06 Technical

Die Einstellungen Seite hat die folgenden Funktionen:

- Update von institution.settings
- Konditionales Anzeigen von Einstellungen je nach Feature Flags
- CRUD von blockierten E-Mails, blockierten Fax und Benutzern
- Anzeigen von ergänzenden Daten damit Nutzer die Settings richtig konfiguieren kann

Domains

Für jede Domain gibts zumindest ein top-level Dto sowie ein eigenes Controller / ControllerService.

Domain	Methoden	MongoDB Dokument	Implementierung	Dto
Institution	GET (list)	Institution	neu	Leicht anders wie DB; weniger properties & ohne subobjekte.
Settings	GET (id), PATCH	Institution. settings	neu	Stark anders wie DB; inkl. flattens & renames. Keine Strukturierung nach UI Gruppierung, sondern flach.
Mailbox	GET (id), PUT	-	MailService	{id, size} PUT übergibt {size: 0}
Tray	GET (list), POST, PATCH, DELETE	Tray	TrayService Aber braucht refactoring / audit	Leicht anders wie DB; weniger properties
Benutzer				
User	GET (list)	LarnagsUser	user Aber braucht refactoring / audit	Stark anders wie DB; weniger properties & flacher.
Permission	GET (list), POST, DELETE	LarnagsUser. UserRole	neu	{id, permission, institutionId, userId}
Signature	GET (list)	LarnagsUser. UserRole	neu	keines; binary blob
Blockierte Absender				
Blacklist E-Mail	GET (list), POST, DELETE	Mailbox	MailboxStorageServiceImpl (gibt aber noch remove)	Zurzeit nur String (kein eigenes Object); künstliches Objekt erstellen (damit zB bei DELETE Zuordnung Institution klappt). {id: institutionId + email, email: email} {id: "55db0d06087a069e3c1e54b4-nope@bluecare.ch", email: "nope@bluecare.ch"}
Blacklist Fax	GET (list), POST, DELETE	FaxBlacklistEntry	FaxBlacklistImpl	Zurzeit nur String (kein eigenes Object); künstliches Objekt erstellen (damit zB bei DELETE Zuordnung Institution klappt). {id: institutionId + fax, email: fax} {id: "55db0d06087a069e3c1e54b4-+41 23 231 2121", fax: "+41 23 231 2121"}

Special URLs:

- /api/me gibt eigene NutzerId und aktive InstitutionId zurück
- /api/institution/current gibt ausgewählte Institution zurück sollte optimalerweise in aufgerufener URL enthalten sein, zurzeit aber leider global state in session. Daher dieser spezielle API node.
- bei GET (id) gibt es nur eine GET single route somit muss die id dieses Objekt über ein anderes erfahren werden. zB muss InstitutionDto eine property settingsId enthalten.

Alternativen:

- Blacklist E-Mail / Fax generisches Objekt erstellen für Code Reuse ({id, entry} Aber reuse minimal.
- Settings dokument weglassen Aber Komplexität Frontend nimmt zu; besonders später.
- User nicht vereinfachen Aber Komplexität Frontend nimmt zu.
- Permission in User integrieren Aber Komplexität Frontend / Backend nimmt zu, weil permissions im User Objekt gefiltert werden müssen
- Setting (einzahl) statt Settings Aber Settings korrekter vom Inhalt her. Mehrzahlform wird voraussichtlich nicht gebraucht.
- Calls on pageload könnten direkt im HTML gecached werden Aber (premature?) performance improvement; zurzeit out of scope.

REST

Foundations:

- Authentication über Header (X-HIN- ...)

- Filters über GET parameter (/api/users?institutionId="55db0d06087a069e3c1e54b4")

API calls wenn institution tab angezeigt wird:

- /api/institution/current für aktuell ausgewählte Institution
- /api/settings/55db0d06087a069e3c1e54b4
- /api/trays?institutionId=55db0d06087a069e3c1e54b4
- /api/mailbox/55db0d06087a069e3c1e54b4

API calls für Benutzer Tab:

- /api/user/self für eigenen Nutzer
- /api/users?institutionId=55db0d06087a069e3c1e54b4
- /api/permissions?institutionId=55db0d06087a069e3c1e54b4&userId[]=5c1a0674b77c130cdf683d3d&userId[]=5db1574ce0d2c561a308fa7d
- /api/signatures?institutionId=55db0d06087a069e3c1e54b4&userId[]=5c1a0674b77c130cdf683d3d&userId[]=5db1574ce0d2c561a308fa7d

API calls für Blockierte Absender:

- /api/blacklistEmails?institutionId=55db0d06087a069e3c1e54b4
- /api/blacklistFaxes?institutionId=55db0d06087a069e3c1e54b4