RBAC Matrix - CareLinkAI

This document defines the role-based access control (RBAC) permissions for all features in CareLinkAI.

User Roles

Role	Description
`FAMILY`	Family members seeking care services
`CAREGIVER` (AIDE)	Individual caregivers offering services
`PROVIDER`	Care provider organizations
`OPERATOR`	Platform operators managing leads and operations
`ADMIN`	System administrators with full access

API Routes Permissions

Authentication Routes

Route	Method	Allowed Roles	Notes
`/api/auth/register`	POST	Public	New user registration
`/api/auth/[...nextauth]`	*	Public	NextAuth handlers
`/api/auth/session`	GET	Authenticated	Get current session

Profile Routes

Route	Method	Allowed Roles	Ownership Check
`/api/profile`	GET	Authenticated	Own profile only
`/api/profile`	PATCH	Authenticated	Own profile only
`/api/profile/photo`	POST	Authenticated	Own profile only
`/api/profile/photo`	DELETE	Authenticated	Own profile only

Family Routes

Route	Method	Allowed Roles	Notes
`/api/family/profile`	GET	`FAMILY`	Own profile
`/api/family/profile`	PATCH	`FAMILY`	Own profile
`/api/family/members/*`	*	`FAMILY`	Own family data
`/api/family/documents/*`	*	`FAMILY`	Own documents
`/api/family/residents/*`	*	`FAMILY`	Own residents

Caregiver (Aide) Routes

Route	Method	Allowed Roles	Notes
`/api/caregiver/profile`	GET	`CAREGIVER`	Own profile
`/api/caregiver/profile`	PATCH	`CAREGIVER`	Own profile
`/api/caregiver/credentials`	GET	`CAREGIVER`, `ADMIN`	Own credentials or admin
`/api/caregiver/credentials`	POST	`CAREGIVER`	Create credential
`/api/caregiver/credentials/[id]`	GET	`CAREGIVER`, `ADMIN`	Own credential or admin
`/api/caregiver/credentials/[id]`	PATCH	`CAREGIVER`	Own credential
`/api/caregiver/credentials/[id]`	DELETE	`CAREGIVER`, `ADMIN`	Own credential or admin
`/api/caregiver/credentials/upload-url`	POST	`CAREGIVER`	Generate upload URL
`/api/caregiver/availability`	*	`CAREGIVER`	Own availability

Provider Routes

Route	Method	Allowed Roles	Notes
`/api/provider/profile`	GET	`PROVIDER`	Own profile
`/api/provider/profile`	PATCH	`PROVIDER`	Own profile
`/api/provider/credentials`	GET	`PROVIDER`, `ADMIN`	Own credentials or admin
`/api/provider/credentials`	POST	`PROVIDER`	Create credential
`/api/provider/credentials/[id]`	PATCH	`PROVIDER`	Own credential
`/api/provider/credentials/[id]`	DELETE	`PROVIDER`, `ADMIN`	Own credential or admin
`/api/provider/credentials/upload-url`	POST	`PROVIDER`	Generate upload URL

Lead Routes

Route	Method	Allowed Roles	Notes
`/api/leads`	POST	`FAMILY`	Create lead
`/api/leads`	GET	`FAMILY`, `OPERATOR`, `ADMIN`	Own leads or staff
`/api/leads/[id]`	GET	`FAMILY`, `OPERATOR`, `ADMIN`	Owner or staff
`/api/leads/[id]`	PATCH	`FAMILY`, `OPERATOR`, `ADMIN`	Owner or staff

Operator Routes

Route	Method	Allowed Roles	Notes
`/api/operator/leads`	GET	`OPERATOR`, `ADMIN`	All leads
`/api/operator/leads/[id]`	GET	`OPERATOR`, `ADMIN`	Lead details
`/api/operator/leads/[id]`	PATCH	`OPERATOR`, `ADMIN`	Update lead
`/api/operator/leads/[id]/notes`	POST	`OPERATOR`, `ADMIN`	Add notes
`/api/operator/homes/*`	*	`OPERATOR`, `ADMIN`	Manage homes

Admin Routes

Route	Method	Allowed Roles	Notes
`/api/admin/users`	GET	`ADMIN`	List all users
`/api/admin/users/[id]`	GET	`ADMIN`	View user
`/api/admin/users/[id]`	PATCH	`ADMIN`	Edit user
`/api/admin/users/[id]`	DELETE	`ADMIN`	Delete user
`/api/admin/caregivers`	GET	`ADMIN`	List caregivers
`/api/admin/caregivers/[id]`	GET	`ADMIN`	Caregiver details
`/api/admin/providers`	GET	`ADMIN`	List providers
`/api/admin/providers/[id]`	GET	`ADMIN`	Provider details
`/api/admin/providers/[id]`	PATCH	`ADMIN`	Update provider
`/api/admin/credentials/[id]`	PATCH	`ADMIN`	Verify credential
`/api/admin/provider-credentials/[id]`	PATCH	`ADMIN`	Verify provider credential

Marketplace Routes

Route	Method	Allowed Roles	Notes
`/api/marketplace/caregivers`	GET	Authenticated	Browse caregivers
`/api/marketplace/caregivers/[id]`	GET	Authenticated	Caregiver details
`/api/marketplace/providers`	GET	Authenticated	Browse providers
`/api/marketplace/providers/[id]`	GET	Authenticated	Provider details
`/api/marketplace/categories`	GET	Authenticated	Get categories

Message Routes

Route	Method	Allowed Roles	Ownership Check
`/api/messages`	GET	Authenticated	Own conversations
`/api/messages`	POST	Authenticated	Can message any user
`/api/messages/threads`	GET	Authenticated	Own threads
`/api/messages/threads/[id]`	GET	Authenticated	Participant only
`/api/messages/[id]`	GET	Authenticated	Participant only
`/api/messages/[id]`	PATCH	Authenticated	Message sender only
`/api/messages/[id]`	DELETE	Authenticated	Message sender or admin

Favorite Routes

Route	Method	Allowed Roles	Ownership Check
`/api/favorites`	GET	Authenticated	Own favorites
`/api/favorites`	POST	Authenticated	Create own favorite
`/api/favorites/[id]`	DELETE	Authenticated	Own favorite
`/api/favorites/all`	GET	Authenticated	Own favorites

File Upload Routes

Route	Method	Allowed Roles	Notes
`/api/upload/presigned-url`	POST	Authenticated	Generate upload URL

UI Pages Permissions

Public Pages

Page	Access
`/`	Public
`/auth/login`	Public
`/auth/register`	Public
`/auth/reset-password`	Public

Authenticated Pages

Page	Allowed Roles	Notes
`/dashboard`	All authenticated	Role-specific dashboard
`/settings`	All authenticated	User settings
`/settings/profile`	All authenticated	Edit profile
`/settings/security`	All authenticated	Security settings
`/messages`	All authenticated	Messaging
`/favorites`	All authenticated	Favorites list

Family Pages

Page	Allowed Roles
`/settings/family`	`FAMILY`
`/homes/*`	`FAMILY`
`/leads`	`FAMILY`

Caregiver Pages

Page	Allowed Roles
`/settings/credentials`	`CAREGIVER`, `PROVIDER`
`/settings/availability`	`CAREGIVER`

Provider Pages

Page	Allowed Roles
`/settings/provider`	`PROVIDER`
`/settings/services`	`PROVIDER`

Operator Pages

Page	Allowed Roles
`/operator/leads`	`OPERATOR`, `ADMIN`
`/operator/leads/[id]`	`OPERATOR`, `ADMIN`
`/operator/homes`	`OPERATOR`, `ADMIN`
`/operator/homes/[id]`	`OPERATOR`, `ADMIN`

Admin Pages

Page	Allowed Roles
`/admin/*`	`ADMIN`
`/admin/users`	`ADMIN`
`/admin/aides`	`ADMIN`
`/admin/aides/[id]`	`ADMIN`
`/admin/providers`	`ADMIN`
`/admin/providers/[id]`	`ADMIN`

Marketplace Pages

Page	Allowed Roles	Notes
`/marketplace`	Authenticated	Browse all
`/marketplace/caregivers`	Authenticated	Browse caregivers
`/marketplace/caregivers/[id]`	Authenticated	Caregiver profile
`/marketplace/providers`	Authenticated	Browse providers
`/marketplace/providers/[id]`	Authenticated	Provider profile

Resource Ownership Rules

Profile Data

- Users can **view** and **edit** their own profile

- Admins can **view** and **edit** any profile
- Operators can **view** any profile

Leads

- Families can **create** leads
- Families can **view** and **edit** their own leads
- Operators can **view** and **edit** all leads
- Operators can **assign** leads to themselves
- Admins have full access

Credentials

- Caregivers/Providers can **create**, **view**, and **edit** their own credentials
- Admins can **view** all credentials
- Admins can **verify** credentials
- Operators can **view** credentials (for verification purposes)

Messages

- Users can **send** messages to any user
- Users can only **view** conversations they are part of
- Users can only **edit/delete** their own messages
- Admins can view and moderate all messages

Favorites

- Users can **create** and **delete** their own favorites
- Users cannot view other users' favorites

Permission Helper Functions

These functions are available in `@/lib/auth/rbac`:

```
// Authentication
const session = await requireAuth();

// Role checks
const session = await requireRole(UserRole.FAMILY);
const session = await requireAnyRole([UserRole.OPERATOR, UserRole.ADMIN]);
const session = await requireAdmin();
const session = await requireOperator();
const session = await requireStaff(); // Admin or Operator

// Ownership checks
requireOwnership(session, resourceUserId); // User or Admin
requireOwnershipOrStaff(session, resourceUserId); // User, Operator, or Admin

// Boolean checks
if (hasRole(session, UserRole.ADMIN)) { ... }
if (hasAnyRole(session, [UserRole.OPERATOR, UserRole.ADMIN])) { ... }
if (isAdmin(session)) { ... }
if (isOperator(session)) { ... }
if (isStaff(session)) { ... }
if (ownsResource(session, resourceUserId)) { ... }
```

Permission Definitions

Predefined permission checks are available in `Permissions` object:

```
import { Permissions } from '@/lib/auth/rbac';

// Profile permissions
if (Permissions.profile.view(session, userId)) { ... }
if (Permissions.profile.edit(session, userId)) { ... }

// Lead permissions
if (Permissions.lead.create(session)) { ... }
if (Permissions.lead.view(session, leadFamilyId)) { ... }
if (Permissions.lead.assign(session)) { ... }

// Credential permissions
if (Permissions.credential.verify(session)) { ... }

// Admin permissions
if (Permissions.admin.manageProviders(session)) { ... }
```

Client-Side Role Gates

For conditional rendering in React components:

```
import { useIsAdmin, useHasAnyRole, RoleGate, AdminOnly } from '@/lib/auth/client-
rbac';

// Hooks
const isAdmin = useIsAdmin();
const isStaff = useHasAnyRole([UserRole.ADMIN, UserRole.OPERATOR]);

// Components
<RoleGate roles={[UserRole.ADMIN, UserRole.OPERATOR]}>
  <StaffOnlyContent />
</RoleGate>

<AdminOnly>
  <AdminPanel />
</AdminOnly>
```

Testing RBAC

Test Cases

1. **Unauthorized Access**
   - Try accessing authenticated routes without login → 401

2. **Role Mismatch**
   - Family user tries to access `/operator/*` → 403
   - Caregiver tries to access `/admin/*` → 403

3. **Ownership Violation**
   - User A tries to edit User B’s profile → 403
   - Family A tries to view Family B’s leads → 403

4. **Admin Override**
   - Admin can view any profile → 200
   - Admin can verify any credential → 200

5. **Staff Access**
   - Operator can view leads → 200
   - Operator can assign leads → 200

6. **Resource Creation**
   - Only Family can create leads → 200 for Family, 403 for others
   - Only Caregivers can create credentials → 200 for Caregiver, 403 for others