**INFICON Assignment Report**

**Methodology Overview**

This project aimed to develop a proof-of-concept (POC) weather analysis application that integrates multiple open data sources, compares short-term forecasts to historical norms, and generates readable, human-centric summaries using a large language model (LLM). A first-pass implementation of optional bonus features was attempted: aggregating weather-related content from online news and social media (Reddit), and standing up a lightweight MCP server with a basic command-line chatbot interface to simulate real-time interaction. Included are also files to support an LLM-backend for the CLI agent. See Jupyter notebook for additional details. Design decisions prioritized demonstrating key capabilities under time constraints while maintaining quality, modularity, and reliability.

**Design Considerations**

Three primary principles guided the design:

1. **Balancing Complexity and Practicality:**  
   Trade-offs were made between showcasing LLM capabilities, building in robust error handling, and maintaining an architecture flexible enough for future extension. In a production setting, such decisions would typically be driven by ROI and business requirements; here, clarity and demonstration value were prioritized.
2. **Clear, Modular Architecture:**  
   The workflow was designed to be comprehensible to developers with general experience but no prior context. Emphasis was placed on modular, well-commented code to support maintainability and iterative development.
3. **Resource-Aware Model Integration:**  
   The system supports both OpenAI’s hosted GPT endpoint for high-quality output and a fallback local GGUF model for cost-effective inference on constrained hardware. This dual approach illustrates the kind of pragmatic model selection often required in applied AI systems.

**System Architecture**

* **Data Sources:** Forecast and historical weather data were sourced from NOAA, Open Meteo, and Weatherbit. NOAA USW stations were prioritized for authoritative observations.
* **News & Social Media:** A preliminary integration of news headlines and weather-related Reddit posts was implemented. Posts were filtered by location and weather-related keywords.
* **Dataflow:** All inputs are retrieved, standardized, and synthesized via the LLM. The architecture enables future extensions with minimal refactoring.
* **Model Selection:** A hybrid approach supports both hosted and local inference, demonstrating real-world adaptability to infrastructure constraints.

**Assumptions**

The implementation assumes a USW station exists within ~1° latitude/longitude of a queried location. While effective for U.S. cities, broader coverage would require interpolation or global station mapping. Future versions could relax this constraint or interpolate between stations.

**Strategies for Handling Data Inconsistencies and LLM Hallucinations**

To reduce hallucination risk and improve coherence:

* **Pre-Aggregated Inputs:** Structured, preprocessed data limits ambiguity and enforces consistency.
* **Expert-Style Prompting:** Prompts emulate domain reasoning, nudging the LLM toward grounded conclusions.
* **Caveats in Output:** Summaries include explicit disclaimers when uncertainty or inconsistency is detected, enhancing user trust.

These measures combine prompt engineering best practices with careful data preprocessing to boost interpretability, resilience, and trustworthiness in LLM-generated output.

**Recommendations for Future Development**

Future improvements should be assessed by impact and effort across key dimensions. Below is a non-exhaustive list of recommendations:

* **Robustness & Reliability:**
  + Add robust exception handling and standardized error responses.
* **Capability Expansion:**
  + **CLI:** Extend with a lightweight conversational agent to guide interaction.
  + **Diagnostics:** Add richer logging and debugging tools.
  + **News & Reddit:** Improve the filtering, deduplication, and summarization of external sources. Output formatting could also be optimized to respect LLM token budgets without degrading informational value.
* **LLM Optimization:**
  + Explore prompt chaining, RAG, or fine-tuning to improve summarization quality and efficiency.
  + Evaluate models based on quality, cost, and deployment constraints.
* **Data Source Strategy:**
  + Investigate whether more nuanced prompt design could reduce dependence on external APIs for context.

**Testing Recommendations**

Future testing should emphasize reliability and robustness. Below is a non-exhaustive list of recommendations.

* **Conflicting Data:** Ensure that discrepancies are surfaced with clear model commentary.
* **Sparse Regions:** Evaluate fallback behaviors when data is missing or low-quality.
* **Input Sensitivity:** Test how input variability affects model stability and guide architectural tuning.

A prioritization matrix could help target refinements with the greatest cost-benefit ratio.