HTTP-JSON-Assistant

Project Overview

HTTP-JSON-Assistant is an embedded system built on an STM32 microcontroller that receives commands over HTTP, extracts JSON payloads, and executes corresponding actions on the hardware.

The project is designed as a **single evolving system**. Its end goal is to function as a personal assistant capable of triggering time-based notifications such as reminders and alarms. Development is intentionally phased so that the communication and execution foundation is validated before any higher-level assistant behavior is added.

Phase-1 Scope

Phase-1 focuses exclusively on establishing a **reliable command execution pipeline** on the STM32.

The objective of this phase is to confirm that the system can:

- Act as a network-accessible endpoint
- Receive HTTP requests
- De-encapsulate data from Ethernet down to JSON
- Execute a deterministic hardware action

An LED blink is used as the execution target in this phase. The LED is not a feature; it is a verification mechanism used to confirm that the end-to-end path from network input to hardware output is functioning correctly.

System Operation (Phase-1)

1. A client on the local network sends an HTTP POST request to the STM32.
2. The request body contains a JSON command:
3. {"cmd":"blink"}
4. The STM32 is connected to the network through an ENC28J60 Ethernet controller.
5. The ENC28J60 receives raw Ethernet frames and forwards them to the STM32 over SPI.

6. A lightweight TCP/IP stack (lwIP) running in bare-metal mode processes Ethernet, IP, and TCP layers.
7. The application layer parses the HTTP request and extracts the JSON payload.
8. The JSON parser decodes the command.
9. The STM32 executes the requested action by blinking an LED.
10. A minimal HTTP acknowledgment is sent to close the connection cleanly.

The system follows a **fire-and-forget command model**. No application-level response data is returned.

Design Intent (Phase-1)

Phase-1 exists to validate infrastructure, not features.

The intent of this phase is to:

- Prove that the STM32 can reliably receive commands over a network
- Validate correct de-encapsulation from Ethernet down to JSON
- Maintain strict separation between drivers, protocol stack, and application logic
- Keep execution deterministic and easy to debug
- Avoid introducing scheduling, RTC logic, or assistant behavior prematurely

The same infrastructure will be reused directly in later phases to implement time-based assistant functionality.

Technical Choices

- **Ethernet (ENC28J60)**
  Used to maintain explicit control over network communication.
- **lwIP (NO_SYS mode)**
  Provides correct TCP/IP handling in a bare-metal environment without an RTOS, ensuring predictable execution.
- **Static IP Configuration**
  Simplifies bring-up and debugging by removing reliance on external services.

- **HTTP and JSON**
  Used strictly as a transport and command encoding mechanism, not as a feature framework.

What Phase-1 Does Not Include

Phase-1 explicitly excludes:

- Scheduling or alarm logic
- RTC configuration
- Persistent storage
- Calendar integration
- User interfaces
- Cloud connectivity

These features depend on the correctness of the underlying command pipeline and will be added only after Phase-1 is complete.

Summary

HTTP-JSON-Assistant is being developed as a staged embedded system. Phase-1 establishes a solid foundation by proving that network-delivered commands can be reliably received, decoded, and executed on the STM32. The LED blink implemented in this phase serves only as validation of the infrastructure that will later support full assistant behavior.