Project Documentation: IoT-based Real-time Energy Monitoring System

1. Project Overview

This project implements a **real-time energy monitoring and theft detection system** using **Streamlit**, **FastAPI**, and **machine learning**. It provides:

- **Real-time energy consumption tracking** (current, voltage, power)
- **Anomaly detection** (power theft, faulty equipment)
- **Automated power optimization** (industrial machines, smart homes)
- **Interactive dashboard** with live updates

Target Applications

- **Industrial:** Reduce power consumption in factories by optimizing machine usage.
- **Commercial:** Monitor energy usage in offices, malls, and data centers.
- **Residential:** Smart home energy management.

2. System Architecture

Core Components

Component	Description
`app.py`	Main Streamlit dashboard (real-time monitoring, alerts)
`api_server.py`	FastAPI backend (data processing, anomaly detection)
`data_simulator.py`	Simulates ESP32 sensor data (fallback if no hardware)
`anomaly_detection.py`	ML model (Isolation Forest) for theft/fault detection
`dashboard_components.py`	UI components (graphs, metrics, alerts)
`api_client.py`	Connects to ESP32/industrial sensors
`utils.py`	Helper functions (data storage, calculations)

Data Flow

```
ESP32 Sensors → FastAPI (Backend) → Streamlit Dashboard → User Alerts
                                ↓
                    (Anomaly Detection)
                                ↓
                (Automated Power Optimization)
```

3. Key Features

1. Real-time Monitoring

- **Live Power Consumption Graphs** (Plotly)
- **Current/Voltage/Power Metrics**
- **Auto-refresh (5s intervals)**

2. Anomaly Detection

- **Machine Learning (Isolation Forest)**
- **Alerts for:**
  - Power theft
  - Equipment malfunction
  - Unexpected consumption spikes

3. Industrial Power Optimization

- **Smart Load Balancing:**
  - If `Machine_1` is idle, reduce its power while maintaining `Machine_2`.
- **Predictive Maintenance:**
  - Detect abnormal power patterns before failure.

4. Alert System

- **Visual (Red/Green indicators)**
- **Threshold-based alerts**
- **Historical anomaly logs**

5. Data Management

- **Json storage**

- **API + Simulation fallback**

4. Project Structure

```
IoT-Energy-Monitoring/
├── app.py                    # Streamlit dashboard
├── api_server.py             # FastAPI backend
├── data_simulator.py         # Simulated sensor data
├── anomaly_detection.py      # ML model training/prediction
├── dashboard_components.py   # UI components
├── utils.py                  # Data helpers
├── api_client.py             # ESP32 communication
├── requirements.txt          # Python dependencies
└── README.md                 # Setup guide
```

5. Installation & Setup

Dependencies (`requirements.txt`)

```
streamlit==1.31.0
fastapi==0.103.0
uvicorn==0.23.2
pandas==2.0.3
plotly==5.18.0
scikit-learn==1.3.0
requests==2.31.0
numpy==1.24.3
```

Run the System

1. **Start Backend (FastAPI)**`python api_server.py`
2. **Run Simulator (if no ESP32)**`python data_simulator.py`
3. **Launch Dashboard**`streamlit run app.py --server.port 5000`
4. **Access Dashboard:**

Open http://localhost:5000 in a browser.

6. Deployment

Streamlit Cloud

1. Push to **GitHub**.
2. Deploy on share.streamlit.io.
3. Set `app.py` as the main file.

7. Future Scope

1. Smart Industry Integration

- **Automated power allocation** based on production demand.
- **Predictive maintenance** using ML.

2. Home Automation

- **Smart plugs** integration (Tuya/Sonoff).
- **AI-based scheduling** (reduce consumption during peak hours).

3. Scalability

- **Cloud (AWS/GCP)** for large-scale deployments.
- **Multi-tenant dashboards** (factories, buildings).

8. Conclusion

This system enables: ✅ **Real-time energy monitoring**

✅ **Power theft detection**

✅ **Industrial load optimization**

✅ **Scalable for smart cities**

Next Steps:

- Integrate with **industrial PLCs**.
- Add **solar/wind energy tracking**.
- Deploy in a **pilot factory**.

GitHub Repo: https://github.com/SreeAditya-Dev/IoT-based-real-time-energy-monitoring-system