Rising Waters: A Machine Learning Approach to Flood Prediction

DATE	28-02-2026
TEAM ID	LTVIP2026TMIDS89043
PROJECT NAME	Rising Waters: A Machine Learning Approach to Flood Prediction
MAXIMUM MARKS	3 MARKS

3.4 TECHNOLOGY STACK:

1. Data Acquisition

Flood prediction relies on diverse datasets. Common technologies and sources include:

- **Remote Sensing & Satellite Data**
  - Platforms: NASA EarthData, Copernicus Sentinel Hub
  - Libraries: sentinelsat, rasterio, GDAL
- **Meteorological Data**
  - Sources: NOAA, Open WeatherMap API, Weather Underground
  - Tools: Python requests, pandas
- **Hydrological Data**
  - River flow, rainfall, water levels from local or national agencies
  - APIs or CSV/NetCDF datasets

2. Data Storage & Management:

Handling large-scale geospatial and temporal data requires:

- **Databases**
  - PostgreSQL + PostGIS – for geospatial data
  - MongoDB – for semi-structured or JSON data
- **Cloud Storage**
  - AWS S3, Google Cloud Storage, Azure Blob Storage
- **Data Formats**
  - CSV, NetCDF, GeoTIFF, HDF5

3. Data Preprocessing & Feature Engineering:

Preparing data for ML models involves:

- **Python Libraries**
  - pandas, numpy – data wrangling
  - scikit-learn – scaling, encoding, and preprocessing
  - xarray, rasterio, geopandas – for geospatial/temporal analysis

    - matplotlib, seaborn, plotly – visualization of flood patterns
- **Techniques**
    - Handling missing data
    - Normalization/standardization
    - Feature extraction (e.g., rainfall accumulation, river level change, slope, soil saturation)

4. Machine Learning & Modeling

Choosing predictive models for flood forecasting:

- **Algorithms**
    - **Classical ML**: Random Forest, Gradient Boosting (XGBoost, LightGBM), Support Vector Machines
    - **Deep Learning**: LSTM, GRU (for time-series prediction)
    - **Hybrid Models**: CNN-LSTM for spatio-temporal patterns
- **Frameworks**
    - scikit-learn
    - TensorFlow / Keras
    - PyTorch
    - XGBoost / LightGBM
- **Model Evaluation**
    - Metrics: RMSE, MAE, Precision/Recall (if classification), ROC-AUC
    - Cross-validation and time-series split

5. Deployment & Visualization:

Once the model is trained, deployment and visualization are key:

- **Web & Dashboard**
    - Streamlit or Dash for interactive flood dashboards
    - Flask or FastAPI for backend APIs
- **GIS & Mapping**
    - Leaflet.js, Folium, Kepler.gl – for real-time flood maps
- **Cloud & CI/CD**
    - AWS EC2, GCP Compute Engine, Azure ML
    - Containerization with Docker, orchestration with Kubernetes
- **Monitoring**
    - Prometheus + Grafana for monitoring model predictions and data pipelines