1. Kaggle database

Environmental sensor telemetry data. (2020, July 20). Kaggle.
https://www.kaggle.com/datasets/garystafford/environmental-sensor-data-132k

2. Briefly describe the overall goal of the data system.

The data system's main job is to handle and organize information from city sensors, covering temperature and humidity for example. It's built to grow easily as more data comes in. Using containers, it's ready to move around wherever it's needed, making it a solid start for working with the data and building user-friendly applications.

3. Choose an appropriate database solution capable of fulfilling all requirements listed above. Describe how your intended data storing solution will be reliable, scalable, and maintainable.

I'm Choosing MongoDB for the data storing solution cause it ensures adaptability to changing sensor data structures, scalability through horizontal distribution, and maintainability with its document-oriented model. Containerization facilitates portability and simplifying transitions between environments. MongoDB's security features and compatibility with cloud platforms contribute to a reliable data processing system.

4. Justify your decision and discuss alternatives and why they might not fit this use case.

MongoDB is great due to its flexible and schema-less design, horizontal scalability, containerization support, robust security features, and compatibility with cloud platforms. Alternatives like relational databases, time-series databases, key-value stores, and traditional SQL databases may face limitations in adapting to the dynamic and growing nature of sensor data or lack the needed flexibility for diverse structures and relationships.

5. Create a plan for implementing your data system prototype.

My plan is to install MongoDB for data storage and Docker for easy movement. Then I'm planning how the sensor data will be organized in the database. I'll preparing the local MongoDB database, creating a space to store sensor data. Finishing my MongoDB database using Docker for simplicity and future scaling.