Project Title: The Global Fleet Orchestrator

Subtitle: *Risk-Aware Workload Scheduling using Time Series Forecasting & Conformal Prediction*

---

1. High-Level Architecture

This project simulates the "brain" of Microsoft's **Project Forge**[1]. It forecasts datacenter conditions (Heat/Load) and uses Conformal Prediction to rigorously assess the risk of "Thermal Runaway" before approving AI workloads.

The Data Flow:

1. **Ingest:** Synthetic Telemetry (Weather, Grid Carbon, IT Load) for 3 Regions.
2. **Forecast:** Predict the next 24 hours of Temperature and IT Load.
3. **Risk Assess:** Apply **Conformal Prediction** intervals to quantifying the uncertainty (e.g., "We are 95% sure temp won't exceed X").
4. **Decide:** The "Scheduler Logic" routes the workload to the safest, most efficient region.
5. **Visualize:** Streamlit Dashboard for the "Human-in-the-Loop."

---

2. Phase-by-Phase Execution Plan

Phase 1: The "Digital Twin" (Data Engineering)

Goal: Generate realistic telemetry based on the Microsoft context documents.

- **Action:** Create a Python script (`data_gen.py`) to simulate 3 regions with distinct physics:
    - **Arizona (West US 3):**
        - *Physics:* High Solar (Low Carbon day), High Water (WUE ~1.52)[2].
        - *Constraint:* High risk of thermal throttling if temp > 40°C.
    - **Ireland (North Europe):**
        - *Physics:* Low Water (WUE ~0.02)[3], Variable Carbon (Wind).
        - *Constraint:* Low cooling capacity (relies on outside air).
    - **Wyoming (West Central US):**
        - *Physics:* "Goldilocks" zone. Low PUE (1.125)[4].
        - *Constraint:* Limited Grid Capacity (simulated).

Phase 2: The "Oracle" (Time Series Forecasting)

Goal: Predict the "Operating Conditions" for the next 24 hours.

- **Problem Statement:** You cannot schedule based on *current* temp; you must schedule based on *future* temp.
- **Methodology:**
    - Treat `External_Temp` and `Grid_Carbon` as time-series targets.
    - **Model:** Use a Regression model (XGBoost or Linear Regression) with time-based features (Hour of Day, Day of Year, Lagged Values).
    - *Why:* Simple, robust, and easy to explain during an interview.

Phase 3: The "Safety Net" (Conformal Prediction)

Goal: Quantify Risk. (This is your "Senior" differentiator).

- **The Logic:** A point forecast of 34°C is dangerous if the safety limit is 35°C. What if the model is off by 2 degrees?
- **Implementation:**
    - **Split Conformal:** Use a calibration set to measure how "wrong" your model usually is.
    - **The Output:** Instead of `34°C`, your model outputs `[33°C, 36°C]`.
    - **The Decision Rule:** "If the **Upper Bound** (>35°C) breaches the Safe Operating Envelope, **DENY** the workload."

Phase 4: The "Command Center" (Visualization)

Goal: A Streamlit Dashboard that tells the story.

---

3. Detailed Visualizations (The "Eye Candy")

You will build three specific views in Streamlit. Each corresponds to a specific interview question you will face.

View A: The "Forecast & Risk" Monitor

- **Purpose:** Proves you understand Reliability and SLAs.
- **Visualization Description:**
    - **X-Axis:** Time (Next 24 Hours).
    - **Y-Axis:** Temperature (°C).
    - **Line 1 (Blue):** The Model Forecast (Expected Temp).
    - **Shaded Area (Light Blue):** The **Conformal Prediction Interval** (95% Confidence).
    - **Line 2 (Red Dashed):** The "Thermal Breaker" Limit (35°C).
- **The "Aha!" Moment:** Highlight a specific hour where the *Forecast* is safe (34°C) but the *Interval* crosses the Red Line (36°C). Label this **"Risk Blocked"**.

View B: The "Carbon vs. Water" Scatter

- **Purpose:** Proves you understand the PUE/WUE trade-offs[5555].
- **Visualization Description:**
    - **Chart Type:** Animated Scatter Plot (Plotly).
    - **X-Axis:** Carbon Intensity (gCO2/kWh).
    - **Y-Axis:** Water Usage (Litres).
    - **Dots:** The 3 Regions (Arizona, Ireland, Wyoming).
    - **Animation:** As you scrub through the "Time of Day" slider, the dots move.
        - *Arizona* moves rapidly up the Y-axis (Water) at noon.
        - *Ireland* moves right on the X-axis (Carbon) when the wind stops.

View C: The "Scheduler" Outcome

- **Purpose:** Proves you can drive "Process Optimization"[6].
- **Visualization Description:**
    - **Metrics Bar:**
        - "Original Carbon Footprint" vs. "Optimized Footprint".
        - "Water Saved (Liters)".
    - **The "Switch":** A toggle button: **"Enable Project Forge Logic"**.
    - **Result:** When toggled, the charts update to show the load moving from Arizona to Wyoming, creating a visible dip in global water consumption.

---

4. Implementation Steps (The "How-To")

1. **Setup Repo:** `git init microsoft-resource-efficiency-portfolio`.
2. **Dependencies:** `streamlit`, `pandas`, `numpy`, `scikit-learn` (for the model), `mapie` (optional, for easy Conformal Prediction), `plotly`.
3. **Data Script:** Write `generate_data.py`. Focus on the "Arizona Water Spike" logic—make sure WUE correlates with Temp[7].
4. **Model Training:**
    - Split data: `Train` (Jan-Oct), `Calibrate` (Nov), `Test` (Dec).
    - Train Regressor on `Train`.
    - Calculate residuals on `Calibrate` to get your "q" (quantile).
5. **Build App:**
    - Use `st.line_chart` for the Time Series.
    - Use `st.plotly_chart` for the Interactive Scatter.
6. **The Write-Up:** Your `README.md` must explain *why* you used Conformal Prediction (Risk aversion in Critical Environments).