Intelligent Cloud Platform

CMPE - 281

Project Deliverable 1

System Design Document

Project: Intelligent Cloud Platform for Smart Homes

Instructor: Dr. Jerry Gao

Team:
Vijaya Sharavan Reddy Baddam - 018321342
Venkata Gowtham Jalam - 018315791
Nikhil Dupally - 018325736
Divyasri Lakshmi Alekhya Nakka - 018291702

Table Of Contents

Section 1: Introduction

1.1 Project Introduction

Smart and safe home environments are becoming more and more necessary in today's world. As more senior individuals choose independent living, it is critical to have a strong monitoring system that guarantees automation, safety, and immediate emergency responses. AI-driven alert detection, advanced IoT device management, and audio/video surveillance are all supported by the **Intelligent Cloud Platform for Smart Homes**, a system that operates over the cloud.

1.2 Project Objectives

- Construct a consolidated cloud platform for senior living facilities and smart homes.
- Use real-time audio and video data to enable remote surveillance.
- Use AI models to identify anomalous, emergency, and safety situations.
- Give different stakeholders (homeowners, cloud employees, and device teams) dashboards according to their roles.
- For the management of numerous residences, guarantee scalability, security, and multi-user tenancy.

1.3 Expected Outcomes

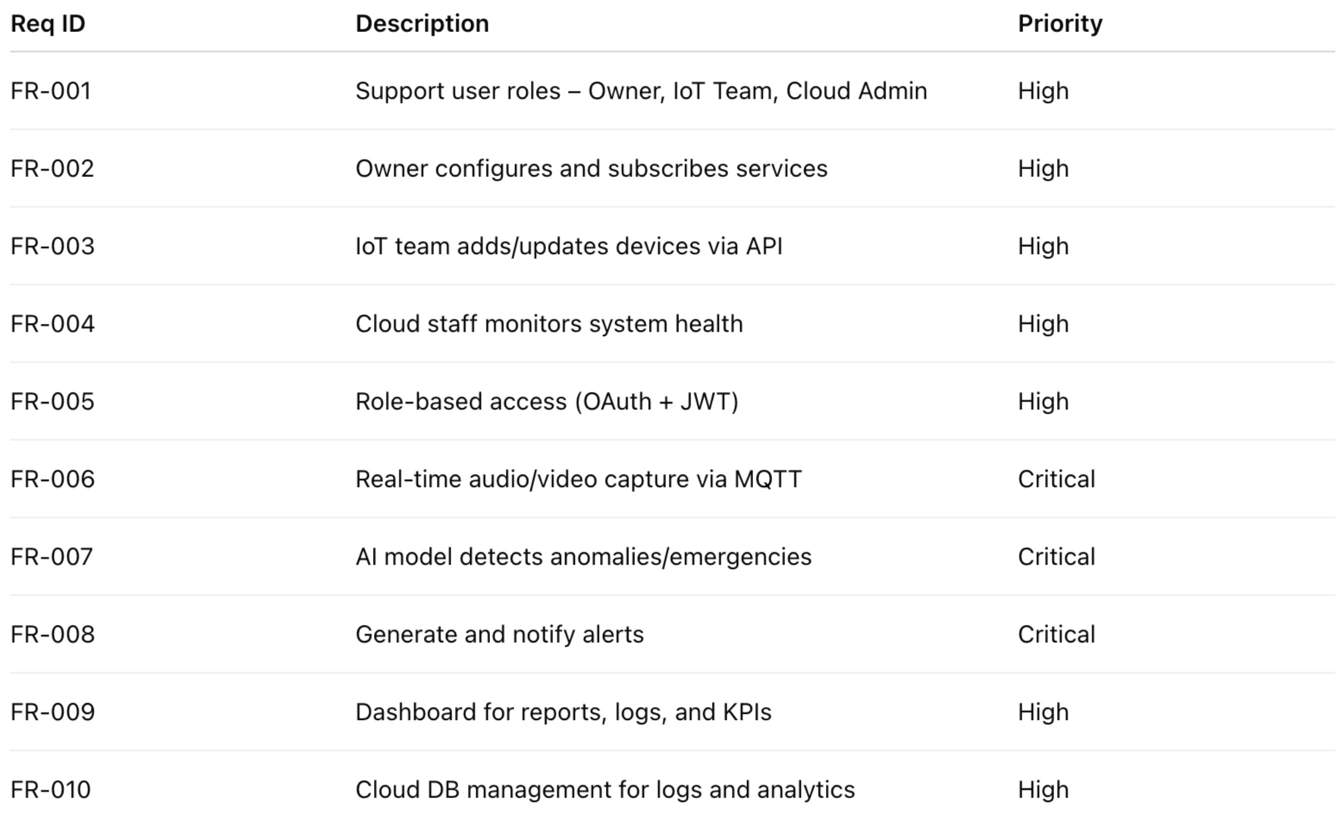
- Amazon EC2-hosted cloud system that is fully functional.
- Alert notifications and real-time smart home monitoring.
- Administrators can monitor and configure system components via the cloud dashboard.
- Database management system for logging and storing user configurations, warnings, and device data.
- UI that is responsive and easy to use for all user groups.

1.4 Scope & Limitations

In Scope: Individual smart homes, IoT sensors, pattern recognition, alerting.
Out of Scope: Cross-house analytics, third-party medical system integration.

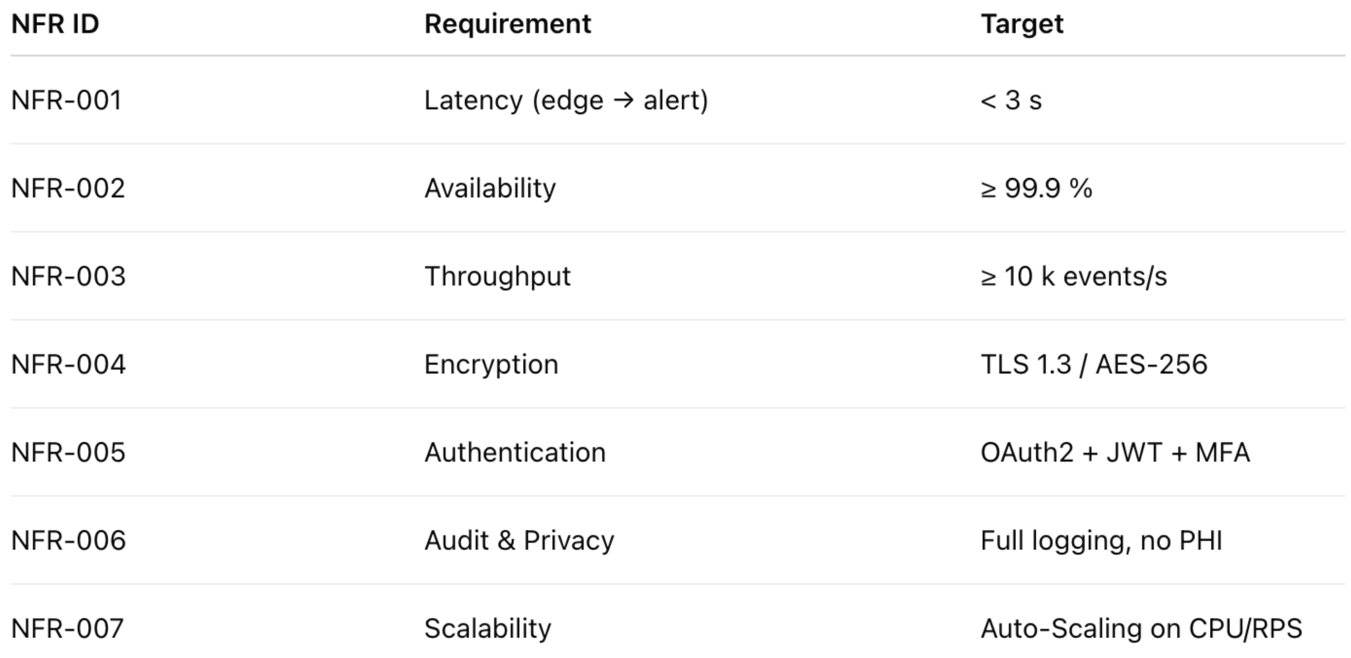
Section 2: System Requirements and Analysis

2.1 Functional Requirements



Req ID	Description	Priority
FR-001	Support user roles – Owner, IoT Team, Cloud Admin	High
FR-002	Owner configures and subscribes services	High
FR-003	IoT team adds/updates devices via API	High
FR-004	Cloud staff monitors system health	High
FR-005	Role-based access (OAuth + JWT)	High
FR-006	Real-time audio/video capture via MQTT	Critical
FR-007	AI model detects anomalies/emergencies	Critical
FR-008	Generate and notify alerts	Critical
FR-009	Dashboard for reports, logs, and KPIs	High
FR-010	Cloud DB management for logs and analytics	High

2.2 Non-Functional Requirements

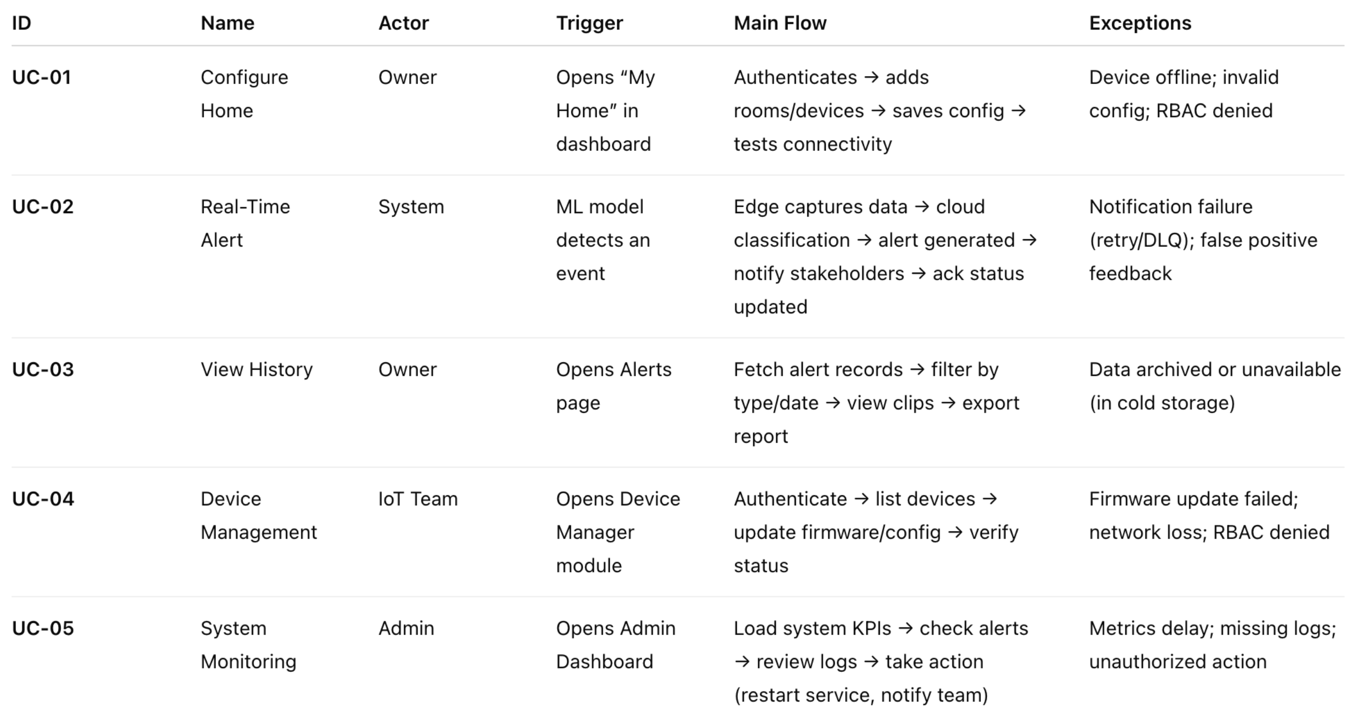


NFR ID	Requirement	Target
NFR-001	Latency (edge → alert)	< 3 s
NFR-002	Availability	≥ 99.9 %
NFR-003	Throughput	≥ 10 k events/s
NFR-004	Encryption	TLS 1.3 / AES-256
NFR-005	Authentication	OAuth2 + JWT + MFA
NFR-006	Audit & Privacy	Full logging, no PHI
NFR-007	Scalability	Auto-Scaling on CPU/RPS

2.3 Stakeholder Roles and User Groups:

1. House Owners:
    - Configure and subscribe to services.
    - View alerts and device status through a personal dashboard.
    - Manage service preferences and receive notifications.
2. Edge-Based IoT Devices Team:
    - Add, delete, or modify IoT device configurations.
    - Monitor device status and perform firmware updates.
    - Manage connectivity and diagnostics.
3. Cloud Service Staff:
    - Maintain overall system health.
    - Monitor and respond to alerts generated by AI modules.
    - Access logs, reports, and analytics through the admin dashboard.

2.4 Key Use Cases



ID	Name	Actor	Trigger	Main Flow	Exceptions
UC-01	Configure Home	Owner	Opens "My Home" in dashboard	Authenticates → adds rooms/devices → saves config → tests connectivity	Device offline; invalid config; RBAC denied
UC-02	Real-Time Alert	System	ML model detects an event	Edge captures data → cloud classification → alert generated → notify stakeholders → ack status updated	Notification failure (retry/DLQ); false positive feedback
UC-03	View History	Owner	Opens Alerts page	Fetch alert records → filter by type/date → view clips → export report	Data archived or unavailable (in cold storage)
UC-04	Device Management	IoT Team	Opens Device Manager module	Authenticate → list devices → update firmware/config → verify status	Firmware update failed; network loss; RBAC denied
UC-05	System Monitoring	Admin	Opens Admin Dashboard	Load system KPIs → check alerts → review logs → take action (restart service, notify team)	Metrics delay; missing logs; unauthorized action

Process Analysis:

- **Data Flow:**
  Audio/video captured by IoT devices → Streamed to cloud servers → Analyzed by ML models → Alerts generated if anomalies detected → Notifications sent to relevant users → Logs stored in the database
- **Service Subscription Flow:**
  Homeowner login → Selects service package → Payment/activation → Devices activated and configured → System goes live.

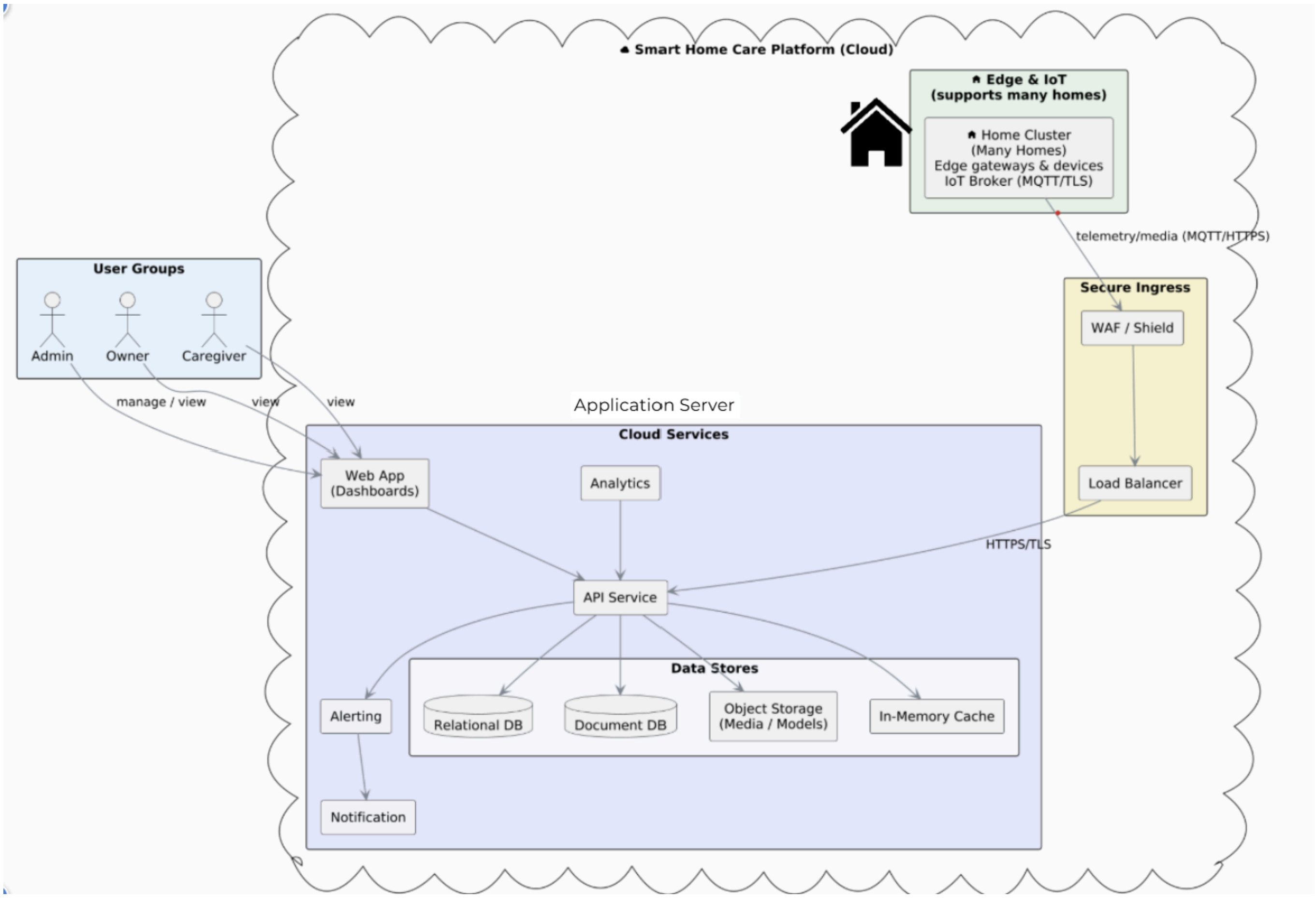
2.5 Assumptions & Constraints

- Stable broadband and secure edge gateway.
- MQTT over TLS for device data.
- Budget optimization (S3 lifecycle, spot instances).
- No PII/PHI stored; data minimization.

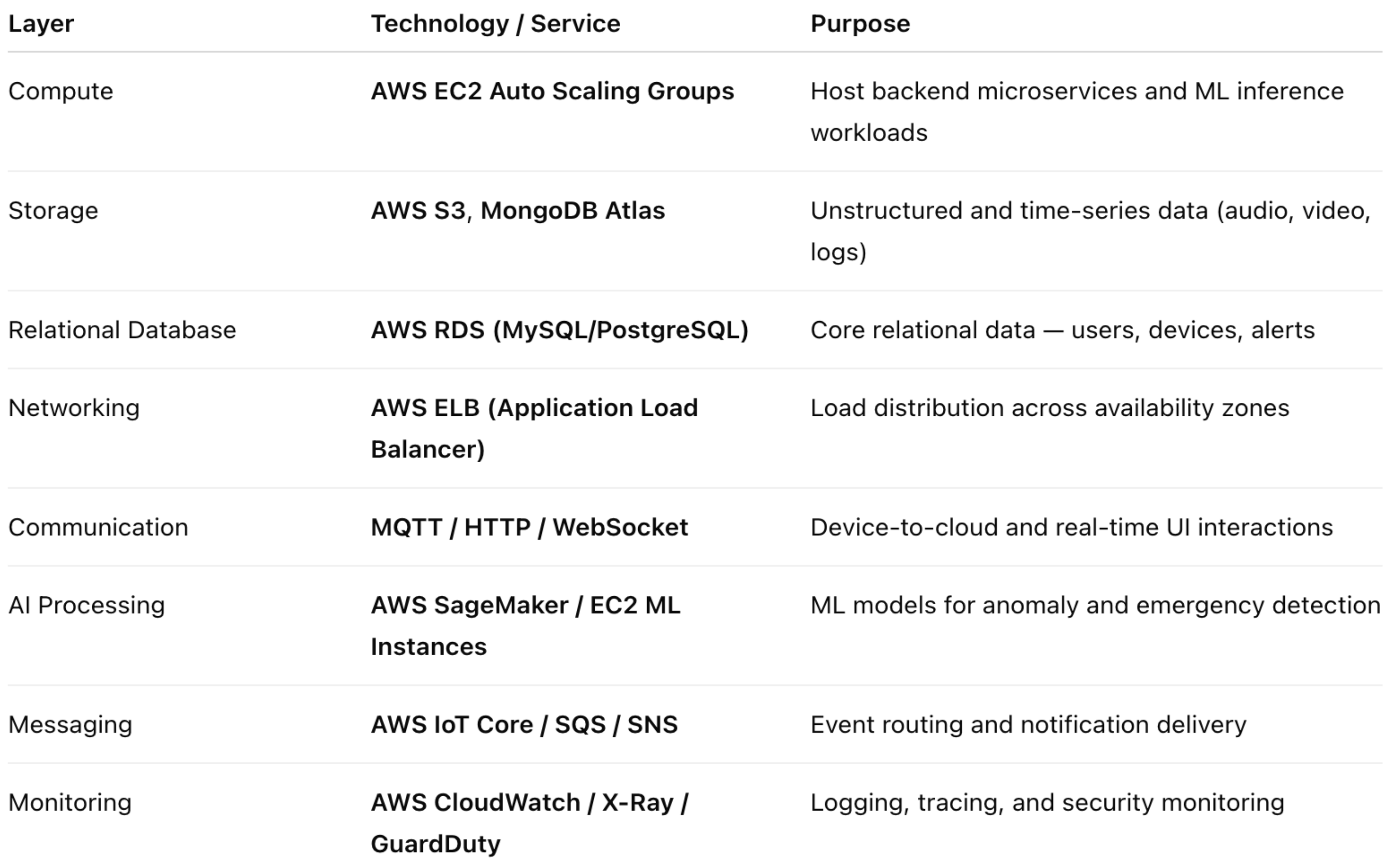
Section 3: System Infrastructure and Architecture

The **Intelligent Cloud Platform for Smart Homes** leverages AWS cloud technologies to ensure scalability, resilience, and real-time responsiveness for smart home automation, monitoring, and AI-powered alerting.

3.1 Cloud-Based System Infrastructure



Technologies Used:



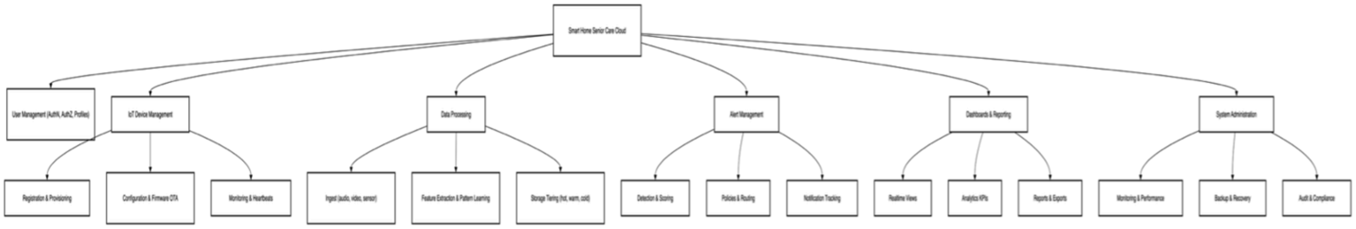
Layer	Technology / Service	Purpose
Compute	**AWS EC2 Auto Scaling Groups**	Host backend microservices and ML inference workloads
Storage	**AWS S3**, **MongoDB Atlas**	Unstructured and time-series data (audio, video, logs)
Relational Database	**AWS RDS (MySQL/PostgreSQL)**	Core relational data — users, devices, alerts
Networking	**AWS ELB (Application Load Balancer)**	Load distribution across availability zones
Communication	**MQTT / HTTP / WebSocket**	Device-to-cloud and real-time UI interactions
AI Processing	**AWS SageMaker / EC2 ML Instances**	ML models for anomaly and emergency detection
Messaging	**AWS IoT Core / SQS / SNS**	Event routing and notification delivery
Monitoring	**AWS CloudWatch / X-Ray / GuardDuty**	Logging, tracing, and security monitoring

System Infrastructure Components:

- **User Devices:** Smartphones, tablets, and desktops access the platform through a web or mobile interface.
- **IoT Devices:** Edge sensors such as smart cameras, microphones, and motion detectors capture environmental data.
- **Edge Gateway:** Local node (e.g., Raspberry Pi or Jetson Nano) aggregates and preprocesses sensor data before securely streaming to the cloud.
- **API Gateway / Load Balancer:** Manages incoming requests and routes them to appropriate backend services using HTTPS and WebSockets.
- **Microservices:** Independent service units (Dashboard, Device Manager, Alert Monitor, etc.) deployed on containerized EC2 instances.
- **Cloud Database Layer:**
  - **RDS (MySQL)** for structured relational data (users, devices, alerts).
  - **MongoDB** for large-scale time-series and unstructured sensor data.
  - **Redis** for caching and session management.
- **AI Model Services:** Hosted inference engines analyze continuous audio/video data to detect anomalies, raise alerts, and store prediction metadata.
- **Notification Subsystem:** Uses AWS SNS, SES, and WebSocket events to push alerts and updates in real time to users' dashboards.

3.2 Component-Oriented Functional Architecture

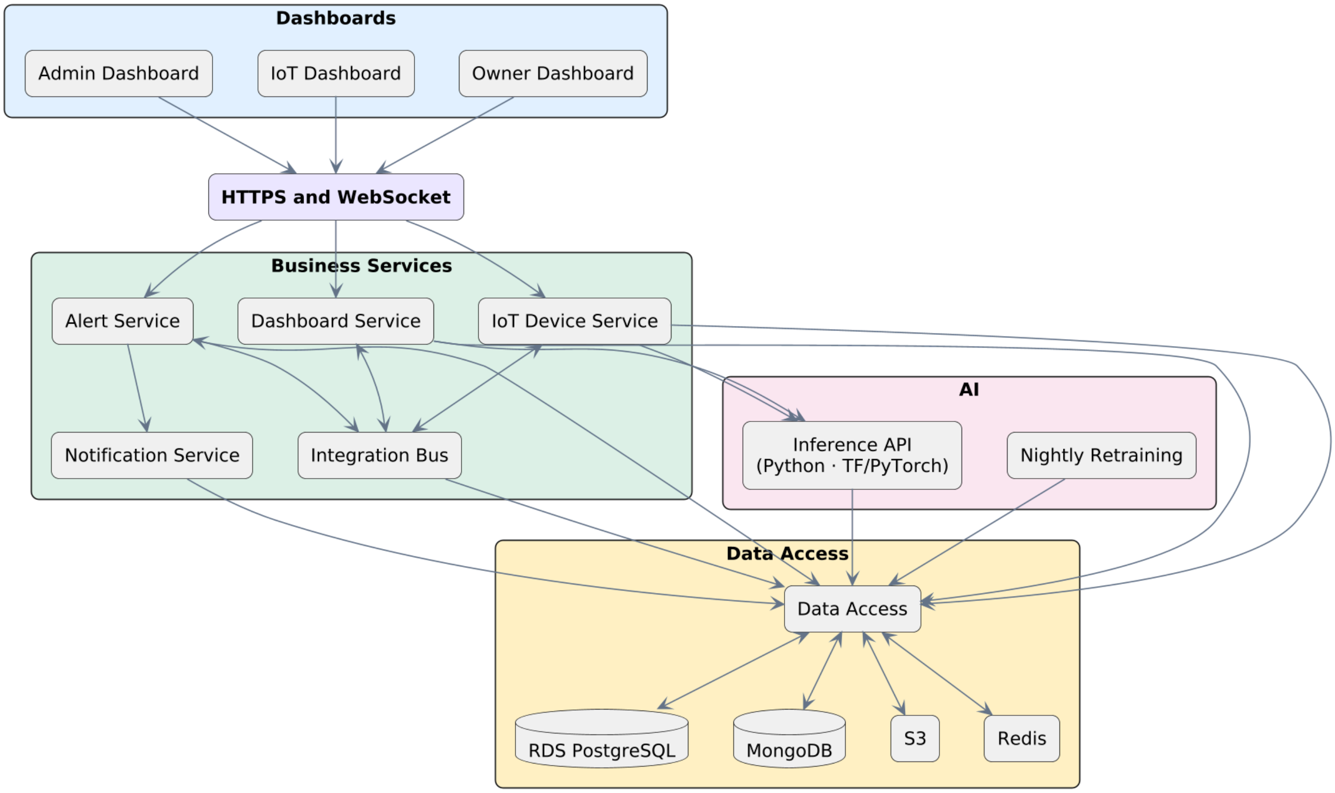
3.2.1 Function-Oriented Tree



Each major system component is modular and independently deployable:

- **System Dashboard** - Offers monitoring and control interfaces (Owner/Admin views)
- **Home-based IoT Device Manager** - Allows adding/configuring devices in each room
- **Alert Tracking & Monitoring System** - Central module for AI-based alert processing
- **Database Manager** - Handles storage, indexing, and retrieval for all data

Functional Interactions:



- The dashboard receives inputs from all other modules.
- Device Manager interfaces directly with IoT Gateway.
- Alert system fetches data from AI Models and notifies Dashboard/Users.
- All data modules communicate with the Database Manager for logging.

3.3 Cloud Design and Component Interactions

3.3.1 Scalability | Load Balance | Multi-Tenancy

Scalability:
The Intelligent Cloud Platform is designed using horizontally scalable microservices.

- **EC2 Auto Scaling Groups (ASG)** automatically adjust compute resources based on metrics such as CPU utilization (>60%), request per second (RPS >1500), and active alert count.
- **MongoDB sharding** is implemented using `{tenant_id, house_id}` as a compound shard key for parallelism and tenant-level isolation.
- **RDS (MySQL)** supports read replicas for analytics and high availability.
- **Redis** handles session caching and real-time metrics aggregation.
- **S3 lifecycle policies** manage cold data transfer from Standard → Infrequent Access → Glacier tiers.

Load Balancing:

- **AWS Application Load Balancer (ALB)** terminates TLS traffic and routes requests via path-based routing (`/api`, `/ws`, `/static`).
- Cross-zone load balancing ensures even request distribution across Availability Zones.
- Dedicated target groups are used for WebSocket connections to reduce congestion on API traffic.
- Health checks: `GET /healthz` (interval 30s, timeout 5s, threshold 2/2).

Multi-Tenancy:

- Each tenant (house or community) is assigned a unique **tenant_id**, embedded in JWT tokens and database schemas.
- RDS tables include tenant_id as a partition key; for small tenants, a **schema-per-tenant** approach may be used.
- S3 paths are isolated per tenant: `s3://intelligent-cloud/{tenant_id}/...`
- Application-level access controls enforce row-level security, ensuring strict tenant isolation.

3.3.2 Black-Box I/O View

Component	Inputs	Outputs	Data Stores	Notes
IoT Device Service	MQTT messages (`device/+/status`), REST API `/devices/{id}`	Device configuration, acknowledgment	RDS (devices), MongoDB (telemetry)	Uses AWS IoT Core for message routing
Alert Service	Audio/video inference events, SQS queue	Alert objects, WebSocket push	RDS (alerts), MongoDB (alert_events)	Idempotent alert creation and classification
Notification Service	Alert objects + routing policy	Email, SMS, push notification results	RDS (audit_log)	SNS/SES integration with DLQ retries
Dashboard Service	HTTP GET requests for KPIs	JSON responses with analytics and summaries	RDS, Redis, MongoDB	Cached results for faster dashboard loads
Inference API (AI/ML)	Audio/video payload or `s3_uri`	`{label, confidence_score}`	S3 (models/artifacts)	TensorFlow/PyTorch models hosted on EC2/SageMaker

3.3.3 High-Level APIs

Endpoint	Method	Auth Scope	Purpose	Request Example	Response (200)
`/auth/login`	POST	Public	User authentication	`{username, password}`	`{token, user}`
`/devices`	POST	Owner/IoT	Register new IoT device	`{tenant_id, house_id, type, room, mac}`	`{device_id}`
`/devices/{id}`	PUT	Owner/IoT	Update device configuration	`{config}`	`{ok: true}`
`/alerts/search`	POST	Owner/Admin	Retrieve alerts	`{filters}`	`{items, next}`
`/alerts/{id}/ack`	POST	Owner	Acknowledge alert	`{note}`	`{status:"acked"}`
`/dashboard/stats`	GET	Any	Retrieve KPI summaries	`–`	`{kpis}`
`/ml/predict`	POST	System	Submit audio/video for ML inference	`{s3_uri}`	`{label, score}`

API Notes:

- All endpoints secured by JWT-based authentication.
- Rate limiting: `30–60 requests/min per user`.
- Idempotency keys are required for POST/PUT to ensure safe retries.
- All responses include an `X-Correlation-Id` for observability tracing.

3.3.4 Network Connectivity Matrix

From	To	Protocol / Port	Direction	Purpose	Control / Security Layer
Browser / Mobile	ALB	HTTPS / 443	Inbound	UI / API access	WAF + SG-Web
ALB	API ASG	HTTP / 8080	East–West	Internal service routing	VPC SG rules
API ASG	RDS	TCP / 5432	East–West	SQL data access	SG-API → SG-DB
API ASG	MongoDB	TCP / 27017	East–West	Unstructured telemetry	SG-API → SG-Mongo
API ASG	Redis	TCP / 6379	East–West	Caching and session store	SG-API → SG-Cache
IoT Devices	AWS IoT Core	MQTT/TLS / 8883	Inbound	Device telemetry stream	IoT policy enforcement
IoT Core	SQS	AWS Private	East–West	Message queuing	IAM roles + VPC endpoints
API → Clients	Clients (Browsers, Mobiles)	WebSocket / 443	Outbound	Real-time alert push	SG-Web outbound rules

3.3.5 Observability, Security, and Compliance

Observability:

- Metrics: p95_latency, ActiveAlerts, WS_connected_clients, alert_delivery_success_rate.
- Logging: Structured JSON logs using `X-Correlation-Id` and tenant context.
- Tracing: AWS X-Ray integrated with API and ML inference microservices.
- Monitoring: CloudWatch dashboards and PagerDuty for alert notifications.

Security:

- IAM least privilege for all roles and services.
- Encryption:
  - TLS 1.3 in transit
  - AES-256 at rest for RDS/Mongo/S3
- Key Management: AWS KMS with tenant-level Customer Master Keys (CMKs).
- Secrets Management: AWS Secrets Manager with automatic rotation.
- WAF & GuardDuty:
  - WAF blocks bots, geo anomalies, and SQLi/XSS patterns.
  - GuardDuty monitors for intrusions and credential anomalies.

Compliance & Recovery:

- Audit Logging: Append-only logs in RDS audit_log table.
- Data Retention: MongoDB TTL indexes (90 days) for volatile data.
- Backup:
  - RDS snapshots every 6 hours

- Cross-region S3 replication

- Disaster Recovery: Infrastructure-as-Code (CloudFormation/Terraform) for rapid redeployment.
- Business Continuity: RTO < 4 hours, RPO < 30 minutes.

Section 4 – Cloud-based System Design and Component Interaction Design

4.1 System Database Design

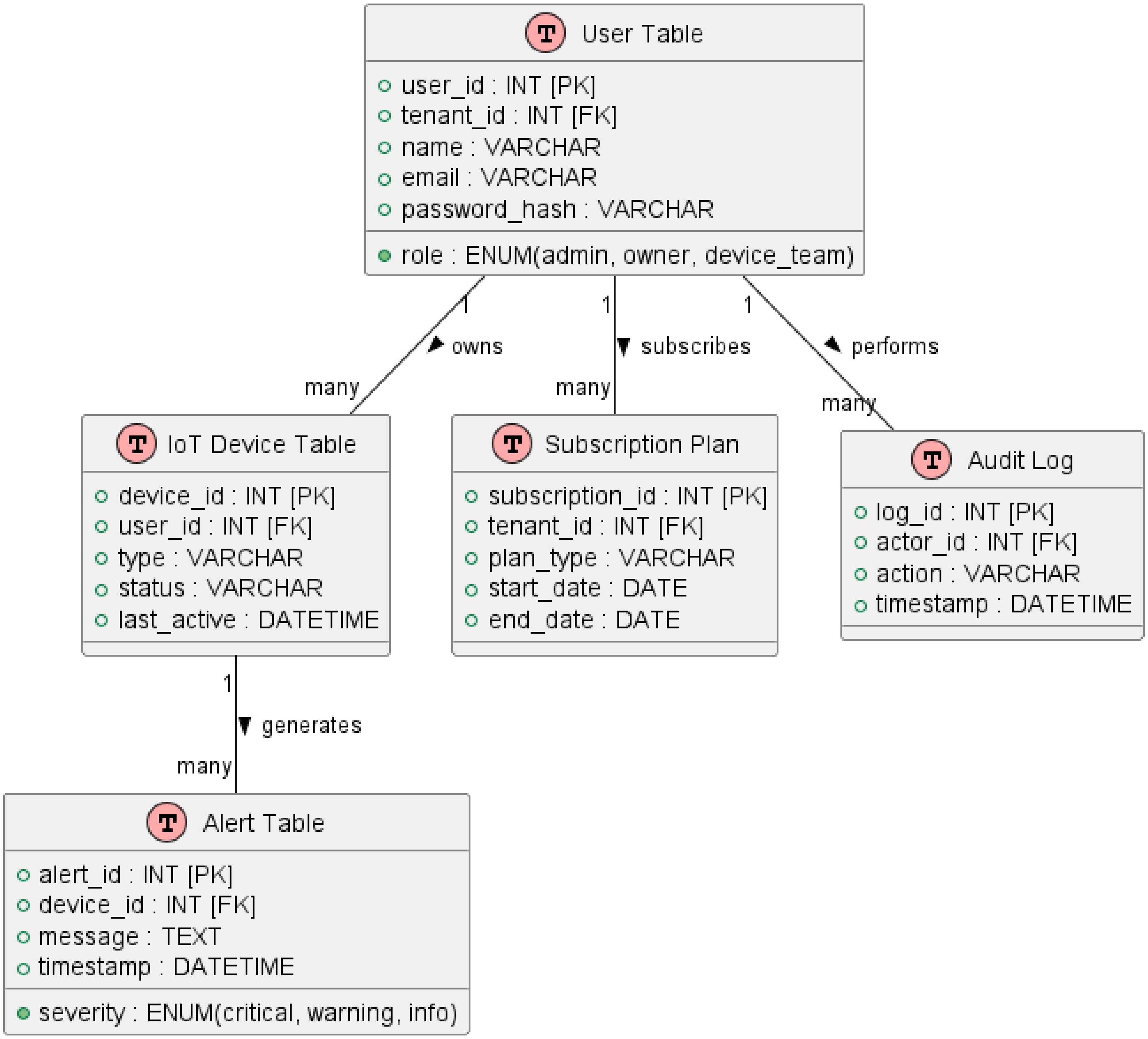
The Intelligent Cloud Platform integrates **three tiers of databases** optimized for specific workloads relational, NoSQL, and edge storage to balance performance, scalability, and data availability.

4.1.1 Relational (RDS MySQL) – ERD

- **Purpose:** Handles structured and transactional data such as user profiles, device registrations, subscription details, alerts, and audit logs.
- **Design:** Implemented in **Amazon RDS (MySQL)** with normalized tables and foreign-key relations.
- **Core Entities:**
  - User: user_id, tenant_id, role, email, password_hash
  - Device: device_id, user_id, type, status, last_active
  - Alert: alert_id, device_id, timestamp, severity, message
  - Subscription: tenant_id, plan_type, start_date, end_date
  - Audit_Log: log_id, action, actor, timestamp
- **Relationships:**
  - One user can own many devices.
  - Devices generate multiple alerts.
  - Each alert links to a tenant for multi-tenancy tracking.

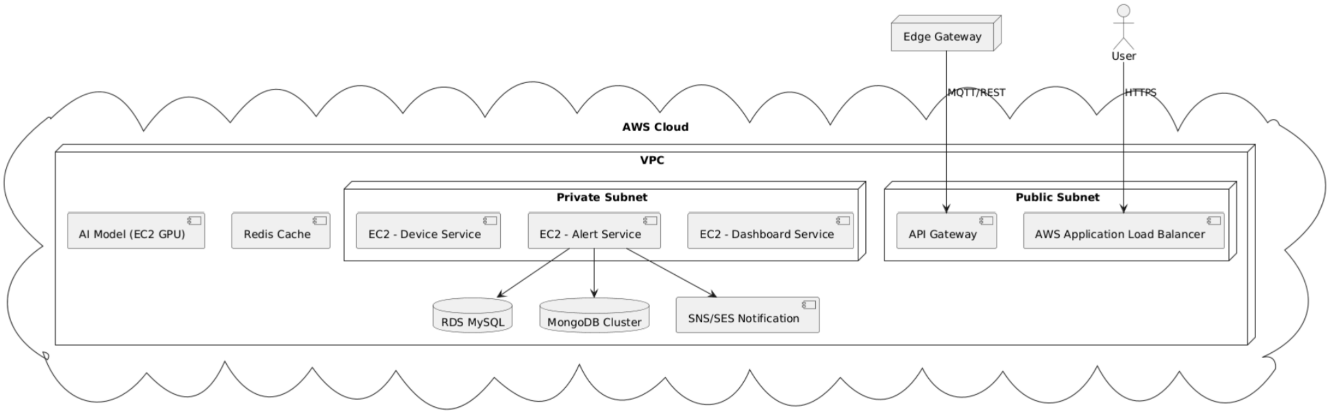
- **Features:** Referential integrity with foreign keys, read replicas for analytics, and RDS snapshots for disaster recovery.

Relational Database Design:



4.1.2 NoSQL DB – MongoDB

- **Purpose:** Stores **unstructured and time-series sensor data** (video frames, audio logs, motion values).
- **Deployment:** Hosted on AWS EC2 with sharding and replica sets.
- **Shard Key:** Compound key {tenant_id, house_id} for parallel queries and tenant isolation.
- **Collections:**
  - sensor_data: raw sensor readings and timestamps.
  - ml_predictions: AI model outputs with metadata (confidence, labels).
  - device_status: heartbeat and connectivity logs.
- **Indexes & Policies:** TTL indexes (90 days retention), JSON schema validation, and AES-256 encryption at rest.
- **Scalability:** Auto-sharding enables independent growth of tenants and regions without downtime.



4.1.3 Edge DB – SQLite

- **Purpose:** Local data cache for offline operation at the home gateway level.
- **Location:** Embedded within the Edge Gateway (e.g., Raspberry Pi or Jetson Nano).
- **Functions:**
    - Stores temporary device state and AI inference results when cloud connection is lost.
    - Performs batch sync to cloud once connectivity returns.
- **Security:** AES-encrypted DB file, signed transactions, automatic flush after upload.

Database	**Type**	**Use Case**	**Example Data**	**Storage Location**
MySQL (RDS)	Relational	User, Device, Alert metadata	User, Alert, Subscription	AWS RDS
MongoDB	NoSQL	Time-series sensor data	sensor_data, ml_predictions	EC2 Cluster
SQLite	Edge DB	Offline cache / temporary storage	Local state, logs	Edge Gateway

4.2 Communication Design

4.2.1 Backend-Frontend Architecture

- **Frontend:** React/Next.js web dashboard with role-based views for homeowners, cloud staff, and IoT teams.
- **Backend:** Node.js and Python microservices running on containerized EC2 instances behind AWS Application Load Balancer (ALB).
- **Communication Flow:**
    - HTTPS requests → API Gateway → Backend Service → Database Layer.
    - Real-time updates delivered via WebSocket channels.

- **Security:** JWT authentication, CORS policies, and IAM role-based access control.

4.2.2 REST APIs

All functional interactions are exposed as RESTful APIs with versioning (/api/v1/).

Core Endpoints:

Service	Endpoint	Method	Description
User Management	/api/users/register	POST	Create new account
Authentication	/api/login	POST	Generate JWT token
Device Manager	/api/devices/:id	GET/PUT	View or update device
Alerts	/api/alerts	GET	Fetch alerts for tenant/user
AI Predictions	/api/predictions	GET	Fetch recent model outputs

API Rules:

- Rate limit 30–60 requests/min per user.
- POST/PUT requests require idempotency keys.
- Responses include X-Correlation-Id for traceability.
- WebSocket support for real-time alert delivery.

4.2.3 End-to-End Interactions

1. **Data Collection:** IoT sensors capture audio/video data.
2. **Edge Processing:** Edge Gateway performs compression and metadata tagging.
3. **Cloud Ingestion:** Data streamed to AWS API Gateway → AI Model Service.
4. **AI Inference:** Anomaly detection and alert generation.
5. **Database Storage:** Results logged in RDS and MongoDB.
6. **Notification:** Alerts pushed to users via SNS/SES/WebSocket.
7. **Dashboard Display:** UI updates in real time with alert details.

4.3 High-Level Cloud Computing Design

4.3.1 Load Balancing

- Implemented using **AWS ALB** with cross-zone routing and health checks (GET /health, interval 30 s, timeout 5 s).
- Separate target groups for REST and WebSocket traffic.
- Ensures uniform distribution across availability zones and rapid failover.

4.3.2 Scalability

- **Auto Scaling Groups (ASG):** Scale up when CPU > 60 % or RPS > 1500.

- **Sharded MongoDB** supports tenant-level parallel processing.
- **RDS read replicas** offload analytical queries.
- **Redis cache** for session and real-time metrics aggregation.
- **S3 Lifecycle Policies** automatically move cold data to Glacier for cost optimization.

4.3.3 High-Level APIs

Provide central integration points for microservices:

- **/api/devices** – IoT device registration and management.
- **/api/alerts** – Alert retrieval and acknowledgement.
- **/api/analytics** – Historical and predictive analytics.
- **/api/admin** – Tenant management and system monitoring.

All APIs use JWT security, support CORS, and return structured JSON responses with standardized status codes.

Section 5 – System Dashboard UI Design (GUI and Flows)

The **System Dashboard UI** provides an intuitive and role-based interface for users to monitor, configure, and manage all smart home and cloud components in real time. It is the central hub that bridges the backend microservices, AI alert systems, and IoT device managers to end users through a secure, responsive, and user-friendly design.

5.1 System User Dashboard Architecture and Roles

The dashboard is **multi-tenant and role-based**, offering customized experiences for three key user groups:

1. Homeowners

- Can view the status of connected IoT devices (cameras, sensors, lights, locks).
- Receive live notifications or alerts about emergencies (motion detection, smoke alarm, intrusion).
- Manage home configurations, schedules, and energy preferences.
- Access AI-generated reports and insights on activity trends.

2. IoT Device Team

- Add or remove edge devices remotely through the dashboard.
- View firmware versions, uptime, and connection stability.
- Run diagnostic checks and push firmware or configuration updates.
- Access sensor health metrics and real-time data streams.

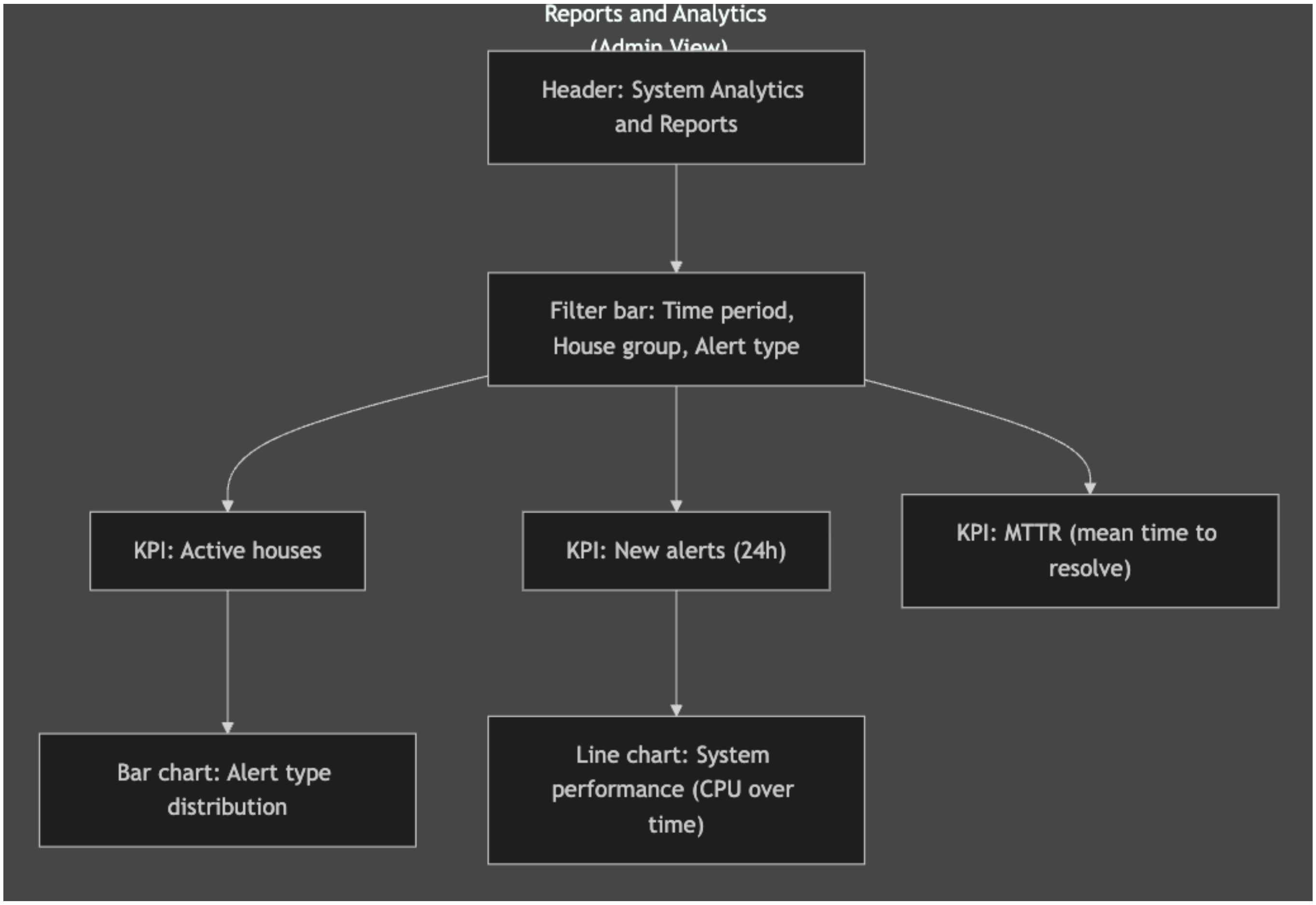
3. Cloud Admins / Service Staff

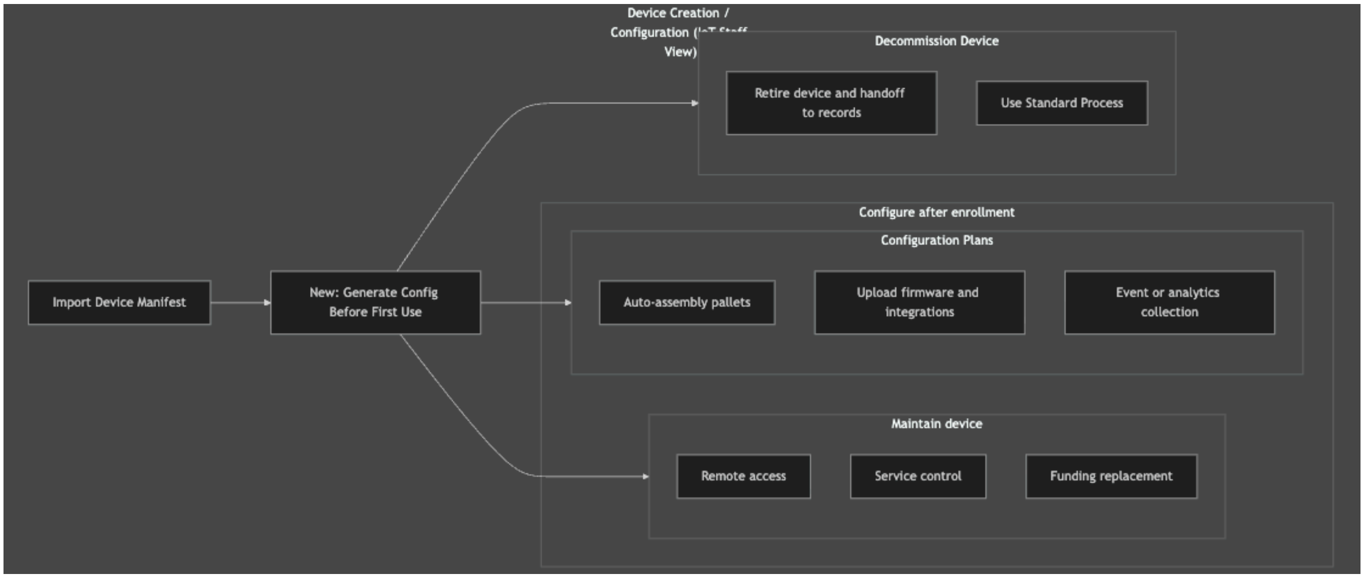
- Oversee entire platform operations across tenants.

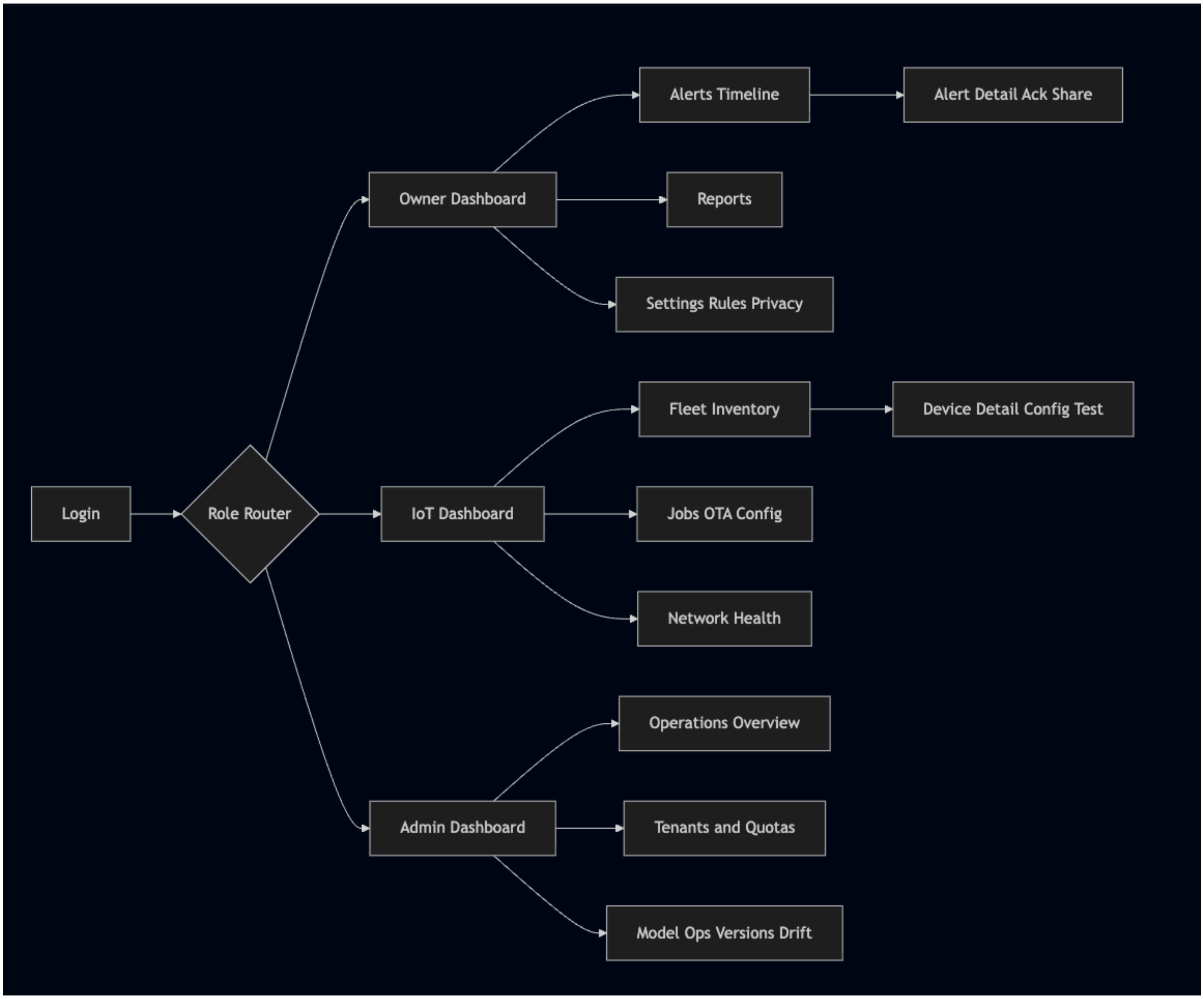
- Monitor AI inference status, alert queue load, and service uptime.
- Manage system logs, tenant onboarding, and policy configurations.
- Use analytics dashboards (integrated with AWS CloudWatch) to visualize system metrics such as latency, throughput, and active device count.

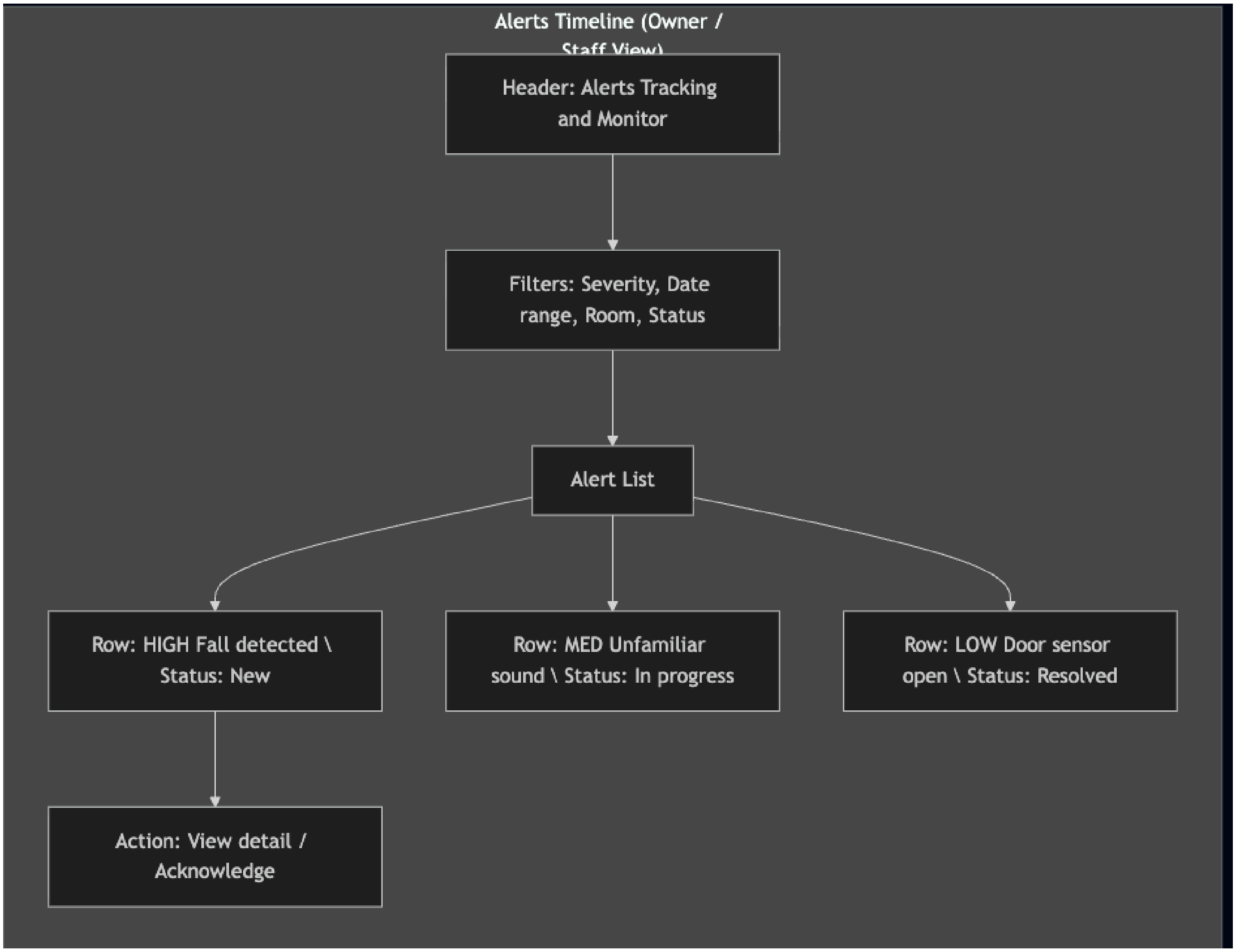
Architecture Overview:

- **Frontend:** Built with ReactJS/Next.js and Tailwind for responsive layouts.
- **Backend Integration:** Communicates via REST and WebSocket APIs.
- **Authentication:** Auth0 Single Sign-On (SSO) and role-based JWT tokens.
- **Visualization:** Charts and heatmaps powered by Recharts and D3.js.
- **Deployment:** Hosted on AWS EC2 with CloudFront CDN for fast content delivery.


Reports and Analytics
(Admin View)
Header: System Analytics and Reports
Filter bar: Time period, House group, Alert type
KPI: Active houses
KPI: New alerts (24h)
KPI: MTTR (mean time to resolve)
Bar chart: Alert type distribution
Line chart: System performance (CPU over time)


Device Creation / Configuration (IoT Staff View)
Decommission Device
Retire device and handoff to records
Use Standard Process
Configure after enrollment
Configuration Plans
Import Device Manifest
New: Generate Config Before First Use
Auto-assembly pallets
Upload firmware and integrations
Event or analytics collection
Maintain device
Remote access
Service control
Funding replacement


Login
Role Router
Owner Dashboard
Alerts Timeline
Alert Detail Ack Share
Reports
Settings Rules Privacy
IoT Dashboard
Fleet Inventory
Device Detail Config Test
Jobs OTA Config
Network Health
Admin Dashboard
Operations Overview
Tenants and Quotas
Model Ops Versions Drift


Alerts Timeline (Owner / Staff View)
Header: Alerts Tracking and Monitor
Filters: Severity, Date range, Room, Status
Alert List
Row: HIGH Fall detected \ Status: New
Row: MED Unfamiliar sound \ Status: In progress
Row: LOW Door sensor open \ Status: Resolved
Action: View detail / Acknowledge

5.2 UI Operations Behavior Designs

The dashboard design ensures **clear interaction flow**, **contextual visibility**, and **real-time feedback** for user actions. It follows **Material UI design principles**, emphasizing accessibility and consistency across roles.

5.2.1 Navigation Flow

Main Navigation Structure:

1. **Login / Authentication Page**
   - Auth0 SSO login → redirects based on user role (Admin / Owner / Device Team).
2. **Home / Overview Page**
   - Displays personalized dashboard summary.
   - Quick stats: “Active Devices,” “Pending Alerts,” “AI Status,” “System Health.”

3. **Device Management Page**
    - Add new IoT devices or modify existing ones.
    - Each device card displays ID, type, uptime, and health.
4. **Alert Center**
    - Real-time alert feed with severity filters (Critical / Warning / Info).
    - Acknowledge or mute alerts directly.
5. **Analytics / Reports**
    - View AI-detected event patterns.
    - Graphs for activity heatmaps, alert frequency, and system uptime.
6. **Settings / Preferences**
    - Update user details, language, and notification preferences.
7. **Logout**
    - Session terminated securely; redirect to login.

Behavioral Flow Example:

- **Event Trigger:** Motion detected by an IoT sensor.
- **Edge Gateway:** Sends event → Cloud API → AI engine confirms anomaly.
- **Dashboard Action:** Displays “Intrusion Detected” alert in real-time → user notified via email/SMS/WebSocket → user acknowledges alert → system logs response.

5.2.2 UI Designs

Key UI Components:

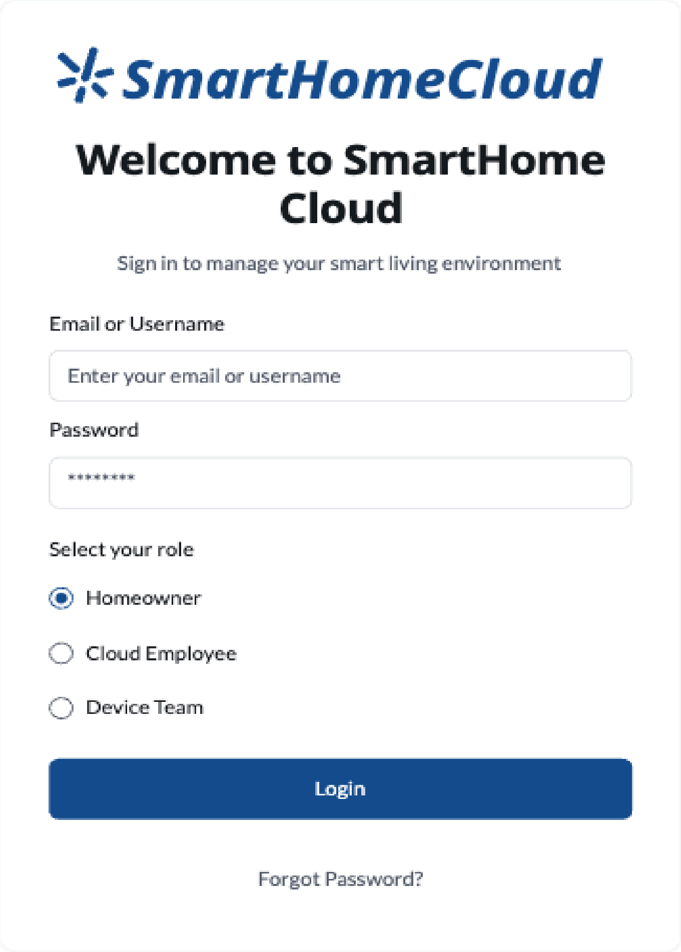
Component	**Description**	**Example Elements**
Header Bar	Global navigation and user status	Logo, role info, notification bell
Side Panel	Context-aware module menu	Dashboard, Devices, Alerts, Reports
Dashboard Cards	Quick view widgets	“Active Alerts,” “Device Uptime,” “Last AI Scan”
Data Tables	Display device and user info	Sortable and searchable tables
Real-Time Charts	Live system analytics	CPU utilization, latency graph
Modal Dialogs	Confirmations and pop-ups	Device deletion, alert acknowledgment

Alert Feed	Priority color-coded alerts	Red (Critical), Orange (Warning), Green (Info)

Design Principles:

- Minimalist and responsive layout for mobile, tablet, and desktop.
- Color-coded components for alert visualization (severity mapping).
- WCAG-compliant for accessibility.
- Consistent iconography (using Lucide and Material Icons).

Login Page



SmartHomeCloud

Welcome to SmartHome Cloud

Sign in to manage your smart living environment

Email or Username

Enter your email or username

Password

****

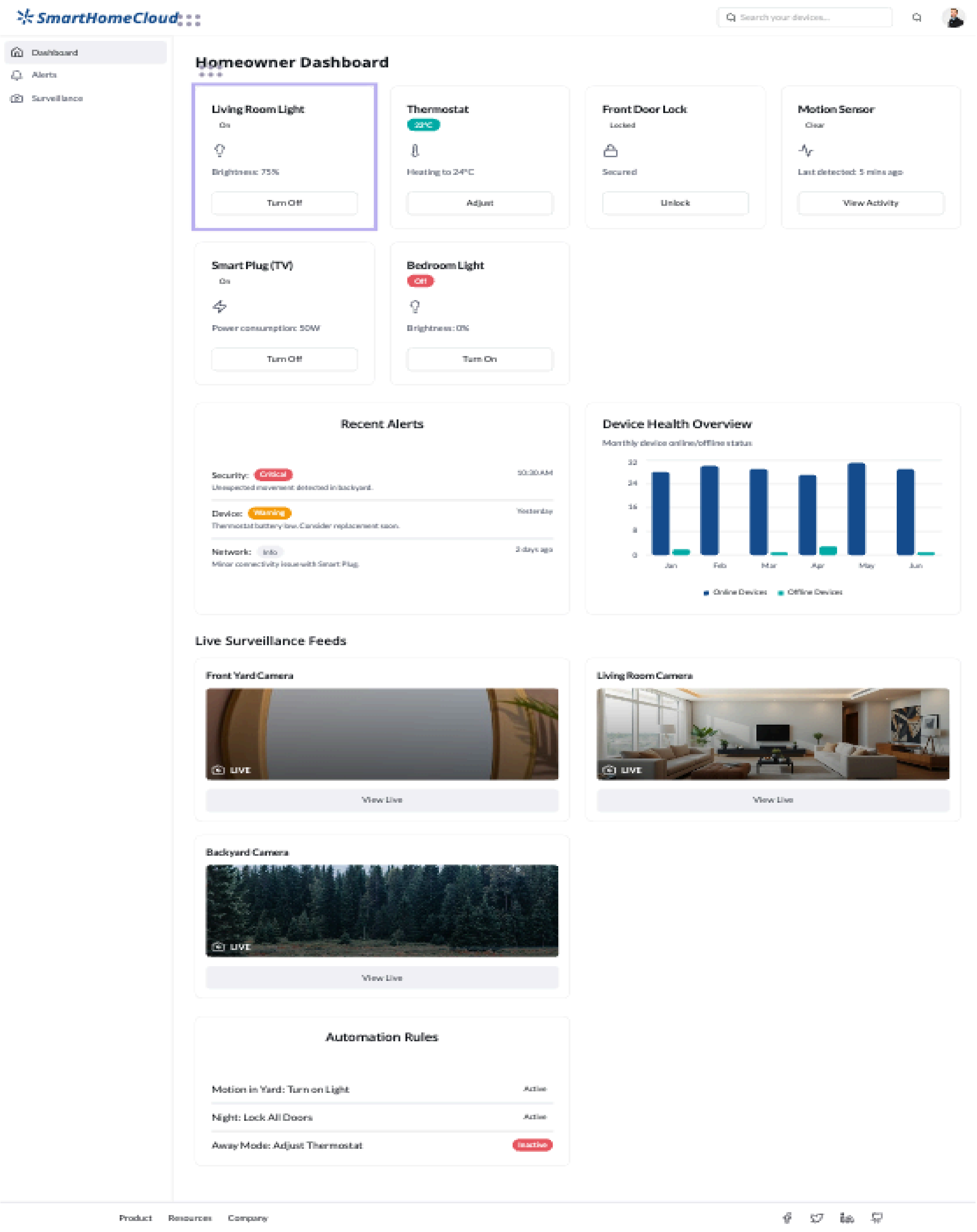
Select your role

● Homeowner

○ Cloud Employee

○ Device Team

Home-Owner Dashboard:



SmartHomeCloud

Search your devices...

- Dashboard
- Alerts
- Surveillance

Homeowner Dashboard

Living Room Light
On
Brightness: 75%
Turn Off

Thermostat
22°C
Heating to 24°C
Adjust

Front Door Lock
Locked
Secured
Unlock

Motion Sensor
Clear
Last detected: 5 mins ago
View Activity

Smart Plug (TV)
On
Power consumption: 50W
Turn Off

Bedroom Light
Off
Brightness: 0%
Turn On

Recent Alerts

Security: Critical — 10:30 AM
Unexpected movement detected in backyard.

Device: Warning — Yesterday
Thermostat battery low. Consider replacement soon.

Network: Info — 2 days ago
Minor connectivity issue with Smart Plug.

Device Health Overview
Monthly device online/offline status

Live Surveillance Feeds

Front Yard Camera
LIVE
View Live

Living Room Camera
LIVE
View Live

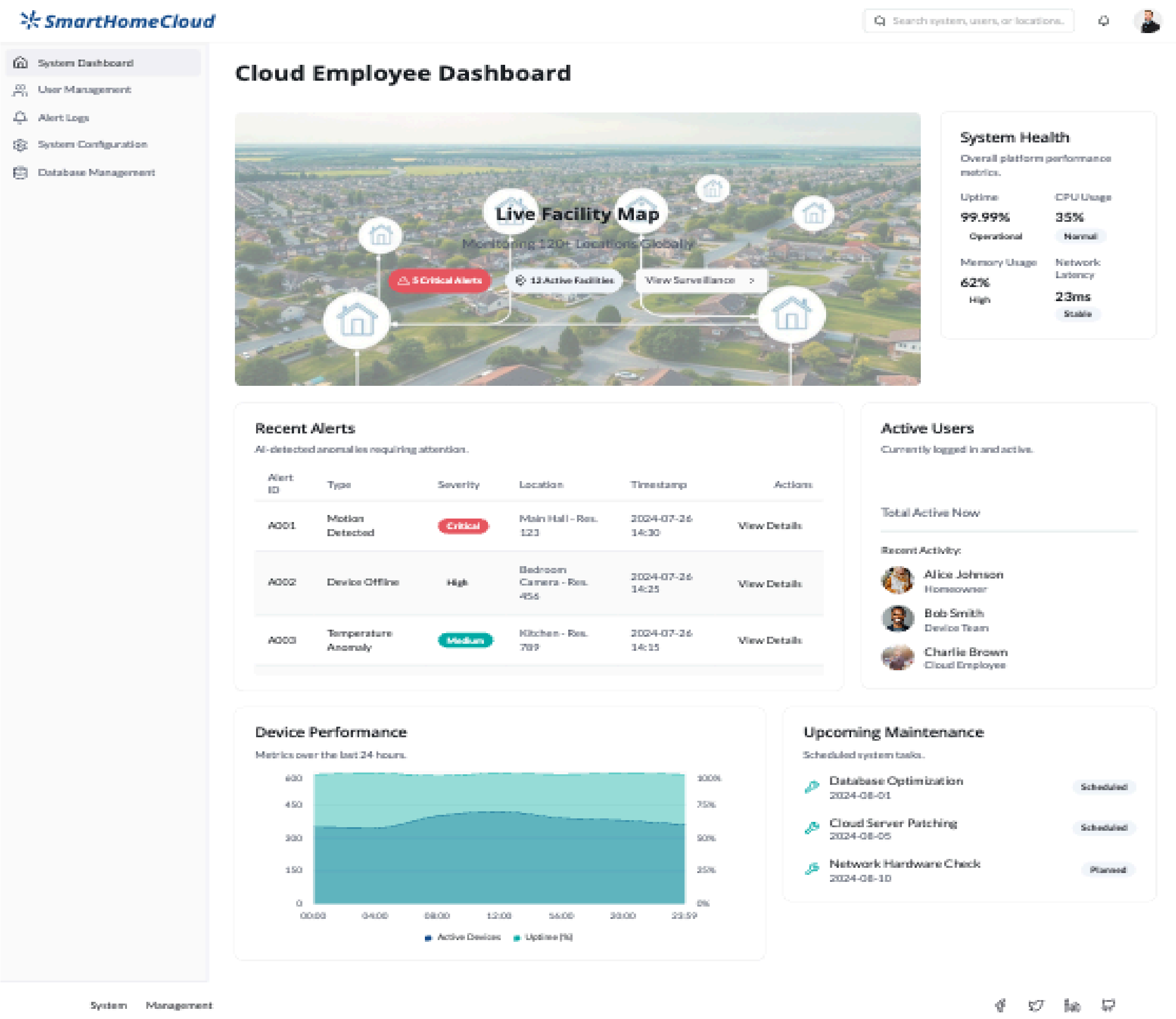
Backyard Camera
LIVE
View Live

Automation Rules

Motion in Yard: Turn on Light	Active
Night: Lock All Doors	Active
Away Mode: Adjust Thermostat	Inactive

Product Resources Company

Cloud Employee Dashboard:



Cloud Employee Dashboard

Live Facility Map

Monitoring 120+ Locations Globally

System Health

Overall platform performance metrics.

Uptime 99.99% Operational	CPU Usage 35% Normal
Memory Usage 62% High	Network Latency 23ms Stable

Recent Alerts

AI-detected anomalies requiring attention.

Alert ID	Type	Severity	Location	Timestamp	Actions
A001	Motion Detected	Critical	Main Hall - Res. 123	2024-07-26 14:30	View Details
A002	Device Offline	High	Bedroom Camera - Res. 456	2024-07-26 14:25	View Details
A003	Temperature Anomaly	Medium	Kitchen - Res. 789	2024-07-26 14:15	View Details

Active Users

Currently logged in and active.

Total Active Now

Recent Activity:

Alice Johnson	Homeowner
Bob Smith	Device Team
Charlie Brown	Cloud Employee

Device Performance

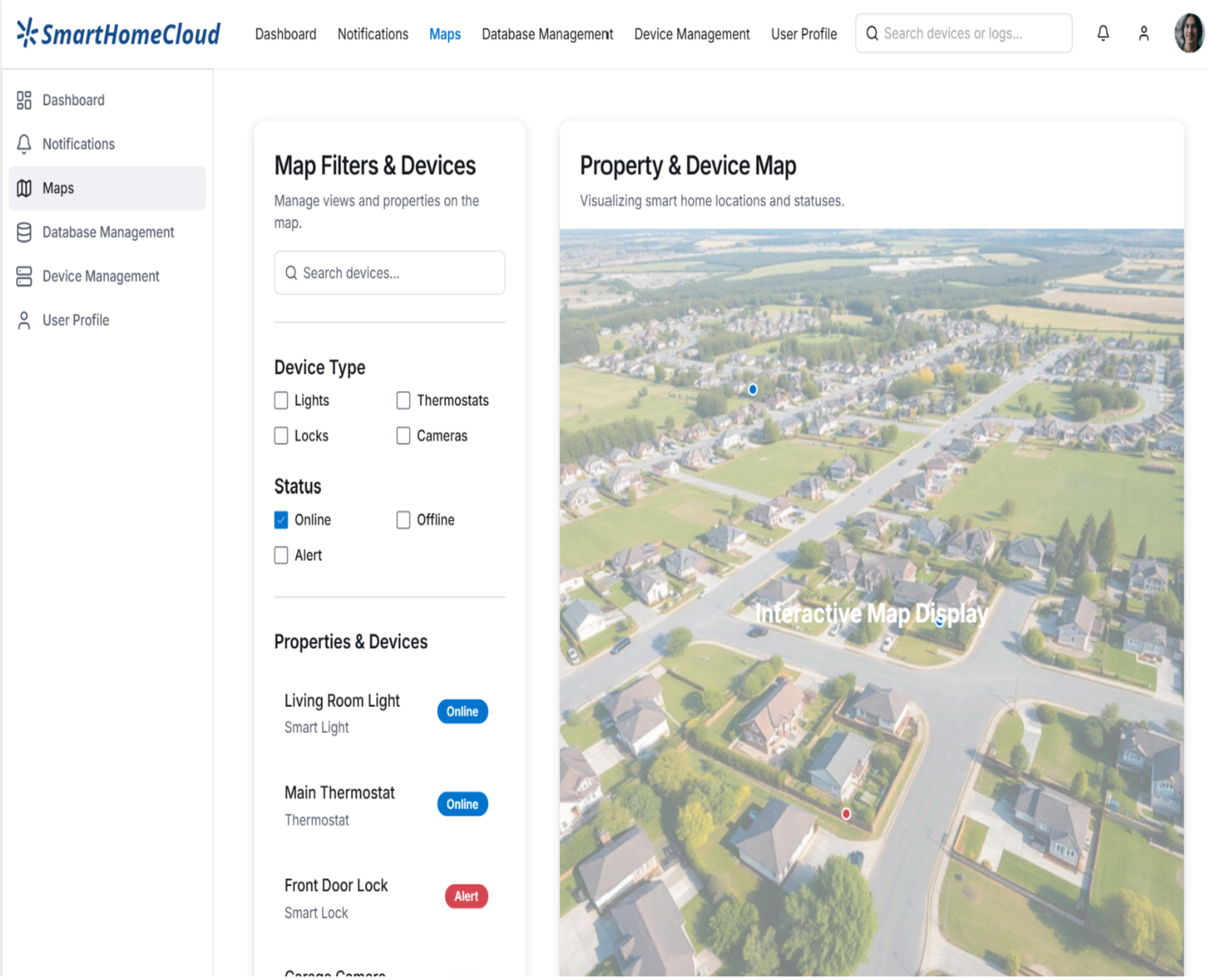
Metrics over the last 24 hours.

Upcoming Maintenance

Scheduled system tasks.

Database Optimization	2024-08-01	Scheduled
Cloud Server Patching	2024-08-05	Scheduled
Network Hardware Check	2024-08-10	Planned

Maps:



SmartHomeCloud

Dashboard Notifications Maps Database Management Device Management User Profile

Search devices or logs...

- Dashboard
- Notifications
- Maps
- Database Management
- Device Management
- User Profile

Map Filters & Devices

Manage views and properties on the map.

Search devices...

Device Type

□ Lights □ Thermostats
□ Locks □ Cameras

Status

☑ Online □ Offline
□ Alert

Properties & Devices

Living Room Light
Smart Light
Online

Main Thermostat
Thermostat
Online

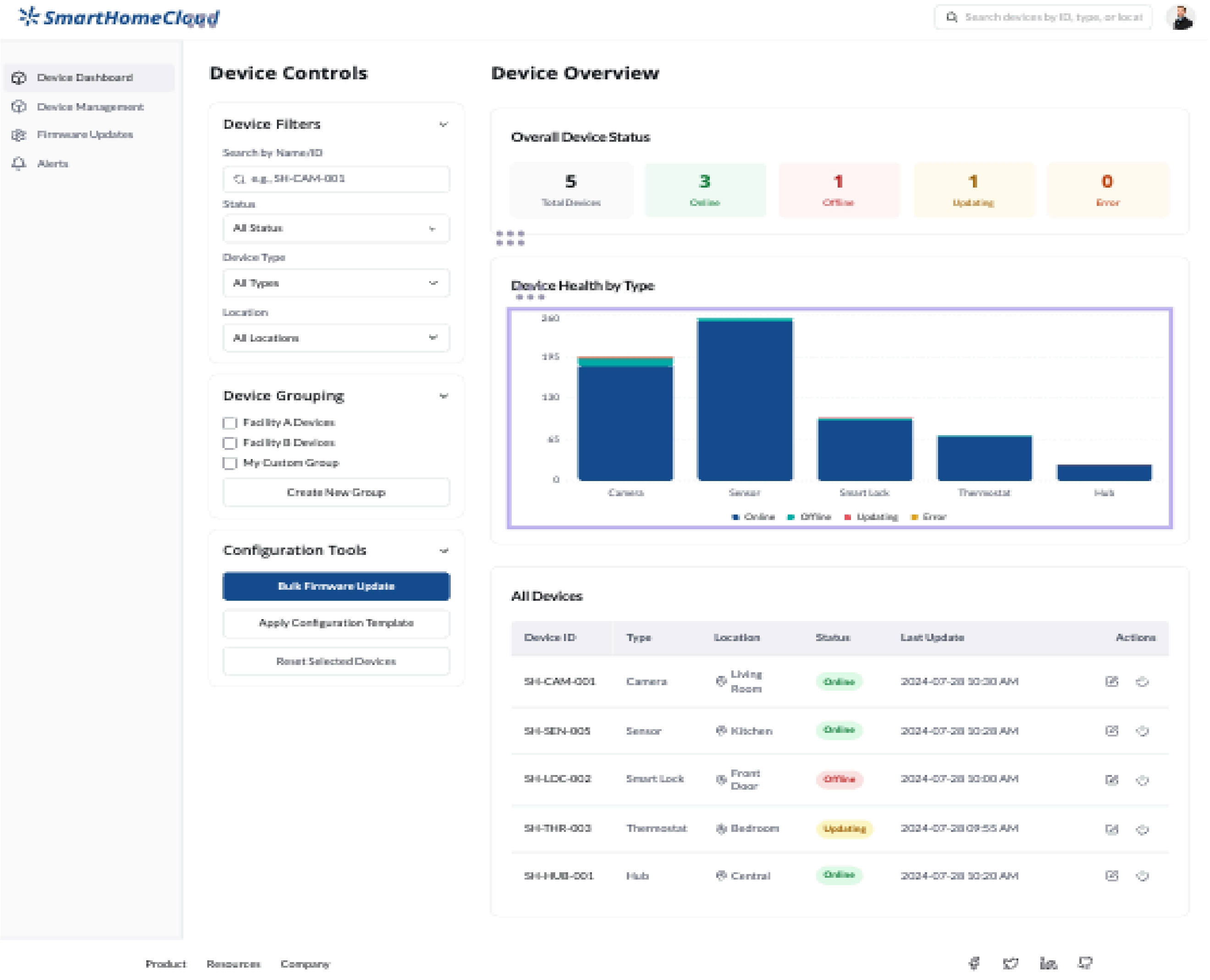
Front Door Lock
Smart Lock
Alert

Garage Camera

Property & Device Map

Visualizing smart home locations and statuses.

Device Controls and Overview:



Device Controls

Device Filters

Search by Name/ID: e.g., SH-CAM-001

Status: All Status

Device Type: All Types

Location: All Locations

Device Grouping

- □ Facility A Devices
- □ Facility B Devices
- □ My Custom Group

Configuration Tools

Device Overview

Overall Device Status

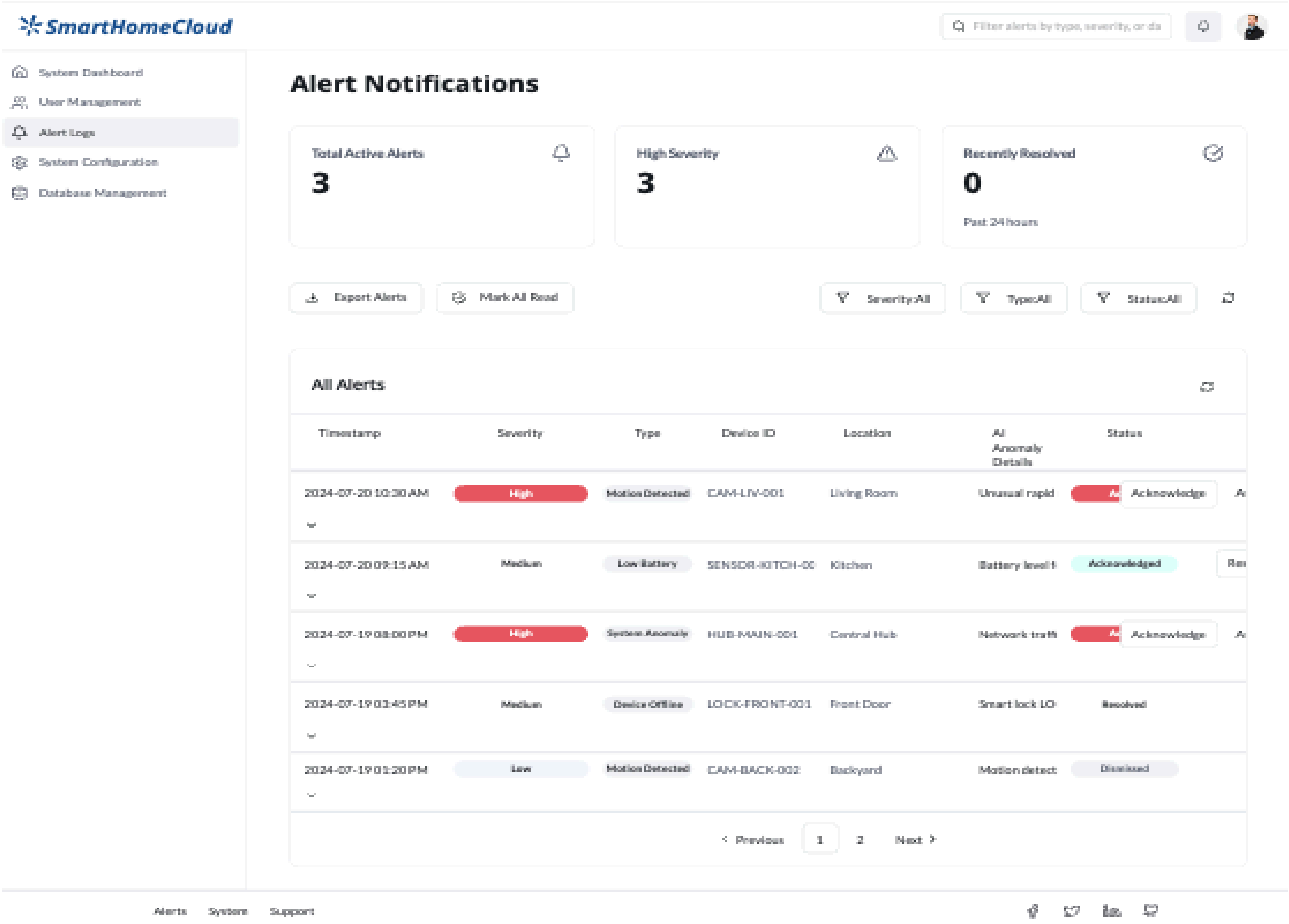
5	3	1	1	0
Total Devices	Online	Offline	Updating	Error

Device Health by Type

All Devices

Device ID	Type	Location	Status	Last Update	Actions
SH-CAM-001	Camera	Living Room	Online	2024-07-28 10:30 AM	
SH-SEN-005	Sensor	Kitchen	Online	2024-07-28 10:28 AM	
SH-LOC-002	Smart Lock	Front Door	Offline	2024-07-28 10:00 AM	
SH-THR-003	Thermostat	Bedroom	Updating	2024-07-28 09:55 AM	
SH-HUB-001	Hub	Central	Online	2024-07-28 10:20 AM	

Alert Notifications:



Alert Notifications

Total Active Alerts
3

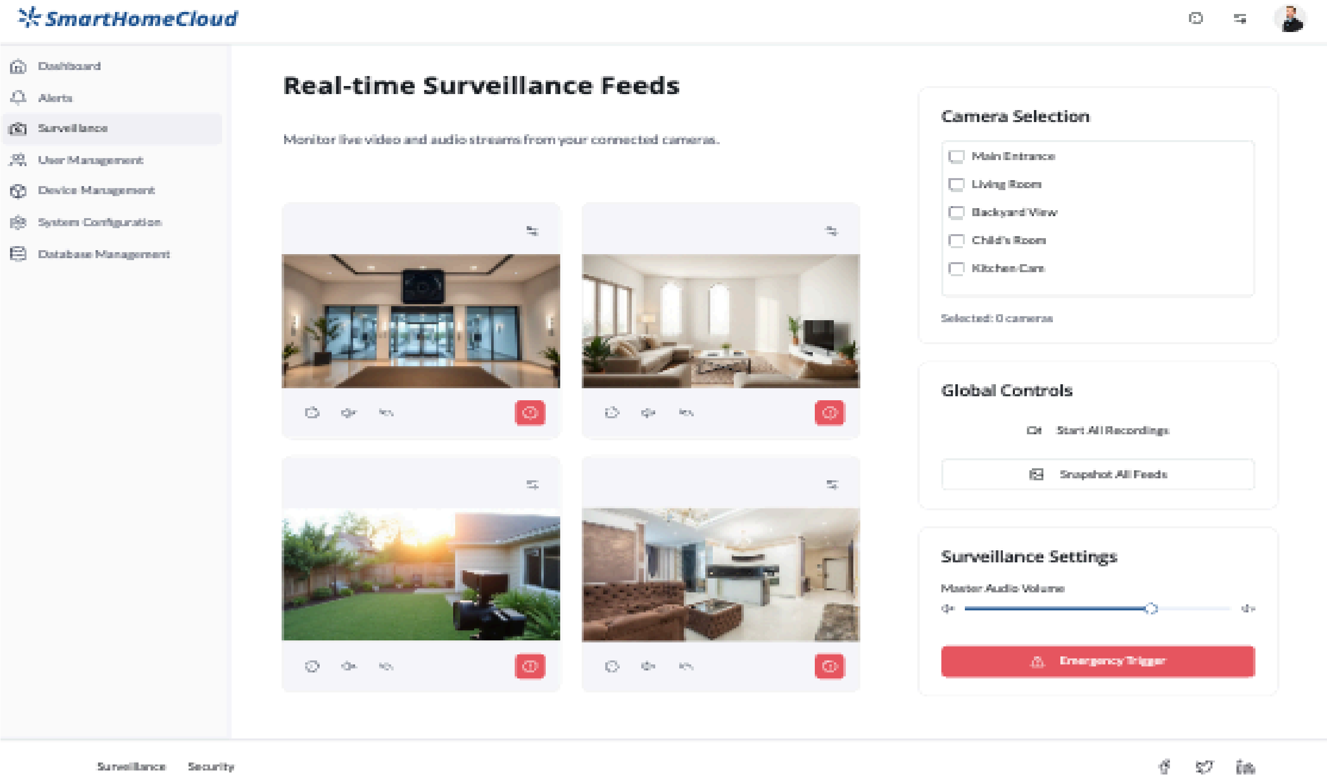
High Severity
3

Recently Resolved
0
Past 24 hours

All Alerts

Timestamp	Severity	Type	Device ID	Location	AI Anomaly Details	Status	
2024-07-20 10:30 AM	High	Motion Detected	CAM-LIV-001	Living Room	Unusual rapid	A Acknowledge	A
2024-07-20 09:15 AM	Medium	Low Battery	SENSOR-KITCH-00	Kitchen	Battery level i	Acknowledged	Re
2024-07-19 08:00 PM	High	System Anomaly	HUB-MAIN-001	Central Hub	Network traffi	A Acknowledge	A
2024-07-19 03:45 PM	Medium	Device Offline	LOCK-FRONT-001	Front Door	Smart lock LO	Resolved	
2024-07-19 01:20 PM	Low	Motion Detected	CAM-BACK-002	Backyard	Motion detect	Dismissed	

Real-time Surveillance Feeds:



Real-time Surveillance Feeds

Monitor live video and audio streams from your connected cameras.

Camera Selection

- □ Main Entrance
- □ Living Room
- □ Backyard View
- □ Child's Room
- □ Kitchen Cam

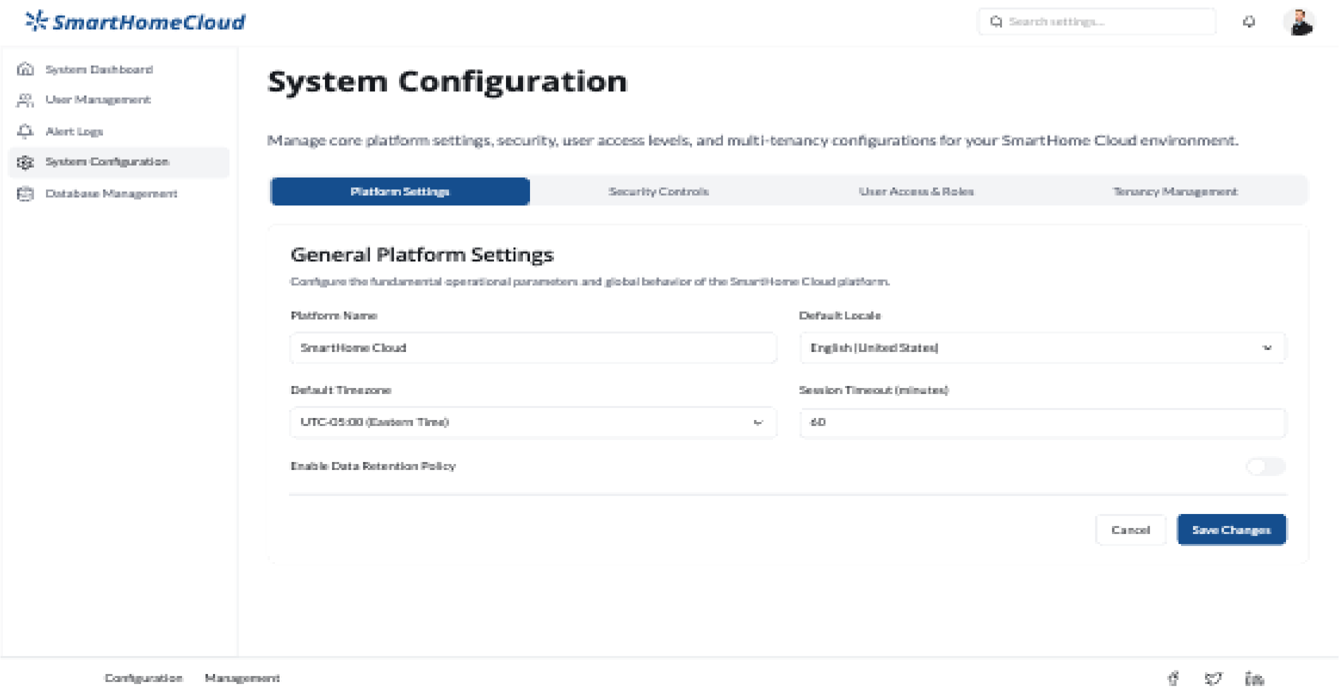
Selected: 0 cameras

Global Controls

Surveillance Settings

Master Audio Volume

System Configuration:



System Configuration

Manage core platform settings, security, user access levels, and multi-tenancy configurations for your SmartHome Cloud environment.

General Platform Settings

Configure the fundamental operational parameters and global behavior of the SmartHome Cloud platform.

Platform Name: SmartHome Cloud

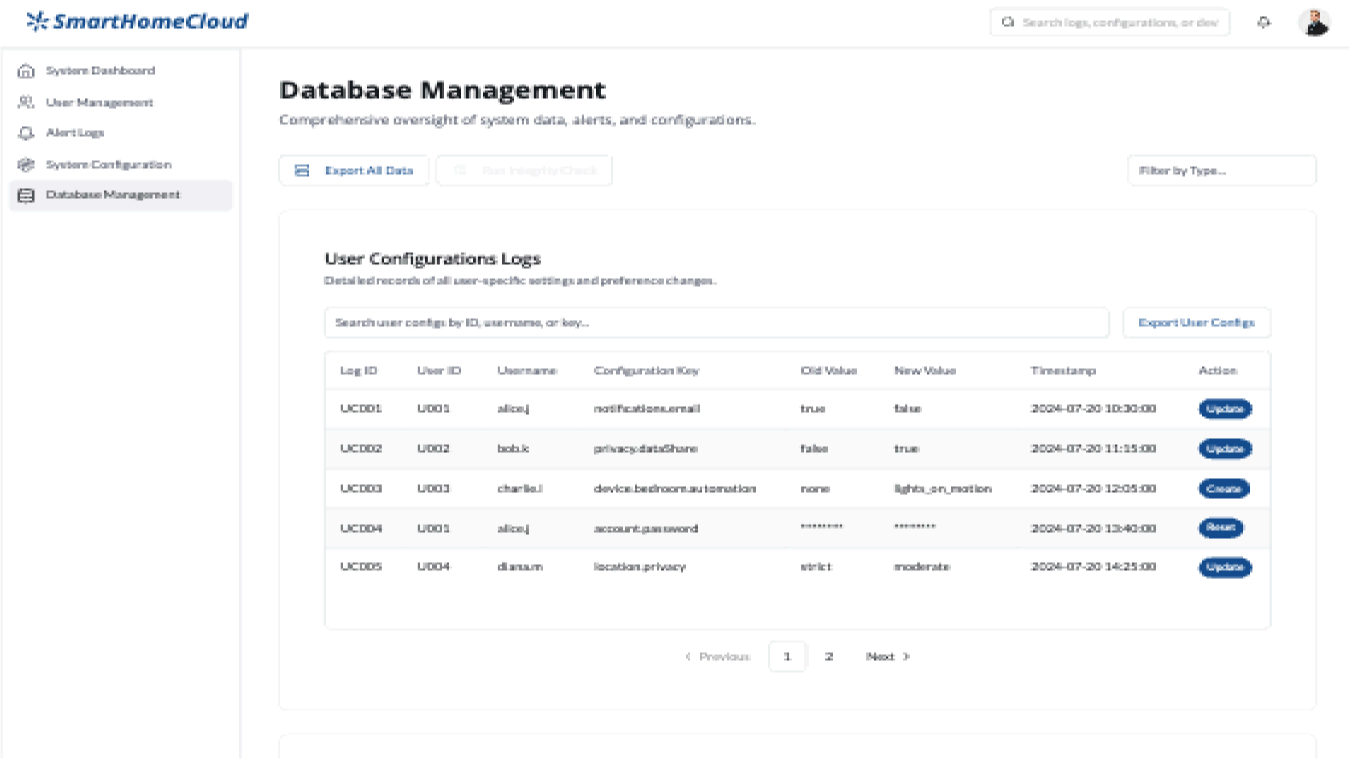
Default Locale: English (United States)

Default Timezone: UTC-05:00 (Eastern Time)

Session Timeout (minutes): 60

Enable Data Retention Policy

Database Management:



Database Management

Comprehensive oversight of system data, alerts, and configurations.

User Configurations Logs

Detailed records of all user-specific settings and preference changes.

Log ID	User ID	Username	Configuration Key	Old Value	New Value	Timestamp	Action
UC001	U001	alice.j	notifications.email	true	false	2024-07-20 10:30:00	Update
UC002	U002	bob.k	privacy.dataShare	false	true	2024-07-20 11:15:00	Update
UC003	U003	charlie.l	device.bedroom.automation	none	lights_on_motion	2024-07-20 12:05:00	Create
UC004	U001	alice.j	account.password	********	********	2024-07-20 13:40:00	Reset
UC005	U004	diana.m	location.privacy	strict	moderate	2024-07-20 14:25:00	Update

5.3 Project Plan and Schedule

Phase	**Task Description**	**Team Members**	**Duration**	**Deliverables**
Phase 1	Requirements Gathering & UI Mockups	All	Week 1–2	Wireframes, role-based UI sketches
Phase 2	Frontend and API Integration	Nikhil, Alekhya	Week 3–4	Functional Dashboard Prototype
Phase 3	Backend Integration (Alert & Device Services)	Vijaya, Gowtham	Week 4–5	Real-time data binding & alert streaming
Phase 4	Testing and UX Optimization	All	Week 6	Responsive UI & usability tests
Phase 5	Deployment and Final Presentation	All	Week 7	Deployed EC2 Dashboard, Demo Video