Project Initialization and Planning Phase

Date	15 March 2024
Team ID	team-739852
Project Title	Natural Disasters Intensity Analysis and Classification using AI
Maximum Marks	3 Marks

Project Proposal (Proposed Solution) template

The proposal report aims to develop a real-time AI-powered system to detect and classify natural disasters using deep learning and image processing techniques, enabling faster emergency response and minimizing potential damage.

Project Overview	
Objective	The primary objective is to build an intelligent system that can detect the type and intensity of natural disasters in real-time using a webcam feed, helping authorities take prompt action to mitigate the impact.
Scope	The project focuses on real-time video frame capture using a webcam, processing the frames using a deep convolutional neural network (CNN), and displaying the disaster classification and intensity on the UI using OpenCV. It is built with Flask, Keras, and OpenCV.
Problem Statement	
Description	Disaster response teams and authorities often lack accurate and fast tools to detect and classify natural disasters from real-time image data. Existing systems struggle with image complexity, class imbalance, and low performance in uncontrolled environments.
Impact	Addressing this issue will allow quicker identification of disaster types and intensity levels, improve emergency preparedness, and reduce the loss of life and property by enabling faster responses and better resource allocation.
Proposed Solution	
Approach	Implementing a deep convolutional neural network (CNN) trained on

	a diverse image dataset of natural disasters, integrated with a Flask-based UI to capture live video frames, analyze them in real time, and showcase results using OpenCV.
Key Features	Real-time disaster detection using an integrated webcam. Classification of disaster type and estimation of intensity using a pre-trained model. Flask-based interface with OpenCV output for visual feedback. Image preprocessing with `ImageDataGenerator` to improve model performance. Scalable and updatable model pipeline using Keras and TensorFlow.

Resource Requirements

Resource Type	**Description**	**Specification/Allocation**
Hardware		
Computing Resources	CPU/GPU specifications, number of cores	T4 GPUs
Memory	RAM specifications	16 GB
Storage	Disk space for data, models, and logs	1 TB SSD
Software		
Frameworks	Python frameworks	Flask
Libraries	Additional libraries	tensorflow, OpenCV, NLTK, pandas, numpy
Development Environment	IDE, version control	Jupyter Notebook, Git
Data		
Data	Source, size, format	Images and videos in labeled formats; size: 20,000 samples