Njangui

Functional Guide (English)

Zekeng Ndadji Milliam Maxime

2026-04-06

Abstract

This functional guide documents Njangui from the operational point of view. It details user journeys, governance rules, tontine and finance workflows, notifications, penalties, and reporting behavior for daily usage and stakeholder alignment.

Contents

1 Introduction

1.1 Why Njangui exists

Njangui solves a real operational problem: helping a community manage its full financial and governance lifecycle in one application.

It covers: - tontine operations, - caisse accounting, - events and main levees, - disciplinary penalties, - staff governance, - financial and disciplinary reporting.

1.2 Product position

Njangui is designed as a **rich client**: - business rules are implemented in the mobile app, - Firebase is used as persistence/synchronization, - architecture keeps room for a future backend replacement without rewriting all user workflows.

1.3 Guiding principles

1. **Business clarity**: users should understand actions and consequences.
2. **Traceability**: sensitive operations are auditable in reports.
3. **Real governance**: staff rights drive operations, not a single super-user model.
4. **Schema compatibility**: staging and historical data must remain usable during evolution.
5. **Safe operations**: confirmations, validation constraints, explicit status transitions.

1.4 Actors and responsibilities

- **Member**: participates in draws, rounds, votes (if eligible), tracks own sanctions and finances.
- **Staff MANAGE_MEMBERS**: member lifecycle, invitations, staff administration.
- **Staff MANAGE_TONTINES**: tontine setup, draw rules, ranking lifecycle.
- **Staff MANAGE_CAISSES**: caisse operations, financial closures and allocations.
- **Staff MANAGE_EVENTS**: events and main levee workflows.
- **Staff MANAGE_PENALTIES**: penalty validation, monitoring and closure.
- **Group creator/admins**: structural administration (group name/admins/currency/delete) under governance constraints.

1.5 Key glossary

- **Tontine group (Njangui)**: the group workspace.
- **Tontine**: contribution cycle with ranking and winners.
- **Round**: one contribution period of a tontine.
- **Caisse**: financial account (individual or shared).
- **Main levee**: dedicated collection campaign.
- **Penalty**: disciplinary item with amount and/or textual sanctions.

1.6 Baseline reference

- versionName: 1.1.0
- versionCode: 4
- code name: AJEMI−UDs

2 User Journeys

2.1 End-to-end view

A complete journey usually follows:

1. authentication,
2. active group selection,
3. role-based operations per module,
4. notification-driven follow-up,
5. filtered report consultation/export.

2.2 Journey 1 - Regular member

2.2.1 A. Access

From Login, the user can: - sign in, - register, - reset password.

2.2.2 B. Join group

- Receives invitation.
- Accepts/declines.
- On acceptance, becomes active member with role-based actions.

2.2.3 C. Daily usage

- Checks current tontine and draw/ranking state.
- Follows round contribution requirements.
- Monitors personal sanctions and finances.

2.3 Journey 2 - Member and staff administration

Actor: staff with MANAGE_MEMBERS.

Can: - invite/remove members, - register initial staff, - create elections (when eligible), - manage replacement workflows requiring approvals.

Critical controls: - mandate state, - post-expiry eligibility window, - confirmation dialogs for high-impact actions.

2.4 Journey 3 - Tontine management

Actor: staff with MANAGE_TONTINES.

Typical flow:

1. create tontine,
2. configure ranking preconditions,
3. run/close ranking,
4. manage rounds until tontine ends.

Business constraints include: - start-date constraints relative to round duration, - participant vs random ranking modes, - ranking reset when rules are changed.

2.5 Journey 4 - Cashier and penalty manager

2.5.1 Cashier (`MANAGE_CAISSES`)

- manages individual/shared caisses,
- records accounting operations,
- closes financial steps for rounds/events,
- exports filtered finance views.

2.5.2 Penalty manager (`MANAGE_PENALTIES`)

- validates/completes penalties,
- tracks due and paid states,
- coordinates recovery closure,
- exports filtered disciplinary reports.

2.6 Journey 5 - Group creator/admin

- updates group display name (stable group ID),
- manages admin list,
- can delete group,

- handles governance fallback actions when staff is missing/expired.

2.7 Navigation map

Drawer modules: - Overview - Tontines - Members & Staff - Caisses - Events - Finance - Penalties - Settings

Theme/language changes are global and should apply without breaking navigation.

2.8 Cross-screen UX expectations

- consistent loaders,
- reliable post-action refresh,
- pull-to-refresh on all operational pages,
- background + foreground notification delivery.

3 Functional Modules

3.1 Dynamic domain map

```
flowchart LR
    A[Overview] --> B[Members & Staff]
    A --> C[Tontines]
    A --> D[Caisses]
    A --> E[Events]
    A --> F[Penalties]
    A --> G[Finance]
    A --> H[Settings]
    C --> D
    C --> F
    D --> G
    E --> D
    E --> F
    F --> G
```

3.2 1. Overview

Operational hub for the active group: - invitations, - governance actions, - group administration, - current-round highlights for active tontines.

3.3 2. Members & Staff

3.3.1 Member lifecycle

- invitation management,
- active/inactive monitoring,
- member removal with lifecycle-aware impact.

Removal rule: - not-started tontines: member is removed from participants/ranking, - ongoing/past tontines: historical data is preserved.

3.3.2 Staff lifecycle

- initial staff registration,

- election creation and voting,
- replacement requests with approval thresholds.

Permissions depend on: - active mandate, - expiry timing window, - delegated rights.

3.4 3. Tontines

3.4.1 Creation inputs

- individualAmountPerRound
- startDate
- roundDuration
- maxWinnerPerRound
- participants (with totalNames support)

3.4.2 Ranking and draw

Preconditions: - multipleNamesStrategy (NONE / BALANCED) - fixed positionning - rankingType (RANDOM_ALGORITHM / PARTICIPANT_SELECTION) - deadlineForParticipantSelection

Core rules: - winners are grouped by maxWinnerPerRound, - closing can auto-complete missing picks, - precondition updates invalidate existing draws and require re-draw.

3.4.3 Rounds

- notify all members at round start,
- send reminder near deadline,
- caisse managers close rounds with contributions/outflows,
- successful rounds are archived,
- tontine ends automatically when all slots are completed.

3.5 4. Caisses

Supported models: - individual caisse (per-member balance), - shared caisse (single collective balance).

Capabilities: - CRUD caisse, - accounting operations (credit/debit), - temporary low-balance alert suspension, - shared caisse distribution, - filtered transaction view and export.

All amounts use the group's configured currency.

3.6 5. Events

Supported flows: - events with individual contributions, - events with collective contribution, - internal/external beneficiaries, - prioritized caisse outflow deduction, - outflow rollback for correction.

Automatic effects: - caisse debits, - penalties generated on deficits/late rules.

3.7 6. Penalties

Penalty lifecycle:

1. generated or manually created,
2. validated,
3. notified,
4. marked paid,
5. recovery closed (with allocation destinations).

A penalty may include: - amount-based fines, - text-based sanctions, - context (tontine/round/free text), - due date.

3.8 7. Finance

Provides: - filtered views by period/member/context, - consolidated aggregates, - single export action based strictly on filtered dataset.

3.9 8. Settings

- profile data,
- password changes,
- preferences (language/theme/default group),
- account logical deactivation.

3.10 9. Notification coverage

Current notification families include: - invitations + invitation feedback, - tontine creation, - ranking lifecycle (pick, deadline, completion, changes, rules updates), - round lifecycle (start, reminder, closure required, completed, ended), - governance lifecycle (election, reminders, results, missing staff, replacements), - penalties lifecycle (review, validation, due, recovery), - low-balance caisse reminders, - events/main levee lifecycle signals.

Notifications are persisted in Firestore and pulled by the background worker for device delivery.

4 Screen Mockups and UX Scenarios

This section provides “living mockups”: structure + behavior expectations + critical states.

4.1 1. Navigation map

```
Auth (Login/Register/Reset)
  -> Overview (active group)
     -> Members & Staff
     -> Tontines
     -> Caisses
     -> Events
     -> Penalties
     -> Finance
     -> Settings
```

4.2 2. Login screen

```
+-------------------------------------------------------+
|                        NJANGUI                        |
|  Subtitle: Organize your community with confidence    |
|                                                       |
|  Email      [_____________________________]           |
|  Password   [_____________________________]           |
|                                                       |
|  [ Sign in ]                                          |
|                                                       |
```

```
|  Create account          Forgot password                |
+---------------------------------------------------------+
```

Expected states: - action loader while authenticating, - clear feedback for unknown/deactivated account, - safe redirect on success.

4.3 3. Overview dashboard

```
+---------------------------------------------------------+
| Active group [Select]                  [Refresh]        |
+---------------------------------------------------------+
| KPI cards: Active members | Invitations | Staff         |
+---------------------------------------------------------+
| Governance: [Register Staff] [Create Election]          |
|             [Vote] [Approve Replacement]                |
+---------------------------------------------------------+
| Group admin: [Rename] [Currency] [Admins] [Leave]       |
+---------------------------------------------------------+
| Active tontines: current round + beneficiaries          |
+---------------------------------------------------------+
```

4.4 4. Tontines screen

```
+---------------------------------------------------------+
| Current tontine [Select by name]                        |
+---------------------------------------------------------+
| Tontine summary card: status, current round, winners    |
+---------------------------------------------------------+
| Actions: [Ranking] [Rules] [Round operations]           |
+---------------------------------------------------------+
| My tontines list (cards)                                |
|  - Name | Status | Rights-based actions                 |
+---------------------------------------------------------+
```

UX notes: - selected tontine must be obvious, - avoid duplicate controls in top area and cards, - clear visual spacing between functional blocks.

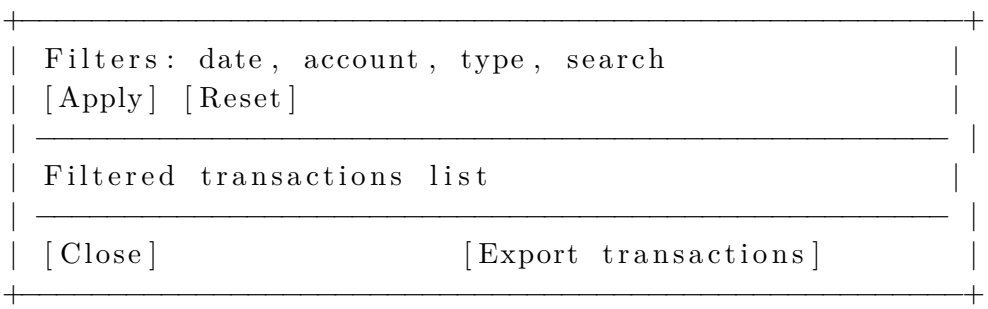
4.5 5. Critical dialogs

4.5.1 Ranking result dialog

```
+---------------------------------------------------------+
| Ranking results                                         |
| ... ordered slots ...                                   |
|                                                         |
| [ Close ]                      [ Close ranking ]        |
+---------------------------------------------------------+
```

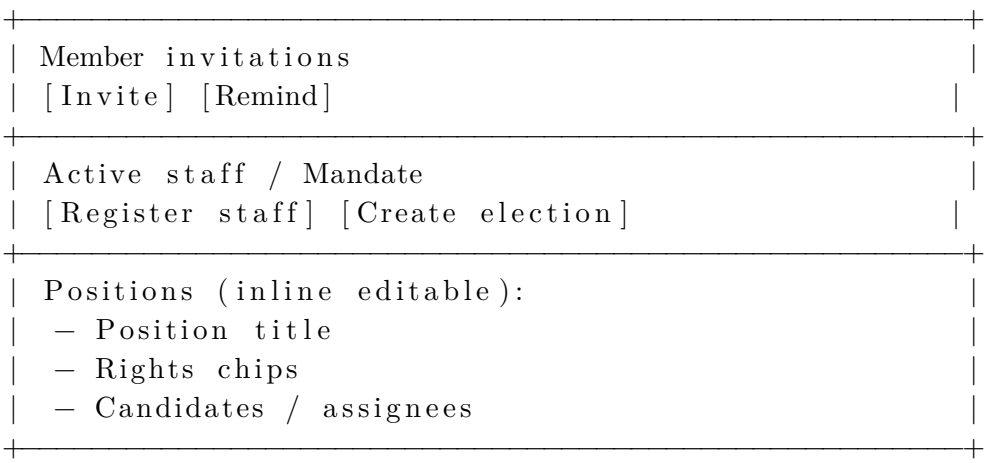
Action hierarchy: - secondary action on left, - destructive/high-impact action emphasized on right, - explicit confirmation before irreversible transition.

4.5.2 Caisse transactions dialog



Export button must stay reachable on small devices.

4.6 6. Members & Staff screen



4.7 7. Cross-module interaction rules

For Caisses, Events, Penalties, Finance: - top filters, - aggregate summary card, - detailed list, - rights-aware actions, - one export button scoped to current filtered data.

4.8 8. Cross-screen states

Every page should expose: - initialization loader, - action loader, - guided empty state, - recoverable error state, - pull-to-refresh.

5 FAQ and Support

5.1 1. First checks before debugging

Always verify: - active group selection, - user effective role and rights, - staff mandate state (active/expired), - network connectivity, - data freshness (pull-to-refresh).

5.2 2. Common questions

5.2.1 I cannot see an expected action

Most frequent cause: rights or timeline condition not satisfied.

Checklist: 1. Is the user an active member? 2. Is required MANAGE_* right granted? 3. Does staff mandate state allow this action now? 4. Is the target tontine/event in the required status?

5.2.2 Data seems stale after an operation

Recommended actions: - pull-to-refresh, - leave/re-enter screen, - verify loaders and completion callbacks, - verify Firestore write actually succeeded.

5.2.3 Notifications are not delivered

Check: - device notification permission, - authenticated user session, - worker scheduling state, - notifications /<groupId> document fields (type, recipients , contentData, …).

5.2.4 Round closure fails

Possible causes: - ranking not complete/closed, - missing MANAGE_CAISSES right, - incomplete contribution/outflow payload, - round already closed.

5.2.5 Why is a penalty not visible to everyone yet?

Penalty visibility is lifecycle-dependent: - creation, - validation, - notification, - payment, - recovery closure.

5.3 3. Quick resolution runbook

5.3.1 Permissions issue

1. Inspect member/staff profile.
2. Confirm rights attached to current position.
3. Check mandate start/end and expiry window.
4. Reproduce with another user having same rights.

5.3.2 Firestore schema issue

1. Verify expected fields presence.
2. Validate fallbacks for missing fields.
3. Re-test with partially migrated documents.
4. Add targeted logs around read/write boundaries.

5.3.3 PDF export issue

1. Confirm filters currently applied.
2. Confirm filtered dataset is non-empty.
3. Confirm storage/sharing permissions.
4. Confirm automatic file opening flow.

5.4 4. Operating best practices

- use confirmation dialogs for high-impact actions,
- avoid changing ranking rules after draw starts unless intentional reset is accepted,
- use filtered reports in governance and finance meetings,
- avoid off-workflow financial edits that bypass accounting history.

5.5 5. Escalation template for bug reports

Capture at minimum: - module + screen, - user role, - exact action taken, - expected vs actual result, - logs/stacktrace with timestamp, - affected group/tontine identifiers.

This structure shortens diagnosis and reduces back-and-forth.

6 Cross-Domain Business Index

Purpose: map business rules to practical usage, with a functional focus (minimal technical vocabulary).

6.1 1. Who does what and why

Domain	Key action	Who can do it	Business impact
Members	Invite or remove member	Member management role	Updates real group composition
Governance	Create election / register staff	Role and mandate dependent	Activates governance rights
Tontines	Configure ranking and close draw	Tontine management role	Defines beneficiary order
Rounds	Close a tontine round	Caisse management role	Archives round and may generate penalties
Caisses	Record credit/debit operations	Caisse management role	Ensures financial traceability
Events	Close contributions/outflows	Caisse + events roles	Direct balance and discipline impact
Penalties	Validate / mark paid / close recovery	Penalty + caisse roles	Updates disciplinary and financial state
Reports	Export report	Authorized module user	Produces filtered audit-ready document

6.2 2. Sensitive actions (confirmation required)

Action	Why sensitive
Update ranking rules	May invalidate already completed picks
Close incomplete ranking	Triggers automatic picks for missing participants
Close round	Creates archives and potential penalties
Delete caisse	Also removes linked accounting entries
Delete tontine / group	Major structural impact
Close penalty recovery	Finalizes monetary allocation

6.3 3. Notification map (functional view)

Family	Examples	Target audience
Invitations	invite, accept, decline	members and coordinators
Tontines	created, ranking updates, position changes	impacted participants
Rounds	started, reminder, closure	participants + caisse managers
Governance	election, vote reminder, results	members and staff
Penalties	validation, due reminder, recovery	impacted member + penalty/caisse staff
Caisses	low-balance warning	impacted members
Events / main levees	outflow required, closure	caisse staff + impacted members

6.4 4. Export contract

Global rule in the app: - one export action per report area, - export must use only currently filtered data.

Module	Typical filters
Caisse transactions	date, account, type, search
Finance	period, member, context
Penalties	type, period, context, member

6.5 5. Support and QA quick path

1. Confirm active group.
2. Confirm effective user right.
3. Confirm lifecycle state (mandate, ranking, round, penalty).
4. Refresh before diagnosis.
5. Reproduce on staging-like data when possible.