**Mentor’s session day -3**

Admin Part: Form to Tag Areas

Create a form for tagging different areas. This will involve collecting information about various locations to enable accurate weather predictions.

User Part: Daily Weather Forecast

Provide daily weather forecasts for users based on the tagged areas. Users can access weather predictions specific to their location.

Machine Learning Model: Training and Prediction

Train a machine learning model to predict the weather. The model will analyze data to forecast the weather for different areas and provide users with accurate predictions.

Database Implementation with MySQL

Two datasets will be used, and they will be implemented in MySQL for backend purposes. This involves storing and managing the data required for the form, user forecasts, and machine learning model predict

**Mentor’s session day -4**

Prediction Model

1. Objective: Predict whether it will rain and if a flood will occur using a forecast dataset.
2. Dataset: Only the provided forecast dataset will be used (no new dataset).
3. Inputs:
   * Date Entry: The date for which the prediction is to be made.
   * City Database: Information about the cities.
4. Outputs:
   * Rain Prediction: Will it rain or not?
   * Flood Prediction: Will a flood happen or not?
5. Location Consideration: The model will calculate flood probability for different areas in Kolkata using user-provided area, city, and location details.
6. Geographical Data: The model should consider latitude and longitude, which can be obtained using plugins or APIs.

Learning About Pickling

1. Pickling in Python: A process to serialize and save models or data so they can be easily loaded later.

Finding Location with Latitude and Longitude

1. Task: Find a location using specified latitude and longitude.
2. Tool: Use the geopy library in Python, which works with geographic locations.
3. API Usage: Your instructor mentioned using an API to obtain latitude and longitude data.

Steps

1. Train a Machine Learning Model from dataset1. Use pickle if required to save the model
2. Generate a dummy records database with around 40-50 records, this dataset will have all the columns from dataset1 along with 2 additional ones-> area and city name
3. For each of these records, calculate flood probability
4. The columns -> "flood probability, area, city, latitude, longitude" should be published to a MySQL DB
5. On the user portal, there should be option to enter location, and flood scores of nearby locations will be displayed on the portal
6. On the admin/forecasting side, we should have a form, that inputs values for all of the columns in dummy records database and finds the probability score and post that, it populates the database with the score, area name, city name, latitude and longitude.
7. n step 3and 6, probability calculation should be by the ML Model trained in step 1

**Mentor’s session day -5**

#### *we are developing a software system for predicting flood probability using LSTM (Long Short-Term Memory) models. The system has two primary components: a weather forecaster module (admin module) and a predictor module (user module).*

#### *Weather Forecaster Module (Admin Module):*

#### *The admin can input data through a form with fields such as city, area, latitude, and longitude.*

#### *This module is responsible for training the model with the provided data.*

#### *The trained model is then saved to the hard drive and connected to the predictor module for making predictions.*

#### *Predictor Module (User Module):*

#### *Users can select a city and area.*

#### *The software loads the pre-trained one model and uses it to predict the flood probability for the selected location.*

#### *Additionally, we want the one model to be highly accurate, focusing specifically on flood probability prediction. To achieve this, you plan to remove irrelevant columns like latitude, longitude, and areas from the dataset before training the model.*

Prediction Module:

The predictor module using a pre-trained the one model is fully implemented and operational.

Contact Us Page:

The "Contact Us" page functionality has been implemented using MySQL to store user inquiries.

Pickling in Python:

Successfully integrated pickling in Python into the project (Forecaster).

API for Live Weather Updates:

Currently researching APIs that can provide live weather updates to enhance prediction accuracy and functionality.

Dashboards:

Creating dashboards for both the forecaster page and the prediction page.

Website Updates:

Working on making changes to the website to improve functionality and user experience.

IMPORTANT NOTES ABOUT WEBSITE

First: Change a photo in the slider on the website.

Second: Add photos of flood-prone areas, organized by state.

Third: Fix the footer on the home and about pages.

Fourth: When making the website responsive, I can't see both points: "Predict" and "forecaster new."

Fifth: "Our Services" section, the last column, "Animal Rescue and Veterinary Services," with the description "Assistance for rescuing and caring for pets and livestock affected by floods," should be centered in the middle.

For the Rescue Request Form, the destination should be limited to Mumbai only, and the areas need to be provided. This should be added to MySQL

**Mentor’s session day -6**

Graph Feature: Keep the graph feature, but make it more attractive (beautify it).

Predictor Output Color: Change the predictor's output color from green to red.

Simplified Input for End Users: End users should only need to input the year and location for flood prediction.

Two Applications: Create two separate applications—one for scientists and one for end users.

Database Integration: Connect the predictor to MySQL to store data.

Dashboard Update: Add URLs to the dashboard as instructed.

Include All Details on the X-Axis and Y-Axis: Make sure that all relevant details are available for selection on both axes. This means listing all possible data points that can be plotted on the x-axis and y-axis.

User Selection via Radio Buttons: Implement radio buttons that allow users to select what data they want to display on the x-axis and y-axis. This gives users flexibility and control over the visualization.

Add Graphs: Integrate graphs into the dashboard to visually represent the data based on user-selected x and y-axis values.

Include URLs for Interaction: Add URLs to the dashboard to enhance user interaction. These could be links to more detailed reports, related websites, or additional data.