Air Quality Prediction System - Complete Code Analysis

🎯 System Overview

This is a comprehensive **Air Quality Monitoring and Prediction Dashboard** that:

- Fetches real-time sensor data from ThingSpeak IoT platform
- Uses machine learning to predict future air quality (PM2.5 levels)
- Provides interactive visualizations and forecasts
- Monitors multiple sensors: Temperature, Humidity, MQ135 (air quality), MQ7 (CO), and Dust

🏗 System Architecture

1. Data Source Configuration

```python
CHANNEL_ID =
READ_API_KEY =
```

- Connected to ThingSpeak IoT platform
- Reads data from 5 sensor fields

2. Model Management

```python
MODEL_DIR = "models"
LINEAR_MODEL_PATH = "linear_model.joblib"
RF_MODEL_PATH = "rf_model.joblib"
```

- Persistent model storage using joblib
- Two ML models: Linear Regression & Random Forest

🔄 Machine Learning Flow

Phase 1: Data Acquisition

`fetch_past_data(results=1000, fields=None)`

Purpose: Retrieve historical sensor data from ThingSpeak

Process:

1. **API Call**: Makes HTTP request to ThingSpeak API
2. **Data Transformation**:
   - field1 → temperature
   - field2 → humidity
   - field3 → mq135 (air quality sensor)
   - field4 → mq7 (carbon monoxide sensor)
   - field5 → dust (PM2.5 particulate matter)
3. **Feature Engineering**:
   - Converts timestamps to Unix format
   - Extracts hour of day (0-23)
   - Extracts day of week (0-6)
   - Handles missing values

Output: Clean pandas DataFrame with sensor readings and time features

Phase 2: Feature Preparation

`prepare_features(df)`

Purpose: Convert raw data into ML-ready features

Feature Selection Logic:

```python
# Temporal features (always included)
- timestamp (Unix timestamp)
- hour (0-23)
- day_of_week (0-6)

# Sensor features (included if >50% valid data)
- temperature
- humidity
- mq135
- mq7
```

Data Quality Checks:

- Minimum 10 valid dust measurements required

- Features included only if >50% data availability
- Automatic fallback to timestamp-only if sensors unavailable

Phase 3: Model Training

`train_prediction_models(force_retrain=False)`

Purpose: Train two complementary ML models

Model Caching Strategy:

- Models saved to disk with joblib
- Automatic reuse if models <1 day old
- Force retrain option available

Training Pipeline:

1. **Data Sampling**: Limits to 500 samples for performance
2. **Train-Test Split**: 80/20 split with random_state=42
3. **Feature Scaling**: StandardScaler normalization
4. **Model Training**: **Linear Regression**:
    - Simple, interpretable baseline
    - Good for linear relationships
    - Fast training and prediction

    Random Forest Regressor:
    - Handles non-linear patterns
    - n_estimators=50, max_depth=10
    - More robust to outliers
5. **Model Evaluation**: R^2 scores logged for both models
6. **Persistence**: Models saved with joblib

Phase 4: Prediction Generation

`predict_future_values(df, periods=24, freq='1H')`

Purpose: Generate future air quality predictions

Prediction Strategy:

1. **Future Timestamps**: Generate next 24 hours

2. **Feature Propagation**: Use last known sensor values
3. **Ensemble Prediction**: Average both model outputs
4. **Post-processing**: Ensure no negative predictions

Key Innovation: Uses ensemble method combining linear and non-linear approaches

📊 Visualization System

Core Prediction Chart

`plot_prediction_chart(prediction_hours=24)`

Components:

- **Historical Data**: Recent 100 data points
- **Linear Prediction**: Dotted line from Linear Regression
- **ML Prediction**: Dashed line from Random Forest
- **Ensemble Prediction**: Bold line (average of both)
- **Confidence Interval**: Shaded area (±20% of ensemble)
- **Prediction Boundary**: Vertical line marking forecast start

AQI Analysis Charts

`plot_aqi_distribution_forecast()`

AQI Categories (EPA Standards):

- Good: 0-12 μg/m³ (Green)
- Moderate: 12-35.4 μg/m³ (Yellow)
- Unhealthy for Sensitive: 35.4-55.4 μg/m³ (Orange)
- Unhealthy: 55.4-150.4 μg/m³ (Red)
- Very Unhealthy: 150.4-250.4 μg/m³ (Purple)
- Hazardous: >250.4 μg/m³ (Maroon)

Creates pie chart showing distribution of forecasted AQI categories

`plot_hourly_forecast()`

- Hour-by-hour bar chart for next 24 hours
- Color-coded bars based on AQI category

- Reference lines at key AQI thresholds

Sensor-Specific Predictions

Individual Sensor Forecasts:

- `plot_mq135_prediction_chart()`: Air quality sensor
- `plot_mq7_prediction_chart()`: Carbon monoxide sensor

Combined Sensor Analysis:

- `plot_combined_mq135_mq7_chart()`: Compare gas sensors
- `plot_mq135_dust_combined_chart()`: Air quality correlation
- `plot_combined_mq135_mq7_dust_chart()`: Full sensor suite

Note: Individual sensors use simplified prediction (normal distribution around recent average)

Analytical Charts

- `plot_sensor_correlations()`: Correlation heatmap between all sensors
- `plot_comparison_chart()`: Past 24h vs Future 24h comparison

🎯 Forecast Metrics System

`generate_forecast_metrics(hours_ahead=24)`

Purpose: Generate comprehensive forecast summary

Metrics Calculated:

1. **Current Status**:
    - Current PM2.5 value
    - Current AQI category
    - Status color coding
2. **Trend Analysis**:
    - Compare first half vs second half of recent data
    - Trend direction: improving ⬇️, stable ➡️, worsening ⬆️
    - 10% threshold for trend detection
3. **Forecast Summary**:
    - Average predicted value

- Maximum value and timestamp
- Minimum value and timestamp
- Most frequent AQI category

Output: Structured JSON with actionable insights

🔧 Technical Implementation Details

Error Handling & Robustness

- Comprehensive try-catch blocks
- Graceful degradation when sensors fail
- Automatic fallback to simpler models
- Logging throughout the pipeline

Performance Optimizations

- Model caching (1-day expiry)
- Limited data sampling (500 points max)
- Efficient pandas operations
- Minimal API calls

Data Quality Assurance

- Missing value handling
- Outlier detection through clipping
- Minimum data requirements
- Feature availability checks

🎨 Visualization Features

Interactive Elements

- Plotly-based interactive charts
- Hover information
- Zoom and pan capabilities
- Color-coded AQI categories

User Experience

- Clear chart annotations
- Prediction boundary markers
- Confidence intervals
- Multiple time perspectives

🔮 Prediction Methodology

Strengths:

1. **Ensemble Approach**: Combines linear and non-linear models
2. **Feature Engineering**: Temporal patterns captured
3. **Real-time Data**: Live sensor integration
4. **Multiple Sensors**: Comprehensive environmental monitoring

Limitations:

1. **Sensor Dependency**: Individual sensors use simplified prediction
2. **Static Features**: Future sensor values assumed constant
3. **Short-term Focus**: Optimized for 24-hour forecasts
4. **Linear Assumptions**: May miss complex atmospheric interactions

🚀 Potential Enhancements

1. **Advanced Models**: LSTM, ARIMA, or Prophet for time series
2. **Weather Integration**: Add meteorological data
3. **Spatial Analysis**: Multiple location support
4. **Alert System**: Threshold-based notifications
5. **Model Updates**: Online learning capabilities

📈 Use Cases

1. **Public Health**: Early warning for sensitive individuals
2. **Urban Planning**: Pollution pattern analysis
3. **Industrial Monitoring**: Compliance tracking
4. **Research**: Environmental trend studies
5. **Personal Wellness**: Daily activity planning

This system represents a complete IoT-to-insights pipeline, transforming raw sensor data into actionable air quality predictions through machine learning and interactive visualization.