**The Complete Authentication Feature Matrix**

**Sign-in / Sign-up (core)**

* Username/email + password (session & API)
* Email verification (mandatory/optional; resend; verification expiry)
  + After verification, send email verified email.
* Password reset via email link
* Change password (with re-auth for sensitive actions)
* Account lock/unlock; disable/enable; soft delete & hard delete
* Remember-me (long-lived refresh or session)
* Login by username **or** email; optional phone-based login
* “Re-authenticate” challenge for sensitive endpoints

**Password less & One-time links/codes**

* Magic link (email login link with single-use token)
* Email OTP (6–8 digits, short TTL, throttled)
* SMS OTP (via Twilio/etc., with regional sender support)
* TOTP (authenticator apps: Google/Microsoft Authenticator, Authy)
* Backup codes (10 single-use)
* **WebAuthn / Passkeys** (FIDO2, device biometrics, security keys)

**Multi-Factor Authentication (2FA/MFA)**

* Step-up MFA on risky events (new device, geo-anomaly)
* Methods: TOTP, SMS, Email OTP, Push (optional via a push provider), WebAuthn
* Enforce per-user/per-group/per-org policy (e.g., admins require MFA)
* Recovery flows (backup codes, admin recovery)

**Social / Federated Login**

* OAuth 2.0 / OIDC: Google, Apple, Microsoft, GitHub, Facebook, Discord, Slack, Twitter(X)
* Google One Tap (OIDC) (optional)
* Enterprise SSO: **OIDC** (recommended), **SAML 2.0** (optional for org tenants)
* PKCE for public clients, nonce/state protection
* Account linking/unlinking (multiple social identities per user)

**API Auth (for SPAs/mobile/services)**

* **JWT** access + **refresh** tokens (rotating refresh recommended)
* Sliding sessions or absolute/relative expiry
* Token blacklisting/denylist & immediate revocation
* Optional **DPoP/Proof-of-possession** (advanced) and **mTLS** (service-to-service)
* Personal **API keys/tokens** (scoped, expiring, rotated)

**Sessions, Devices, and Security Posture**

* Session store (server-side), device tracking (UA, IP, geo, last seen)
* Device management UI (sign out specific device, revoke refresh tokens)
* Concurrent session limits, session idle timeout vs absolute lifetime
* Geo/IP allow/deny lists (per org/user)
* Suspicious login detection & alerts (new device/location)

**Permissions & Multi-tenancy**

* RBAC (roles: owner/admin/manager/member) at org/team/project scopes
* Optional **ABAC** (attribute-based rules) for advanced policies
* Object-level permissions (e.g., per Company/Farm/Project)
* Organization & team memberships; invitations with **magic links**
* Domain restrictions (e.g., only @company.com sign-ups)

**Account Lifecycle, Compliance & UX**

* Email/phone verification, consent capture (TOS/Privacy), marketing opt-in
* Age gating / parental consent (COPPA-style flows)
* Data export (user portal) & **account deletion** (GDPR/CCPA support)
* Password policy: strength (zxcvbn), breach checks (HIBP k-anon), history
* Username policy & reserved names
* Localized emails/SMS; theming
* Accessibility, CAPTCHA after failures (hCaptcha/reCAPTCHA), rate limits

**Observability & Admin Ops**

* Audit log (logins, MFA enrollments, policy changes, token events)
* Admin dashboard: search users, status/MFA flags, impersonation (with audit)
* Alerts: excessive failures, impossible travel, brute force lockouts
* Secrets rotation: JWT signing keys, OAuth creds, API keys

**🧩 1. Registration & Login**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /register/ | POST | RegisterView | Register a new user and send verification email (via allauth). |
| /login/ | POST | LoginView | Login via email/username/phone + password (supports Remember Me + MFA). |
| /logout/ | POST | LogoutView | Logout (clears session/JWT). |
| /me/ | GET | MeView | Returns current authenticated user details. |

✅ **All implemented correctly**.

**✉️ 2. Email Verification**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /email/verify/ | POST | VerifyEmailView | Verifies user email with UID/token. |
| /resend-verification/ | POST | ResendVerificationEmailView | Re-sends verification email (works logged-in or by email). |

✅ Already implemented and fully working.

**🔑 3. Password Management**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /password/forgot/ | POST | ForgotPasswordView | Sends reset email with token. |
| /password/reset/ | POST | ResetPasswordView | Confirms reset with token. |
| /password/change/ | POST | ChangePasswordView | Change password after login. |

✅ Fully implemented.

**✨ 4. Passwordless Login (Magic Link, OTP, etc.)**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /magic/request/ | POST | MagicLinkRequestView | Request a magic login link via email. |
| /magic/consume/ | POST | MagicLinkConsumeView | Consume a magic link (JWT/session issued). |
| /otp/email/request/ | POST | EmailOTPRequestView | Request email-based OTP. |
| /otp/email/verify/ | POST | EmailOTPVerifyView | Verify email OTP and login. |
| /otp/sms/request/ | POST | SMSOTPRequestView | Request SMS-based OTP via Twilio. |
| /otp/sms/verify/ | POST | SMSOTPVerifyView | Verify SMS OTP and login. |

✅ **Implemented and consistent.**

**🔒 5. MFA / Authenticator Apps**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /mfa/totp/setup-begin/ | POST | TOTPSetupBeginView | Generate a secret + otpauth URL for app (Google Authenticator, etc.). |
| /mfa/totp/confirm/ | POST | TOTPConfirmView | Confirm TOTP code to enable 2FA. |
| /mfa/verify/ | POST | MFAVerifyView | Verify MFA during login. |
| /mfa/backup-codes/generate/ | POST | BackupCodesGenerateView | Generate 10 backup codes. |
| /mfa/backup-codes/verify/ | POST | BackupCodeVerifyView | Login via backup code. |

✅ Fully implemented.

**🪪 6. WebAuthn / Passkeys**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /webauthn/register/begin/ | POST | WebAuthnRegisterBeginView | Start passkey registration (creates challenge). |
| /webauthn/register/complete/ | POST | WebAuthnRegisterCompleteView | Complete passkey registration. |
| /webauthn/auth/begin/ | POST | WebAuthnAuthBeginView | Begin WebAuthn authentication challenge. |
| /webauthn/auth/complete/ | POST | WebAuthnAuthCompleteView | Complete authentication and issue JWT. |

✅ These are **stubbed correctly**, can later integrate python-fido2.

**🧱 7. Account Management (Admin & User-Initiated)**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /accounts/<id>/lock/ | POST | AccountLockView | Admin locks account. |
| /accounts/<id>/unlock/ | POST | AccountUnlockView | Admin unlocks account. |
| /accounts/<id>/disable/ | POST | AccountDisableView | Admin disables account. |
| /accounts/<id>/enable/ | POST | AccountEnableView | Admin re-enables account. |
| /accounts/<id>/soft-delete/ | POST | AccountSoftDeleteView | Admin soft deletes account. |
| /accounts/<id>/restore/ | POST | AccountRestoreView | Admin restores deleted account. |
| /accounts/<id>/hard-delete/ | DELETE | AccountHardDeleteView | Admin permanently deletes account. |
| /suspend/ | POST | SuspendAccountView | User self-suspends account (requires reauth). |
| /request-delete/ | POST | RequestDeleteAccountView | User requests account deletion (cancelable). |
| /cancel-delete/ | POST | CancelDeleteAccountView | User cancels deletion request. |
| /delete/ | DELETE | HardDeleteAccountView | User self-deletes permanently (requires reauth). |

✅ All implemented, integrated with notify\_user() and reauth protection.

**🔁 8. Re-Authentication**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /api/auth/reauth/ | POST | ReAuthView | Forces users to confirm credentials before sensitive actions (e.g. delete/suspend). |

✅ Implemented and active.

**🧰 9. Admin / Utility**

| **Endpoint** | **Method** | **View** | **Description** |
| --- | --- | --- | --- |
| /unlock/ | POST | unlock\_user() | Manual unlock endpoint for admin (redundant if using /accounts/<id>/unlock/). |
| /accounts/<id>/<action>/ | POST | AccountActionView | Generic admin action handler (alternative to individual lock/unlock routes). |

🟡 **Note:** these are redundant given the dedicated admin routes.  
👉 You can **keep one unified system** — preferably the AccountActionView to manage all via action param.

**🧩 10. Missing (Suggested Additions)**

| **Missing Endpoint** | **Description** | **Suggested Path** |
| --- | --- | --- |
| GET /accounts/deleted/ | List of all soft-deleted or suspended accounts for admin. | /accounts/deleted/ |
| GET /accounts/suspended/ | Admin can view suspended users. | /accounts/suspended/ |
| GET /accounts/pending-deletion/ | List of user accounts that requested deletion. | /accounts/pending-deletion/ |
| POST /auth/webauthn/challenge/verify/ | Optional endpoint to validate browser passkey challenges. | /webauthn/verify/ |

Would you like me to generate the **missing admin listing endpoints** (deleted / suspended / pending delete) next?  
They’ll include:

* pagination,
* print/log statements for debugging,
* optional filters (status, date, email),
* and admin-only permissions.