Project Proposal: Decentralized Agricultural Sentinel (DAS)

1. Executive Summary: The Autonomous Economic Agent

The **Decentralized Agricultural Sentinel (DAS)** is a groundbreaking project that demonstrates the full lifecycle of an autonomous economic agent within the physical world. It is a proof-of-concept for the **Agentic Economy** where intelligence, physical action, and decentralized finance converge.

The DAS system features a **Nullshot AI Agent** that monitors a micro-farm environment via a custom **Model Context Protocol (MCP) Utility**. The Agent makes a complex, conditional decision based on both **simulated sensor data** and **real-time visual analysis** (via ESP-CAM). Crucially, the Agent executes an **on-chain micro-transaction** to "pay" for resources before triggering a **physical action** (irrigation via pump/relay). This architecture is the "1%" innovation, showcasing a fully autonomous, economically self-sufficient entity.

2. Hackathon Alignment and Track Selection

This project is a perfect fit for **Track 1: MCPs/Agents using the Nullshot Framework**, as it directly implements the core technologies of the hackathon.

Requirement	DAS Project Implementation	Innovation Focus
Track 1 Focus	The core decision-making logic is implemented in an **AI Agent** built using the **Nullshot Framework**.	Focus on the Agent's complex, multi-data-source decision logic.
MCP Utility	A custom **MCP Utility** is developed to expose both simulated data and physical hardware control functions.	The MCP acts as the secure, standardized API for the physical world.
Web3 Ecosystem	Integration with a Web3 platform (e.g., Thirdweb) is used for **autonomous micro-transactions** and **decentralized event logging**.	The Agent is an **Autonomous Economic Entity** that pays for its own actions.

Innovation	Demonstrates the full **Sense (Simulated/Visual) -> Analyze (AI) -> Transact (Web3) -> Act (Physical)** loop.	Proves the Agentic Economy extends beyond software to control real-world assets.

3. Technical Architecture and Component Breakdown

The DAS system is a three-layer architecture, specifically designed around the user's available components.

3.1. Embedded Layer (The Sentinel)

This layer uses the available hardware to create a functional, low-cost micro-farm control unit.

Component	Role in DAS	Justification for Use
Gateway/Bridge	**LilyGO TTGO T-Call ESP32 SIM800L**	Acts as the primary communication hub, running the client-side code to interface with the MCP Utility and providing robust connectivity (Wi-Fi/GSM).
Actuator	**Relay + Pump**	Provides the essential **physical action** component, simulating an irrigation system.
Visual Input	**ESP-CAM**	Provides the **real-time visual data** stream, which is critical for the Agent's conditional decision-making.
Control Hub	**Arduino Nano RP2040**	Used for precise timing and control of the pump/relay, communicating with the ESP32.
Simulated Data	**ESP32 Firmware**	Generates pre-defined or randomized values for `soil_moisture` and `temperature` to feed the Agent.

3.2. Bridge Layer (The Custom MCP Utility)

This layer is the core of the hackathon submission, standardizing the hardware's capabilities for the AI Agent.

MCP Function Name	Description	Data Source/Action
`get_simulated_data()`	Retrieves the latest simulated `soil_moisture` and `temperature` readings.	ESP32 generates and sends data to the MCP Utility server.
`capture_image()`	Triggers the ESP-CAM to take a photo and returns a secure URL for the Agent to access.	ESP-CAM captures image, ESP32 uploads it to a temporary cloud service.
`trigger_irrigation(duration_s)`	Activates the pump/relay for a specified duration.	MCP Utility sends command to ESP32, which relays to the Nano RP2040 to activate the relay.

3.3. Agentic Layer (The Nullshot AI Agent)

This is the intelligence layer, built using the Nullshot Framework, demonstrating complex autonomy.

Agent Name: `DAS_Economic_Manager`

Core Decision-Making Flow:

1. **Sense:** Agent calls `get_simulated_data()` via the MCP.
2. **Analyze (Condition 1):** If `soil_moisture` is below the critical threshold (e.g., 15%):
   - Agent calls `capture_image()` via the MCP.
   - Agent performs a simple **Visual Check** on the image (e.g., checks for image clarity/color to simulate pest/disease analysis).
3. **Transact (Condition 2):** If the visual check is clear (no disease) and the moisture is low, the Agent initiates a **micro-transaction** (e.g., 0.001 ETH) to a simulated water utility smart contract (Thirdweb).
4. **Act:** Upon successful transaction confirmation, the Agent calls `trigger_irrigation(5)` via the MCP.
5. **Log:** Agent logs the entire decision-making process, the transaction hash, and the action taken to the blockchain.

4. Web3 Integration: The Autonomous Economic Entity

The project's highest innovation is the integration of the physical action with a decentralized economic model.

- **Autonomous Payment:** The Nullshot Agent is pre-funded and programmed to execute a transaction *before* calling the physical action function. This is the definitive proof of an **Autonomous Economic Entity** that manages its own resources.
- **Decentralized Logging:** The Agent uses the blockchain to log its actions, creating an immutable, auditable record of its decisions and physical interventions.

5. Demo Video Plan (3-5 Minutes)

The video will be structured to highlight the full loop and the "wow" factor.

Time	Segment	Focus
0:00 - 0:30	**The Vision**	Introduce the problem and the solution: **Intelligent, Autonomous, Economically Self-Sufficient Agents.**
0:30 - 1:30	**The Bridge & Hardware**	Showcase the physical setup (ESP32, Pump/Relay). Explain the custom MCP functions and how they expose the hardware.
1:30 - 3:00	**The Agentic Loop in Action**	**Live Demo:** Show the Agent's log. Show the simulated data trigger the conditional check. Show the Agent calling the MCP functions.
3:00 - 4:00	**The Web3 "Wow"**	Show the **on-chain transaction** being executed by the Agent. Show the **pump physically activating** immediately after the transaction confirms.
4:00 - 5:00	**Conclusion**	Summarize the full-stack integration and the potential for scaling this model to real-world, high-impact applications.

This proposal is now complete and ready to guide your development.