

JSUI Requirements Specification

Version 2026-02-05

This document supersedes `improvements.md` and reflects the completed implementation status as of February 5, 2026.

1. Architecture

1.1 Deployment Model

- **User SPA:** `index.html` — general user interface for account management, repository operations, SPARQL queries
- **Admin SPA:** `admin.html` — administrative interface for system-wide management
- **Shared Infrastructure:** Router, authentication, state management, and layout components are shared between both SPAs
- **Backend:** SPOCQ server providing `/system/` API endpoints

1.2 Local Routes vs. Remote Endpoints

Local routes are client-side SPA paths matched by the router to render page components. **Remote endpoints** are HTTP URLs served by the SPOCQ backend.

User App Routes

Local Route	Page Class	Remote Endpoint(s)
<code>/login</code>	LoginPage	POST <code>/system/accounts/{acct}/configuration</code>
<code>/</code>	HomePage	(static content)
<code>/signup</code>	SignupPage	POST <code>/system/accounts</code>
<code>/reset_password</code>	ResetPasswordPage	POST <code>/system/accounts/{acct}/password-reset</code>
<code>/invite</code>	InvitationsNewPage	POST <code>/invitations</code>
<code>/invite/success</code>	InvitationsSuccessPage	(static)
<code>/invitations</code>	InvitationsIndexPage	GET <code>/invitations</code>
<code>/account/:acct</code>	AccountShowPage	GET <code>/system/accounts/{acct}/configuration</code>
<code>/account/:acct/edit</code>	AccountEditPage	GET/POST <code>/system/accounts/{acct}/configuration</code>
<code>/account/:acct/auth_token</code>	AccountAuthTokenPage	POST <code>/system/accounts/{acct}/token-reset</code>
<code>/account/:acct/repositories</code>	RepositoriesIndexPage	GET <code>/system/accounts/{acct}/repositories</code>

Local Route	Page Class	Remote Endpoint(s)
/account/:acct/repository/show	RepositoryShowPage	GET .../configuration, /storage, /service_history, etc.
/account/:acct/repository/edit	RepositoryEditPage	GET/POST .../configuration, .../collaboration
/account/:acct/repository/import	RepositoryImportPage	POST /{acct}/{repo}/service
/account/:acct/repository/query	RepositoryQueryPage	POST /{acct}/{repo}/sparql
/account/:acct/repository/queries/show	RepositoryQueryShowPage	GET view./views/{view}, POST .../sparql
/account/:acct/repository/queries/log	RepositoryQueryLogPage	GET .../service_history
/account/:acct/repository/show/status	RepositoryShowStatusPage	(redirects to repository show)
/account/:acct/repository/show	RepositoryShowPage	(redirects to repository show)
/account/:acct/repository/show/data	RepositoryShowDataPage	(redirects to repository show)

Admin App Routes

Local Route	Page Class	Remote Endpoint(s)
/login	AdminLoginPage	POST /system/accounts/{acct}/configuration, GET /system/users/{acct}/configuration
/	AdminDashboardPage	Multiple (lazy-loaded per tab)

The admin dashboard includes six tabs: - **Accounts**: GET /system/accounts, POST /system/accounts, DELETE /system/accounts/{acct} - **Repositories**: GET /system/accounts/{acct}/repositories, DELETE .../repositories/{repo} - **Invitations**: GET /invitations, POST /invitations, DELETE /invitations/{email} - **Query History**: GET /system/accounts/{acct}/repositories/{repo}/service_history (per selected repo) - **Transaction History**: GET /system/service_history/transactions - **Import History**: GET /system/service_history/imports

1.3 Graph Store Protocol

Method	Endpoint	Purpose
GET	/ {acct} / {repo}	Export repository (content-negotiated)
POST	/ {acct} / {repo}	Add statements (merge import)
PUT	/ {acct} / {repo}	Replace all statements
DELETE	/ {acct} / {repo}	Clear repository

Method	Endpoint	Purpose
GET/POST/PUT/DELETE	<code>/api/{acct}/{repo}/service?graph={graph}</code>	Named graph operations

2. Completed Features

2.1 Repository Operations

- **Export:** Format selection popup (Turtle, N-Triples, N-Quads, RDF/XML, JSON-LD, TriG, CSV), content-negotiated GET, blob download
- **Import:** File selection, media type detection/selection, async option with `AcceptAsynchronous: notify` header, XMLHttpRequest with upload progress (bytes/total/percentage), abort capability
- **Clear:** DELETE with confirmation dialog
- **Delete:** DELETE with confirmation dialog, N-Triples result display

2.2 Import/Export Button Behavior

- Import and export buttons are cross-disabled during either operation
- Import button shows progress: **Importing...** X/Y (Z%)
- Import button becomes “Cancel Import” during upload; click aborts
- Export button shows “Exporting...” during download

2.3 View Operations

- **Execute link:** Each view in the query catalog has an external link icon that opens `/api/{acct}/{repo}/{view}` in a new tab with `Accept: text/html`
- **Delete:** Trash icon with confirmation dialog

2.4 Query History

- **User app:** `RepositoryQueryLogsPage` fetches from `/system/accounts/{acct}/repositories/{repo}` with sortable columns (timestamp, elapsed_time, run_time, signature)
- **Signature links:** Each signature value links to the query text via `GET /system/service_history/queries/{account}/{signature}` with `Accept: application/sparql-query`, displayed in a modal dialog
- **Admin app:** Query History tab aggregates across selected repositories with sortable columns; content auto-refreshes when repository selection changes

2.5 Admin Authorization Model

- Any authenticated user can access the admin SPA
- Admin status determined by `administrator_of` field from `/system/users/{acct}/configuration`
- Non-admin users see all data but cannot see admin-only controls:
 - Account delete buttons
 - Repository delete buttons
 - New account button
 - New repository button
 - New invitation button
 - Send invitation links
 - Delete invitation links

2.6 Admin Dashboard Tabs

1. **Accounts:** List, refresh, details modal, create (admin), delete (admin)
2. **Repositories:** Filtered by selected accounts, create (admin), delete (admin)
3. **Invitations:** List, send (admin), delete (admin), create (admin)
4. **Query History:** Filtered by selected repositories, sortable columns
5. **Transaction History:** System-wide, sortable columns (timestamp, UUID, account, repository, insert/remove counts, agent)
6. **Import History:** System-wide, sortable columns (timestamp, UUID, account, repository, agent, source URL, quad count, success)

2.7 SEO

`index.html` includes: - Open Graph meta tags (title, description, image, URL, type) - Twitter Card meta tags (card, title, description) - Improved meta description - Canonical URL - JSON-LD structured data (WebApplication schema) - Noscript fallback with links to documentation and blog

2.8 User Experience

- **Invite link:** Login page includes “Request an invitation” link pointing to `/invite`
 - **Cross-app navigation:** User app header has Admin link; Admin app header has User App link
-

3. CSS Dependencies

3.1 Deleted (Unused)

- `css/bootstrap*.css` — all Bootstrap files
- `css/tablesorter/theme.bootstrap.css`
- `stylesheets/jquery-ui/` — entire Aristo theme directory

- `stylesheets/jquery-ui-1.8.24.min.css`

3.2 Retained

- `stylesheets/style.css` — inherited from Rails, minified
 - `stylesheets/jsui-overrides.css` — custom overrides and replacement CSS for legacy class references
-

4. Server-Side Requirements (SPOCQ)

The following endpoints require SPOCQ server implementation:

4.1 Password Reset

POST `/system/accounts/{acct}/password-reset`

Body: { "email": "user@example.com" }

→ Generate token, store in DB, send email

→ 202 Accepted / 404 Not Found

POST `/system/accounts/{acct}/password-reset`

Body: { "token": "...", "password": "...", "password_confirmation": "..." }

→ Validate token, update password, clear token

→ 200 OK / 422 Unprocessable

4.2 Auth Token Reset

POST `/system/accounts/{acct}/token-reset`

→ Generate new authentication token

→ { "authentication_token": "new-token" }

4.3 Email Sending

SPOCQ should use `run-program` with the external `mail` utility for: - Password reset instructions - Invitation request notifications to admin - Invite code emails to users

5. Not Implemented (By Design)

- **Email confirmation:** Not implemented in Rails; token usage is implicit confirmation
- **Account-account collaboration:** Empty stubs in Rails; not a regression
- **Payment history:** Not intended for current deployment
- **Concurrent import guard:** Server-side concern
- **Import status tracking:** Server-side state machine

6. API Endpoint Reference

6.1 Authentication

Endpoint	Method	Purpose
/system/accounts/{acct}/config	POST	Authenticate, retrieve token and config
/system/users/{acct}/config	GET	Retrieve user privileges (admin check)

6.2 Accounts

Endpoint	Method	Purpose
/system/accounts	GET	List all accounts
/system/accounts	POST	Create account
/system/accounts/{acct}	DELETE	Delete account
/system/accounts/{acct}/config	GET/POST	Read/update account settings
/system/accounts/{acct}/password-reset	POST	Password reset flow
/system/accounts/{acct}/token-reset	POST	Reset auth token

6.3 Repositories

Endpoint	Method	Purpose
/system/accounts/{acct}/repositories	GET	List repositories
/system/accounts/{acct}/repositories	POST	Create repository
/system/accounts/{acct}/repositories/{repo}	DELETE	Delete repository
/system/accounts/{acct}/repositories/{repo}/config	GET/POST	Read/update settings
/system/accounts/{acct}/repositories/{repo}/collaboration	GET/POST	Collaboration management
/system/accounts/{acct}/repositories/{repo}/history	GET	History
/system/accounts/{acct}/repositories/{repo}/storage	GET	Storage
/system/accounts/{acct}/repositories/{repo}/source-statistics	GET	Source statistics
/system/accounts/{acct}/repositories/{repo}/source-history	GET	Source history
/system/accounts/{acct}/repositories/{repo}/revisions	GET	Revisions
/system/accounts/{acct}/repositories/{repo}/views/management	GET/DELETE	Views management

6.4 History

Endpoint	Method	Purpose
/system/service_history/transactions	GET	System transaction history
/system/service_history/imports	GET	System import history
/system/service_history/queries/{acct}/{sig}	GET	Query text by signature

6.5 Invitations

Endpoint	Method	Purpose
/invitations	GET	List invitations
/invitations	POST	Create/send invitation
/invitations/{email}	DELETE	Delete invitation

6.6 Graph Store

Endpoint	Method	Purpose
/ {acct} / {repo}	GET	Export (Accept header selects format)
/ {acct} / {repo}	POST	Import (merge)
/ {acct} / {repo}	PUT	Replace all
/ {acct} / {repo}	DELETE	Clear
/ {acct} / {repo} / sparql	POST	Execute SPARQL query