

RMS Content

## **RMS – Production Robot Management System**

Full-Stack Ownership • Live Production • Robotics & Automation

### **Overview**

RMS (Robot Management System) is a production-grade, enterprise-level platform that I designed, built, deployed as part of my work at the company.

The system is live in production and is actively used by 5+ hotels to manage their robotic fleets, handle robot calls, and automate real-world operational tasks.

RMS is a mission-critical system that directly controls physical robots operating in hospitality environments and supports 24/7 reliability.

This project is not a demo, prototype, or academic exercise — it is a real business system, used daily by real customers.

### **Project Purpose & Business Impact**

The purpose of RMS is to provide hotels with a centralized, secure, and scalable control platform for:

Managing service robots across multiple hotels and locations

Dispatching robot calls and operational tasks in real time

Automating robot actions based on live external data

Monitoring robot state, availability, and execution

Enforcing strict security, permissions, and auditability

RMS bridges the gap between software systems and physical robotics, enabling hotels to operate robots efficiently, safely, and at scale.

### **Real-World Production Usage**

Deployed across multiple hotels

Controls live robots in real environments

Handles robot calls, task orchestration, and automation

Designed for continuous operation and operational stability

Maintained and evolved based on real customer needs

### **Core System Capabilities**

Robot & Machine Management

Centralized management of robots and machines

Assignment of robots to hotels, locations, and zones

Tracking robot availability and operational state

Support for multi-hotel, multi-robot deployments

### **Robot Calls & Task Orchestration**

Creation and dispatch of robot calls from the dashboard

Task lifecycle management:

Created

Dispatched

In progress

Completed / Failed

Safe handling of concurrent robot requests

Prevention of task collisions and invalid dispatches

### **Third-Party Robotics Integration**

RMS includes a deep production integration with Yunji Technology, a Chinese company specializing in autonomous hospitality robots.

This integration allows RMS to act as a vendor-aware orchestration layer between hotel operations and physical robots.

Capabilities

Secure communication with Yunji robot APIs

Dispatching robot commands and operational tasks

Mapping RMS business logic to vendor-specific robot commands

Fetching robot status and availability

Synchronizing robot state with backend systems

Supporting multiple robots per hotel and per location

The system is designed to be vendor-agnostic, allowing future robot providers to be integrated without rewriting business logic.

### **Automation & Web-Driven Task Pipelines**

RMS includes a dedicated automation layer that enables robots to act based on live external information.

Automation Flow

Puppeteer-based web automation and scraping

Extraction of data from external web systems

Validation and normalization of scraped data

Conversion of data into structured robot tasks

Secure dispatch of tasks to robots via backend APIs

End-to-end pipeline:

### **External Web Source**

→ Automation (Puppeteer) → RMS Backend Processing → Robot Task Dispatch → Physical Robot Execution

This allows robots to perform dynamic, data-driven actions, not static or manual commands.

### **Authentication, Authorization & Security**

Security was a core design principle, not an afterthought.

Firebase Authentication as the identity provider

Token-based authentication (ID tokens verified on every request)

Strict ownership-based authorization enforced server-side

Users can only access entities they own or are assigned to

All permission checks are validated in the backend

Role-aware access (admin, operator, scoped users)

Rate limiting and request validation

Protection against unauthorized access and privilege escalation

### **Admin Dashboard & Operational UI**

RMS includes a full admin and operations dashboard built for real usage:

User and access management

Robot and machine control

Live robot call monitoring

Task execution visibility

Operational data inspection

Soft deletion with full history preservation

The UI is optimized for clarity, speed, and operational efficiency, not just appearance.

### **Audit, Logging & Accountability**

RMS was built with production accountability and traceability in mind:

Dedicated UserActions / audit logging system

Full tracking of:

Authentication events

Robot calls and task dispatches

Entity creation, updates, and deletions

Soft-delete support with metadata (who, when, why)

Enables debugging, compliance, and operational transparency

### **Architecture & Technology Stack**

Backend

Node.js

Express

Firebase Admin SDK

Firestore

RESTful API architecture

Modular structure (controllers, services, routes, middleware)

Secure authorization middleware

Automation services integrated into backend workflows

## **Automation**

Puppeteer  
Headless browser automation  
Web scraping pipelines  
Task generation and dispatch logic  
Robust error handling and retries  
Frontend  
React / React Native  
Firebase Auth integration  
Secure API communication using ID tokens  
Dashboard-oriented UI architecture  
Engineering Practices  
Environment-based configuration  
Clear separation of concerns  
Defensive programming for production failures  
Scalable and extensible data model  
Designed for long-term maintainability

## **My Role & Ownership**

I was responsible for the entire lifecycle of the system, including:

System architecture and data modeling  
Backend API design and security  
Authentication and authorization strategy  
Third-party robotics API integration  
Automation and scraping pipelines  
Robot task orchestration  
Dashboard and operational tooling  
Production deployment, monitoring, and maintenance  
Supporting real customers in live hotel environments

This role required full ownership, production responsibility, and direct interaction with real operational constraints.

This project demonstrates my ability to:

Build and own real production systems  
Design secure, scalable, and auditable architectures  
Integrate software with physical robotics  
Automate complex workflows using live web data  
Deliver systems that are actively used by real businesses  
Operate and maintain mission-critical software in production  
RMS reflects my approach as a full-stack engineer with a production mindset, capable of taking systems from concept to real-world deployment and long-term operation.