CLIENT VOLUME 3

KESHI ECTS

ECTS Technical Overview

Client-safe technical reference describing the architectural structure, runtime boundaries, security model, reporting and realtime flows, and deployment shape of the platform. It is suitable for architecture review without exposing developer-only internals.

AUDIENCE
Client IT, architects, governance

FOCUS
System design and runtime behavior

VERSION
1.0 | March 10, 2026

Architecture Security Reporting Realtime Deployment

1. Technical Scope

This guide explains the platform from an architectural and operational design perspective. It stays above source code level and focuses on the structures and flows that matter during client technical review.

Covered in this volume

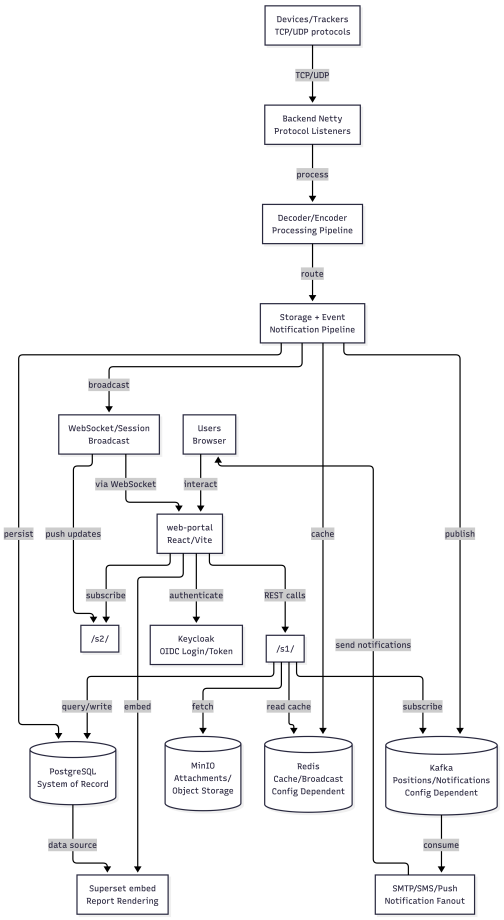
- System context and component boundaries
- Authentication and authorization model
- Portal, backend, storage, and reporting interactions
- Realtime and telemetry-oriented flow patterns
- Deployment shape and supporting infrastructure

Deliberately excluded

- Repository walkthroughs and class-level implementation
- Frontend or backend local setup
- Private extension points and engineering maintenance internals

2. System Context and Core Components

ECTS is centered on a secured web portal backed by business services, identity controls, data stores, and supporting notification, telemetry, and reporting capabilities.



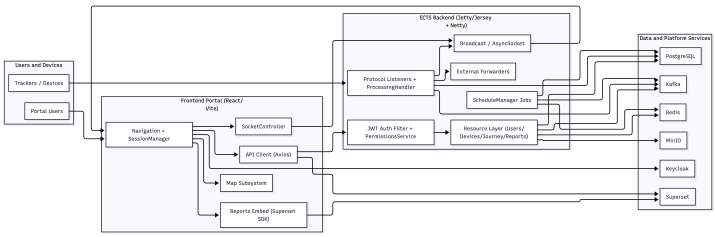
High-level system context connecting end users, the portal, platform services, and runtime support layers.

Main component groups

- **User-facing portal:** operational interface for maps, lists, detail pages, and reports.
- **Business service layer:** workflow rules, validation, orchestration, and data access.
- **Identity services:** authentication, session, and token issuance.
- **Persistence layer:** transactional records, configuration, documents, and historical data.
- **Operational data propagation:** alerts, telemetry, notifications, and analytics views.

3. Runtime Architecture

The runtime architecture separates user interaction, service processing, storage, and supporting infrastructure so each concern can be controlled, monitored, and secured independently.



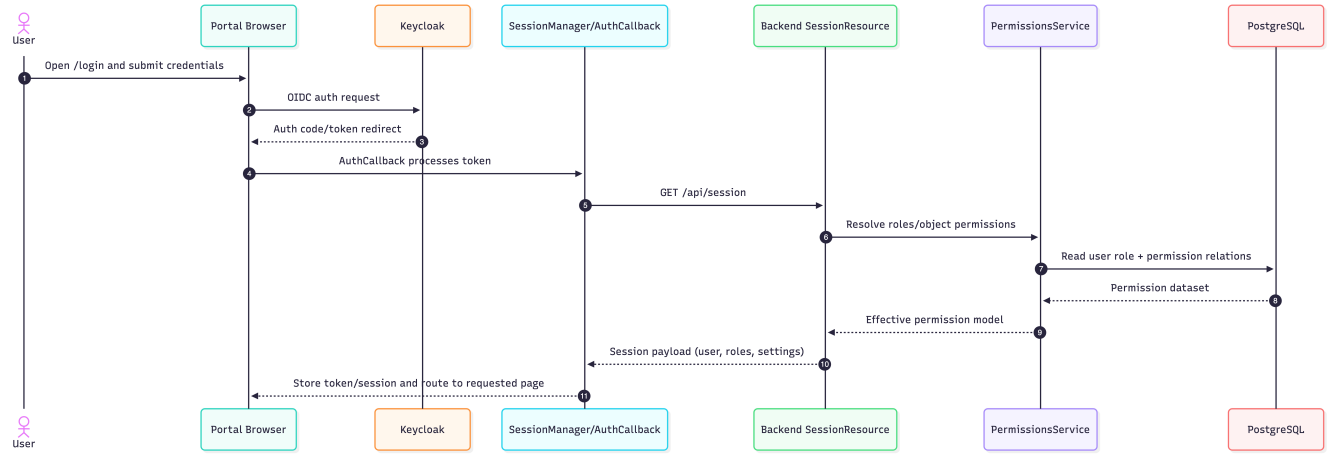
Logical runtime component view showing interaction boundaries and responsibilities.

Why this separation matters

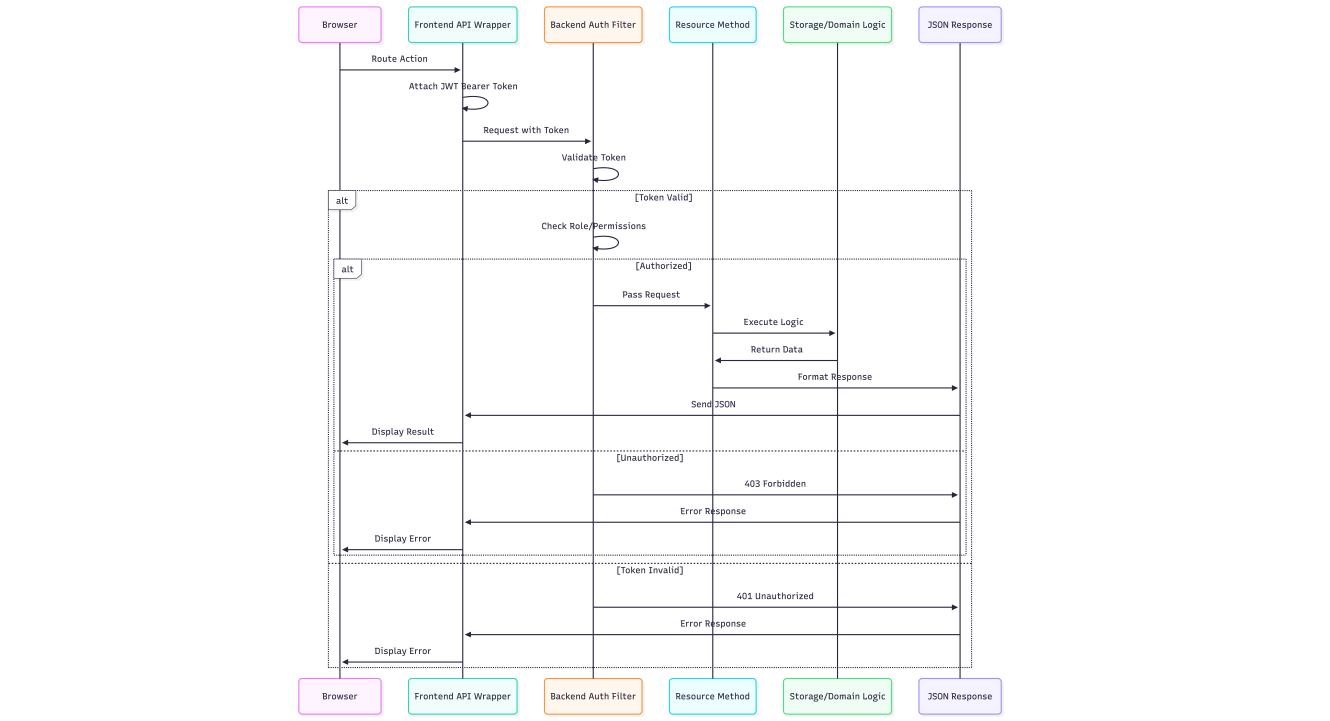
- Portal failures and backend failures can be analyzed independently.
- Identity and analytics paths can be monitored as separate risk areas.
- Operational scaling and recovery decisions can be targeted at the affected layer.
- Security controls can be enforced at both the user boundary and service boundary.

4. Security and Access Flow

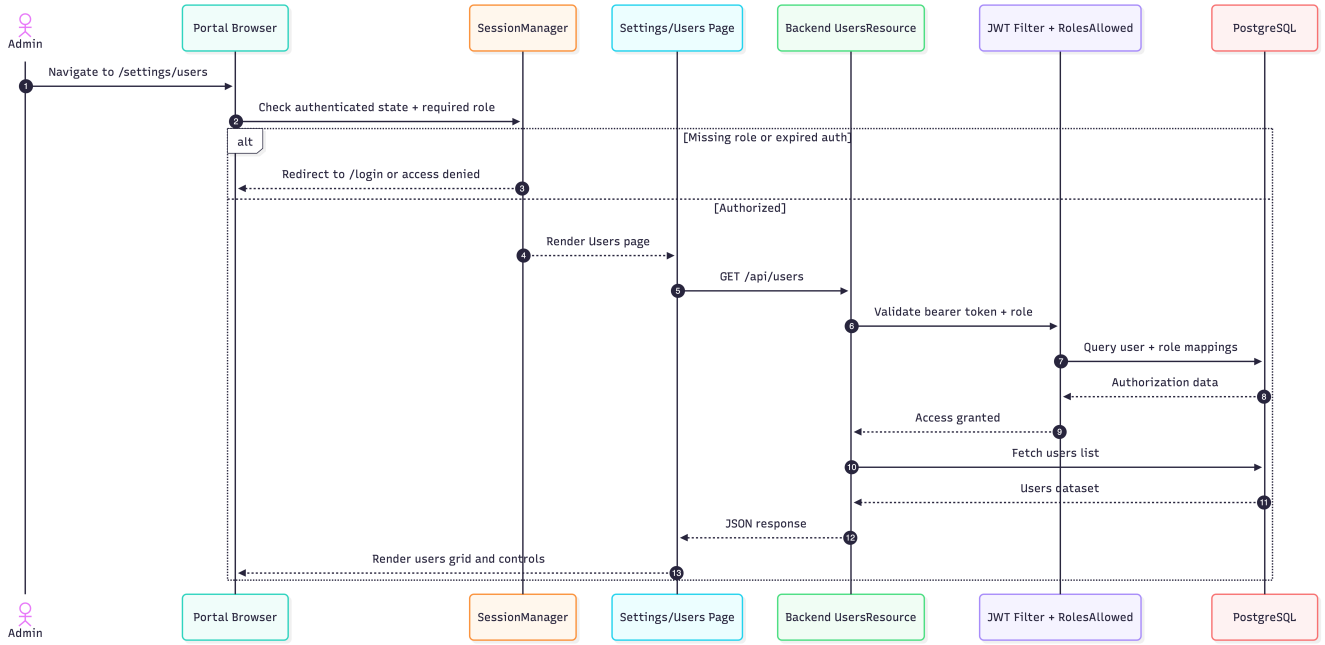
Secure access is based on centralized authentication followed by backend-side authorization enforcement. The portal does not treat page visibility alone as sufficient security; protected access is validated downstream as well.



User authentication and authenticated session establishment.



Token issuance, propagation, and validation during service access.



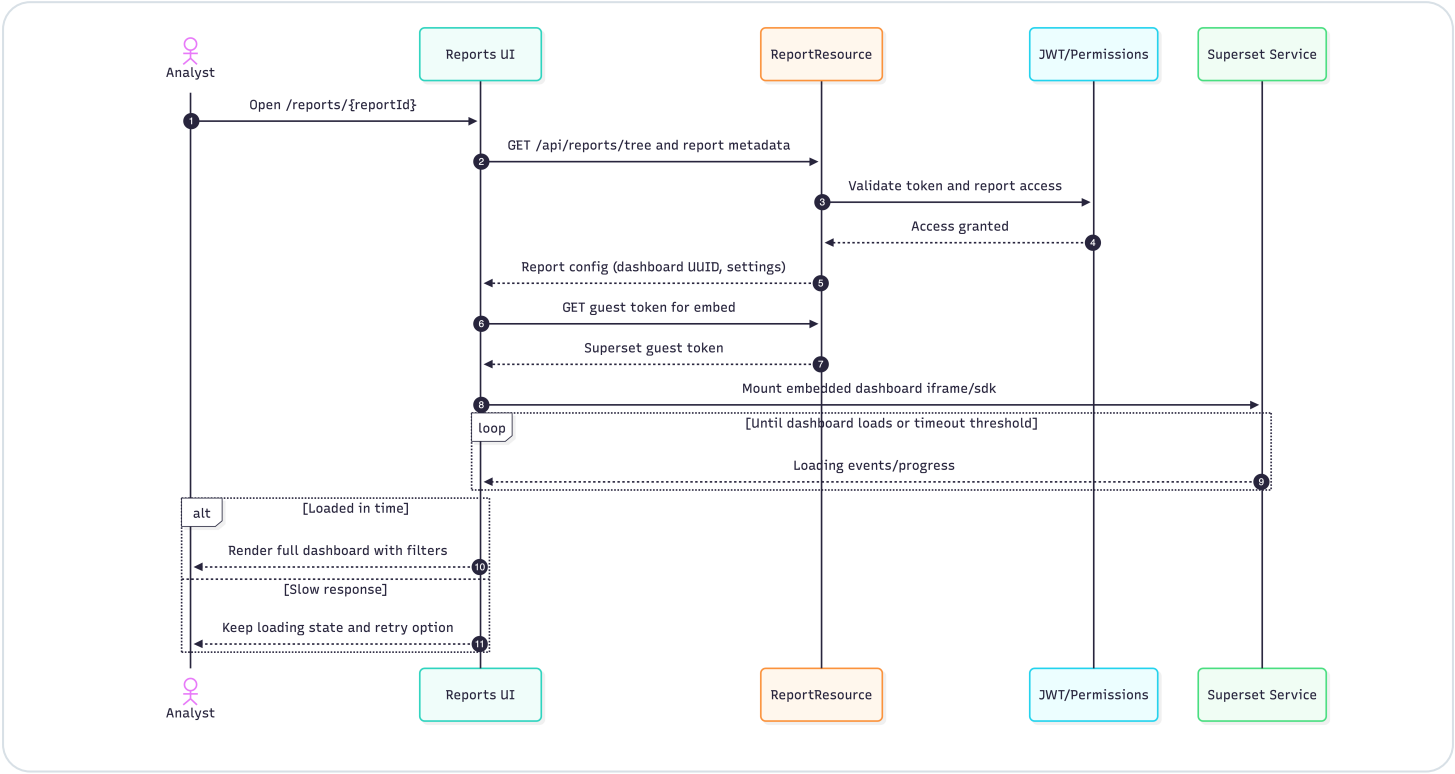
Access and administrative controls around user governance.

Security design principles

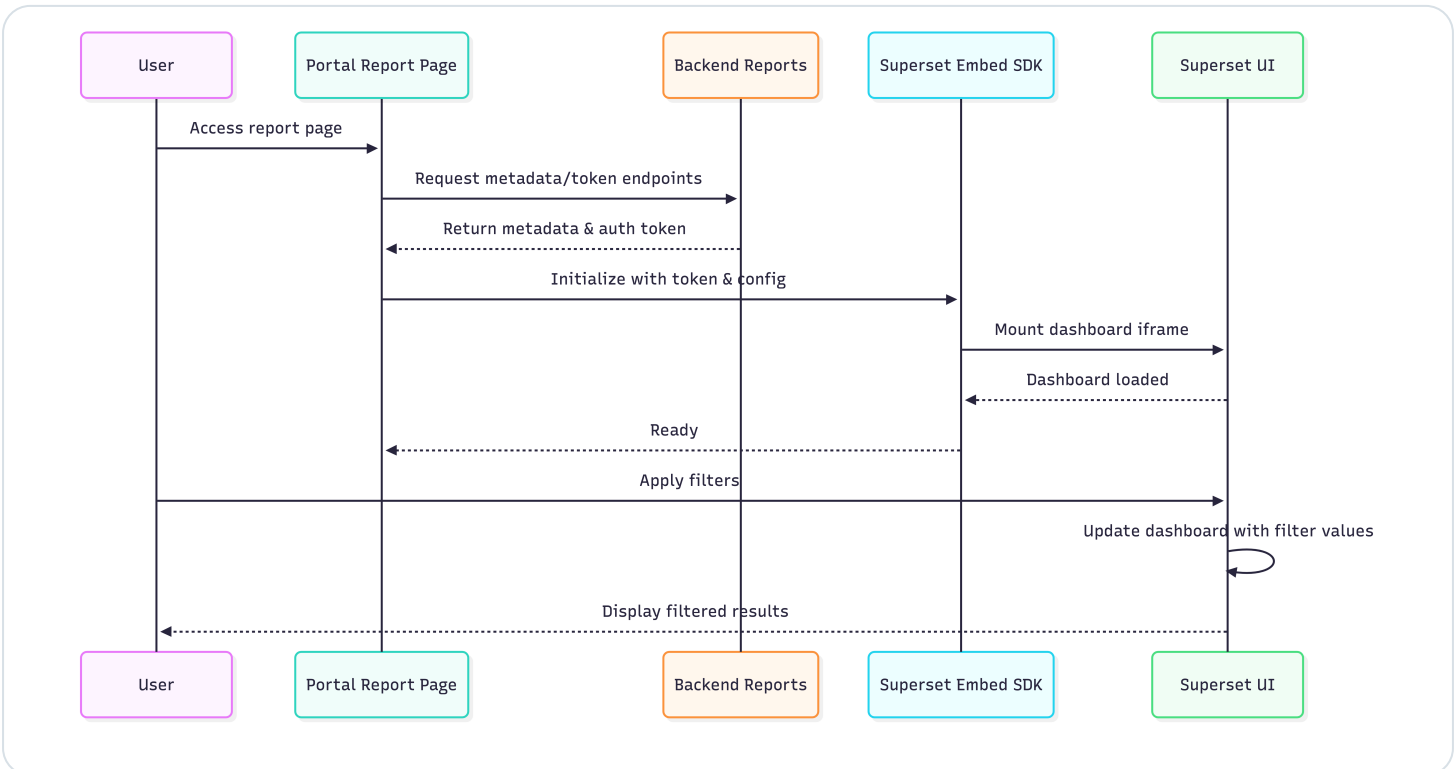
- Centralized identity establishes the session.
- Backend authorization constrains data visibility and actions.
- Protected architecture content and private documentation paths are gated.
- Operational roles determine usable workflows and administrative scope.

5. Reporting and Realtime Flows

Two flows deserve special attention during client review: reporting, because it follows a distinct authorization and rendering path; and realtime telemetry, because it turns backend events into live operational views.

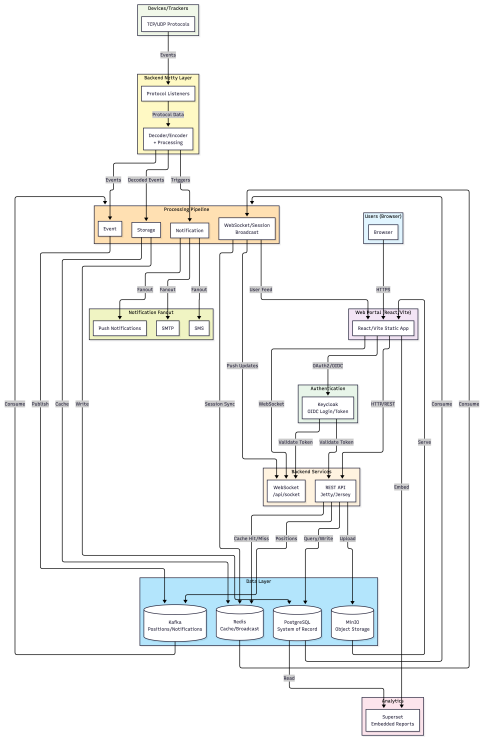


Authorization and token exchange pattern for accessing reporting content.



Portal-side report metadata lookup and embedded analytics rendering flow.

Device telemetry and message fan-out into the operational experience.

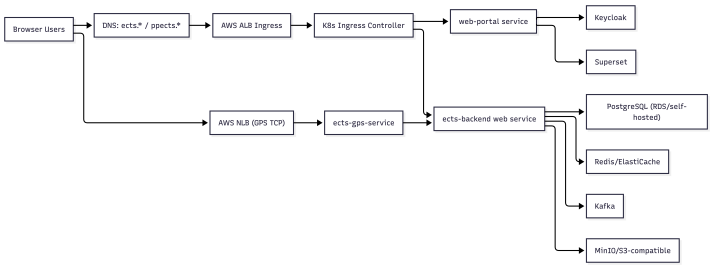


Notification propagation into the portal experience.

These two flows explain why reports may behave differently from standard screens and why backend-side event reliability directly shapes what end users see in live monitoring views.

6. Deployment Shape and Integration Posture

The platform is deployed as a set of cooperating services and supporting infrastructure layers. Public entry points are mediated by gateway or proxy layers, while application and data services remain in controlled runtime zones.



Deployment-oriented architecture for frontend, backend, identity, data, and supporting control points.

Integration posture

- Client users interact through browser-based portal access.
- Identity service handles user authentication and token issuance.
- Backend services expose the operational business capability consumed by the portal.
- Reporting and device-related integrations operate through dedicated service paths.
- Observability and backup functions support service continuity and governance.

7. Technical Review Summary

From a client architecture perspective, the platform is best understood as a secured operational portal backed by distinct service, identity, data, reporting, and observability layers.

What to confirm during review

- Authentication and authorization boundaries are aligned with the client security model.
- Reporting behavior and extended load characteristics are understood.
- Monitoring, incident response, and recovery assumptions are explicit.
- Component boundaries are sufficient for governance and operational support.

Companion volumes

- Client Documentation Pack for the full external set
- Client User Guide for workflow onboarding and screenshots
- Operations and Deployment Guide for continuity and support focus

CLIENT VOLUME 3

Architecture Without Internal Leakage

This volume is intended for client-side technical review and governance conversations where a clear architectural explanation is needed without exposing internal engineering material.

Use this volume for

- Architecture review meetings
- Technical due diligence
- Client IT alignment

Publication

Publication: ECTS documentation portal

Prepared by: Keshi ECTS Engineering Team

Revision: 1.0, March 10, 2026