Project Requirements Document (PRD)

Project Name: GoCloak

Prepared by: Sajib Jahan

Date: April 21, 2025

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Purpose

The purpose of this document is to define the product requirements for GoCloak, an open-source, Go-based Identity and Access Management (IAM) platform. This document translates business goals outlined in the Business Requirements Document (BRD) into actionable, user-focused requirements. It sets the stage for system design (SDD) and technical specifications (TSD), without specifying implementation-level details.

# 1. Project Overview

**GoCloak** is a lightweight, high-performance alternative to Keycloak, written in Go, designed to offer authentication, authorization, and user management functionality for modern applications and services. The goal is to deliver a modular, scalable Identity and Access Management (IAM) solution with a web-based Admin Panel and REST APIs to manage **realms** and other identity-related entities.

The system will be self-hosted and easy to deploy, with a minimal setup and a focus on extensibility and performance. It targets developers and system administrators who need fine-grained control over authentication and authorization infrastructure without the overhead of Java-based platforms.

**Keycloak overview:** <https://www.youtube.com/watch?v=z8gxQr6LGG4>

### What GoCloak Will Look Like

* A **central IAM server** with:  
  + A **REST API** for creating and managing realms, users, clients, and roles
  + An **Admin Panel** to manage these visually (only internal access)
  + A **token service** for OAuth2/OIDC-based login and access control
* **Login and registration UIs** for end-users
* A modular architecture for future expansion: external IdPs, MFA, audit logs, etc.

# 2. Scope

GoCloak aims to be a modern, lightweight alternative to Keycloak. It provides authentication, authorization, and user management features through a RESTful API and an internal Admin Panel.  
  
**In-Scope Features:**  
• Multi-tenant support using isolated Realms  
• Admin Panel for managing realms, users, roles, clients, and audit logs  
• REST API for integrating with external services  
• JWT-based authentication and token management (OAuth2/OIDC)  
• Basic MFA support and account lockout mechanisms  
  
**Out-of-Scope (Initial Release):**  
• Social logins (Google, Facebook, etc.)  
• External IdP federation (LDAP, SAML)  
• Advanced/custom authentication flows  
• Full-featured end-user portal

# 3. Feature List

**3.1 Realms (Multi-Tenancy)**

* Create, update, delete, and retrieve isolated identity domains
* Realm-level configurations (token settings, password policies)

**3.2 User Management**

* CRUD operations on users within realms
* Grouping users and assigning roles
* Password reset and account status handling

**3.3 Role and Permission Management**

* Create roles and map them to users or groups
* Role-based access control (RBAC)

**3.4 Client Management**

* Register, update, and delete client applications per realm
* Manage client secrets and redirect URIs

**3.5 REST API (External Use)**

* OAuth2 and OIDC-compliant authentication endpoints
* Endpoints for managing users, roles, clients, and realms
* Token introspection and revocation support
* Secure API access using JWT and RBAC enforcement

**3.6 Authentication Flows**

* Password Grant: Secure login with username/password
* Authorization Code Flow: For web and mobile app authentication
* Implicit Flow: For public clients requiring direct token issuance
* Token Refresh and Revocation mechanisms

**3.7 Token and Authentication Services**

* JWT issuance and validation (RS256/HS256)
* Support for OAuth2 and OpenID Connect protocols
* Centralized logout and session management

**3.8 Admin Panel (Internal Use Only)**

* Manage realms, users, clients, and roles via UI
* Role-restricted admin access with RBAC
* View audit logs and basic usage metrics

**3.9 Security Features**

* Secure password storage (bcrypt/argon2)
* Enforced HTTPS on all endpoints
* CSRF protection in Admin Panel
* Brute-force protection (login thresholds, lockouts)
* Session expiry and idle timeout policies

# 4. User Stories / Use Cases

1. **Administrator**:

**User Management:**

* As an admin, I want to create a new user account so that a new employee can access the application.
* As an admin, I want to update a user's profile information so that the user's details are accurate.
* As an admin, I want to assign roles to a user so that the user has the correct permissions.
* As an admin, I want to reset a user's password so that the user can regain access to the application.
* As an admin, I want to enable or disable a user account to control access to the application.
* As an admin, I want to delete a user account when a user leaves the organization.
* As an admin, I want to view a user's login history and session information to monitor user activity.

**Role and Permission Management:**

* As an admin, I want to create new roles to define specific permissions.
* As an admin, I want to update existing roles to modify access levels.
* As an admin, I want to delete roles that are no longer in use.
* As an admin, I want to assign roles to groups to apply permissions at a group level.
* As an admin, I want to create composite roles that aggregate multiple roles.

**Realm Management:**

* As an admin, I want to create a new realm to manage authentication and authorization for a specific application.
* As an admin, I want to configure authentication flows for a realm to define the login and registration processes.
* As an admin, I want to set up identity providers for a realm to enable third-party authentication (e.g., SAML, OIDC).
* As an admin, I want to configure session timeout settings for a realm to control user sessions.

**Client Management:**

* As an admin, I want to register a new client application to integrate with the identity provider.
* As an admin, I want to configure client credentials (client ID, secret) to secure client communication.
* As an admin, I want to define client scopes to control the information that client applications can access.
* As an admin, I want to manage client session limits to prevent excessive resource consumption.

**Event Management and Monitoring:**

* As an admin, I want to view system logs and audit events to track key actions and security events.
* As an admin, I want to configure event listeners to send notifications when specific events occur.

**Identity and Access Management (IAM):**

* As an admin, I want to define custom user attributes to store additional user information.
* As an admin, I want to set up password policies to enforce strong password requirements.
* As an admin, I want to configure multi-factor authentication (MFA) to enhance security.

1. **System Designer**

**Performance and Scalability:**

* As a system designer, I want the system to support a concurrent user limit of X users to handle peak loads effectively.
* As a system designer, I want the system to maintain response times under Y milliseconds for authentication requests to ensure optimal performance.
* As a system designer, I want to define rate-limiting rules for login attempts to prevent abuse and mitigate DDoS attacks.
* As a system designer, I want to implement session clustering to ensure high availability and fault tolerance.
* As a system designer, I want the system to support horizontal scaling to handle increasing user loads.

**Data Consistency and Integrity:**

* As a system designer, I want the user data to be synchronized across all nodes in real-time to maintain data consistency.
* As a system designer, I want to ensure that user role assignments are atomic transactions to prevent inconsistent state updates.
* As a system designer, I want the system to provide rollback capabilities for user data changes to recover from erroneous operations.

**Security and Compliance:**

* As a system designer, I want to enforce OAuth 2.0 and OIDC standards for authentication flows to adhere to security best practices.
* As a system designer, I want to implement data encryption for user credentials and sensitive information to protect data at rest and in transit.
* As a system designer, I want to configure audit logging to capture all admin actions for compliance and monitoring.
* As a system designer, I want to implement session timeout policies to minimize the risk of session hijacking.

**System Monitoring and Alerts:**

* As a system designer, I want to define performance metrics for login requests, session creation, and role assignment to monitor system health.
* As a system designer, I want to set up alert thresholds for high CPU, memory usage, and network latency to proactively detect anomalies.
* As a system designer, I want to implement distributed tracing to identify performance bottlenecks in multi-node deployments.

**Extensibility and Integration:**

* As a system designer, I want to design an API layer for custom authentication mechanisms to accommodate third-party integrations.
* As a system designer, I want to define integration points for external identity providers (SAML, LDAP, OIDC) to ensure interoperability.
* As a system designer, I want to provide extension points for custom event listeners and policies to allow custom business logic execution.

**Data Storage and Management:**

* As a system designer, I want to define data retention policies for session data and login history to comply with data protection regulations.
* As a system designer, I want to implement data partitioning and sharding to distribute load and optimize query performance.
* As a system designer, I want to provide caching mechanisms for frequently accessed user data to reduce database load.

**Disaster Recovery and Failover:**

* As a system designer, I want to define failover procedures for authentication nodes to ensure uninterrupted service during outages.
* As a system designer, I want to implement data backup strategies with periodic snapshots to enable data recovery in case of corruption or data loss.
* As a system designer, I want to maintain read replicas for user data to provide fallback options in case the primary database fails.

1. End User

**Self-Managed Authentication Server:**

* As an end user, I want to download and install GoCloak to set up my own authentication and authorization server without complex configurations.
* As an end user, I want to create a new tenant in GoCloak to isolate my application's user data and manage access policies.

**Secure Access and Identity Management:**

* As an end user, I want to define custom authentication flows (e.g., login, MFA) to provide a tailored user access experience.
* As an end user, I want to connect GoCloak with external identity providers (SAML, OIDC) to allow users to log in using their existing accounts.
* As an end user, I want to implement role-based access control (RBAC) to enforce permission levels across resources.

**Authorization Policies and Session Management:**

* As an end user, I want to define resource access policies to control how users interact with application APIs and endpoints.
* As an end user, I want to monitor active sessions and revoke access tokens to maintain security and prevent unauthorized access.- As an end user, I want to define custom authorization policies to control access to resources.
* As an end user, I want to set up permissions for resources (APIs, endpoints) to restrict access.
* As an end user, I want to monitor user sessions and revoke tokens to maintain security.- As an end user, I want to update my profile information, such as email and phone number, to keep my account up-to-date.
* As an end user, I want to manage my security settings, such as changing my password or configuring MFA settings.

1. Developer/Integrator

**API-Driven User Management:**

* As a developer, I want to leverage GoCloak’s REST APIs to manage users, roles, and sessions programmatically.
* As a developer, I want to implement custom authentication flows via APIs to extend GoCloak’s functionality.
* As a developer, I want to integrate third-party identity providers to allow SSO and centralized authentication.

**Extensibility and Custom Plugins:**

* As a developer, I want to develop custom plugins to add new authentication mechanisms or event listeners.
* As a developer, I want to define custom event handlers to trigger business logic on key user actions (e.g., login, logout).- Develop custom plugins
* Define custom event listeners

1. Compliance and Security Analyst

**Security Policy Enforcement:**

* As a security analyst, I want to define password complexity policies to prevent weak passwords.
* As a security analyst, I want to enforce session timeout policies to minimize security risks.

**Data Compliance and Audit Reporting:**

* As a compliance analyst, I want to configure data retention policies to align with regulatory requirements.
* As a compliance analyst, I want to generate audit reports detailing user access and admin actions for security reviews.- Configure data retention policies
* Generate audit reports

1. Operations/DevOps

**Monitoring and Alerts:**

* As an operations engineer, I want to set up performance monitoring in GoCloak to detect potential system bottlenecks.
* As an operations engineer, I want to define alert thresholds to be notified of unusual activity or resource consumption.

**Backup and Recovery:**

* As an operations engineer, I want to implement data backup strategies to ensure data integrity during system failures.
* As an operations engineer, I want to define failover procedures to maintain service availability during outages.- Implement data backup strategies
* Define failover procedures

1. Business Stakeholder

**Reporting and Analytics:**

* As a business stakeholder, I want to generate user activity reports in GoCloak to track system usage patterns.
* As a business stakeholder, I want to track SLA adherence to ensure that the authentication services meet expected performance levels.

**User Growth Monitoring:**

* As a business stakeholder, I want to monitor user registration metrics to assess user acquisition trends.
* As a business stakeholder, I want to analyze login success and failure rates to identify potential authentication issues.- Track user registration metrics
* Monitor login success/failure rates

# 5. Functional Requirements

• REST API for CRUD operations on realms, users, clients, and roles  
• JWT token generation and validation endpoints  
• Admin UI for internal management  
• Support for multiple OAuth2/OIDC grant types  
• Password encryption and session handling  
• Audit logging for administrative actions

# 6. Success Metrics

• 90% test coverage for backend and admin panel  
• API response time under 200ms at 1k RPS  
• MVP successfully tested with 3+ realms and 1,000+ users each  
• Positive feedback on ease of use and performance from internal testers

# 7. UX/UI Requirements (Admin Panel)

• Simple dashboard with realm selector  
• User and client management views  
• Audit log viewer with filtering  
• Admin login with RBAC-based access control

# 8. Assumptions

• Developers have familiarity with OAuth2 concepts  
• Initial deployments are internal or small-scale open source  
• Admin Panel will not be exposed to public internet in early versions

# 9. Risks

• Scope creep from adding advanced identity features too early  
• Security issues due to improper token handling  
• Performance bottlenecks under concurrent load

# 10. Appendix / References

• [GoCloak BRD v1.0.0 - April 2025]  
• OAuth2 Spec: https://datatracker.ietf.org/doc/html/rfc6749  
• OIDC Spec: https://openid.net/specs/openid-connect-core-1\_0.html