In-Class Summative Assignment – Part 1

System Architecture Design for a Contact-Tracing Backend

Scenario

Your team has been hired to prototype a **backend component** of a contact-tracing system that tracks which users have been near each other (within 6 feet for over 15 minutes). The data must be:

- Queried efficiently
- Stored securely
- Used ethically

Today you will design the **high-level system architecture** for this application. In our next session, we'll zoom in on one part of the system and design its internal data structures and algorithms.

Part 1: System Diagram

Using a modeling tool of your choice (e.g., draw.io, Lucidchart), create a **container-level system diagram** showing how your backend system is organized.

Include components like:

- **Data Ingest** – receives location data from user devices
- **Storage Layer** – saves location histories securely
- **Contact Checker** – computes whether users have overlapped in time and space
- **API / Interface** – external access (e.g., public health dashboards or apps)

Diagram Requirements:

- Use standard symbols: boxes (components), arrows (data/control flow), clouds (external services), etc.
- Label each component clearly
- Label all arrows with the type of communication or data
- Make the layout as clean and understandable as possible

Collected Work: Submit a photo or exported image of your diagram by the end of class.

Assessment: based on:

- Includes all required components
- Arrows and data flow are labeled
- Standard depictions of components
- Diagram is clear, complete, and readable

In-Class Timeline

- **20 min:** Begin drafting system diagrams individually
- **10 min:** Turn to a partner or small group — explain your diagram and discuss improvements
- **20 min:** Refine your diagram based on discussion, submit
- **15 min: Show-and-Tell**: Volunteers share diagrams with class; short instructor feedback

Be thoughtful, creative, and clear. Focus on architecture, not internal code. This is early-stage design work—your goal is to communicate the system's structure and logic effectively.