Landslide triggering system

# Abstract

This topic release when I am struggling with the assignment in my college while my provided mentor forced me to have at least 1 microcontroller or an embedded device.

I was searching for the idea on the internet for a capable combination between embedded IoT & Data Engineer which could best suit for the time. Then I read 2 papers discussing about the relation between environment metrics and landslide phenomenon. One show that most of the hazard such as earthquake, flood, and landslide have the natural origin, the other represented some special pattern in temperature and soil moisture in which led to the failure of rock and mud. They use some statistic data and graphs for analyzing and this is the missing part. As I found that even NASA is trying to collect data about all landslide occurrences in the world for the purpose of building the landslide database for future forecasting model.

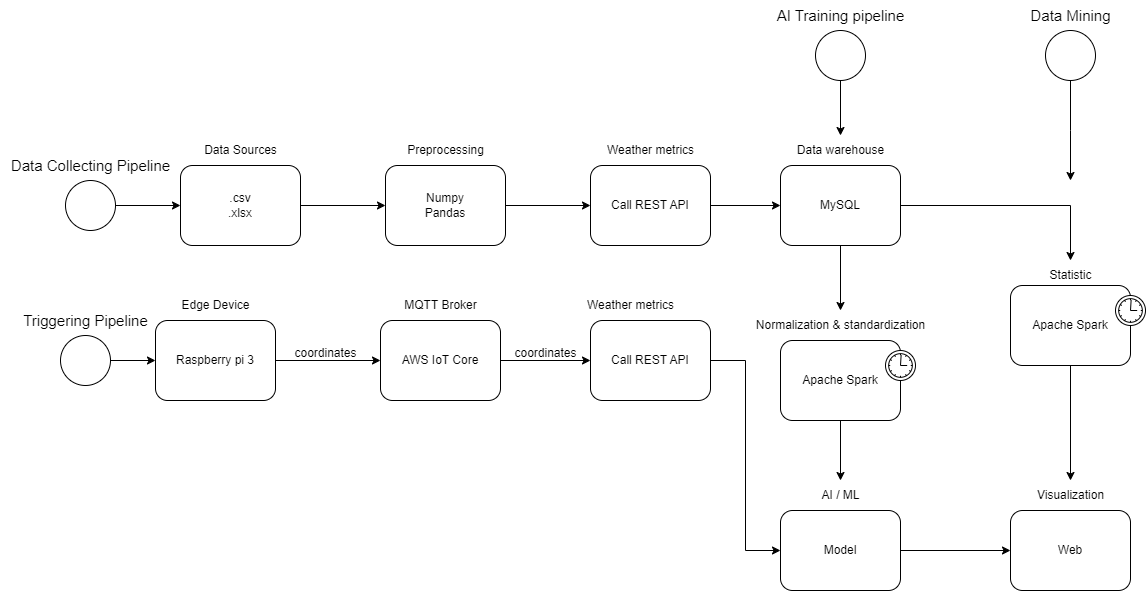
Then I realized that we should build a sample system which a small scale, low accuracy and high latency but could satisfy all demand of a combined assignment.

# Objective

A fulfillment system including data collecting, data warehouse and data processing in order to construct predicted model and risk triggering should be finished after this project. These are some sub targets.

* Applying learned knowledge about designing, deploying and maintain data warehouse.
* Building reliable, stable, and low latency collecting and processing data pipeline.
* Some statistical analysis, visualized graphs or plotting the data for special patterns identifying.
* Might contains AI/ML model for prediction solution.

# System Diagram

To give you a basic concept, I made a high-level diagram with components and relations. This will also give a simple explanation about the idea.

The diagram describes 4 main flows refer to 4 pipelines in which could be 1-time running pipeline or scheduled pipeline. First 3 pipelines could be monitored via Airflow.

The first data pipeline is used to collect and gather data for the data warehouse, the sources is Catalogue provided by NASA about recorded landslide appearances. Once the file is extracted by a python script, the coordinate is sent to the API calling service and the other needed data go to the database. The next task then send request to the Open Meteo weather API to gather and store the natural metrics in about 3 years before the landslide in that location. This pipeline might run once only and provide failure cases about ground hazard.

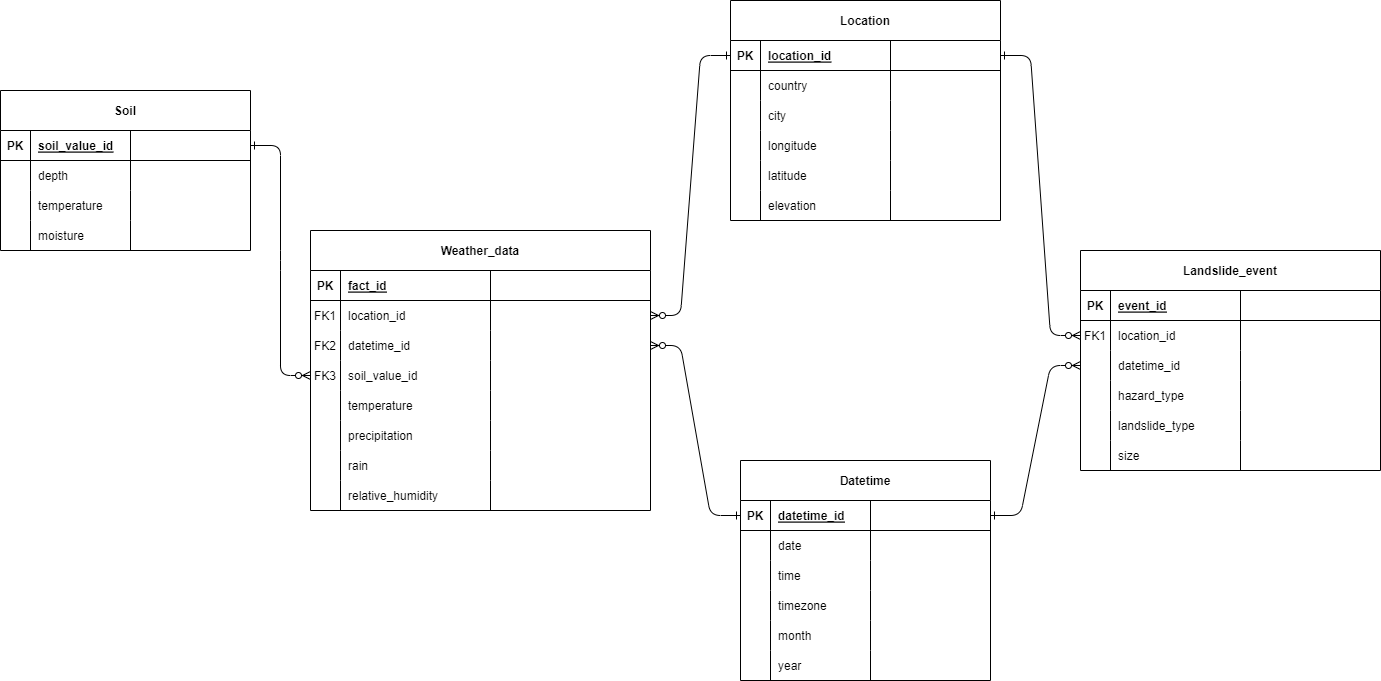
The second pipeline is an AI training pipeline which could automatically run whenever an update to the database appears. There is a Spark job for data retrieving, normalization & standardization before training process.

There is also a Spark job belongs to the third pipeline for Analyzing purpose. It will access the data and do some statistical calculation in which could further be represented via graphs in a Web UI.

The last pipeline is triggering pipeline where a raspberry pi 3 is an edge device uses a GPS module to get the current coordinate and send to MQTT Broker of AWS IoT Core. The coordinate then be received and used for weather data query before moving through the AI model for classification. The predicted result provided by AI is showed in the Web UI.

# Datawarehouse Schema Design

To store the weather data of recorded landslide occurrences, I recommend the design below. This is a galaxy schema design with 2 fact tables play the central role. (Weather data & landslide event) The dimension tables include Datetime, Soil, and Location.

The Datetime table contains time values related of the event or a set of weather data, so it has the 1-many relation with 2 fact tables. The Location table contains information about a specific geographical point with coordinate, elevation metrics while the Soil is a sub table divided from weather data and consist of more detail about soil temperature & soil moisture in different depth.

# Technology usage

## Data warehouse & processing

* MySQL or other type of RDBMS
* Python for simple tasks and Spark for complex tasks

## Protocol & Infrastructure

* HTTP, MQTT
* Docker, AWS

## UI & Monitoring

* Airflow for pipelines monitoring and HTML, CSS, JS for web UI.

## AI model

* Logistic regression or other types of binary classifier.