We have limited time during the hackathons, we are going to do the most we can do based on our skill level.

Implementing AI and IoT to the system is very complex, so let's aim for a Web App Interface that displays fire risk levels on a map

- Data Integration that fetches fire/weather data from APIs (NASA FIRMS, NOAA)
- Basic Prediction Model that analyzes the simple risks based on temperature, wind speed, humidity, etc…
- Alert System if the risks are high, then it should trigger a warning.

** If possible, let's add comments on our code so we can screenshot and remind ourselves what that piece of code does for our website **

FRONTEND:

- We will use HTML/CSS/JavaScript for simplicity

BACKEND:

- We can use Flask(Python) ← Preferable
    - pip install flask requests
- or Firebase (for real-time data)

DATA:

- NASA FIRMS API & NOAA Weather API

What are some features to include?:

- Interactive Fire Risk Map (We can use Google Maps API & Fire Data)
  Using folium instead of google maps api because: **Cons of Google Maps API:**

1. **Costs**: While Google Maps has a **free tier**, heavy usage (e.g., lots of requests or usage of certain advanced features) can quickly lead to costs. Depending on the scale of your project, this could become expensive.

    - The free tier offers a limited number of requests per day. If you're using features like geocoding or directions, you may exceed the limit quickly.
2. **Requires API Key**: To use the Google Maps API, you need to get an API key and potentially set up billing, which can be an additional setup step.

3. **Requires More Complex Setup**: You might need a more complex setup if you want to embed the Google Maps API into a Python project, as it generally requires using **JavaScript** (for a web-based project). While there are Python libraries like **gmplot**, they are more limited in features compared to the full JavaScript-based API

   (we want to make a properly functioning platform over a featureful one that stops working)

- Real-time API Integration (NASA FIRMS for wildfire hotspots)
- Risk calculation like if there is high temp + low humidity = high risk!
- Emergency Alert System, like an email or a text to alert others if the risk is extreme
- Multiple languages for anyone to read based on their preference. (language API)
- NOTE TO AK - Leaflet works on **both desktop and mobile devices**. It's designed to be responsive, meaning it adapts to different screen sizes and works well on smartphones and tablets. - folium(for the map) is adapted from Leaflet.js

4. Potential mobile access if we launch it?! NOTES!:-

How Does Leaflet.js Work?:

- You use **Leaflet.js** in a web-based application to create a **map container** (a `<div>` element in HTML), and then you use Leaflet's functions to add map layers, markers, and other elements.

  Example: A basic Leaflet map setup:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Leaflet Map Example</title>
    <link rel="stylesheet"
href="https://unpkg.com/leaflet/dist/leaflet.css" />
    <script src="https://unpkg.com/leaflet/dist/leaflet.js"></script>
</head>
<body>
    <div id="map" style="width: 100%; height: 500px;"></div>
    <script>
        // Initialize the map and set its view
        var map = L.map('map').setView([51.505, -0.09], 13); //
Coordinates for London, UK

        // Add a tile layer (background map)

L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
```

```
            attribution: '&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors'
        }).addTo(map);

        // Add a marker
        L.marker([51.505, -0.09]).addTo(map)
            .bindPopup('A pretty CSS3 popup.<br> Easily
customizable.')
            .openPopup();
    </script>
</body>
</html>
```

This code:

- Creates a map centered on the coordinates **51.505, -0.09** (London).
- Adds **OpenStreetMap** tiles as the background.
- Places a **marker** at the coordinates and attaches a **popup** that shows when clicked.

How Does Leaflet.js Relate to Folium?

Folium is a **Python wrapper** around Leaflet.js. It allows you to use **Leaflet.js**'s features inside Python applications, especially within **Jupyter Notebooks**.

So, instead of writing JavaScript code directly, you can use Folium's **Python syntax** to generate Leaflet maps and visualize geographic data.

For example, this Python code with Folium:

```
import folium

# Create a map centered at a specific latitude and longitude
m = folium.Map(location=[51.505, -0.09], zoom_start=13)

# Add a marker with a popup
folium.Marker([51.505, -0.09], popup="London").add_to(m)

# Save the map as an HTML file
m.save("map.html")
```

This will generate an HTML file containing a Leaflet map with a marker at the given coordinates.