**Global Solution 2025.1 – Grupo 11**

**Integrantes**

* Alexandre Oliveira Mantovani - RM560355
* Edmar Ferreira Souza - RM560406
* Enyd Bentivoglio – RM560234
* Jose Andre Filho - RM87775
* Ricardo Lourenço Coube - RM559871

**GitHub com código-fonte:**

https://github.com/alexomantovanIA/1TIAOR20242-Global-Solutions-GS-20251

**Wokwi Project:**

https://wokwi.com/projects/432026417622874113

**Vídeo demonstrativo:**

https://youtube.com/nao-listado-exemplo

**1. Introdução**

A intensificação de eventos naturais extremos tem representado riscos severos para a vida humana e para a infraestrutura urbana e rural. Entre esses eventos, os deslizamentos de terra figuram como uma das ocorrências mais perigosas em regiões com alta pluviosidade e declividade do solo.

Neste contexto, o presente projeto propõe uma solução digital baseada em sensores e algoritmos de machine learning para detecção precoce do risco de deslizamentos. A proposta integra sensores simulados via ESP32, um pipeline de ingestão via MQTT, persistência em banco de dados, análise em tempo real e envio automático de alertas via SNS da AWS, garantindo uma resposta rápida diante de condições críticas.

**2. Desenvolvimento**

**2.1 Arquitetura da Solução**

A arquitetura da solução é composta por cinco camadas principais:

* Simulação via ESP32 (Wokwi): sensores simulados (chuva, umidade e vibração no eixo Z) enviam dados em tempo real.
* Transmissão MQTT: os dados são publicados em um tópico MQTT público.
* Coletor e Inferência Local: script Python coletor\_inferente\_mqtt.py assina o tópico MQTT, grava as leituras no banco SQLite e executa inferência imediata via modelo de machine learning.
* Alerta via SNS: quando a classe prevista é "risco", um alerta é automaticamente enviado via AWS SNS.
* Visualização Analítica: um Jupyter Notebook permite análises e gráficos sobre os dados e inferências realizadas.

**2.2 Banco de Dados**

O banco SQLite gs\_deslizamento.db contém as seguintes tabelas:

* **leituras**: armazena os dados brutos recebidos dos sensores.
* **inferencias**: registra as inferências do modelo (classes: ok, atenção, risco).
* **alertas**: armazena os alertas enviados via SNS com status de envio.

**2.3 Modelo de Machine Learning**

* **Algoritmo**: RandomForestClassifier
* **Entradas**: umidade, chuva, acc\_z
* **Saídas**: classe predita (ok, atenção, risco)
* **Treinamento**: realizado com dados simulados representando cenários distintos.
* **Persistência**: modelo salvo em modelo\_multiclasse\_ajustado\_v2.pkl

**2.4 Segurança e Organização**

Uso de variáveis sensíveis com .env (AWS credentials, ARN)

Estrutura modular com scripts bem definidos

Separação de código, dados e modelo em diretórios distintos (data/, ml/, src/)

**3. Resultados Esperados**

Funcionamento em tempo real: com a execução do script, qualquer dado recebido é imediatamente processado e inferido.

Precisão e agilidade: o modelo é capaz de classificar corretamente cenários de risco com base em parâmetros ambientais.

Disparo de alertas confiável: o sistema já foi validado com o envio de alertas reais via SNS para simulações de deslizamento.

Dashboard analítico: visualizações gráficas permitem análise de padrões, comportamento temporal dos sensores e distribuição das classes inferidas.

**4. Conclusões**

A solução proposta atende plenamente ao objetivo da GS 2025.1 ao integrar:

* Programação lógica em Python
* Machine Learning funcional
* Uso de sensores (simulados via ESP32)
* Comunicação em tempo real via MQTT
* Persistência local estruturada
* Integração com nuvem via AWS SNS
* Análise visual em Jupyter Notebook

Ela demonstra como a combinação de técnicas computacionais pode auxiliar na prevenção de desastres naturais, criando um ecossistema digital de alerta antecipado com potencial de aplicação real.