OKTA Implementation for Insight Applications

Implement robust and scalable identity and access management (IAM) architecture using **Okta Enterprise Edition** to centralize authentication across multiple applications with varied tech stacks (Java/Spring Boot, Node.js, Angular, React) and a common SQL Server database.

This new architecture is designed to replace legacy user management flows handled via:

- **Entitlements App** (external user onboarding and approvals)
- **MIS App** (internal user provisioning and AD sync)

The new Okta-based system will unify and modernize authentication and authorization, eventually enabling **decommissioning** of Entitlements and MIS systems.

Legacy System Overview

Entitlements App (External Users)

- Users register via application login pages
- Approval workflow managed in Entitlements UI
- On approval, user added to central Users table with IsInternal = 0
- Fixed permissions assigned automatically

MIS App (Internal Users)

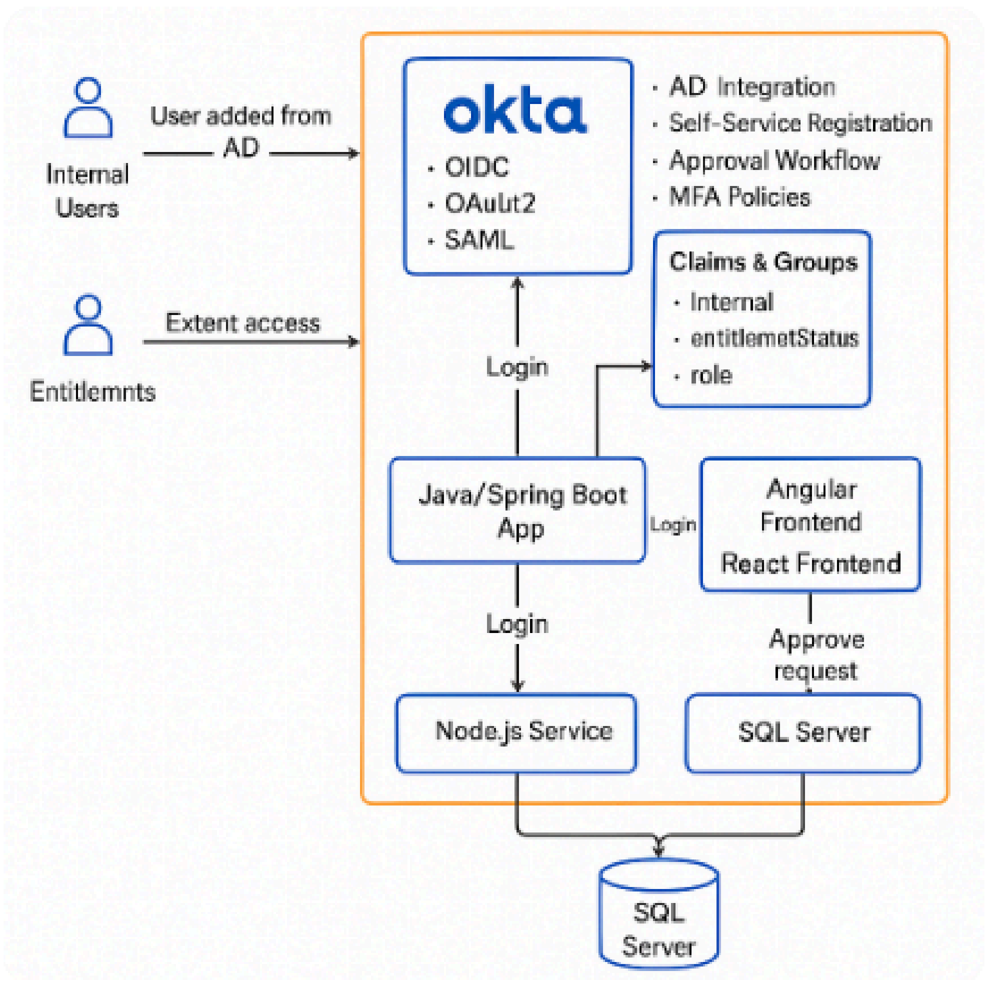
- Admins search AD from MIS UI to add internal users (Internal and Analyst users)
- Permissions managed and assigned within MIS
- Users added to central Users table with IsInternal = 1

Shared User & Permissions DB

- Common Users table stores all identities
- IsInternal boolean flag differentiates user source
- Application-level permissions mapped via roles and permission tables

Revised IAM Model with OKTA Integration

Legacy Module	Replacement in New Architecture
ntitlements	Okta Self-Service Registration + SCIM + Workflows
MIS	Okta AD Agent + Universal Directory
Permission Logic	Centralized in Okta Groups & Claims
Users Table	Synced with Okta via API/Webhook or minimal reference only



High-Level Workflow Mapping

Internal Users

- ❖ Synced from Active Directory into Okta using Okta AD Agent
- ❖ Dynamic group assignment based on AD OU or group rules
- ❖ Application access based on group membership and claims

- Login via SSO (OIDC)

External Users

- Register via Okta-hosted sign-up or custom UI (SPA)
- Approval request sent via Okta Workflow to designated reviewers
- On approval, user is provisioned with group assignment (e.g., App_External_Users)
- Login via OIDC and access determined by group claims

Detailed Component Architecture

Identity Provider: Okta (Enterprise Edition)

- SSO, OIDC/OAuth2, MFA, AD Integration
- Centralized user directory with profile enrichment
- Dynamic group assignment rules
- Custom claim definitions per app

Internal Users Flow (via AD + Okta)

- Okta AD Agent syncs user objects from on-prem AD
- Internal users marked with isInternal = true in profile
- Group membership reflects role/department
- Access token includes group/role information used by downstream apps

External Users Flow (via Registration Portal + Okta)

- User initiates registration via custom or Okta-hosted page
- Profile stored in Okta with isInternal = false
- Okta Workflow triggers approval chain
- Admin approval assigns correct group and activates user

Integration with Legacy User Table (Decommissioning Plan)

- Short-term sync with central Users table continues
  - isInternal, userSource, entitlementStatus, and other metadata recorded
  - No permission logic in DB — enforced via Okta token claims
- Plan to deprecate Entitlements & MIS functionality step-by-step after full Okta rollout

Okta Configuration Guide

Main Area	**Details**
1. Universal Directory Setup	• Enable profile schema with custom attributes: - isInternal (Boolean) - userSource (Enum: AD, SelfSignup, SCIM) - entitlementStatus (Pending, Approved)
2. Authorization Server Setup	• Create EnterpriseAppsAuthServer • Add claims: - groups: Filter = starts with App_ - isInternal: Expression = user.profile.isInternal - roles, tenant_id, entitlementStatus as required
3. Applications Registration (OIDC)	• Register each client app: - SPA: React, Angular - Web/API: Spring Boot, Node.js • Use Authorization Code with PKCE flow • Define redirect URIs and post-logout URIs • Assign apps to appropriate Okta groups based on access control
4. Group and Role Design	• Application-level groups: - App1_Admin, App1_User, App2_ReadOnly • User-type groups: - Internal_Users, External_Users • Dynamic group rules: - Based on isInternal, userSource profile attributes
5. Okta Workflows Setup	• Build onboarding flow for external users: - Trigger: New external registration - Actions: • Notify approver • Update entitlementStatus • Assign to group - Optional: • Trigger webhook for downstream systems • Send confirmation email

Tech Stack Integration Details:

React (SPA)

- Use @okta/okta-auth-js and @okta/okta-react
- Handle login via Redirect or Popup
- Secure routes with <SecureRoute>
- Access token includes group/role for API consumption

Angular (Admin Panel)

- Use @okta/okta-angular
- Configure OktaAuthService for routing guards
- Inject access token in API interceptor

Node.js (API Gateway)

- Use @okta/jwt-verifier
- Middleware to validate JWT and extract claims
- Fine-grained permission enforcement based on groups/roles

Java Spring Boot (API Services)

- Use okta-spring-boot-starter
- YAML Config changes
- Annotate with @PreAuthorize("hasAuthority('App1_Admin')")

Migration & Decommission Plan

Phase	Task	Details
1	SSO Enablement for all apps	Configure Okta login flows and test token validation
2	Sync Internal Users from AD	Okta AD Agent setup with group/OU filters
3	Build External Registration Portal	Self-service flow + approval via Okta Workflows
4	Create Dynamic Groups & Claims	Implement isInternal, userSource, etc.
5	Migrate Permissions from MIS/Entitlements to Okta	Redesign access roles in Okta groups
6	Refactor Applications to Use Okta Tokens	Replace custom user tables with token-based roles
7	Final Testing & Validation	Audit logins, group mapping, claims accuracy
8	Decommission MIS and Entitlements	Post-successful cutover & freeze legacy systems