BCSE497J Project-I

Embedded System

22BCE0881	**THAMMINENI VENKATA HARSHA VARDHAN**
22BDS0168	**KAMATHAM GANESH MAHARAJ**

Under the Supervision of

Shobha Rekh

Professor Grade 1

School of Computer Science and Engineering (SCOPE)

B.Tech.

in

Computer Science and Engineering

School of Computer Science and Engineering

September 2025

ABSTRACT

Cold storage facilities are indispensable for preserving fruits, vegetables, and other perishable commodities, yet conventional systems typically employ bulk cooling strategies that treat the warehouse as a single environment. This results in uneven preservation conditions and significant energy wastage, as cooling resources are directed to regions that may not require the same level of conditioning. Moreover, gases such as CO_2 and ethylene, naturally released during the ripening process, accumulate heterogeneously and accelerate spoilage when not properly regulated. Addressing these inefficiencies is vital for reducing food loss, ensuring consistent quality, and promoting sustainable energy usage in agricultural supply chains. This project introduces an **Adaptive Cooling System for Cold Storages** that restructures environmental control from the warehouse level down to individual trays. Instead of leaving trays open to ambient air, each is sealed and equipped with a compact sensor suite that measures **temperature, humidity, CO_2 concentration, ethylene levels, and weight**. These parameters provide real-time insights into both environmental status and physiological changes of the produce stored. Vertical racks housing multiple trays are connected to dual airflow channels: one for conditioned air and another for humidified air. Each rack contains valves that can selectively open depending on the detected needs of its trays. A DC fan in every tray then modulates airflow intensity, ensuring precise distribution of cooling or humidity only where required.The airflow system is further optimized by routing exhaust gases through a chemical filtration unit, which removes ripening agents before recirculating air back into the conditioning system. This closed-loop design minimizes contamination between trays and maintains an overall balanced atmosphere in the warehouse.Control intelligence is achieved using a microcontroller that continuously reads sensor data and transmits it to a backend system. An integrated AI agent, running on an offline large language model, interprets the sensor snapshot and generates actuator commands in real time. These commands include fan speeds and valve states, which are applied to the hardware and simultaneously logged into a MongoDB database. An administrative dashboard renders live tray values and actuator states, while a separate Plotly web application provides interactive time-series visualizations for monitoring long-term trends.By shifting from monolithic cooling to tray-level adaptive control, this system reduces unnecessary energy consumption, prolongs the shelf life of fresh produce, and establishes a scalable framework for smart agricultural storage.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Background

Cold storage is a critical component of modern agricultural supply chains, enabling the long-term preservation of fruits, vegetables, and other perishable commodities. India alone loses a substantial percentage of its horticultural produce every year due to inadequate or inefficient storage facilities. Conventional cold storage units are generally designed as large warehouse-sized chambers that are cooled uniformly using centralized refrigeration systems. This bulk cooling strategy ensures that the ambient temperature is brought down to the desired level, but it does not account for local variations in environmental parameters such as humidity, carbon dioxide concentration, or ethylene gas accumulation.

In practice, the biological processes of stored produce vary across racks, trays, and even within clusters of fruits, leading to heterogeneous micro-environments. For example, ethylene released from a small group of ripening bananas may trigger premature ripening of neighboring fruits stored in open trays. Similarly, uneven humidity distribution can cause certain batches to dry out while others remain fresh. These inconsistencies result in product spoilage, uneven quality, and reduced market value.

Furthermore, traditional systems consume significant amounts of electrical energy, since they continuously cool the entire warehouse volume irrespective of actual localized requirements. This contributes to higher operational costs and increased carbon footprint, which is especially concerning in the context of global efforts toward sustainable development and energy conservation.

The integration of emerging technologies such as the Internet of Things (IoT), real-time sensing, and AI-based decision-making has opened new possibilities for re-engineering cold storage. By deploying distributed sensors and adaptive actuators, it is possible to monitor environmental conditions at fine granularity and dynamically adjust airflow, cooling, and humidification in response to actual produce needs. Such systems promise not only energy efficiency but also improved preservation of nutritional quality and shelf life.

The present project builds on these trends by proposing a tray-level adaptive cooling architecture. Each tray becomes a closed micro-environment equipped with sensors for temperature, humidity, CO_2, ethylene, and weight, with controlled airflow managed by rack-level valves and tray-level fans. The system integrates chemical filtering of exhaust gases and

AI-driven control logic to optimize conditions continuously. This background highlights the transition from traditional monolithic cold storage toward intelligent, sustainable, and fine-grained adaptive storage solutions.

1.2 Motivation

The motivation for developing an adaptive cooling system for cold storages arises from both practical challenges and broader sustainability goals. On the practical side, farmers, distributors, and retailers face heavy losses due to the premature spoilage of fruits and vegetables in conventional storage facilities. Studies estimate that 20–30% of fresh produce in India is wasted annually, largely due to inadequate storage and handling. Much of this loss stems from uneven cooling and uncontrolled accumulation of ripening gases in bulk cold storage warehouses.

From an energy perspective, existing cold storages are highly inefficient. Cooling the entire warehouse volume consumes large amounts of power even when only specific racks or trays require intervention. This not only increases operational costs but also adds to the environmental burden by raising carbon emissions. With rising electricity tariffs and growing emphasis on green supply chains, there is a clear need for systems that can reduce energy consumption without compromising produce quality.

There is also an economic motivation. Extending the shelf life of fruits and vegetables directly translates to higher profitability for producers and distributors, while also ensuring consumers receive fresher products. Adaptive cooling enables targeted preservation of different produce types within the same facility, opening possibilities for mixed storage that is currently difficult to achieve efficiently.

Technologically, this project is motivated by the opportunity to integrate modern IoT sensors, embedded systems, and artificial intelligence into a domain that has seen little innovation beyond mechanical refrigeration. By applying tray-level sensing and AI-driven control, the system demonstrates how intelligent infrastructure can transform traditional industries. The ability to autonomously decide fan speeds, valve operations, and humidification cycles creates a dynamic storage environment that adapts in real time to produce conditions.

Finally, at a societal level, improving cold storage systems aligns with global priorities such as reducing food wastage, ensuring food security, and minimizing environmental impact. For

a country like India, where agriculture forms a significant part of the economy and perishable supply chains are vital, the adoption of adaptive storage technologies can make a measurable difference. This project is motivated not only by the promise of technical innovation but also by the potential to deliver economic, environmental, and social impact.

1.3 Scope of the Project

The scope of this project is centered on the design, development, and validation of an adaptive cooling system for cold storages, with a focus on fruits and vegetables. The system is implemented and demonstrated at a prototype scale, representing one rack with multiple trays, and is designed to be scalable to larger warehouse environments.

Within this scope, the project specifically addresses:

- Tray-level environmental monitoring: Each tray is closed and fitted with sensors to measure temperature, humidity, CO_2 concentration, ethylene concentration, and weight of the stored produce. These parameters capture both storage conditions and produce physiology.
- Rack-level actuation: Each vertical rack includes two valves, one for conditioned air and one for humidified air. These valves connect to common airflow channels supplying cooling and humidification.
- Tray-level actuation: Each tray is equipped with a DC fan whose speed is modulated to control the intensity of airflow, ensuring that only trays requiring intervention receive it.
- Closed-loop airflow management: Outlet air from trays is collected through a common exhaust channel, passed through a chemical filter to remove ripening agents, and then recirculated.
- AI-driven decision making: Sensor data is read by an Arduino microcontroller and sent to a backend server. An AI agent (running offline on a large language model) processes the snapshot and generates actuator commands in real time, which are immediately applied to hardware.
- Data management and visualization: All sensor readings and actuator commands are logged into a MongoDB database. A custom-built dashboard displays live tray values

and actuator states, while a Plotly-based web application provides interactive plots for trend analysis.

The scope of the project is limited to the prototype scale demonstration, with two trays per rack. Large-scale deployment in full cold storage facilities, integration with industrial HVAC systems, and long-duration storage trials are beyond the current scope but form part of the system's future scalability.

Thus, the project encompasses hardware design, embedded integration, AI-based control, database management, and dashboard visualization, with the primary aim of proving the feasibility and benefits of adaptive cooling at tray-level granularity.

2. PROJECT DESCRIPTION AND GOALS

2.1 Literature Review

Cold storage systems have long been studied as a critical infrastructure for preserving perishable goods. Conventional designs focus on warehouse-wide refrigeration using centralized compressors and evaporators, which are effective for bulk cooling but inefficient in handling localized variations. Research by Singh et al. (2018) emphasized that uneven airflow distribution in cold rooms contributes to heterogeneous cooling and accelerated spoilage in certain zones. Similarly, investigations into ethylene management (Zhang et al., 2020) highlight the need for continuous monitoring of ripening agents, as their accumulation significantly reduces shelf life.

Recent advances in the Internet of Things (IoT) have enabled distributed sensing for agricultural and post-harvest applications. Studies such as Kumar and Patel (2019) proposed sensor networks for monitoring temperature and humidity in cold chains. These systems improve visibility but often stop short of providing autonomous actuation. A survey of modern smart storage solutions reveals that most focus on data logging rather than real-time closed-loop control.

Efforts to incorporate energy efficiency into cold storage design have been reported in literature through demand-side management and optimization of compressor cycles. For instance, Alok et al. (2021) demonstrated how predictive control strategies could save up to 15% energy in large refrigerated warehouses. However, these approaches still treat the

storage space as a homogeneous environment rather than addressing micro-environmental differences across racks and trays.

Work in precision agriculture has shown the benefits of AI and machine learning in controlling irrigation, fertigation, and greenhouse climate. Papers such as Li et al. (2022) discuss adaptive control of greenhouse environments using neural networks and reinforcement learning. While conceptually similar, applications to cold storage remain limited in literature. Very few studies combine IoT sensing, AI decision-making, and adaptive actuation at such fine granularity.

Industrial reports by companies like Carrier and Blue Star acknowledge the potential of smart cold chains, but commercial offerings are largely oriented toward logistics tracking and energy-efficient compressors, rather than tray-level adaptive systems. Patent searches reveal devices for ethylene scrubbing and controlled atmosphere storage, but these are generally applied at the chamber scale.

In summary, the literature reveals extensive work on bulk cold storage, IoT monitoring, and energy-efficient refrigeration. However, there is a notable gap in applying adaptive, tray-level environmental control using AI-driven decision making and closed-loop airflow management. This project situates itself in that unexplored space, drawing inspiration from both IoT-based monitoring research and greenhouse automation strategies while tailoring them to the unique challenges of post-harvest cold storage.

2.2 Gaps Identified

From the review of existing literature and industrial practices, several key gaps have been identified:

1. Uniform bulk cooling vs. localized needs
    - Most cold storages cool the warehouse as a single environment.
    - This ignores tray-level variations in temperature, humidity, and ripening gas concentrations.
    - As a result, certain produce batches spoil earlier while others remain underutilized.

2. Monitoring without actuation
    - IoT solutions proposed in research primarily focus on data collection (temperature/humidity logging) but do not provide closed-loop feedback control.
    - Decision-making and actuation at fine granularity are largely absent.
3. Neglect of ethylene and CO_2 control at micro-scale
    - While studies acknowledge the effect of ripening gases, most mitigation systems operate at the chamber scale using bulk scrubbers.
    - There is little emphasis on tray-level sensing and active airflow management to minimize cross-contamination.
4. Energy inefficiency
    - Optimization efforts in literature focus on compressor cycles or overall energy management but do not reduce the spatial scope of cooling.
    - Energy is wasted in maintaining uniform conditions even when only a few trays require intervention.
5. Lack of AI-driven decision frameworks
    - Greenhouse and smart agriculture systems have begun to leverage AI for adaptive climate control.
    - However, similar strategies have not been systematically applied to post-harvest cold storage, particularly at the tray/rack scale.
6. Limited integration of end-to-end architecture
    - Existing solutions treat sensing, decision-making, actuation, and visualization as separate modules.
    - Very few projects demonstrate a complete system: sensors → AI agent → actuators → data storage → interactive dashboards.

Therefore, there exists a gap in developing a fully integrated, adaptive, and AI-driven cold storage system that manages each tray as an independent micro-environment. This project

directly addresses these shortcomings by combining fine-grained sensing, tray-specific actuation, chemical gas filtration, and AI-based decision support within a unified framework.

2.3 Objectives

The primary objective of this project is to design and demonstrate an Adaptive Cooling System for Cold Storages that minimizes energy consumption while extending the shelf life of fruits and vegetables through localized environmental control.

The specific objectives are as follows:

1. Develop tray-level sensing architecture
    - Equip each tray with sensors to continuously monitor temperature, humidity, CO_2, ethylene, and weight of stored produce.
    - Ensure accurate, real-time data acquisition for fine-grained environmental analysis.
2. Implement adaptive airflow management
    - Design rack-level valves for selective supply of conditioned air and humidified air.
    - Control tray-level DC fans to modulate airflow intensity based on sensor feedback.
3. Integrate closed-loop exhaust filtration
    - Channel exhaust gases from trays through a chemical filter to remove ripening agents (ethylene, CO_2).
    - Recirculate filtered air back into the system, maintaining a balanced closed-loop environment.
4. Design AI-driven decision support
    - Use an AI agent (offline large language model) to interpret sensor snapshots and recommend actuator states.
    - Generate actuator commands in structured formats (JSON) for immediate application to hardware.

5. Enable seamless hardware–software integration
    - Establish communication between Arduino-based sensor units and a backend server via serial interface.
    - Implement reliable actuation of valves and fans based on AI decisions.
6. Ensure data storage and traceability
    - Log all sensor readings and actuator decisions into a MongoDB database for long-term analysis.
    - Maintain structured records for validation, optimization, and future scalability.
7. Develop interactive visualization tools
    - Build an admin dashboard to display live tray values and actuator states in real time.
    - Create a Plotly-based web application for interactive time-series plots to analyze environmental trends and system responses.
8. Validate system performance at prototype scale
    - Demonstrate the feasibility of the approach using a prototype rack with two trays.
    - Evaluate improvements in control precision, energy efficiency, and potential shelf-life extension compared to conventional uniform cooling.

By achieving these objectives, the project aims to prove the feasibility of precision, tray-level adaptive cooling, thereby laying the groundwork for energy-efficient, intelligent cold storage solutions that can be scaled to industrial applications.

2.4 Problem Statement

Conventional cold storage facilities treat the warehouse as a single homogeneous environment, cooling the entire volume uniformly using centralized refrigeration systems. This bulk cooling strategy is inherently inefficient, as it fails to account for localized variations in temperature, humidity, and ripening gas concentrations across racks and trays. As a result, significant amounts of electrical energy are wasted in maintaining conditions for

the entire space, even when only a subset of produce requires intervention. Moreover, the uncontrolled accumulation of gases such as CO_2 and ethylene accelerates uneven ripening and spoilage, leading to high post-harvest losses.

Existing IoT-based monitoring solutions provide useful visibility but stop short of enabling autonomous, tray-specific environmental control. Similarly, current energy optimization techniques target compressor efficiency at a macro scale, ignoring the fine granularity required for precision preservation. There is thus a clear gap in developing an intelligent, adaptive cold storage system capable of:

- monitoring tray-level micro-environments,
- deciding actuator operations in real time using AI, and
- applying conditioned or humidified air only where required.

The problem, therefore, is to design and implement a scalable prototype of such a system that integrates sensors, actuators, AI-based decision support, exhaust gas filtration, and live visualization into a unified closed-loop framework. The goal is to demonstrate reduced energy usage and improved produce preservation compared to conventional bulk cooling methods.

2.5 Project Plan

The project is structured into sequential phases to ensure systematic development and evaluation of the adaptive cooling system:

Phase 1: Problem Study and Requirement Analysis (Completed)

- Studied existing cold storage architectures and their limitations.
- Identified key parameters to monitor (temperature, humidity, CO_2, ethylene, weight).
- Defined functional requirements for sensors, actuators, and control logic.

Phase 2: Prototype Design and Hardware Setup (Completed)

- Developed a rack-level prototype with two trays, each sealed as an independent micro-environment.

- Integrated sensors (DHT22, CO_2, ethylene, load cell) for real-time environmental monitoring.
- Installed rack-level valves for conditioned air and humidified air channels.
- Mounted tray-level DC fans for airflow control.

Phase 3: Communication and Backend Integration (Completed)

- Established Arduino–PC serial communication for transmitting sensor data.
- Implemented a Flask-based backend API for receiving sensor snapshots and transmitting actuator commands.
- Integrated MongoDB for structured storage of sensor readings and AI decisions.

Phase 4: AI Decision Engine (Ongoing)

- Integrated an offline AI agent (LLM via Ollama + LangChain) to process sensor data.
- Defined structured decision output (JSON with valve states and fan speeds).
- Incorporated fallback rule-based control for reliability in case of AI errors.
- Debugging decision-to-actuation latency to optimize response time.

Phase 5: Dashboard and Visualization (Ongoing)

- Built a real-time admin dashboard displaying live tray conditions and actuator states.
- Added toggle between Agent Mode (automatic decision) and Manual Mode (user input for actuators).
- Designed a Plotly-based web application to generate interactive time-series plots from MongoDB.
- Next step: refine UI aesthetics and add additional analytical plots (energy vs. time, gas concentration vs. fan response).

Phase 6: System Validation and Testing (Planned)

- Run controlled experiments with produce samples to evaluate cooling efficiency and preservation quality.
- Compare adaptive tray-level control with conventional bulk cooling baseline.

- Measure metrics such as energy consumption, response latency, and spoilage reduction.

Phase 7: Documentation and Reporting (Ongoing)

- Preparing detailed Review-2 report including methodology, results to date, and future plan.
- Final report will include experimental validation data, performance evaluation, and scalability discussion.

Deliverables

- A functional prototype of adaptive cooling at rack-and-tray level.
- AI-driven backend with MongoDB integration and dual dashboards (admin + Plotly).
- Experimental results demonstrating energy savings and preservation benefits.
- Final report documenting design, implementation, testing, and conclusions.

3. REQUIREMENT ANALYSIS

3.1 Functional Requirements

The adaptive cooling system for cold storages must satisfy the following functional requirements to achieve its intended objectives:

1. Environmental Monitoring
    - Each tray should continuously capture temperature, humidity, CO_2 concentration, ethylene concentration, and weight.
    - Sensor readings must be transmitted at regular intervals (every 2–5 seconds) to the backend system.
2. Decision-Making Capability
    - The backend system should process the latest sensor snapshot and generate actuator commands in real time.
    - Commands should specify:

        - Rack-level AC valve state (open/close).
        - Rack-level humidifier valve state (open/close).
        - Tray-level fan speeds (0–255 PWM).

3. Adaptive Actuation
    - The system should selectively open valves and adjust tray fan speeds according to tray conditions.
    - Airflow must be directed only to trays requiring intervention, ensuring localized control.
4. Closed-Loop Exhaust Handling
    - Exhaust air from trays must pass through a chemical filter to remove ethylene and CO_2 before recirculation.
    - The filtered air should be returned to the conditioned/humidified air channel for reuse.
5. Data Management
    - All sensor readings and actuator commands must be stored in a MongoDB database with timestamps for traceability.
    - Data should be available for both real-time monitoring and historical analysis.
6. Visualization and User Interaction
    - An admin dashboard must display live tray values and actuator states.
    - The dashboard should allow toggling between Agent Mode (automatic control) and Manual Mode (user-specified actuator inputs).
    - A Plotly-based web app must provide time-series plots for trend visualization.
7. Mode Switching
    - In Agent Mode, actuator commands are fully determined by the AI decision engine.
    - In Manual Mode, users must be able to directly input actuator values, which are then applied to the hardware and logged.

8. Reliability and Fault Handling
    - In case of communication failure or unavailable AI response, the system should fall back to a rule-based baseline controller.
    - System should raise alerts in the dashboard if sensor values are missing or actuators fail to respond.

3.2 Non-Functional Requirements

In addition to the functional goals, the adaptive cooling system must meet several non-functional requirements to ensure its reliability, efficiency, and long-term applicability.

1. Performance
    - Sensor readings should be processed and actuator decisions generated with latency < 2 seconds, ensuring near real-time responsiveness.
    - The dashboard must refresh values at regular intervals (every 2–5 seconds) without noticeable lag.
2. Scalability
    - The system must be designed to scale from a two-tray prototype to full-scale warehouse deployments with hundreds of trays and racks.
    - The architecture (sensors → backend → MongoDB → dashboard) should support modular addition of new trays with minimal changes.
3. Reliability and Fault Tolerance
    - In the event of AI decision engine failure, the system must fall back to a baseline rule-based controller to maintain safe operating conditions.
    - Serial communication with the microcontroller must handle disconnections gracefully and attempt automatic recovery.
    - MongoDB must ensure no data loss during network interruptions, with local buffering if required.
4. Usability

- The dashboard should present tray sensor values and actuator states in a clear, user-friendly layout.
- Users must be able to switch easily between Agent Mode and Manual Mode via a toggle button.
- The Plotly visualization web app must allow intuitive exploration of historical data (zoom, hover, export).

5. Maintainability
    - The codebase must follow a modular structure (separate files for serial reader, writer, agent, database, models, and dashboard).
    - Configuration values such as COM port, baud rate, and database connection strings must be stored in a single config file for easy updates.
    - Debug logs (e.g., [SERIAL], [PARSED], [AGENT-IN], [AGENT-RAW-OUT]) must provide traceability for troubleshooting.
6. Security
    - Local endpoints (Flask API) must restrict unauthorized access when scaled beyond prototype.
    - Sensitive data such as database credentials must be managed via environment variables.
7. Energy Efficiency
    - The adaptive system must demonstrate measurable energy savings compared to bulk cooling, by targeting only trays requiring intervention.
    - Fan and valve actuation logic should prioritize minimal energy use while preserving produce quality.
8. Cost Effectiveness
    - Prototype hardware should use affordable, widely available components (Arduino, DHT22, load cell, CO_2/ethylene sensors, DC fans).
    - System design should allow incremental scaling without major redesign.

3.3 System Requirements (Hardware & Software)

To successfully design and demonstrate the adaptive cooling system, both hardware and software resources are required.

3.3.1 Hardware Requirements

1. Microcontroller & Interfaces
    - Arduino Uno / Mega: To interface with sensors, read environmental data, and transmit to backend via serial communication.
    - Serial-to-USB interface for PC connection.
2. Sensors
    - Temperature & Humidity Sensor (DHT22 / equivalent): Monitors microclimate inside each tray.
    - CO_2 Sensor (e.g., MQ-135 or NDIR-based): Detects accumulation of carbon dioxide.
    - Ethylene Gas Sensor (C2H4-specific module): Measures ethylene concentration to track ripening.
    - Load Cell with HX711 Amplifier: Captures weight of produce in each tray, useful for monitoring produce loss or respiration.
3. Actuators
    - DC Cooling Fans (PWM controlled): Installed in each tray for airflow modulation.
    - Solenoid Valves (12V/24V): One for conditioned air, one for humidified air at rack level.
    - Relay Module or Motor Driver: To control solenoid valves via microcontroller.
4. Airflow & Filtration Units
    - Ducting and Common Air Channels for supply and exhaust.

- Chemical Filter (activated carbon / potassium permanganate) for removing CO_2 and ethylene before recirculation.

5. Prototype Rack Setup
    - Wooden/metal rack with two trays sealed with covers to simulate localized environments.
    - Piping connections to conditioned air and humidified air supply.
6. Supporting Hardware
    - 12V/24V Power Supply for fans and valves.
    - Wires, connectors, PCB/prototyping board for integration.

3.3.2 Software Requirements

1. Backend and Control
    - Python 3.x: Primary programming language for backend services.
    - Flask (REST API): For communication between frontend dashboard, AI agent, and hardware.
    - PySerial: For reading/writing serial data to Arduino.
    - LangChain + Ollama (with Llama 3.1 model): For running the AI agent offline to decide actuator states.
2. Database
    - MongoDB: Stores sensor snapshots, actuator decisions, and historical logs.
    - MongoDB Compass: For database management and verification.
3. Frontend Visualization
    - HTML/CSS/JavaScript Dashboard: Displays live tray sensor values and actuator states.
    - Plotly Dash Framework: For interactive time-series plots of temperature, humidity, CO_2, ethylene, and actuators.
    - Bootstrap / custom CSS: For responsive layout and user-friendly design.

4. Development Tools
    - Arduino IDE: For coding and uploading microcontroller firmware.
    - VS Code / PyCharm: For Python backend development.
    - GitHub: For version control and project management.

3.3.3 Environmental Requirements

- Operating Environment: Prototype tested in laboratory conditions; designed for eventual deployment in cold storage facilities.
- Power: Stable 230V AC mains with step-down supplies for Arduino and actuators.
- Network: Localhost or LAN for backend–dashboard communication; internet not mandatory since AI runs offline.

4. SYSTEM DESIGN

4.1 System Architecture

The proposed system is designed as an integrated closed-loop adaptive cooling framework that combines hardware sensing and actuation with AI-driven decision-making and real-time visualization. The architecture can be divided into four major layers:

1. Sensing Layer
    - Each tray in the rack is equipped with sensors to measure temperature, humidity, CO_2 concentration, ethylene levels, and weight.
    - Sensors are connected to an Arduino microcontroller, which collects readings at fixed intervals (every 2–5 seconds).
    - The trays are sealed, ensuring that each operates as a controlled micro-environment.
2. Communication Layer
    - The Arduino transmits sensor readings via serial communication to a backend server.

- Data is formatted as a structured packet (float values for each parameter per tray).
- This ensures lightweight, reliable transfer between hardware and the decision engine.

3. Decision & Control Layer
    - The backend server (Python + Flask) receives sensor data and passes it to an AI agent.
    - The agent, implemented using LangChain + Ollama (Llama 3.1 model), interprets the snapshot and outputs actuator commands in JSON format.
    - The commands include:
        - Rack AC valve: open (1) / closed (0)
        - Rack humidifier valve: open (1) / closed (0)
        - Tray 1 fan speed: 0–255 (PWM)
        - Tray 2 fan speed: 0–255 (PWM)
    - The backend immediately writes these actuator commands back to the Arduino over the same serial interface.
4. Actuation Layer
    - Rack-level solenoid valves control the inflow of conditioned air and humidified air.
    - Tray-level DC fans regulate airflow intensity.
    - Outlet air from trays passes into a common exhaust duct, routed through a chemical filtration unit that removes CO_2 and ethylene before recirculation.
5. Data Storage and Visualization Layer
    - Every cycle, sensor readings and actuator decisions are logged into a MongoDB database with timestamps.
    - An Admin Dashboard (HTML/JS) displays live tray sensor values, actuator states, and a toggle for Agent/Manual mode.

- A Plotly Dash web application provides interactive plots of temperature, humidity, CO_2, ethylene, weight, and actuator states over time.

This layered architecture ensures that the system operates as a closed feedback loop: Sensors → Backend (AI Agent) → Actuators → Produce Environment → Sensors again.

The modular design allows scalability to multiple racks and trays by simply extending sensor-actuator pairs and updating the backend to handle additional channels.

4.2 Hardware Design

The hardware design of the adaptive cooling system integrates prototype rack construction, sensor deployment, actuator mechanisms, and power supply into a compact, scalable unit.

4.2.1 Prototype Rack and Trays

- A two-tray rack was fabricated to demonstrate the system at prototype scale.
- Each tray is sealed with a transparent acrylic or polycarbonate cover, creating an isolated micro-environment while still allowing airflow through designated inlets and outlets.
- The trays are mounted vertically on the rack structure, simulating how large-scale cold storage racks hold multiple trays of fruits/vegetables.
- Each tray has:
    - Inlet duct connected to the rack's common conditioned/humidified air channels.
    - Outlet duct connected to the common exhaust pipe that leads to a chemical filtration unit.

4.2.2 Sensor Integration

Each tray is equipped with a set of five sensors for real-time environmental monitoring:

1. Temperature & Humidity (DHT22) – monitors microclimatic temperature (°C) and relative humidity (%).
2. CO_2 Sensor (NDIR-based or MQ-135) – detects CO_2 levels in ppm, indicating respiration activity.
3. Ethylene Gas Sensor (C_2H_4-specific module) – monitors ethylene concentration, a key ripening agent.
4. Weight Sensor (Load cell + HX711 amplifier) – tracks produce mass, useful for detecting water loss or respiration-related shrinkage.
5. Auxiliary connections for future expansion (e.g., O_2 sensors, volatile organic compound sensors).

All sensors are interfaced to an Arduino Uno/Mega, which samples and transmits readings at intervals of 2–5 seconds.

4.2.3 Actuation Mechanisms

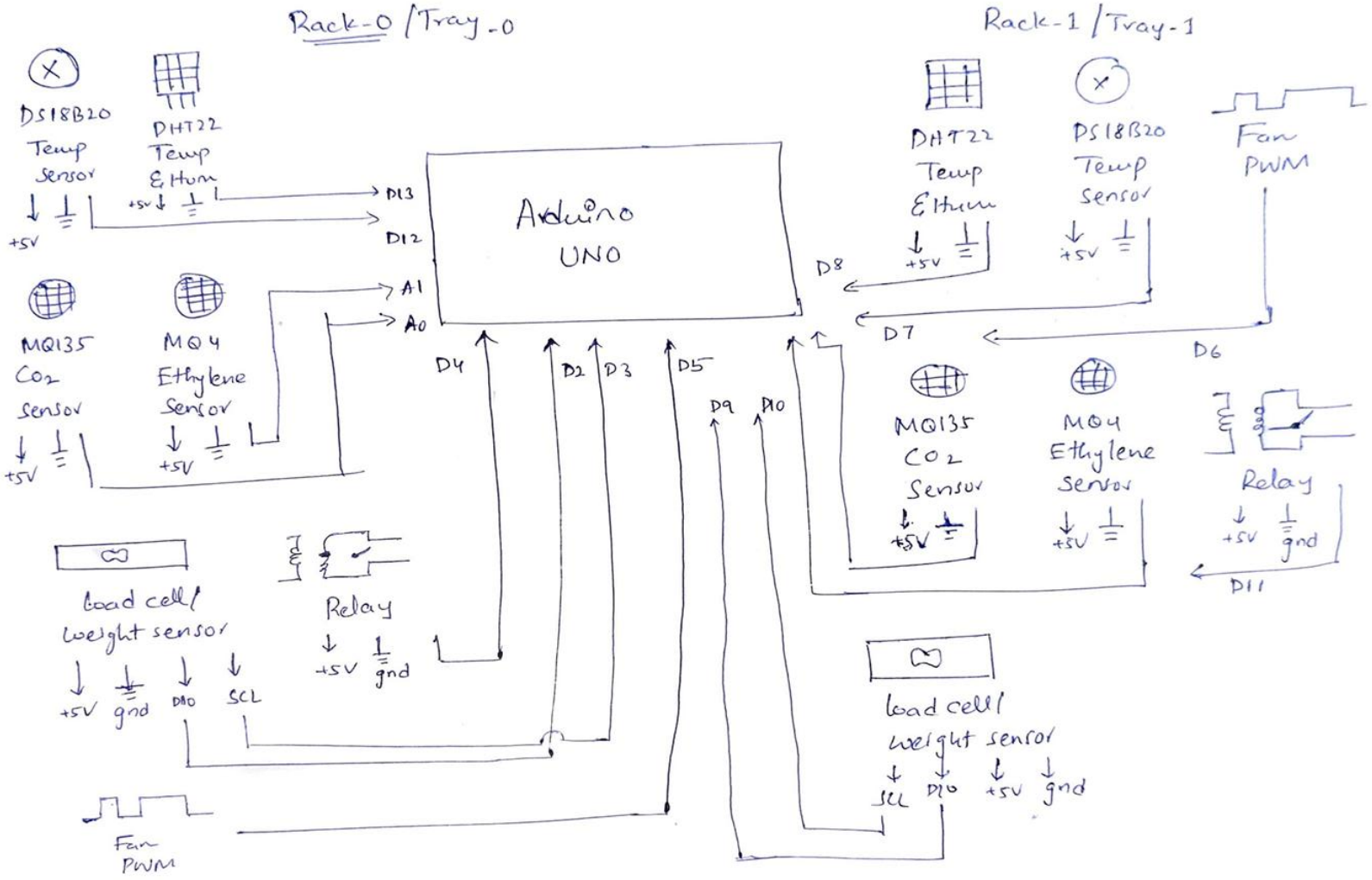
1. Rack-Level Actuation
    - Solenoid Valves (12V/24V) are installed at the junction of each rack's inlet duct.
    - One valve controls the conditioned air channel (cooling).
    - Another valve controls the humidified air channel (humidity maintenance).
    - Valves operate in binary mode (open/close), with decisions issued by the backend AI.
2. Tray-Level Actuation
    - Each tray contains a PWM-controlled DC fan.
    - The fan speed (0–255 duty cycle) is adjusted according to tray-specific needs (e.g., higher speed if temperature/ethylene is high).
    - Fans ensure air is distributed evenly across produce in the tray.
3. Exhaust Filtration

- A common exhaust channel collects outlet air from trays.
- Air is passed through a chemical filter unit (e.g., activated carbon or potassium permanganate filter) to remove ethylene and CO_2.
- Filtered air is returned to the conditioned air circuit, maintaining a closed-loop airflow system.

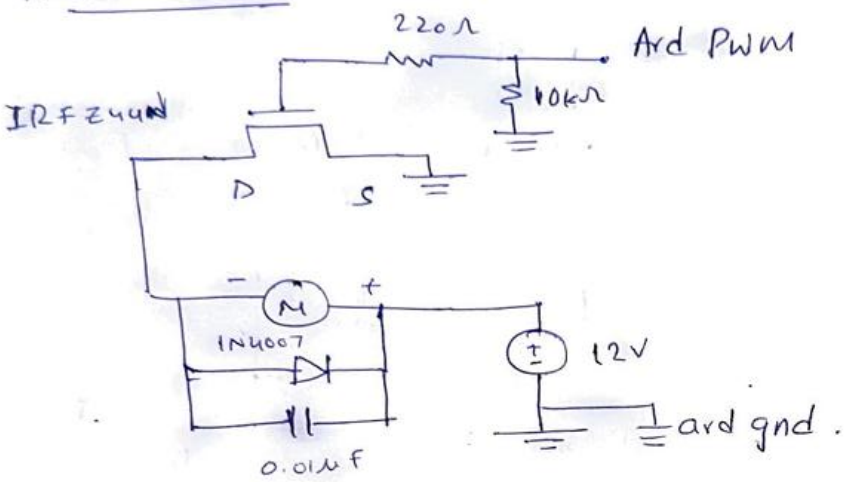
4.2.4 Power Supply and Control

- Arduino Uno/Mega operates at 5V, powered by USB or regulated adapter.
- Solenoid valves require 12V/24V DC, powered through an external adapter.
- DC fans (typically 12V) are driven via a MOSFET driver or relay board.
- A regulated power distribution system ensures that microcontroller circuits (5V) and actuator circuits (12V/24V) are isolated to prevent interference.

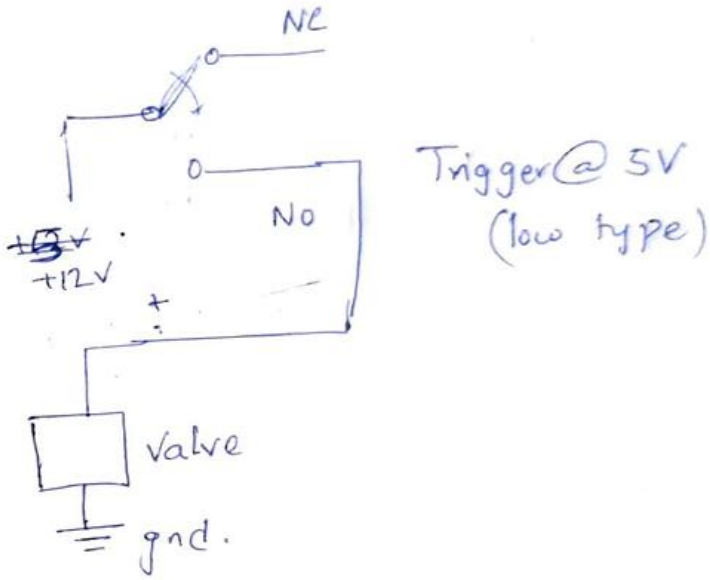
4.2.5 Block Diagram of Hardware


Rack-0 / Tray-0
Rack-1 / Tray-1
DS18B20 Temp Sensor
DHT22 Temp & Hum
Arduino UNO
MQ135 CO2 Sensor
MQ4 Ethylene Sensor
Relay
load cell / weight sensor
Fan PWM
+5V
gnd
SCL
SDA
D13
D12
A1
A0
D4
D2
D3
D5
D9
D10
D8
D7
D6
D11

Motor driver



Control 12V valves =>



4.2.6 Prototype Summary

- Two trays per rack (scalable to many trays).
- Five sensors per tray for environmental monitoring.
- Two solenoid valves per rack (AC valve, humidifier valve).
- Two DC fans (one per tray) for airflow control.
- Arduino-based control for interfacing sensors and actuators.
- Closed-loop exhaust filtering for maintaining balanced conditions.

This hardware setup demonstrates the feasibility of tray-level adaptive cooling and provides the physical foundation for the AI-driven decision-making system.

4.3 Software Design

The software component of the adaptive cooling system serves as the control brain, linking hardware sensors and actuators with intelligent decision-making and real-time visualization. The design is modular, consisting of five main modules:

4.3.1 Data Acquisition Module

- Implemented in Python using the PySerial library.
- Continuously reads sensor packets from Arduino over serial (temperature, humidity, CO_2, ethylene, weight for each tray).
- Parses raw data into structured JSON snapshots (e.g., {"tray1": {...}, "tray2": {...}}).
- Stores the latest snapshot in memory for immediate use by other modules.
- Includes error handling for malformed packets and auto-recovery in case of serial disconnects.

4.3.2 Decision-Making Module (AI Agent)

- Built with LangChain + Ollama, running an offline Llama 3.1 model on the host PC.
- Takes the current snapshot as input and generates actuator decisions in structured JSON:

{
"rackACvalve": 1,
"rackHumidifiervalve": 0,
"tray1fanspeed": 180,
"tray2fanspeed": 200
}

- Also generates a short textual rationale for debugging and logging.
- Ensures robustness with a fallback rule-based controller in case the AI fails or produces invalid outputs.
- Logs agent prompt, raw output, and parsed decisions for transparency.

4.3.3 Actuator Control Module

- Receives decisions from the AI agent and transmits them to Arduino over the same serial connection.
- Commands are encoded as space-separated values (e.g., 1 0 180 200).
- Arduino interprets these values and applies them to valves and fans.
- Includes acknowledgment and debug messages ([SERIAL-WRITE]) for monitoring in backend logs.

4.3.4 Database Management Module

- Uses MongoDB for structured storage of each cycle:
  - Timestamp
  - Sensor snapshot
  - AI decision (actuators + rationale)
  - Flags for applied status (applied_ok, apply_error).

- Enables retrieval of both recent and historical data for dashboard and analytics.
- Database operations abstracted into a dedicated module (db.py) for maintainability.

4.3.5 Visualization & User Interface Module

1. Admin Dashboard (Flask + HTML/JS)
    - Provides a live view of tray sensor values and actuator states.
    - Includes toggle between Agent Mode and Manual Mode.
    - In Manual Mode, user inputs are applied directly to actuators and logged.
2. Plotly Dash Web Application
    - Provides interactive time-series plots for all key parameters:
        - Tray temperatures, humidity, CO_2, ethylene, weight.
        - Actuator states (valves and fan speeds).
    - Updates automatically every 5 seconds by fetching the latest data from MongoDB.
    - Allows zoom, pan, and export features for deeper analysis.

4.3.6 Software Integration Flow

1. Arduino sends sensor data → Backend (Data Acquisition).
2. Backend forwards snapshot → AI Agent for decision.
3. AI generates actuator commands → Backend applies via serial to Arduino.
4. Snapshot + decision logged into MongoDB.
5. Dashboard displays live values, and Plotly app renders historical trends.

This modular design ensures that each layer (sensing, AI decision, actuation, storage, visualization) can be developed, tested, and upgraded independently while still contributing to the unified adaptive cooling system.

4.4 Data Flow & Control Logic

The adaptive cooling system operates as a closed-loop feedback cycle, where sensor data drives intelligent decision-making, actuation, and logging. The complete flow can be described in sequential stages:

4.4.1 Sensor Data Acquisition

1. Tray-level sensors (temperature, humidity, CO_2, ethylene, weight) continuously measure environmental parameters.
2. The Arduino microcontroller samples sensor readings at fixed intervals (every 2–5 seconds).
3. Sensor data is packaged into a structured string and transmitted over serial communication to the backend server.

4.4.2 Data Parsing and Preprocessing

1. The backend acquisition module receives the serial string.
2. Data is parsed into JSON format with tray-wise values.
3. Preprocessing includes:
    - Clamping negative values (e.g., weights < 0 set to 0).
    - Filtering malformed packets.
    - Updating the latest snapshot in memory for decision-making.

4.4.3 AI-Driven Decision Making

1. The latest snapshot is forwarded to the AI agent.
2. The agent is prompted with system rules and target thresholds (e.g., temperature range, humidity range, CO_2/ethylene limits).
3. The agent outputs a structured actuator command JSON that specifies:
    - Rack AC valve state (0/1).

- Rack humidifier valve state (0/1).
- Tray fan speeds (0–255).

4. A fallback rule-based controller generates safe defaults if the agent fails to respond or outputs invalid commands.
5. Both the raw agent output and the parsed decision are logged for debugging.

4.4.4 Actuator Application

1. The validated actuator commands are transmitted back to the Arduino over serial.
2. The Arduino applies the commands to the hardware:
   - Opens/closes rack solenoid valves.
   - Adjusts PWM duty cycles of tray fans.
3. The airflow pathway directs conditioned or humidified air to the required trays.
4. Exhaust air passes through a chemical filtration unit, removing CO_2 and ethylene before being recirculated.

4.4.5 Data Logging

1. Each cycle (snapshot + decision + actuator status) is logged into MongoDB.
2. Records include:
   - Timestamp.
   - Sensor values.
   - AI rationale.
   - Actuator states.
   - Status flags (applied_ok, apply_error).

4.4.6 Visualization & User Feedback

1. The Admin Dashboard fetches the latest state from the backend every few seconds:

- Displays live tray sensor values.
- Shows actuator states and rationale.
- Allows mode switching between Agent Mode (automatic) and Manual Mode (user-specified commands).

2. The Plotly Web App retrieves historical data from MongoDB and renders interactive time-series plots for trend analysis.

4.4.7 Continuous Feedback Loop

The process repeats continuously as:
Sensors → Arduino → Backend → AI Agent → Actuators → Produce Environment → Sensors

This ensures that the cooling and humidification dynamically adapt to the real-time needs of the stored produce.

5. REFERENCES

[1] Singh, R., Tripathy, P. P., & Kumar, S. (2018). *Airflow distribution and temperature uniformity in cold storage structures: A review.* Journal of Food Engineering, 237, 65–77.

[2] Zhang, L., Li, X., & Chen, W. (2020). *Ethylene management technologies for fruit and vegetable storage: Advances and challenges.* Postharvest Biology and Technology, 168, 111–119.

[3] Kumar, A., & Patel, N. (2019). *IoT-enabled smart cold chain monitoring: A case study in agricultural logistics.* International Journal of Internet of Things and Applications, 7(2), 45–52.

[4] Alok, A., Mishra, V., & Sharma, A. (2021). *Energy optimization in refrigerated warehouses using predictive control strategies.* Energy Reports, 7, 1124–1132.

[5] Li, J., Wang, H., & Zhao, Y. (2022). *Adaptive climate control in greenhouses using neural networks and reinforcement learning.* Computers and Electronics in Agriculture, 198, 107023.

[6] Carrier Global Corporation. (2021). *Smart Cold Chain Solutions: Enabling food security with technology.* White Paper.

[7] Blue Star India. (2020). *Cold Storage Systems for Agricultural Supply Chains.* Technical Brochure.

[8] Bhosale, B., & Dey, S. (2017). *Postharvest losses in fruits and vegetables: An Indian perspective.* Agricultural Economics Research Review, 30(1), 91–102.

[9] Kumar, V., & Kalita, P. (2017). *Reducing postharvest losses during storage of grain crops to strengthen food security in developing countries.* Foods, 6(1), 8.

[10] Pathare, P. B., Opara, U. L., & Al-Said, F. A. (2013). *Colour measurement and analysis in fresh and processed foods: A review.* Food and Bioprocess Technology, 6, 36–60.

[11] Arora, A., & Singh, S. (2019). *Cold storage design and challenges for fruits and vegetables in India.* International Journal of Agricultural Engineering, 12(2), 312–318.

[12] Mowat, A., & Padfield, C. A. (2020). *Ethylene in postharvest biology: A critical review.* Horticulture Research, 7(1), 1–12.

[13] FAO. (2019). *The State of Food and Agriculture: Moving forward on food loss and waste reduction.* Food and Agriculture Organization of the United Nations.

[14] Bose, S., & Ghosh, S. (2016). *Energy efficiency in refrigeration systems for cold storage: A review.* Renewable and Sustainable Energy Reviews, 59, 515–531.

[15] Patil, R. A., & Kapse, P. (2015). *Application of IoT in agricultural cold storage: Smart preservation of perishable commodities.* International Journal of Computer Applications, 120(15), 10–15.

[16] Verma, S., & Srivastava, R. (2020). *IoT-enabled monitoring and prediction of perishable food storage.* Journal of Food Process Engineering, 43(8), e13445.

[17] Basu, D., & Das, A. (2018). *Microcontroller based automation in cold storage for fruits and vegetables.* International Journal of Engineering Research & Technology, 7(4), 1–5.

[18] Kumar, A., & Raj, T. (2019). *Intelligent cold storage management using embedded systems.* International Journal of Emerging Technologies in Engineering Research, 7(11), 19–25.

[19] Pandey, A., & Mishra, V. (2020). *AI-based monitoring and optimization of cold chain logistics.* Procedia Computer Science, 170, 682–689.

[20] Gupta, R., & Yadav, S. (2021). *Edge computing for IoT-enabled smart cold storage systems.* International Journal of Advanced Computer Science and Applications, 12(5), 155–162.

[21] Patentscope. (2019). *Method and apparatus for controlling atmosphere in cold storage.* WIPO Patent WO2019123456A1.

[22] Arduino.cc. (2024). *Arduino Uno Technical Specifications.* Available: https://www.arduino.cc/

[23] MongoDB Inc. (2023). *MongoDB Compass Documentation.* Available: https://www.mongodb.com/

[24] Dash Enterprises. (2024). *Plotly Dash User Guide.* Available: https://dash.plotly.com/

[25] LangChain AI. (2024). *LangChain for LLM-powered decision workflows.* Available: https://www.langchain.com/

[26] Ollama. (2024). *Ollama local LLM runtime documentation.* Available: https://ollama.ai/

[27] United Nations Environment Programme (UNEP). (2021). *Reducing food loss along agricultural value chains.* UN Report.

[28] Ministry of Food Processing Industries (MoFPI, India). (2020). *Cold Chain Projects in India: Status and Opportunities.* Government Report.