

# IoT Sensor MCP

## Control IoT System via LLMs

Tingjun Chen

Nortel Networks Assistant Professor  
ECE Department, Duke University

Yiran Chen

John Cocke Distinguished Professor  
Fellow of AAAS/ACM/IEEE/NAI, Member of EASA  
ECE Department, Duke University

# Overview

- IoT Sensors and LLM-Based Sensor Reading System
- Introduction of the Model Context Protocol (MCP)
- Building MCP Servers for IoT Sensor Reading System
- LLM-Based Sensor Reading System: A Demo

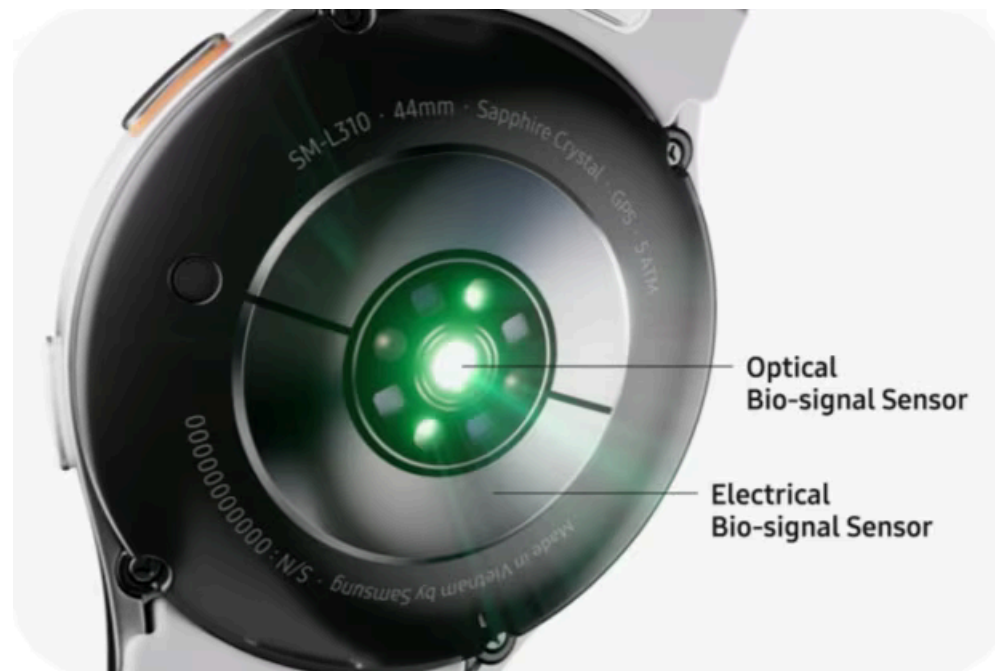
# **IoT Sensors and LLM-Based Sensor Reading System**

## **Section 1**

# What is IoT sensors?

## Definition\*

- A sensor is often defined as a device that receives and responds to a signal or stimulus.



Bio-signal Sensor on smart watch  
(Galaxy Watch 7 as shown in the picture)



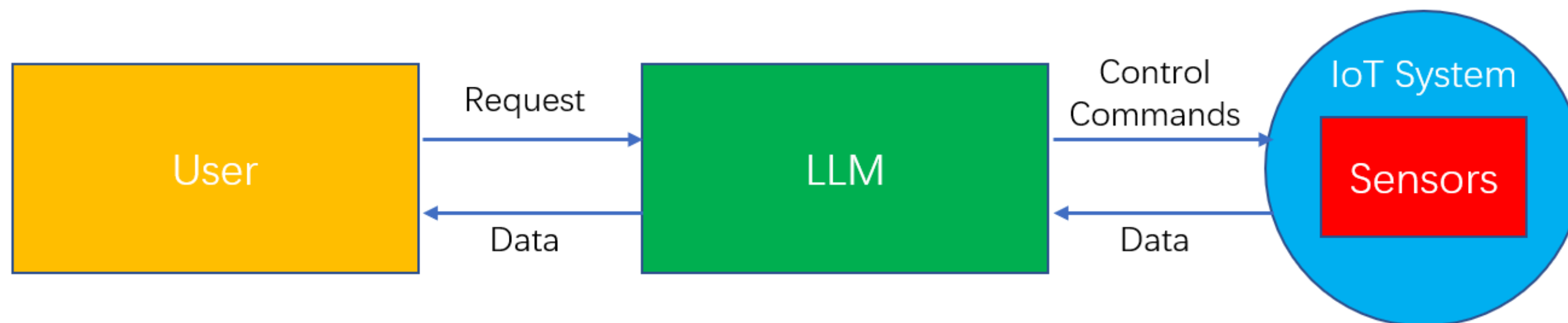
Cameras, LiDAR Scanner, Ambient Light Sensor, etc. on mixed reality headset  
(Apple Vision Pro as shown in the picture)

\*Source: Wikipedia – Sensor (retrieved Aug 3, 2025)

# LLM-Based Sensor Reading System

A LLM-Based Sensor Reading System

- Takes natural language request from user
- Read the data on sensor
- Present the data to the user



# How to enable LLM to interact with IoT System?

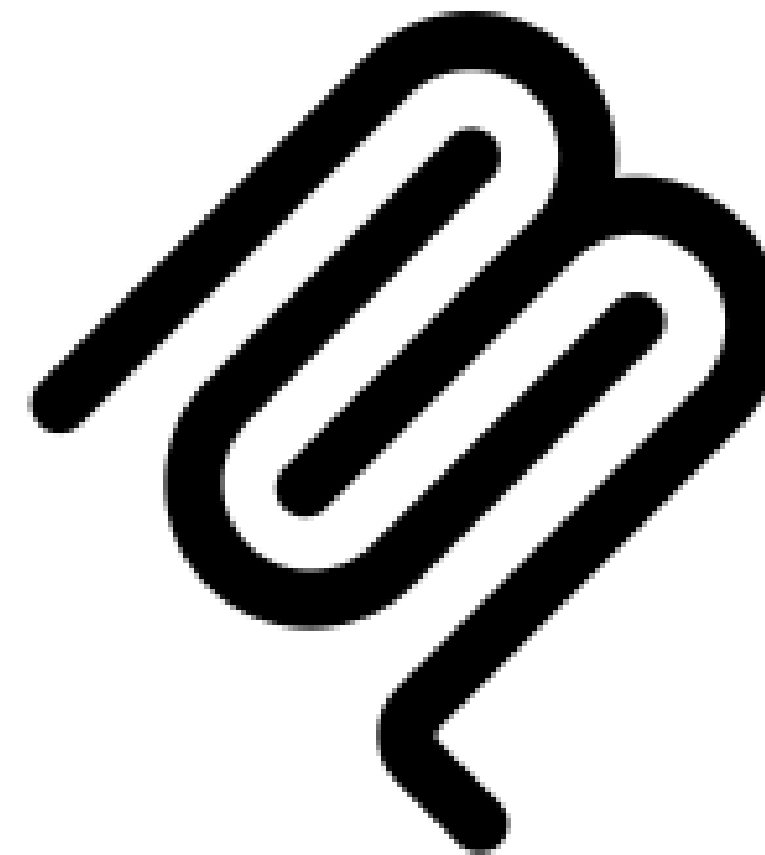
To address this, we introduce the  
**Model Context Protocol**

# Model Context Protocol

## Section 2

# What is MCP?

Model Context Protocol (MCP) is a standard that enables large language models to interact with external tools, resources, and devices in a **structured and coordinated manner**.\*

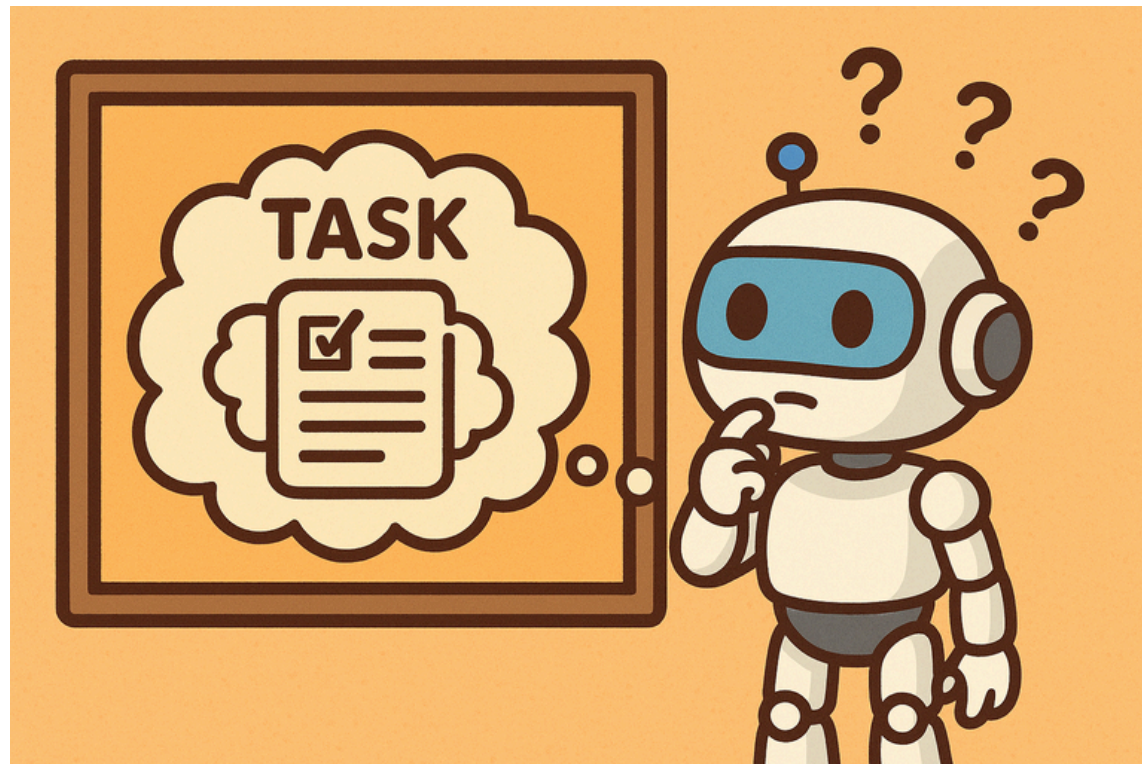


\* Source: Model Context Protocol – Architecture overview (retrieved Aug 5, 2025)

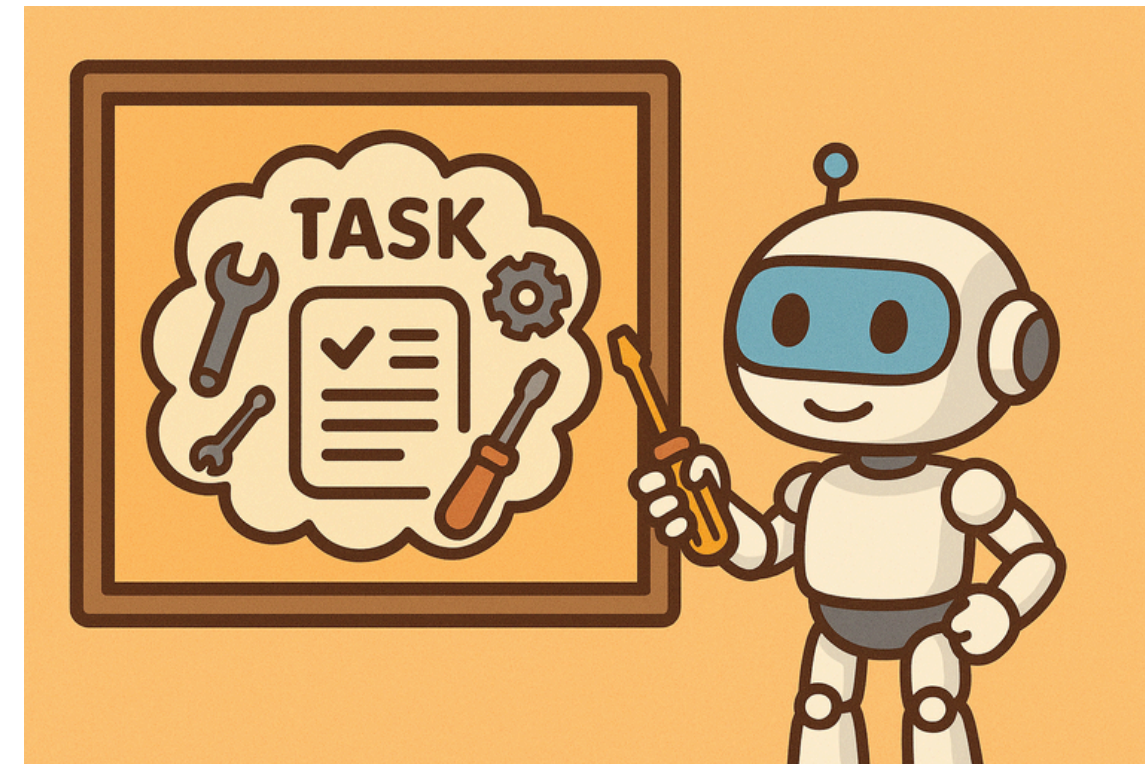


# MCP: A Comparison

- Without MCP
  - LLM complete task based on their learned knowledge
- With MCP
  - LLM complete task via pre-defined modular context protocol.



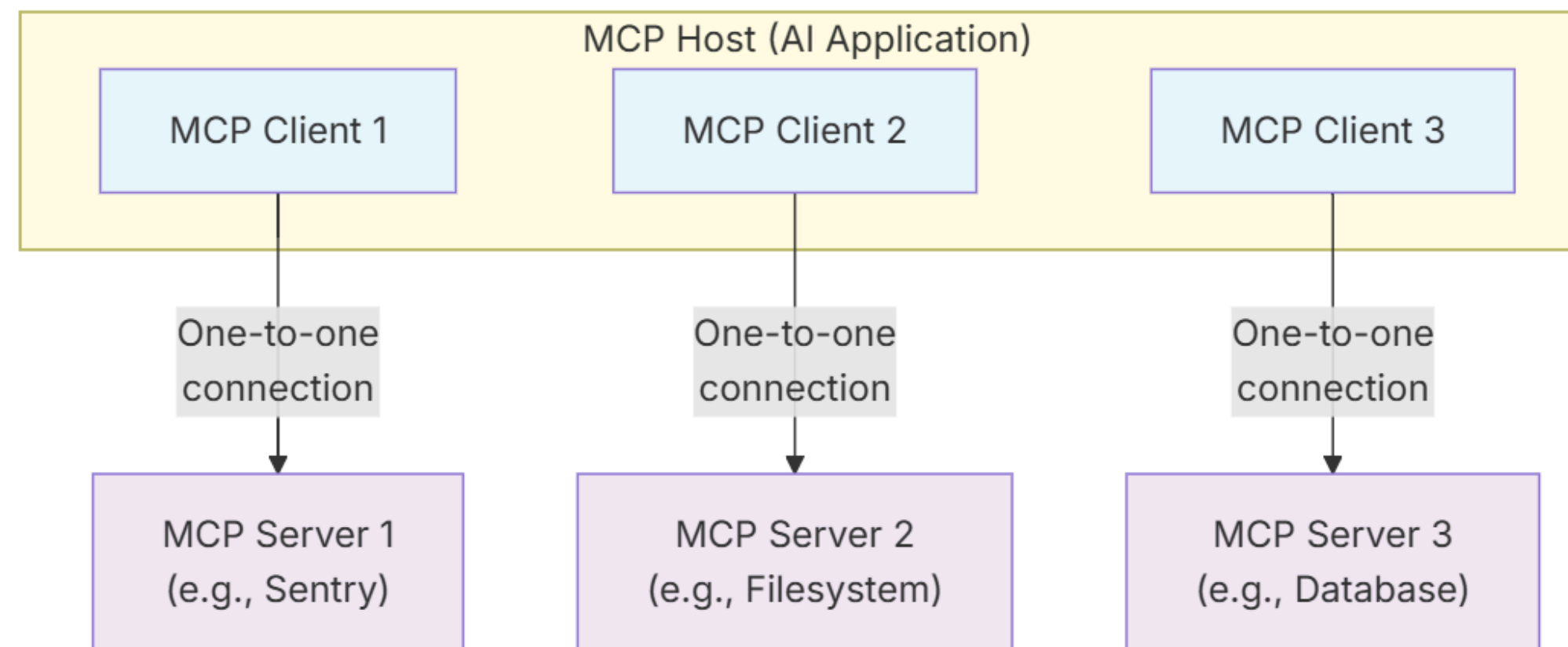
Without MCP



With MCP

# MCP Architecture

- MCP Host: The AI application (ChatGPT, Claude, etc.)
- MCP Client: The MCP Client connects to the MCP Server to get task information for the AI to use. Simply, it is like chat session between LLM and the MCP server.
- MCP Server: A program that provides context to MCP clients

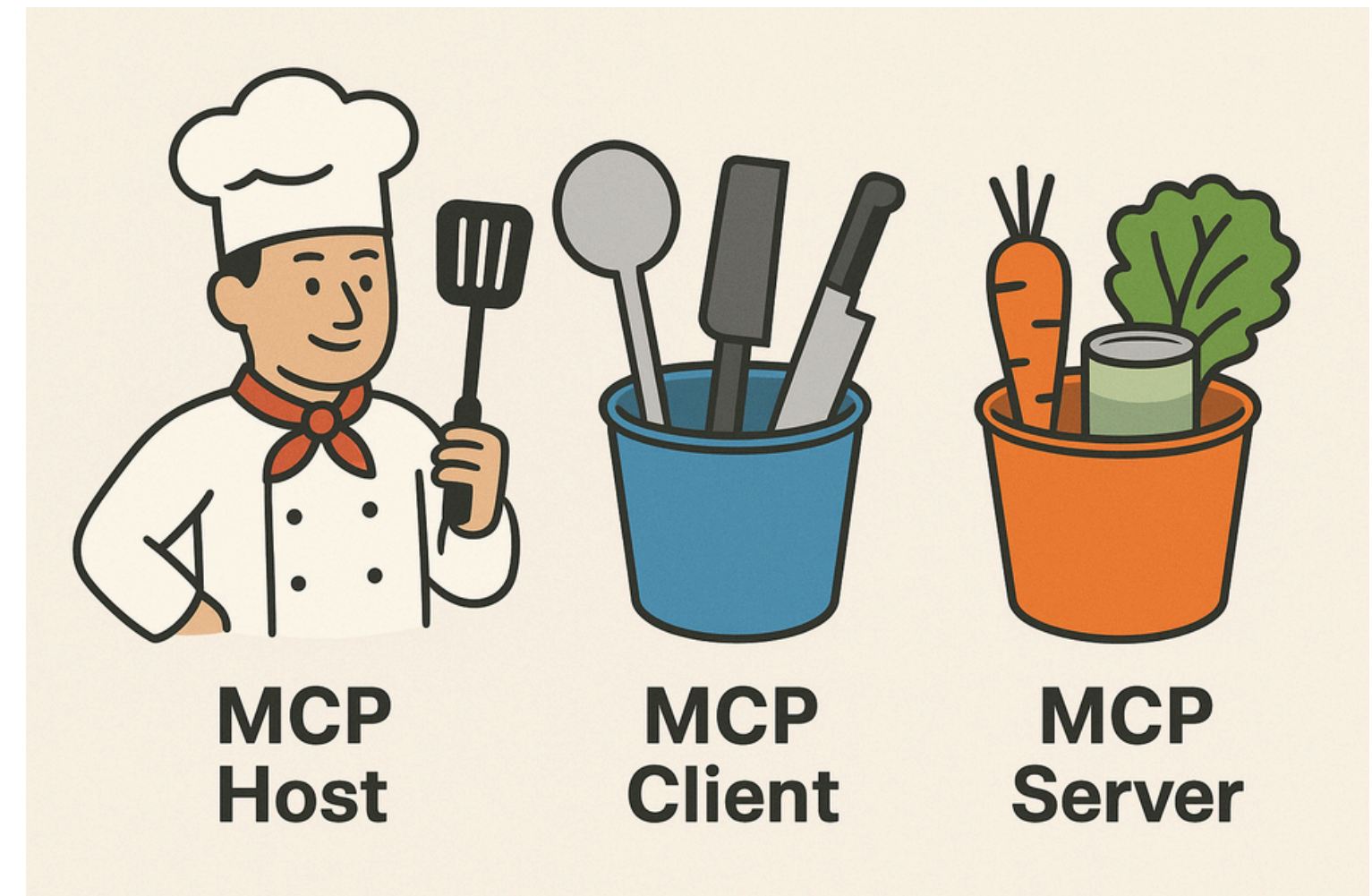


\* Source: Model Context Protocol – Architecture overview (retrieved Aug 5, 2025)



# MCP Architecture

- Think of the MCP Host (AI application) as a **chef**.
- The MCP Client is like a **set of cookers** the chef uses to interact with the kitchen.
- The MCP Server is the **pantry** — it gives the chef the ingredients (context, goals, tools) needed to cook up a response.



# Building MCP Server for IoT Sensor Reading System

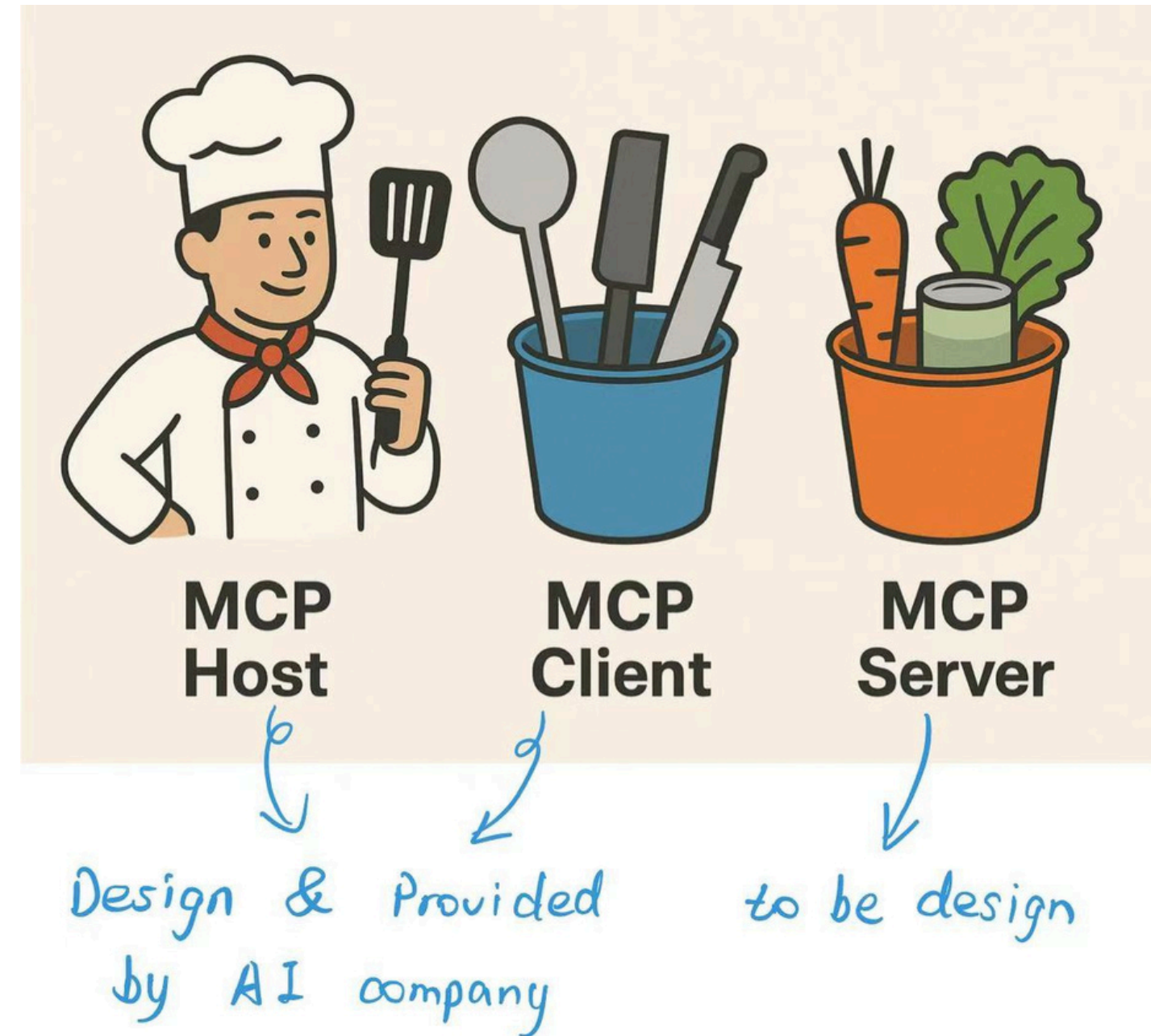
## Section 3

# Designed & To be Design

Recall the three key components of MCP:

- MCP Host
- MCP Client
- MCP Server

With the MCP Host and Client provided by the AI company, the key task is to **develop valid MCP servers.**





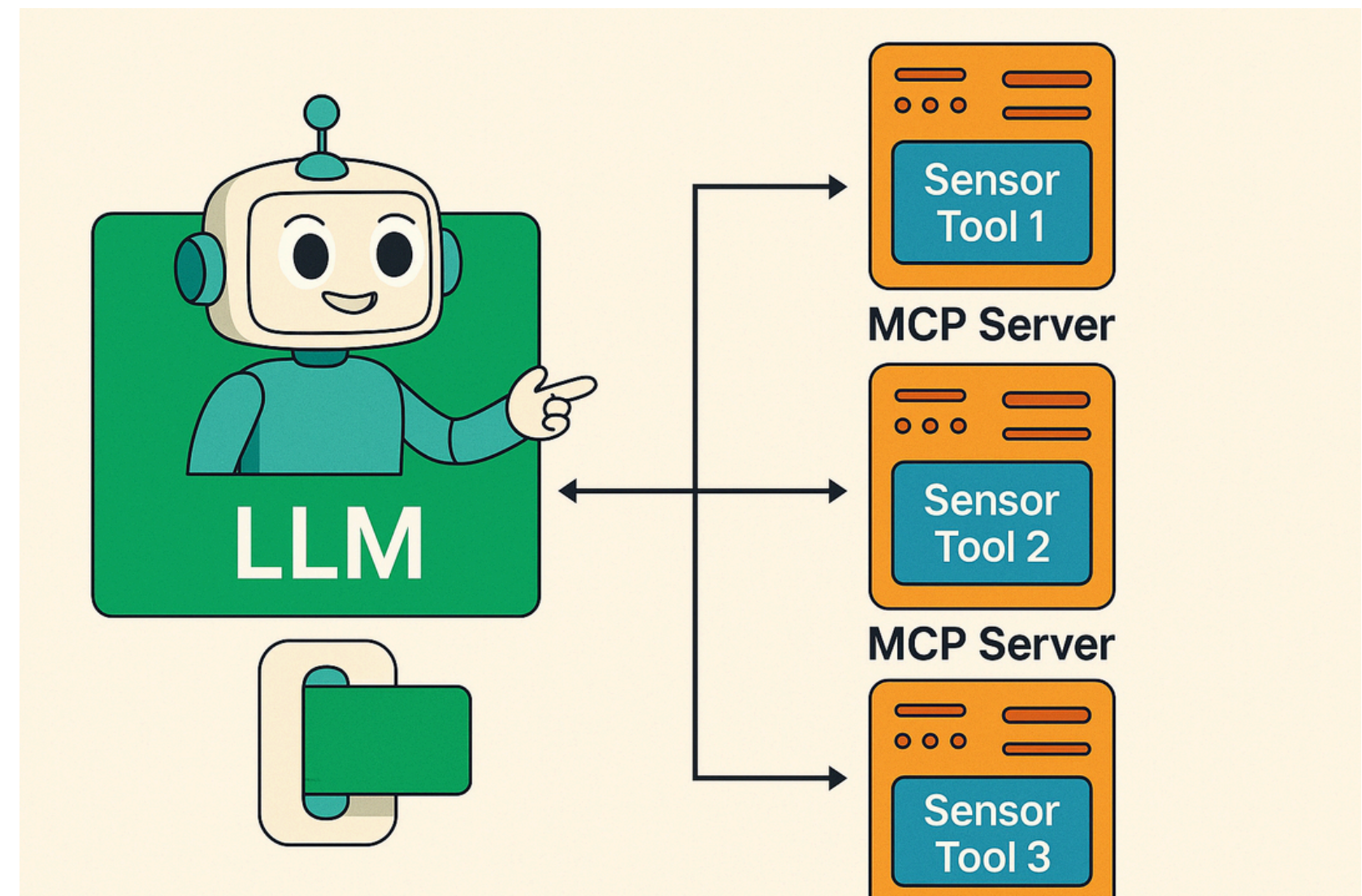
# Design an MCP Server

## MCP Server Design

- Each MCP Server corresponds to a specific type of sensor
- Inside each server, encapsulate each sensor function into a modular, callable tool.

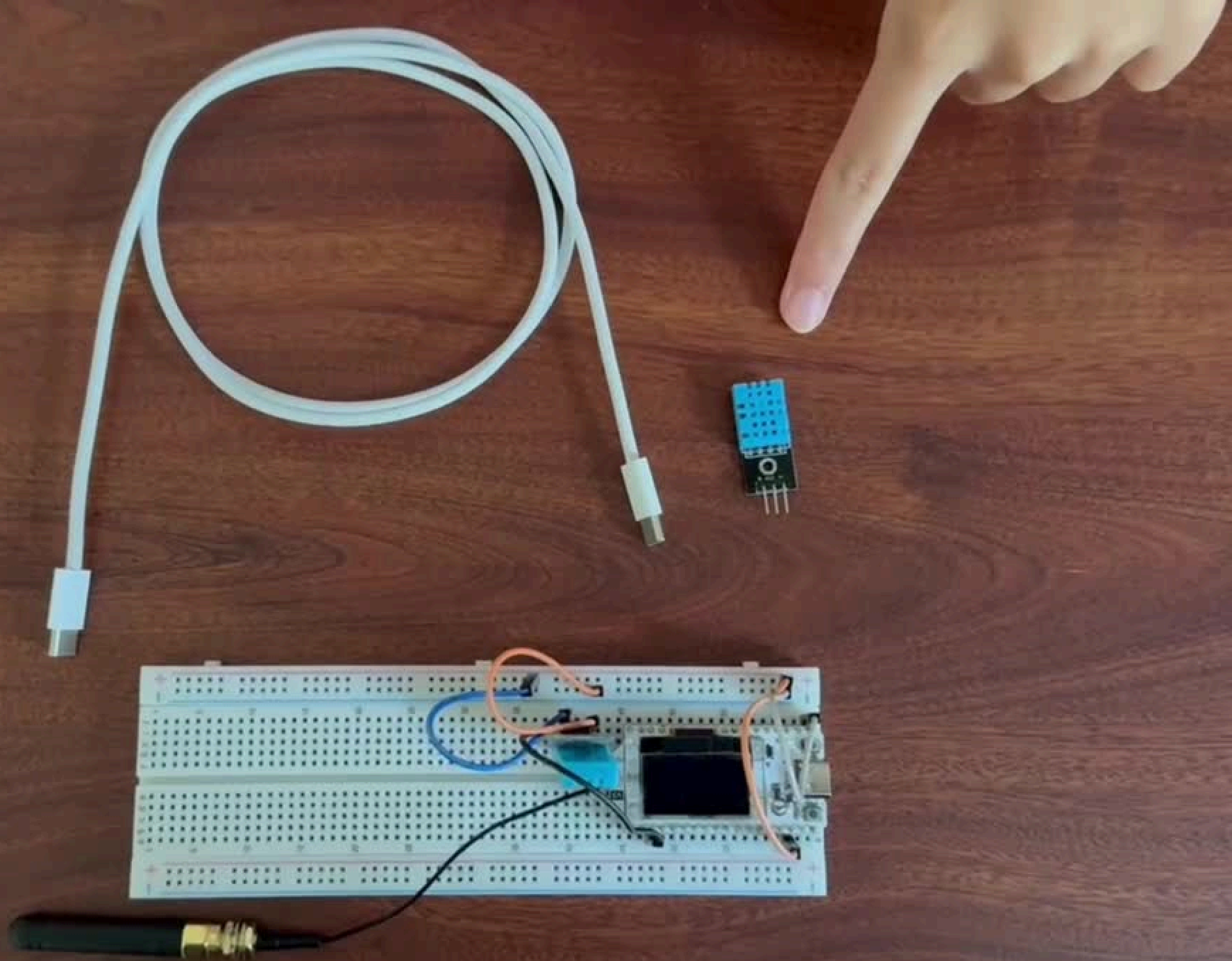
## Code Implementation

See code implementation instruction in lecture note.



# LLM-Based IoT Sensor Reading System: a Demo

## Section 4





# Conclusion

- LLMs can control IoT systems via the Model Context Protocol (MCP)
- MCP enables structured, reliable interaction with external systems
- MCP server design is key for IoT sensor integration
- Demonstrated a working LLM-based IoT sensor reading system

# Reference

Wikipedia contributors. (n.d.). Sensor. In Wikipedia, The Free Encyclopedia. Retrieved August 3, 2025, from <https://en.wikipedia.org/wiki/Sensor>

Model Context Protocol. (n.d.). \*Architecture overview\*. Retrieved August 5, 2025, from <https://modelcontextprotocol.io/docs/learn/architecture>

Figures generated with assistance from ChatGPT (OpenAI, 2025).

The slide features a white background with a yellow vertical bar on the left and a dark gray vertical bar on the right. Both bars contain white geometric shapes, including triangles and diamonds, some with black outlines. The text "Thank You" is centered in a large, bold, black sans-serif font.

# Thank You

**For your attention**

