SMIS SSO Implementation Snapshot

Quick Install & Setup

```
npm install @smis/sso-client
```

Create a client

```javascript
import { SmisSsoClient } from '@smis/sso-client';

const client = new SmisSsoClient({
  appKey: 'pp-demo-123456',
  authBaseUrl: 'http://localhost:3000', // auth-gateway base URL
  probePath: '/sso/probe',              // optional, defaults to /sso/probe
  storageKey: 'my-app-session',         // optional; defaults to `smis-sso:
<appKey>`
  timeoutMs: 15000,                     // popup timeout (ms)
  pollIntervalMs: 60000                 // polling after popup close (ms)
});
```

Core API

Call	Description
`await client.ensureSession()`	Reuses a cached session if valid; otherwise opens the auth popup/probe.
`await client.loadAuthorizations(session?)`	Retrieves roles/permissions; `session` optional (will call `ensureSession` if omitted).
`client.getCachedSession()`	Returns the cached session if still valid, otherwise `null`.
`client.clearSession()`	Removes the cached session without contacting the server.

Convenience Actions (new)

Call	Description
`await client.signIn({ force?: boolean })`	Forces a fresh login when `force` is `true`; otherwise behaves like `ensureSession`.
`await client.signOut(session?)`	Calls `/auth/logout` (if a refresh token is present) and clears local cache.

Call	Description
`await client.switchUser()`	Clears the current session and immediately triggers a new sign-in flow.

Example Usage

```javascript
// Ensure a session (opens popup if needed)
const session = await client.ensureSession();

// Fetch roles/permissions
const authz = await client.loadAuthorizations(session);

// Get cached session (if still valid)
const cached = client.getCachedSession();

// Clear cached session without contacting server
client.clearSession();

// Sign in (forces fresh login)
await client.signIn({ force: true });

// Sign out (clears server & local state)
await client.signOut();

// Switch accounts explicitly
await client.switchUser();
```