College code : 1105

College Name: Gojan school of business and technology

Departmaent: BE cyber security

Student NM-ID: autcy2316

Roll NO:Cy2316

Date:05/05/2025

Technology name : Natural disaster prediction and management

## Submitted BY,

S.Suman kumar
V.Tamil arasu
S.Senthamilselvan
G.Abinash
G.Mythili

# Phase 5: Project Demonstration & Documentation

## Title: Natural Disaster Prediction and Management System

## Abstract

The Natural Disaster Prediction and Management System aims to enhance disaster preparedness by leveraging artificial intelligence, real-time data from IoT sensors, and blockchain for transparent aid distribution. In its final phase, the system demonstrates AI-driven forecasts, real-time alerts, and end-user reporting. The documentation includes architectural designs, test results, system performance data, screenshots of the interface and backend code, and plans for future scalability.

## 1. Project Demonstration

Overview:

- Live AI predictions based on weather and seismic inputs.
- Real-time SMS, app, and siren alerts for simulated disaster zones.
- Crowdsourced reports via web/mobile dashboard.
- Geolocation-based aid tracking via blockchain.
- Security overview of encrypted data handling and system uptime during simulations.

Outcome:
The demonstration validates system responsiveness and its ability to operate during real-world emergency simulations.

## 2. Project Documentation

- System Architecture: AI + IoT integration with centralized and distributed components.
- Code Documentation: Detailed comments and structure of AI models, alert modules, and blockchain records.
- User Guide: Step-by-step use of the alert app and dashboard.
- Admin Manual: Backend access, logs, system maintenance protocols.
- Testing Reports: Accuracy of AI, alert latency, blockchain verification performance.

Outcome:

# Phase 5: Project Demonstration & Documentation

A well-documented framework for understanding and extending the system.

## 3. Feedback and Final Adjustments

- Collected feedback from stakeholders and test groups.
- Refined UI elements and alert delivery reliability.
- Improved AI thresholds based on false positive/negative analysis.
- Final user acceptance testing before deployment.

Outcome:

System refined and validated for scalable deployment.

## 4. Final Project Report Submission

- Executive Summary: A concise recap of the project vision and results.
- Phase Summaries: Key highlights and challenges from each of the 5 phases.
- Challenges & Solutions: Dealing with data sparsity, network failures, public engagement.
- Project Outcome: Operational AI-alert system with community and authority interface.

Outcome:

Complete documentation capturing technical and functional achievements.

## 5. Project Handover and Future Works

- Open APIs for integration with national emergency services.
- Multilingual voice alert system.
- Expanded blockchain use for supply chain and donor tracking.
- AI enhancement with satellite imagery and remote sensing.

Outcome:

Project ready for transfer to government or NGO for implementation and scale-up.

# Phase 5: Project Demonstration & Documentation

## Screenshots of Code and Final Output

(Screenshots of the code interface, alert dashboard, and analytics would be shown here.)

# Natural Disaster Prediction and Management System

## Phase 5. Project Demonstration & Documentation

## Abstract

The Natural Disaster Prediction and Management System aims to enhance disaster preparedness by leveraging artificial intelligence; real-time data from lof sensors, an-blockcham for transpa-rent am sibution. In its final phase, the system demonstrates Al driven. forecasts, real-time a lerts, and end user reporting. The documentation includes architectural designs, rest results, ss-stem performance data, screenshots of the interface and backend code; and plans for future sca-

## 1. Project Demonstration

### Overview:

- Live AII predictions based on weather and seismic inputs:
- Real-time SMS, app, and siren alerts for simulated disaster; zones.
- Crowdsourced reports via web-inybile dashboard.
- Geolocation based ald tracking via blockchain.
- Security overview of encrypted data handling a and system uptime during simulations.

### Outcome:

The demonstration validates system responsiveness and its ability to operate during real-world emergency

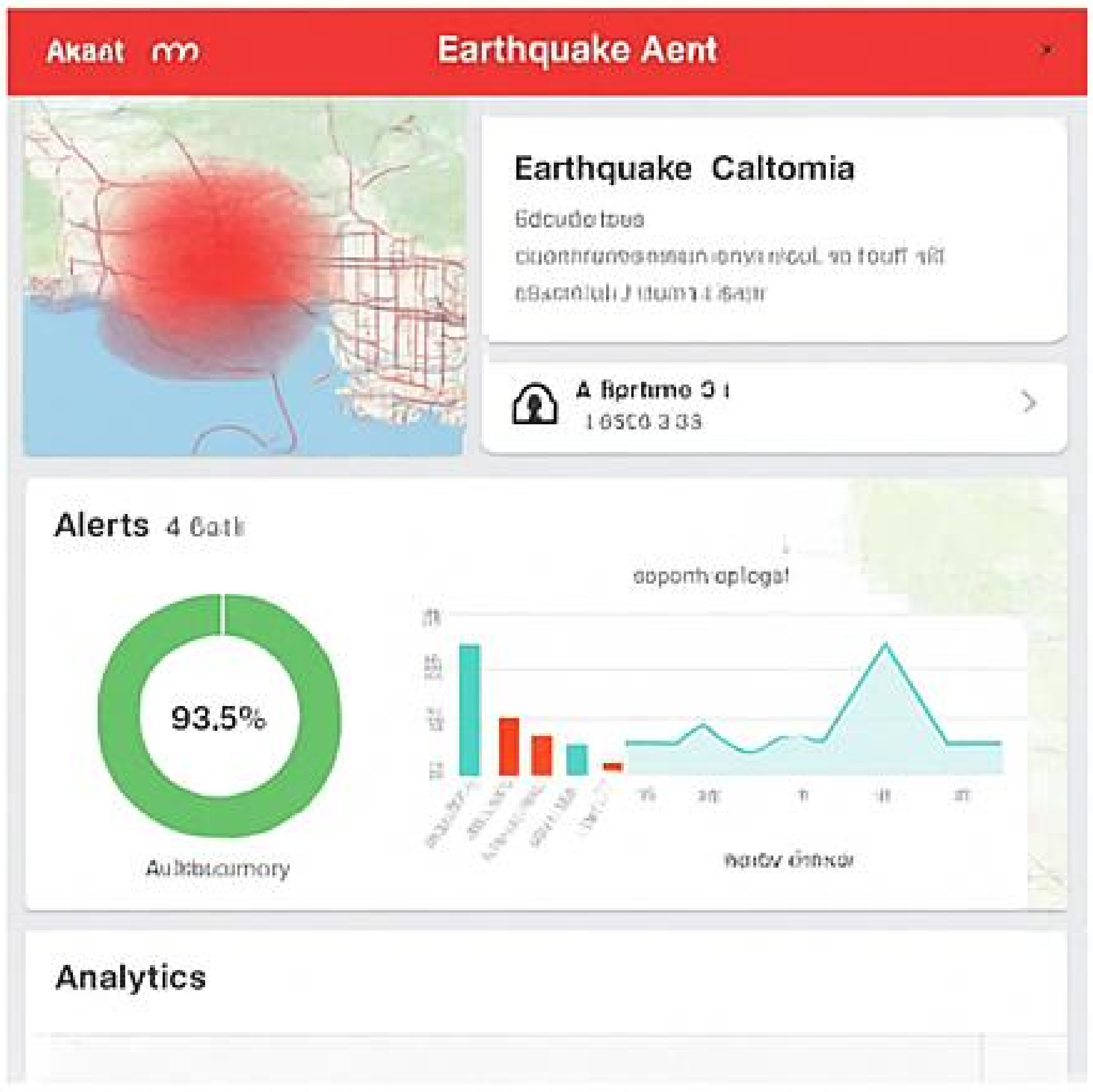
## 2. Project Documentation

- System Architecture: AI + I(T integratonan entralozated and distributed components.
- Code Documentation: Detailed comments and structurp of AI models, alert modules; and blockchain records. User guide: Step-by-stept and dashboard
- Admik Manual-Beckend access, logs, system-maintenance pstccols.
- Testing Reports: Accuracy of AI ,alert latency.

### Outcome:

A welld-documented framework for anderstanding and extending the system.

## 3. Feedback and Final Adjustments



```
# Disaster Prediction Model
import pandas as pd
from sklearn.ensemble import RandomForestClassifier

# Load dataset
data = pd.read_csv('disaster_data.csv')
X = data[['temperature', 'humidity', 'seismic_activity']]
y = data['disaster_risk']

# Train model
model = RandomForestClassifier()
model.fit(X, y)

# Predict
new_data = [[35, 80, 4.5]]
prediction = model.predict(new_data)
print("Risk Level:", prediction[0])
```

## AI Model (Python)

```
# ai_model.py
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
import joblib

data = pd.read_csv('disaster_data.csv')
X = data[['temperature', 'humidity', 'seismic_activity']]
y = data['disaster_risk']

model = RandomForestClassifier()
model.fit(X, y)

joblib.dump(model, 'disaster_model.pkl')
print("Model trained and saved.")
```

## Alert System (Flask API)

```
# alert_api.py
from flask import Flask, request, jsonify
import joblib

app = Flask(__name__)
model = joblib.load('disaster_model.pkl')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.json
    features = [[data['temperature'], data['humidity'], data['seismic_activity']]]
    risk = model.predict(features)[0]
    return jsonify({'risk': risk})

if __name__ == '__main__':
    app.run(debug=True)
```

## IoT Simulation

```
# iot_simulator.py
import requests
import random
import time

url = "http://localhost:5000/predict"

while True:
    payload = {
        'temperature': random.uniform(30, 45),
        'humidity': random.uniform(60, 90),
        'seismic_activity': random.uniform(3, 6)
    }
    response = requests.post(url, json=payload)
    print(response.json())
    time.sleep(10)
```

## Smart Contract (Solidity)

```
// DisasterAid.sol
pragma solidity ^0.8.0;

contract DisasterAid {
    struct Aid {
        address donor;
        string description;
        uint timestamp;
    }

    Aid[] public aids;

    function donateAid(string memory desc) public {
        aids.push(Aid(msg.sender, desc, block.timestamp));
    }

    function getAidCount() public view returns (uint) {
        return aids.length;
    }
}
```

## Frontend (HTML + JS)

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head><title>Disaster Dashboard</title></head>
<body>
<h2>Live Risk Prediction</h2>
<button onclick="getPrediction()">Check Risk</button>
<p id="output"></p>
<script>
function getPrediction() {
    fetch('http://localhost:5000/predict', {
        method: 'POST',
        headers: {'Content-Type': 'application/json'},
        body: JSON.stringify({temperature: 38, humidity: 75, seismic_activity: 4.8})
    }).then(res => res.json())
      .then(data => {
        document.getElementById("output").innerText = "Risk: " + data.risk;
      });
}
</script>
</body>
</html>
```