (opcional)
# ATEN˙ˆO: requer permissıes específicas na assinatura para criar exports.
# Habilite com -var="create_cost_export=true"
resource "azurerm_cost_management_export" "daily" {
  count = var.create_cost_export ? 1 : 0

  name  = "${var.prefix}-cost-export"
  scope = "/subscriptions/${data.azurerm_client_config.current.subscription_id}"

  recurrence = "Daily"

  recurrence_period {
    from = "2025-09-01T00:00:00Z"
    to   = "2030-01-01T00:00:00Z"
  }

  delivery_info {
    destination {
      resource_id        = azurerm_storage_account.sa.id
      container          = azurerm_storage_container.cost.name
      root_folder_path   = "exports"
    }
  }

  format = "Csv"
  time_period {
    type = "BillingMonthToDate"
  }
}


===== ARQUIVO: infra/databricks.tf =====
# Criaçªo opcional de um workspace Databricks (muitos ambientes jÆ possuem um)
resource "azurerm_databricks_workspace" "dbw" {
  count               = var.create_databricks ? 1 : 0
  name                = "${var.prefix}-dbw"
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  sku                 = "premium"
  tags                = var.tags
}

output "databricks_workspace_url" {
  value       = try(azurerm_databricks_workspace.dbw[0].workspace_url, null)
  description = "URL do workspace Databricks (se criado)"
}


===== ARQUIVO: infra/outputs.tf =====
```

```
output "resource_group" {
  value = azurerm_resource_group.rg.name
}

output "storage_account_name" {
  value = azurerm_storage_account.sa.name
}

output "storage_container_cost_exports" {
  value = azurerm_storage_container.cost.name
}

output "function_app_name" {
  value = azurerm_linux_function_app.func.name
}

output "log_analytics_workspace_id" {
  value = azurerm_log_analytics_workspace.law.id
}

output "application_insights_connection_string" {
  value = azurerm_application_insights.appi.connection_string
}


===== ARQUIVO: infra/README_INFRA.md =====
# Infraestrutura - AI_O11y_Custos (Terraform)

## O que este IaC cria
- Resource Group
- Storage Account (ADLS Gen2) + Container ‘cost-exports‘
- Log Analytics Workspace + Application Insights (workspace-based)
- Service Plan (Linux, Consumption) + Function App (Python 3.11) com identidade
gerenciada
- RBAC: Function -> Storage Blob Data Reader
- Diagnósticos do Storage para Log Analytics
- (Opcional) Workspace Databricks
- (Opcional) Export DiÆrio do Cost Management para o ADLS

## Como aplicar
‘‘‘bash
cd infra
terraform init
terraform apply -auto-approve
‘‘‘

### Opçıes œteis
- Criar Databricks: ‘-var="create_databricks=true"‘
- Criar Export de Custos: ‘-var="create_cost_export=true"‘
- Ativar diagnostics da Function: ‘-var="enable_function_diagnostics=true"‘

Exemplo completo:
‘‘‘bash
terraform apply -auto-approve       -var="prefix=aio11y"
-var="location=brazilsouth"       -var="create_cost_export=true"
-var="enable_function_diagnostics=true"
‘‘‘
```

> Dica: o workflow GitHub Actions ‘.github/workflows/deploy-infra.yml‘ jÆ roda ‘terraform init/validate/apply‘ automaticamente ao modificar arquivos em ‘infra/**‘.

===== ARQUIVO: jobs/train_job.json =====
(arquivo nªo encontrado)

===== ARQUIVO: notebooks/train_forecast.py =====
(arquivo nªo encontrado)

===== ARQUIVO: src/ai_o11y_custos/__init__.py =====
(arquivo nªo encontrado)

===== ARQUIVO: src/ai_o11y_custos/pipeline.py =====
(arquivo nªo encontrado)

===== ARQUIVO: src/ai_o11y_custos/scoring_function/__init__.py =====
(arquivo nªo encontrado)

===== ARQUIVO: src/ai_o11y_custos/scoring_function/function.json =====
(arquivo nªo encontrado)

===== ARQUIVO: .github/workflows/ci.yml =====
(arquivo nªo encontrado)

===== ARQUIVO: .github/workflows/deploy-infra.yml =====
(arquivo nªo encontrado)

===== ARQUIVO: .github/workflows/deploy-databricks.yml =====
(arquivo nªo encontrado)

===== ARQUIVO: .github/workflows/deploy-functions.yml =====
(arquivo nªo encontrado)