Embedded Borrower Wireframe Explanation and Lender Cloud System Recommendation for Kiva

Bradley D. Saucier

Southern New Hampshire University

CS-319 UI/UX Design and Development

Professor Jasmine

February 16, 2026

Introduction

The borrower and lender interfaces share one dataset but operate under fundamentally different constraints. The borrower interface is an embedded, USSD-style workflow intended for low bandwidth, short session timeouts, and numeric-only input. USSD is menu-driven and session-based, so each additional screen and keystroke increases timeout risk. The lender interface is a cloud application built for exploration and forecasting, but nonprofit budgets require cost discipline in every architectural choice. The dataset's primary constraint is the absence of repayment transaction history. Repayment progress therefore relies on interval and term as proxies; consequently, all derived outputs require "EST" (estimate) labels to maintain data honesty. This constraint shaped every layout decision in the borrower wireframe and every forecast view in the lender recommendation.

Embedded Borrower Wireframe - Purpose and Function of Each Screen

Screen 1 - Entry Dashboard (KIVA USSD)

Screen 1 delivers actionable status immediately with minimal navigation. The objective: communicate status and enable action before the session times out.

1. Context and group (Header and GRP: WOMEN ENTERPRISE)

   The header establishes service context at entry. The group indicator is critical because group loans create shared-liability obligations, and the borrower must confirm the correct account context before acting.

2. Status and directive (STATUS: ACTIVE; NEXT: PAY BY OCT 15)

   STATUS provides current loan state. NEXT replaces raw system states with a single instruction the borrower can act on. The borrower receives a clear directive rather than a raw status code, saving both keystrokes and critical seconds.

3. Repayment structure (INTERVAL: MONTHLY; TERM: 12 MO)

   Interval and term are critical because many borrowers do not have steady monthly income. Showing repayment structure on the first screen supports immediate planning for seasonal or irregular earnings without deeper navigation.

4. Loan identifier (ID: 1234567)

   LOAN_ID is a critical support reference. It accelerates dispute resolution and partner support by giving the borrower one stable identifier to report when sessions drop, or records must be reconciled offline.

5. Numeric menu and soft keys (1-3 options; 0 Exit; LSK Select; RSK Back)

   USSD interaction is numeric selection. The menu is intentionally flat and predictable to reduce keystrokes and avoid deep navigation. The "0 Exit" function provides a clean termination path and reduces disclosure risk on shared devices. The design assumes shared-device use and treats fast exit as a privacy control.

Screen 2 - Payment Schedule

Screen 2 is the single secondary screen. It supports repayment planning and safe entry for a high-stakes action.

1. Repayment schedule fields (DUE, AMT, REM)

   The screen presents the next due date and schedule-derived values marked EST, consistent with the snapshot constraint described above.

2. EST labeling (AMT: 500 KES (EST))

   The EST label enforces data honesty. Without that label, a borrower could mistake a schedule estimate for a confirmed balance - the kind of silent error that erodes trust.

3. Amount entry and confirmation (Enter payment amt; LSK Confirm)

   Numeric keypad entry is error prone. The workflow uses an echo-back confirmation step to prevent accidental overpayment or underpayment before any commitment is made.

How the Design Benefits the Borrower and Kiva

Borrowers benefit

The interface supports a predictable sequence under stress: confirm context (group and loan ID), confirm state (status), read directive (next), then select an action. This reduces decision time and input burden, making it more likely the borrower finishes before the session drops. Displaying interval and term on Screen 1 reduces confusion for borrowers with seasonal income by making repayment structure explicit at first contact.

Kiva benefit

Kiva sees two operational returns: less support overhead and clearer repayment signaling. The interface replaces raw system states with one actionable instruction, reduces entry errors through confirmation, and preserves trust by labeling estimates when the dataset cannot provide confirmed repayment history. For operations, fewer mis-keyed payments mean fewer reconciliation tickets for field partners.

Innovations and Research Support - Constraint-Driven Design

1. Critical-data-first, two-screen architecture

   Screen 1 presents status, directive, repayment structure, and ID at entry. Screen 2 contains the only high-stakes action, which controls timeout risk and keeps navigation shallow.

2. Selection over typing with mandatory confirmation

   Stable numeric options reduce input friction. Payment entry utilizes echo-back confirmation - the borrower sees the amount repeated before the system commits it.

3. Resilience

   An SMS summary after key actions gives the borrower a receipt when the session drops - continuity without requiring a reconnect.

4. Privacy

   A clean exit option matters because borrowers in group-lending programs may share a household phone - fast logout limits what the next user can see.

Lender Cloud-Based System Recommendation - Next Steps for Planning and Building

Mission alignment and data honesty

The lender dashboard should lead with the borrower's stated loan purpose and verifiable terms - not curated narratives - so lenders evaluate facts, not framing. Forecast views must follow the same EST discipline - if the borrower side labels estimates, the lender side cannot silently drop those labels.

Recommended core modules (screens)

1. Marketplace discovery

   Provide search and filtering that helps lenders find loans aligned to their goals (sector, activity, country, loan use, amount, status).

2. Portfolio dashboard

   Show allocation breakdowns - by sector, country, activity, repayment interval, and status - built from precomputed aggregates. When filters produce zero results, the dashboard should say why (e.g., "No active loans in Agriculture + Kenya") rather than showing a blank screen.

3. Loan detail and recycling forecast

   On a loan detail view, display contract terms (term and interval) and present an estimated return window derived from those proxies. Label estimates consistently and avoid language that implies confirmed repayment progress.

Component Communication and Architecture

The cloud plan should separate user-facing views from heavy aggregation work. The recommended approach is:

1. Web UI loads core dashboard elements first and fetches additional panels asynchronously.
2. API layer provides separate endpoints for lightweight aggregates and heavier loan-level detail.
3. Data store holds raw snapshot data in a structured format. Each loan record carries sector, country, activity, use, term, and interval - six fields that drive every aggregation, so the schema should index on them first.
4. Aggregation jobs build precomputed summary tables (counts and sums by sector, country, status, and repayment interval) so dashboards do not scan raw records on each request.

This maintains dashboard responsiveness by offloading heavy compute to periodic batch runs rather than per-click queries - a necessary cost discipline for a nonprofit.

Varied lender goals and how the UI supports them

1. Casual lenders

   Give them fast discovery and clear loan purpose to make a quick decision. Keep the path to funding short.

2. Impact-focused lenders

   Surface targeted comparisons across sectors, activities, and geographies. Use the same sector and country labels the borrower sees, so a lender comparing two loans is comparing the same categories.

3. Portfolio planners

   Provide diversification views and estimate-based recycling timelines derived from term and interval, labeled as estimates.

4. Team coordinators

   Expose group totals and contribution summaries driven by the loan-to-lender mapping and aggregated team metrics.

Conclusion

The borrower gets task completion under constraint; the lender gets honest data at scale. Both depend on the same discipline: label what you know, flag what you estimated, and never ask the user - borrower or lender - to guess what the system cannot confirm.