Terms of Reference

Reusable Component: User & Access Administration Framework

Mendix Reusable Components Initiative - ISAA

Version	Status
0.0.1	Draft

1. Background

This Terms of Reference document is issued as part of ISAA's (Information Systems Agency of Armenia) initiative to build a standard, reusable foundation of Mendix components for public digital services. Full context, goals, and target users for the overall initiative are described in the accompanying **Concept Note**.

Public digital services built on Mendix routinely need to manage application users, application roles, and access assignments, and to authenticate users through the national **YesEm** digital identity platform (OIDC). This component addresses the gap where each project would otherwise implement its own user provisioning, role model, and access logic independently, leading to duplicated effort and inconsistent handling of identity, provisioning, and authorization.

2. Objective

Develop a configurable, reusable **User & Access Administration Framework** for Mendix applications that provides a standardized approach to managing application users, application roles, and access assignments across public digital services. The framework shall integrate with **YesEm** to leverage authenticated user identity information, and provide reusable capabilities for user provisioning, role management, and access administration.

3. Scope of Work

3.1 Functional Requirements

YesEm Integration

The framework should integrate with **YesEm**, the national digital identity platform, as the authentication source for authenticated user identity information.

Vendors should describe:

- How the framework integrates with YesEm OIDC to authenticate users and retrieve identity claims;
- How YesEm claims (e.g., national ID, name) are mapped to local user attributes, and whether this mapping is configurable rather than hardcoded per application.

User Provisioning

The framework should support multiple, configurable provisioning approaches, suitable for both public-facing and internal applications:

- **JIT (just-in-time) provisioning** - a local User record is automatically created on first successful YesEm login, using identity claims returned by YesEm;
- **Initial user creation** - setup of initial user profiles based on configuration at application startup;
- **Administrator-managed provisioning** - accounts created in advance by an organization administrator, for users who need access before they ever authenticate (e.g., internal civil servants);
- **Approval-based onboarding** - a user requests access, but the resulting role/access grant only becomes active after administrator approval (e.g., a business representative requesting access to act on behalf of their company).

Applications should be able to configure which of these approaches applies, and under what conditions.

User Lifecycle Management

The framework should support the full user lifecycle, not only account creation: deactivation, suspension, reactivation, and offboarding.

Identity Reconciliation

The framework should address how the same physical person appearing through different contexts (e.g., authenticating as a private citizen versus as a government employee) is handled, without creating duplicate or conflicting user records. Vendors should describe their proposed approach.

Application-Defined Roles

The framework should provide a reusable mechanism for applications to configure their own:

- Application roles;
- Access attributes;

- Access assignment rules.

Assignment of roles/attributes to users should support both manual assignment by an administrator and rule-based automatic assignment (e.g., derived from a YesEm claim or an organization attribute).

Attribute-Based Data Access

The framework should support filtering which records a user can see or act on based on attributes such as organization, region, or department, not solely their role. Vendors should describe the proposed mechanism (e.g., via Mendix entity access XPath constraints referencing attributes of the current user) and how it is exposed for reuse by other applications/components without requiring each to reimplement the pattern independently.

Delegation

The framework should support a user acting on behalf of another party (e.g., a legal representative or employee acting for a business, or for another citizen). Vendors should describe how delegation is modeled, granted, and revoked.

Organization-Scoped Administration

Where an application serves multiple organizations, the framework should support an organization's own administrator managing and provisioning users within their own organization only, rather than across all organizations in the system.

Reusable Administration UI

The framework should provide a reusable, drop-in screen set for administering users, roles, and access assignments, so that applications do not need to build their own administration UI.

Audit Integration

User, role, and access assignment changes should be recordable through the **Audit & Activity History Framework**, consistent with that framework's non-CRUD business event mechanism, rather than through separate, application-specific logging.

Self-Service Access Requests

The framework should support users requesting access or roles themselves (feeding into the approval-based onboarding flow above), rather than requiring all access to be admin-initiated.

3.2 Non-Functional Requirements

- **Security**: Identity and access data handled in line with applicable data protection requirements; no unnecessary storage of sensitive identity data beyond what is required for authorization; access control enforced consistently regardless of access path (UI, API).
- **Scalability**: User, role, and access assignment management should scale to a large number of users and organizations without degrading administration or authentication performance.
- **Availability**: Authentication and access-check paths should fail predictably and securely if YesEm is temporarily unavailable.
- **Compatibility**: Must run on *Mendix version > 11.10.0* and be upgradable across supported Mendix versions.

4. Deliverables

Vendors are expected to provide the following, in addition to the specific requirements above:

Standard Implementation Deliverables

#	Deliverable	Description
1	Mendix Source	Mendix application source (Team Server/Git) and deployment package.
2	Technical Documentation	Module purpose, architecture, dependencies, exposed entities, microflows, Java actions, and configuration.
3	Architecture Documentation	Architecture overview and key design decisions.
4	Security Configuration	Module roles, user role mappings, authentication and authorization configuration.
5	API Documentation	Interface specifications, request/response schemas, sample payloads, and error handling.
6	Configuration Guide	Environment-specific constants and configuration settings.
7	Third-Party Component Inventory	Marketplace modules and external dependencies used.
8	Code Quality Report	Static code analysis and Mendix quality metrics.
9	Test Plan	Unit, integration, system, UAT, regression, and non-functional testing approach.
10	Test Results	Test cases and execution evidence.
11	Security Test Report	Security testing and vulnerability assessment results (where applicable).
12	Performance Test Report	Performance and load testing results (where applicable).
13	User Documentation	User and administrator guides.
14	Training Materials	Training guides and supporting materials (where applicable).

Additional Deliverables for Reusable Components

#	Deliverable	Description
1	Module README	Purpose, prerequisites, installation and upgrade instructions.
2	Interface Contract	Public microflows, entities, Java actions, parameters, and expected outputs.
3	Configuration Guide	Post-import configuration instructions.
4	Sample Application	Reference application demonstrating component implementation, including at least one worked example covering YesEm login, provisioning, and attribute-based access.

5. Vendor Response Requirements

Vendor proposals should address:

1. **Technical approach** - proposed architecture, how it maps to the functional requirements in Section 3.
2. **Provisioning & lifecycle approach** - how JIT, administrator-managed, and approval-based provisioning are implemented and configured, and how the full user lifecycle is supported.
3. **Access model approach** - how application-defined roles, attribute-based data access, delegation, and organization-scoped administration are implemented and configured.
4. **YesEm integration approach** - authentication flow and claims-mapping approach.
5. **Reuse & extensibility** - how the framework will be consumed by other components in this initiative (e.g., Audit & Activity History Framework), and how it avoids tight coupling to any single application's user model.
6. **Team & experience** - Experience with Mendix (certificates, if available) and completed relevant projects.
7. **Timeline** - proposed delivery schedule against the deliverables in Section 4.
8. **Indicative Cost** - cost breakdown by phase/deliverable.
9. **Assumptions & risks** - any assumptions made and risks identified.