AI-Powered Early Warning System for Extreme Climate Events Using LSTM: Data Collection, Model Development, and Deployment

Step 1: **Problem Definition & Data Collection**

Objective:

Develop an AI-powered Early Warning System (EWS) utilizing Long Short-Term Memory (LSTM) to predict extreme climate events, including droughts, floods, and heat waves.

Data Requirements:

We will need historical time-series climate data, which can be obtained from:

- Meteorological Stations (e.g., Kenya Meteorological Department, Eldoret Meteorological Station)
- Remote Sensing & Satellites (e.g., NASA, Copernicus, NOAA)
- IoT Sensors (real-time environmental monitoring)
- Open-source Climate Data (e.g., ERA5, World Bank Climate Portal)

Key Climate Variables:

- Temperature (°C)
- Rainfall (mm)

Step 2: **Data Preprocessing**

Tasks:

- Data Cleaning: Handle missing values (e.g., interpolation, mean imputation).
- Feature Engineering: Create lag features, moving averages, and seasonal trends.
- Normalization: Scale data using MinMaxScaler to improve model convergence.
- Train-Test Split: Typically, use 80% for training and 20% for testing.
- Reshape Data: Convert to 3D format (samples, time steps, features) for LSTM input.

Step 3: **Building the LSTM Model**

LSTM Architecture:

- Input Layer: Takes the scaled climate data
- LSTM Layers: Capture long-term dependencies in climate patterns
- Dropout Layers: Prevent overfitting
- Dense Layer: Output final prediction

Example Model (Python - TensorFlow/Keras):

Step 4: **Model Training & Evaluation**

Training:

- Use Mean Squared Error (MSE) or Mean Absolute Error (MAE) as the loss function.
- Use Adam optimizer for better convergence.

Evaluation:

- Metrics: RMSE, MAE, and R^2
- Visualization: Compare actual vs. predicted values

Step 5: **Deployment & Real-Time Monitoring**

Integrate with:

- Web App & API for user access (Flask/Django FastAPI)
- Mobile Alerts (SMS, WhatsApp, Telegram)
- Cloud Deployment (Google Cloud, AWS, Azure)