

InputValidation.pdf

TP1 - Input validation rules + error conditions (New User + Admin flows).

Validation Rules + Field Inventory

Rule ID	Field / Feature	Rules (summary)	Example error message
VAL-00	All text inputs	Trim; required if field required; reject control chars; enforce max length	"Invalid input." / "Field is required."
VAL-01	Email	Non-empty; trimmed; email format; lowercase; max length; uniqueness when registering/updating	"Please enter a valid email address."
VAL-02	Password	8-32 chars; >=1 upper, >=1 lower, >=1 digit; allowed specials only; not common/blocked; detect too-long	"Password must be 8-32 chars incl. upper/lower/digit."
VAL-03	Username / Name	Trimmed; 3-20; letters/numbers/_; unique	"Username must be 3-20 letters/numbers/underscore, unique."
VAL-04	Shared Validator	Use central helpers/constants so UI + services enforce same rules + messages	Messages reused everywhere

Error Menu (what can be thrown / shown)

Keep UI messages short; store them in one shared place. For login, prefer a generic message.

All text fields (VAL-00)

- ERR_EMPTY / required - blank or null
- ERR_TOO_LONG - exceeds max length (define per-field; also enforce a hard cap)
- ERR_INVALID_CHAR - contains control chars (tabs/newlines/null) or disallowed symbols
- ERR_WHITESPACE_ONLY - only spaces after trim (if treated as empty)

Email (VAL-01)

- ERR_EMAIL_EMPTY - required
- ERR_EMAIL_TOO_LONG - exceeds max
- ERR_EMAIL_INVALID - fails email format check (missing @, bad domain, etc.)
- ERR_EMAIL_TAKEN - already in use (registration/change email)

Username / Name (VAL-03)

- ERR_USERNAME_EMPTY - required
- ERR_USERNAME_TOO_SHORT / ERR_USERNAME_TOO_LONG - outside 3-20
- ERR_USERNAME_INVALID_CHAR - only allow letters, digits, underscore (team rule)
- ERR_USERNAME_TAKEN - uniqueness violation

Password (VAL-02)

Allowed special characters: ~ ! @ # \$ % ^ & * () _ - + { } [] | : ; , . ? /

- ERR_PASSWORD_EMPTY - required
- ERR_PASSWORD_TOO_SHORT - fewer than 8 characters

- ERR_PASSWORD_TOO_LONG - exceeds max (team chose 32; still enforce hard cap)
- ERR_PASSWORD_MISSING_UPPER - no A-Z
- ERR_PASSWORD_MISSING_LOWER - no a-z
- ERR_PASSWORD_MISSING_DIGIT - no 0-9
- ERR_PASSWORD_INVALID_CHAR - contains a character outside [A-Z a-z 0-9 allowed special]
- ERR_PASSWORD_COMMON - on blocked/common list (if used)
- ERR_CURRENT_PASSWORD_INCORRECT - change-password flow (don't over-explain)

Roles, selections, and admin targets

- ERR_SELECTION_REQUIRED - role not selected / missing dropdown value
- ERR_TARGET_NOT_FOUND - target user id does not exist
- ERR_PROTECT_LAST_ADMIN - cannot deactivate or demote the last admin

Login / account state

- ERR_CREDENTIALS_INVALID - username/email + password don't match (generic message)
- ERR_ACCOUNT_DISABLED - deactivated/locked account

User Story Validation Coverage

New User Feature Validations

Story	Fields	Validation
NU-01 Register	Email, Password, Username/Name	Apply VAL-01/02/03; enforce uniqueness; hash password
NU-02 Sign in	Email, Password	Apply VAL-01/02; show generic "Invalid credentials" on failure
NU-03 Update Profile	Username	Apply VAL-03; check duplicates
NU-04 Change Password	Current PW, New PW	Current must match; New must satisfy VAL-02; force re-login/session refresh
NU-05 Change Email	New Email	Apply VAL-01; (optional) mark unverified until confirmed

Admin Feature Validations

Story	Fields	Validation
ADM-01 List Users	Filter Email / Role	If email provided: VAL-01; role must be valid enum
ADM-02 Activate/Deactivate	Target User ID	Target must exist; protect last admin; avoid self-lockout
ADM-03 Change User Role	Target User ID, Role	Target must exist; role valid enum; protect last admin
ADM-04 Reset Password	Target User ID, Temp PW	Target must exist; Temp must satisfy VAL-02; user must change after login (if implemented)

Cross-Cutting

- XQ-01: All error messages pulled from a single source (constants or validator API).
- XQ-02: Security-sensitive actions logged minimally (no passwords; no full PII).
- XQ-03: Keep validation helpers consistent across UI + service layers.