PROJECT CODE:

```
from flask import Flask, render_template, request
import pickle
import numpy as np

app = Flask(__name__)

# Load the trained model and scaler
with open('random_forest_model.pkl', 'rb') as f_model:
    model = pickle.load(f_model)

with open('scaler.pkl', 'rb') as f_scaler:
    scaler = pickle.load(f_scaler)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Feature names in the same order used during training
        feature_names = [
            'temp', 'rain', 'snow', 'holiday_No', 'weather_Clouds',
            'hour', 'day', 'month', 'year'
        ]
        features = []
        for feature in feature_names:
            value = request.form.get(feature)
            if value is None or value.strip() == '':
                return render_template('no_chance.html', prediction_text="Invalid input.")
            try:
```

```
                features.append(float(value))
            except ValueError:
                return render_template('no_chance.html', prediction_text=f"Invalid numeric value for {feature}.")

        # Convert to numpy array and reshape for scaler/model
        input_data = np.array(features).reshape(1, -1)
        # Apply scaling
        input_scaled = scaler.transform(input_data)
        # Predict
        prediction = model.predict(input_scaled)
        pred_value = prediction[0]

        # Threshold logic
        threshold = 5000
        if pred_value > threshold:
            text = f"Traffic volume is high: {int(pred_value)}"
            return render_template('chance.html', prediction_text=text)
        else:
            text = f"Traffic volume is low: {int(pred_value)}"
            return render_template('no_chance.html', prediction_text=text)

    except Exception as e:
        # In case of unexpected error, show a generic error page or message
        error_msg = f"Error occurred: {str(e)}"
        return render_template('no_chance.html', prediction_text=error_msg)

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