

MILESTONE 3

This Flask-based deployment provides an interactive way to use the trained Ridge Regression model for Fire Weather Index (FWI) prediction and risk assessment.

Application Design

The app is built with Flask and uses a pre-trained Ridge Regression model and scaler saved as `ridge.pkl` and `scaler.pkl`. At startup, `app.py` loads both objects so that incoming user data can be scaled in the same way as the training data and passed directly to the model for prediction.

User Interface

The `index.html` template presents a modern single-page form where users input all required features: Temperature, Relative Humidity (RH), Wind Speed (Ws), Rain, DMC, DC, ISI, BUI, and FFMC. All fields are required and validated as numeric; invalid or missing inputs trigger a small alert message displayed using Flask's flash mechanism, avoiding server crashes and improving user experience.

Prediction Workflow

When the user submits the form, the `/predict` route in `app.py` collects the values, converts them to floats, and arranges them into a feature vector. This vector is scaled with the loaded `scaler` and passed to the loaded `ridge_model`, which outputs a numeric FWI prediction. The value is rounded to three decimal places and then mapped to a qualitative risk category (Low, Moderate, High) using simple thresholds based on the FWI range.

Result Display and Risk Levels

The `home.html` template shows the predicted FWI along with a color-coded pill indicating the risk level: green for Low, amber for Moderate, and red for High. Beneath the prediction, short

guidance text explains how to interpret each category, suggesting when normal monitoring is sufficient and when heightened fire readiness may be needed. This combination of numeric output and clear labels improves interpretability for non-technical users.

Execution and Usage

To run the application, the user places `app.py`, `ridge.pkl`, `scaler.pkl`, and the `templates` folder in a single project directory, installs Flask and scikit-learn, and starts the server with `python app.py`. Visiting `http://127.0.0.1:5000` then opens the FWI input form, and submitting valid values returns the prediction screen with both FWI and the corresponding risk level, completing the prediction and deployment pipeline.