Diseño de Arquitectura del Sistema

**Fecha:** 05/09/2025

**Autor:Erick Aguila Barrientos, Pool Apaza Aguirre, Victor Condori Mamani, Crystian Garcia Huallpa**

**Versión:** 1.0

# 1. Introducción

Este texto explica de manera minuciosa la estructura de la arquitectura destinada al Invernadero Automatizado en Chihuaco, un prototipo regional que se encarga de la automatización del riego y la regulación climática de acuerdo a los requerimientos funcionales y los requerimientos no funcionales. El sistema se operará de manera local sin necesidad de conexión a la nube para las funciones esenciales; almacenará datos históricos de forma local en una tarjeta microSD y proporcionará una interfaz local (con pantalla y botones) para control y mantenimiento. La elección de los componentes se basa en su disponibilidad y en su bajo costo de mantenimiento.

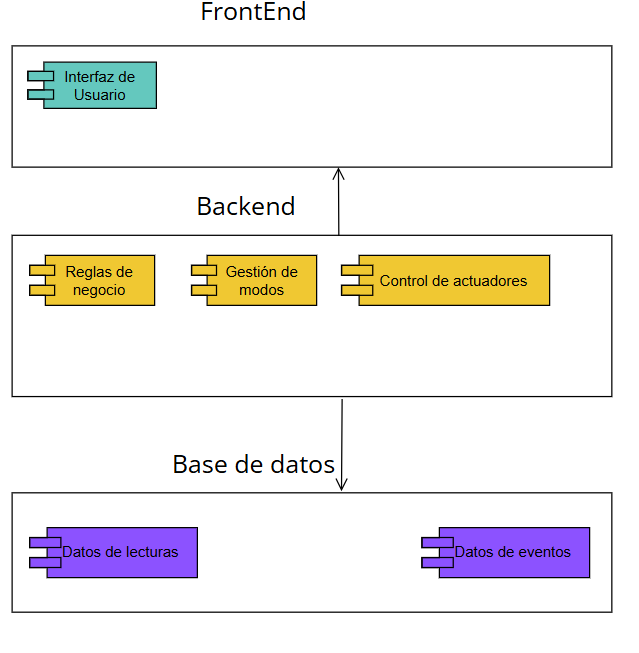
# 2. Visión General de la Arquitectura

## 2.1. Estilo Arquitectónico

El sistema adopta una **arquitectura en capas** basada en el **patrón cliente-servidor local (edge-centric)**. Lógicamente se organiza en tres capas principales:

* **Capa de Presentación (Frontend)**: Interfaz con la que interactúa el usuario — pantalla local (LCD), LEDs y botones.
* **Capa de Lógica de Negocio (Backend)**: Firmware que ejecuta la lógica de control (muestreo de sensores, filtrado, reglas de riego y clima, gestión de alarmas,, prioridad manual/automático). Este componente debe residir en el controlador embebido (ESP32) para asegurar operación autónoma offline.
* **Capa de Persistencia de Datos (Almacenamiento local)**: Archivo/DB local que guarda histórico y logs (microSD en ESP32 para la versión inicial).

## 2.2. Diagrama General de Arquitectura:



# 3. Descripción de Componentes Principales

## 3.1. Frontend (Presentación)

* **Pantalla LCD 16×2 con interfaz I2C**: muestra en tiempo real humedad, temperatura, estado de actuadores (bomba, ventilador, calefactor) y mensajes de alerta.
* **Botones físicos**: permiten navegar opciones básicas (consultar histórico, cambiar umbrales, activar riego manual).
* **LEDs indicadores**: verde = sistema normal, rojo = fallo, azul = riego activo.
* **Zumbador (buzzer)**: alerta sonora cuando parámetros salen de rango o hay fallo en sensores.

Responsabilidades:

* Renderizar información en la pantalla de forma clara y sencilla para el agricultor.
* Capturar entradas del usuario a través de botones físicos (activar bomba manual, confirmar cambio de umbral).
* Mostrar mensajes de estado, errores y validaciones en la interfaz local.
* Proveer **alertas inmediatas** (visuales y sonoras) en caso de valores críticos o fallos de sistema.

## 3.2. Backend (Lógica de negocio)

* Lenguaje y Entorno: C/C++ con framework Arduino Core para ESP32.
* Librerías:Wire.h para I2C (pantalla LCD, RTC DS3231), OneWire.h y DallasTemperature.h para sensores DS18B20, SD.h para módulo microSD, RTClib.h para el RTC, EEPROM.h o Preferences.h para almacenar configuraciones (umbrales definidos por el usuario).

Responsabilidades:

Adquisición y validación de datos: leer periódicamente los sensores de humedad y temperatura; verificar valores fuera de rango o fallos de sensor.

**Aplicación de reglas de negocio (control agrícola):**

* Activar bomba de agua si humedad < umbral definido.
* Apagar bomba al llegar a nivel óptimo.
* Activar ventiladores si la temperatura supera el umbral máximo.
* Activar la calefacción si la temperatura baja del umbral mínimo.
* Prevenir conflictos ( calefacción y ventilación nunca deben estar activos a la vez).

**Gestión de modos de operación:**

* **Automático:** control basado en sensores y umbrales.
* **Manual de emergencia:** el botón físico tiene prioridad sobre el control automático.
* **Fail-safe:** si falla un sensor o la SD, detener actuadores y notificar con LED o buzzer.

**Persistencia de datos:** registrar lecturas y eventos en microSD en formato CSV con marca de tiempo (fecha/hora del RTC).

**Gestión de energía y fiabilidad:**

* Monitorear estado de batería de respaldo.

### 3.2.1. Módulos del Backend

* sensorModule.cpp / sensorModule.h: Encargado de leer los datos de los sensores (humedad capacitiva, DS18B20 de temperatura).
* controlModule.cpp / controlModule.h:
  + Activación automática de bomba, ventiladores y calefactores según umbrales.
  + Prevenir conflictos (ventilación y calefacción simultánea).
  + Ejecutar prioridades (manual de emergencia > automático).
* persistenceModule.cpp / persistenceModule.h:
  + Maneja el registro en microSD.
  + Guarda archivos CSV diarios con lecturas y eventos (timestamp, valor, unidad).
  + Controla la rotación de archivos y verificación de integridad.
* configModule.cpp / configModule.h:
  + Administra parámetros configurables (umbrales de humedad y temperatura).
  + Guarda y lee valores en EEPROM o Preferences (persisten tras reinicio).
* uiModule.cpp / uiModule.h:
  + Gestiona la interacción con el usuario en LCD, LEDs y buzzer.
  + Muestra lecturas actuales, estado de actuadores, alarmas y mensajes.
  + Captura entradas de los botones para menú o activación manual.
* powerModule.cpp / powerModule.h:
  + Supervisa batería de respaldo y estado de la fuente de 12V.
  + Activa estrategias de ahorro si cae tensión.
  + Integra el watchdog para reinicios seguros.
* main.cpp:
  + Punto de entrada.
  + Inicializa todos los módulos.
  + Ejecuta el loop principal de lectura → lógica → control → registro → presentación.

## 3.3. Base de Datos (Persistencia)

* Sistema Gestor: PostgreSQL
* ORM: Sequelize o Prisma
* Esquema de Datos:
  + Tabla ***users***: id, nombre, correo, contraseña (encriptada)
  + Tabla ***tasks***: id, título, descripción, fecha\_vencimiento, estado, user\_id, list\_id
  + Tabla ***lists***: id, nombre, user\_id

# 4. Integraciones Externas (Opcionales)

A pesar de que el sistema ha sido creado para operar completamente de forma local y sin conexión (una condición vital para garantizar su eficacia en Chihuaco), se contemplan integraciones externas en ediciones venideras que mejorarían la supervisión, evaluación y gestión a distancia del invernadero.

### Servicios de Notificación

* **SMS / WhatsApp API (WhatsApp Business API):** permitirían enviar alertas inmediatas al agricultor cuando la humedad o la temperatura sobrepasen valores críticos, incluso si no está en el invernadero.
* **Correo electrónico (Gmail API):** envío de reportes diarios con las estadísticas de humedad, temperatura y riegos ejecutados.
* **Push Notifications (Firebase Cloud Messaging):** si se implementa una aplicación móvil, se podrían enviar notificaciones en tiempo real sobre el estado del invernadero.

### Integración con Plataformas IoT

* **Blynk o ThingsBoard:** plataformas de IoT que ofrecen dashboards listos para visualizar lecturas históricas, gráficas y estados de actuadores en tiempo real.
* **MQTT Broker en la nube (HiveMQ, Mosquitto en servidor VPS):** permitiría que el ESP32 publique datos de sensores y eventos en un broker remoto, que el agricultor consulte desde cualquier lugar con acceso a Internet.
* **Firebase:** envío automático de lecturas hacia una hoja de cálculo en la nube para análisis y respaldos.

# 5. Seguridad

El sistema de invernadero automatizado en Chihuaco pone como prioridad la seguridad en su funcionamiento y la consistencia en su rendimiento, siguiendo los criterios no funcionales establecidos . A continuación, se describen los mecanismos a implementar:

### Seguridad Operativa (Offline – MVP)

* **Watchdog hardware y software:** el ESP32 reinicia automáticamente en caso de bloqueo o error crítico.
* **Fail-safe en actuadores:** si falla un sensor o se pierde comunicación con la SD, la bomba y calefactores se apagan automáticamente y se genera una alerta visual/sonora.

### Seguridad de Datos

* **Integridad en registros:** cada escritura en la SD se valida con “flush” inmediato y uso de archivos temporales antes de renombrar (evita corrupción de datos por apagado inesperado).
* **Control de acceso físico:** la microSD se retira únicamente por el agricultor o personal autorizado; se recomienda almacenar copias periódicas en otro dispositivo.
* **Formato de datos estandarizado (CSV con timestamps RTC):** facilita trazabilidad y reduce errores en análisis posterior.

### Seguridad Física y de Mantenimiento

* **Caja de control cerrada :** protege la electrónica contra humedad, polvo e ingreso no autorizado.
* **Etiquetado de cables y módulos:** reduce riesgos de errores en mantenimiento.
* **Manual de operación en español:** instruye sobre manejo seguro del sistema y protocolos de emergencia.

# 6. Escalabilidad y Despliegue

El sistema se instalará físicamente en el **invernadero de Chihuaco**, con los siguientes pasos:

### Despliegue Inicial en Campo

El sistema se instalará físicamente en el **invernadero de Chihuaco**, con los siguientes pasos:

1. **Montaje de sensores de humedad** en distintos puntos del suelo, calibrados según cultivo.
2. **Instalación de sensores de temperatura (DS18B20)** en la parte central y lateral del invernadero.
3. **Caja de control IP65** con ESP32, módulo SD, relés y fuente de alimentación protegida.
4. **Conexión a actuadores:** bomba de agua de 12V, ventiladores y calefactores.
5. **Pruebas de integración** de riego automático y control climático antes de la operación real.

### Escalabilidad Funcional (Software)

El sistema está diseñado para crecer en funcionalidades mediante actualizaciones de firmware y módulos adicionales:

* Soporte para más sensores: el ESP32 permite hasta 8–10 sensores de humedad.
* Gestión avanzada de datos: la microSD puede almacenar años de registros; en una versión extendida, un Raspberry Pi puede centralizar la información en SQLite o PostgreSQL local.
* Dashboards remotos opcionales: integración futura con plataformas IoT para visualización en tiempo real.

### Posibles Problemas durante el Desarrollo y Despliegue

* Ruido eléctrico y humedad: puede afectar lecturas de sensores → mitigación: sensores capacitivos sellados, cables blindados, caja.
* Cortes de energía: riesgo de detener el sistema → mitigación: batería de respaldo con autonomía de 2 horas.
* Corruptela de datos en SD: posible pérdida de registros en apagados bruscos → mitigación: uso de archivos temporales y validación de escritura.
* Mantenimiento del hardware: reemplazo frecuente de sensores si no se calibra adecuadamente → mitigación: plan de mantenimiento trimestral y guía de usuario.

# 7. Conclusiones

La estructura propuesta para el invernadero automatizado en Chihuaco establece un balance entre la facilidad de implementación, la solidez en la operación continua y la capacidad de ampliación para mejoras futuras. Con una base de tres niveles bien diferenciados interfaz, sistema de control y almacenamiento de datos, se facilita la integración eficiente de sensores y actuadores, permite el procesamiento de la información en tiempo real a través del ESP32 y asegura el almacenamiento histórico mediante soluciones locales. El diseño incluye sistemas de energía de respaldo y autodiagnóstico que garantizan la fiabilidad en áreas rurales, además de permitir el crecimiento hacia más módulos de control y conectividad a IoT. Así, la arquitectura no solo se encarga de la automatización del riego y el control ambiental del invernadero, sino que también sienta las bases para una agricultura sostenible y su avance tecnológico en la comunidad de Chihuaco.