**ALERT-AI: Sistema de Alerta Inteligente para Deslizamentos de Terra**

**Integrantes**

* Fábio Marcos Pedroso Filho – RM 560665

**Repositório GitHub:**

https://github.com/pedrosof/Challenge---Global-Solutions---Sprint-1

**Demonstração no YouTube:**

https://github.com/pedrosof/Challenge---Global-Solutions---Sprint-1

**Introdução**

O projeto ALERT-AI foi desenvolvido como parte do desafio da Global Solution da FIAP com o objetivo de aplicar Inteligência Artificial e Internet das Coisas (IoT) para prever e mitigar riscos de deslizamentos de terra. Utilizando sensores conectados via ESP32, uma rede neural leve e integração com notificações AWS SNS, o sistema fornece alertas automáticos com base em parâmetros ambientais como umidade do solo, inclinação do terreno e presença de chuva.

**Desenvolvimento**

**Arquitetura da Solução**

A solução foi construída com os seguintes componentes:

* **Sensor ESP32** com sensores de umidade do solo, chuva e MPU6050 para inclinação.
* **Backend em Flask** com endpoint /prever que recebe dados, aplica um modelo de ML e persiste no SQLite.
* **Modelo de IA (Random Forest)** treinado em dados simulados com scikit-learn.
* **Notificações AWS SNS** configuradas via boto3 com credenciais lidas de config.ini.
* **Dashboard Flask** para visualização de gráficos gerados em R e salvos no banco de dados.

**Justificativas Técnicas**

* **ESP32** foi escolhido por sua capacidade de conectividade Wi-Fi e leitura analógica.
* **Random Forest** pela boa performance com baixo custo computacional, ideal para classificação de risco.
* **SQLite** por sua leveza e integração local com o backend.
* **Flask** por ser leve e fácil de integrar com APIs REST.
* **R** para análise estatística e visualização, com os gráficos exportados e armazenados diretamente no banco.

**Circuitos Utilizados**

* Sensor capacitivo de umidade do solo
* Sensor analógico de chuva
* Sensor de movimento MPU6050
* LED azul para indicar atividade de coleta

**Códigos Implementados**

* sketch.ino: Código do ESP32 responsável pela coleta e envio dos dados.
* backend.py: Servidor Flask que aplica IA e envia alerta para SNS.
* simula\_sensor.py: Simula leituras do sensor para testes locais.
* treinar\_modelo.py: Gera o modelo Random Forest e o salva com joblib.
* analise\_alerta.R: Gera gráficos e salva imagens binárias no SQLite.
* frontend.py: Visualiza gráficos gerados a partir do banco.

**Resultados Esperados**

* Previsão de risco com base em três parâmetros ambientais.
* Envio automático de alertas para SNS em casos de risco alto (com intervalo mínimo de 30 min).
* Dashboard visual acessível para acompanhamento dos dados.
* Sistema modular, com possibilidade de expansão para sensores reais em campo.

**Conclusões**

O ALERT-AI demonstra como a integração entre sensores, machine learning e cloud computing pode ser aplicada na prevenção de desastres naturais. A escolha de tecnologias leves e de fácil manutenção permite a replicação do sistema em áreas de risco real. Futuras melhorias podem incluir GPS para georreferenciamento, expansão para mais sensores, e armazenamento em nuvem para dados históricos.

**Anexos:**

* Código completo do ESP32 (sketch.ino)
* Backend com ML e SNS (backend.py)
* Modelo treinado (modelo\_risco\_deslizamento.joblib)
* Simulador de sensores (sensor\_simulador.py)
* Scripts em R para geração dos gráficos
* Visualizador Web (frontend.py)