CMIPS Security Architecture Analysis & Enhanced Proposal

Document Overview

This document provides a detailed analysis of the current California IHSS CMIPS security architecture and proposes an enhanced, modernized approach using **API Gateway, JWT tokens, and configurable policy-based authorization**.

Table of Contents

1. Executive Summary

The current CMIPS system utilizes a traditional **LDAP-based authentication system** with role-based access control. This proposal outlines a modernized architecture using **JWT tokens, API Gateway pattern, and configurable policy engines** to enhance security, scalability, and maintainability.

Key Benefits of Enhanced Architecture

- Improved security through token-based authentication
- Granular, configurable access controls
- Better audit trails and compliance
- Reduced dependency on legacy systems
- Enhanced performance through caching

2. Current Architecture Analysis

Existing Security Flow

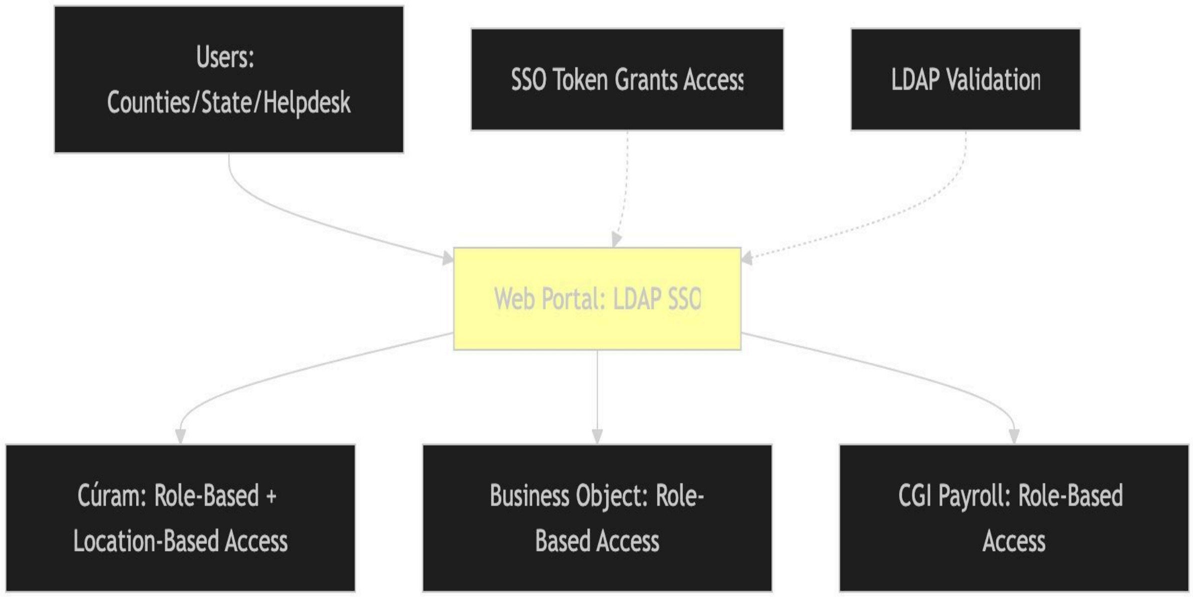
Authentication Process:

- Users access the Web Portal
- Credentials validated against LDAP directory
- LDAP issues SSO token upon successful authentication
- SSO token grants access to backend systems

Authorization Process:

- Role-Based Access Control (RBAC) managed within Cúram
- Location-based restrictions within business objects

Current system Architecture:


Users:
Counties/State/Helpdesk
SSO Token Grants Access
LDAP Validation
Web Portal: LDAP SSO
Cúram: Role-Based + Location-Based Access
Business Object: Role-Based Access
CGI Payroll: Role-Based Access

System Integration:

- Web Portal + LDAP act as central authentication hub
- Cúram handles complex business logic and authorization
- BusinessObjects provides reporting with location filters
- CGI Advantage manages payroll processing

3. Proposed Architecture

3.1 Authentication Layer

- Web Portal: Handles user credential validation
- JWT Token Generation: Creates secure tokens with user claims

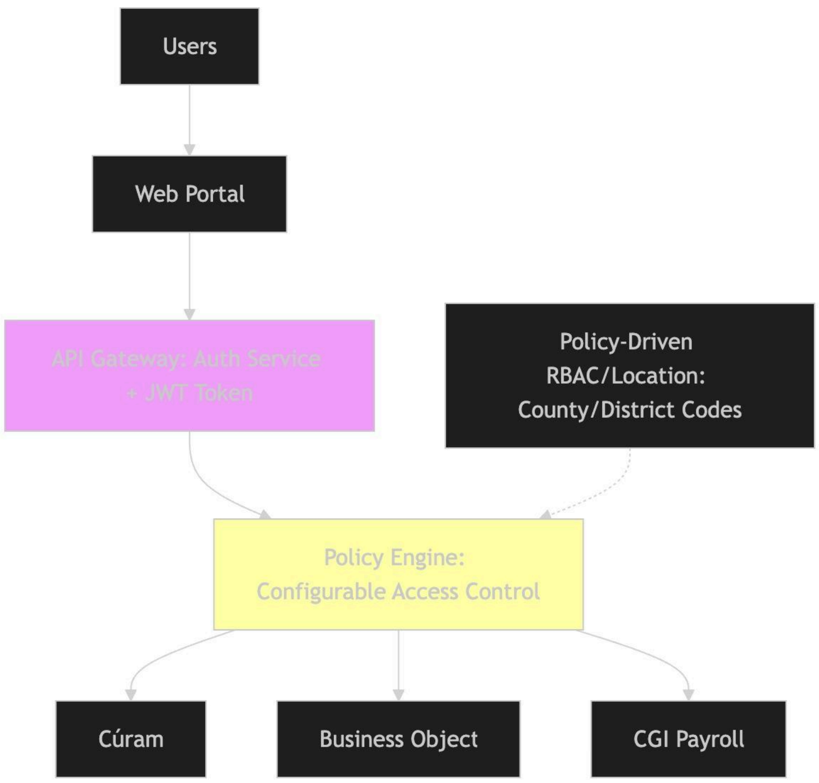
3.2 Authorization Layer

- API Gateway: Central entry point with JWT validation
- Policy Engine: Real-time access decision making
- Policy Database: Configurable rule storage

3.3 Backend Systems

- Cúram System: Case management with policy-driven access
- Business Objects: Reporting with dynamic filters
- CGI Payroll: Secure payroll processing

Architecture:

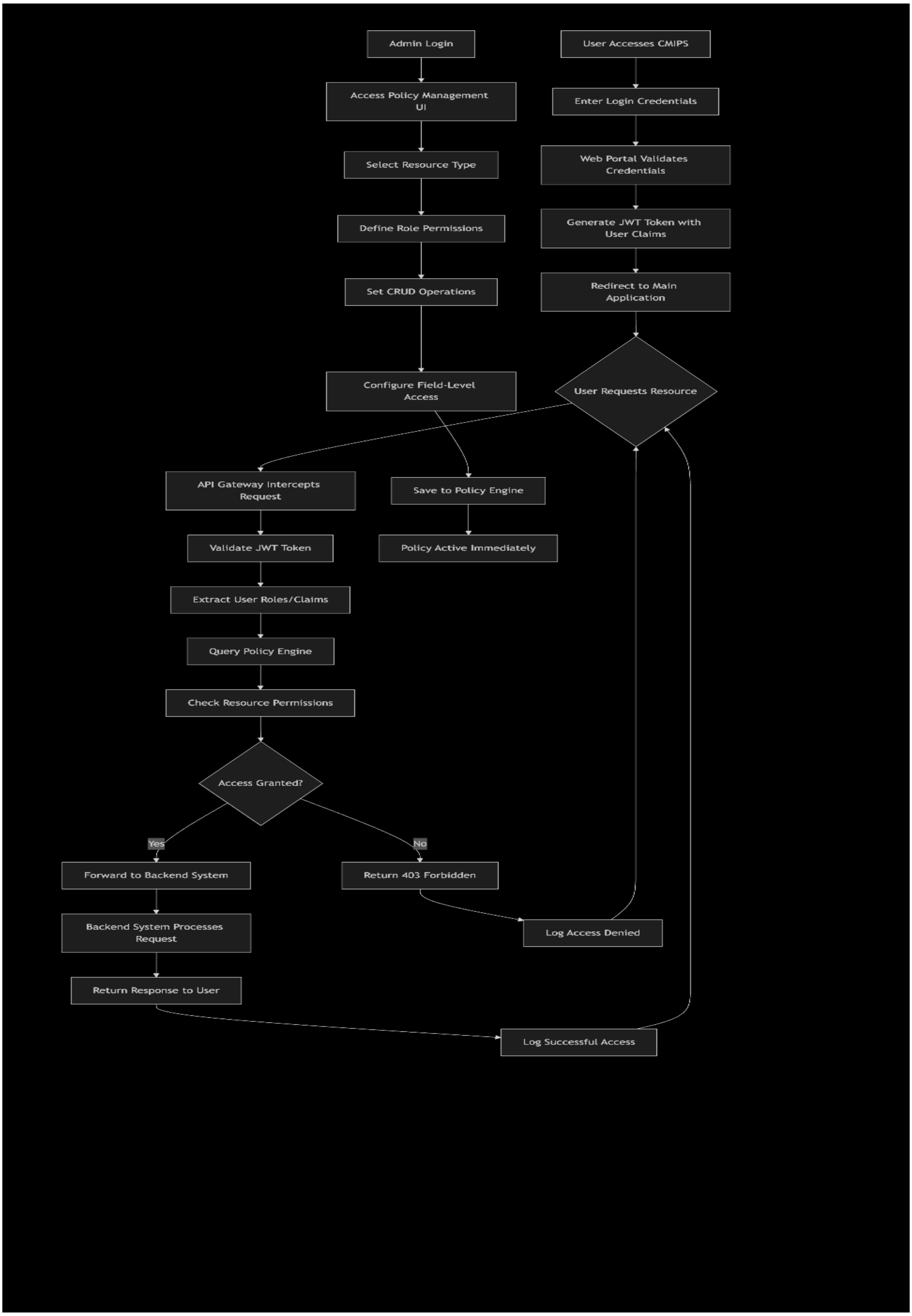

Users
Web Portal
API Gateway: Auth Service + JWT Token
Policy-Driven RBAC/Location: County/District Codes
Policy Engine: Configurable Access Control
Cúram
Business Object
CGI Payroll

Key Features

- Token-Based Authentication: JWT/OAuth2 standards
- Configurable RBAC/ABAC: Dynamic policy management
- Granular Authorization: API-level and action-level controls
- Centralized Auditing: Comprehensive access logging
- Policy Simulation: Test capabilities before deployment

4. Technical Diagrams & Flows

4.1 Activity Diagram – User Authentication & Authorization Flow

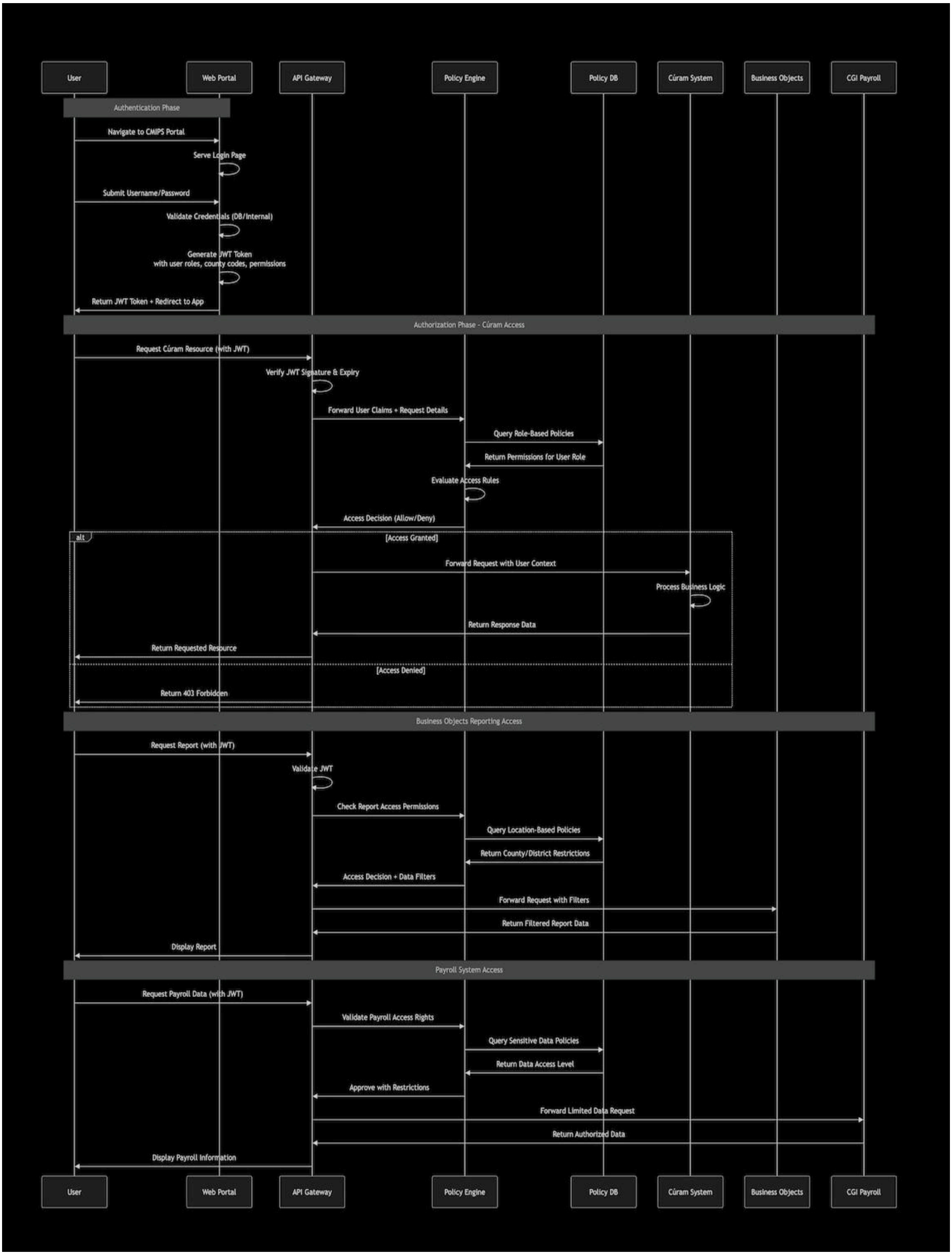

Admin Login
Access Policy Management UI
Select Resource Type
Define Role Permissions
Set CRUD Operations
Configure Field-Level Access
Save to Policy Engine
Policy Active Immediately
User Accesses CMIPS
Enter Login Credentials
Web Portal Validates Credentials
Generate JWT Token with User Claims
Redirect to Main Application
User Requests Resource
API Gateway Intercepts Request
Validate JWT Token
Extract User Roles/Claims
Query Policy Engine
Check Resource Permissions
Access Granted?
Yes
No
Forward to Backend System
Backend System Processes Request
Return Response to User
Return 403 Forbidden
Log Access Denied
Log Successful Access

Explanation:

This activity diagram illustrates the complete user journey from initial login through resource access. The flow demonstrates:

- Credential validation and JWT token generation
- Policy engine consultation for authorization decisions
- Real-time access control evaluation
- Comprehensive logging for audit purposes

4.2 Sequence Diagram – Complete Authentication & Authorization Flow


User
Web Portal
API Gateway
Policy Engine
Policy DB
Cúram System
Business Objects
CGI Payroll
Authentication Phase
Navigate to CMIPS Portal
Serve Login Page
Submit Username/Password
Validate Credentials (DB/Internal)
Generate JWT Token
with user roles, county codes, permissions
Return JWT Token + Redirect to App
Authorization Phase - Cúram Access
Request Cúram Resource (with JWT)
Verify JWT Signature & Expiry
Forward User Claims + Request Details
Query Role-Based Policies
Return Permissions for User Role
Evaluate Access Rules
Access Decision (Allow/Deny)
alt
[Access Granted]
Forward Request with User Context
Process Business Logic
Return Response Data
Return Requested Resource
[Access Denied]
Return 403 Forbidden
Business Objects Reporting Access
Request Report (with JWT)
Validate JWT
Check Report Access Permissions
Query Location-Based Policies
Return County/District Restrictions
Access Decision + Data Filters
Forward Request with Filters
Return Filtered Report Data
Display Report
Payroll System Access
Request Payroll Data (with JWT)
Validate Payroll Access Rights
Query Sensitive Data Policies
Return Data Access Level
Approve with Restrictions
Forward Limited Data Request
Return Authorized Data
Display Payroll Information
User
Web Portal
API Gateway
Policy Engine
Policy DB
Cúram System
Business Objects
CGI Payroll

Explanation:

This sequence diagram details the interaction between system components during a typical user session.

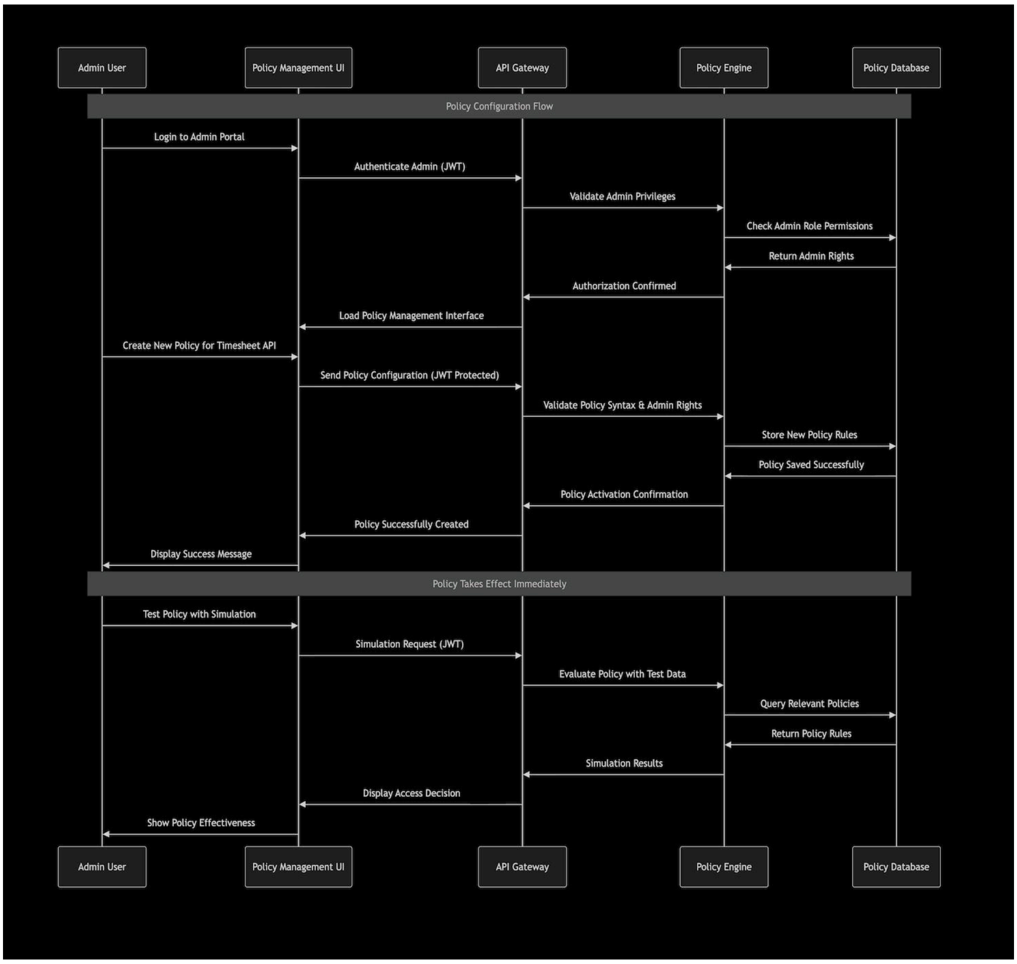
Authentication Phase:

- User credentials are validated using any IAM and generates SSO
- JWT token is generated containing user roles and claims bassed on SSO
- Token is used for subsequent API calls

Authorization Phase:

- API Gateway validates JWT and extracts claims
- Policy Engine evaluates access rights against stored policies
- Request is forwarded to backend systems only if authorized
- Comprehensive logging occurs at each step

4.3 Enhanced Sequence Diagram – Policy Configuration by Admin


Admin User
Policy Management UI
API Gateway
Policy Engine
Policy Database
Policy Configuration Flow
Login to Admin Portal
Authenticate Admin (JWT)
Validate Admin Privileges
Check Admin Role Permissions
Return Admin Rights
Authorization Confirmed
Load Policy Management Interface
Create New Policy for Timesheet API
Send Policy Configuration (JWT Protected)
Validate Policy Syntax & Admin Rights
Store New Policy Rules
Policy Saved Successfully
Policy Activation Confirmation
Policy Successfully Created
Display Success Message
Policy Takes Effect Immediately
Test Policy with Simulation
Simulation Request (JWT)
Evaluate Policy with Test Data
Query Relevant Policies
Return Policy Rules
Simulation Results
Display Access Decision
Show Policy Effectiveness
Admin User
Policy Management UI
API Gateway
Policy Engine
Policy Database

Explanation:

This diagram shows how administrators can dynamically configure security policies.

Policy Management Flow:

- Admin authenticates and accesses policy management interface
- New policies are created and validated for syntax
- Policies are stored in the policy database
- Immediate activation allows real-time testing
- Simulation capabilities enable policy validation before production use

5. Security Benefits

5.1 Enhanced Protection

- **Token Security:** JWT tokens provide secure, stateless authentication
- **Policy Flexibility:** Dynamic rules adapt to changing requirements
- **Granular Control:** Field-level security prevents data leakage
- **Audit Trail:** Comprehensive logging supports compliance

5.2 Risk Mitigation

- **Reduced Attack Surface:** Centralized security controls
- **Quick Response:** Rapid policy updates for emerging threats
- **Access Monitoring:** Real-time detection of suspicious activities
- **Compliance Assurance:** Built-in regulatory requirements