# The First MicroFinance Bank – Loan Management System

**1. Version**

Version: 1.1

Date: 6/19/2025

Changes: Version control included

**2. Project Overview**

Title:

The First MicroFinance Bank – Loan Management System

Description:

A Laravel-based system streamlining small-business loan management, with secure multi-role access, regional filtering, repayment scheduling, and printable documentation.

**3. Objectives & Problem Solving**

Secure role-based access: Prevents unauthorized data access.

Streamlined loan entry: Receptionist form with validation ensures accurate data.

Efficient loan tracking: Status-based organization provides visibility across loan lifecycle.

Zone-specific staff access: Zip-code filtering ensures staff view only region-appropriate data.

Standardized paper records: Printable customer pages support physical documentation needs.

Clear repayment schedules: Users have transparent repayment info, reducing miscommunication.

**4. Success Criteria & Validation**

Automated testing:

PHPUnit/Pest tests for authentication, data entry, printing, schedule access.

Manual QA:

Receptionist/officer workflows tested via realistic scenarios.

UAT (User Acceptance Testing phase)sign-off:

Business stakeholder approval granted for each core functionality.

Metrics tracking:

Login success rates

Form validation errors

Print jobs completed

Schedule accuracy vs. input JSON

Deployment readiness:

All code, tests, documentation, and checklist finalized before launch.

**5. Scope & Tech Stack**

Frontend:

*Languages & Markup*: HTML, CSS, JS (with @media print)

*Framework & Libraries*: React &Node.js

Backend:

Framework: Laravel (PHP), with MVC, Services, Policies

Database : MySQL; migrations for staff, customer, and loan entities

Printing JavaScript + print-specific CSS

Testing PHPUnit or Pest

Authentication Laravel auth with Receptionist / Loan Officer roles

**Project Structure:**

/microfinance-bank

├── app/Http/Controllers

├── app/Http/Middleware

├── app/Http/Requests

├── app/Models

├── app/Services

├── database/migrations

├── database/seeders

├── resources/views

├── public/css/

├── public/js/

├── routes/web.php

└── config/auth.php

**6. Timeline & Detailed Tasks**

Week 1:

Setup & Authentication

Initialize Laravel project and configure .env and database connection

Scaffold authentication system with roles and permissions

Create migrations and seeders for staff and roles

Implement login functionality with role-based redirects

Week 2:

Reception & Data Entry

Build receptionist form to register customer data

Add loan_amount and repayment_schedule to customer schema

Apply validation using FormRequest classes

Week 3:

Loan Officer Module

Develop Loan Officer dashboard view

Implement filters for zip code and loan status

Add “Pay Schedule” button on approved loans

Week 4:

Customer Detail & Printing

Develop customer detail view with editable status and print layout

Build status transition logic via service layer

Implement secured route/controller/view for repayment schedule

Week 5:

Schedule Rendering & Print Styling

Build schedule.blade.php to decode and show repayment JSON

Add “Pay Schedule” access control and schedule button

Design print-friendly CSS to remove UI clutter

Week 6:

Testing, QA & Deployment

Create comprehensive feature tests: auth, forms, statuses, printing, schedule

Conduct manual testing and complete UAT

Fix bugs and finalize deployment documentation and checklist

**7. Feature Breakdown**

Authentication & Roles – Secure login system with distinct user roles

Data Model – Customer entity with loan amount and repayment JSON

Receptionist UI – Form-based entry with validation

Loan Officer Dashboard – Zip and status filters, schedule access

Customer Details – Status updates and print layout

Repayment Schedule – JSON decoding and protected UI

Print Output – CSS-driven layout and hidden UI components

Security – Role-based policies and secure queries

Testing & Deployment – Test suite and go-live checklist

**8. Risk & Mitigation**

Timeline delays: Use daily sprints with buffer days.

Data validation issues: Enforce validation via FormRequests and database constraints.

Authorization lapses: Strict middleware and policy enforcement.

JSON errors in schedule: Validate JSON input and include error handling.

Print layout inconsistencies: Test print styles across major browsers/formats.

## Lack of Laravel/Node.js experience:

Unfamiliarity with frameworks can slow progress.

Mitigation:

Dedicate 2–3 hours/day during Weeks 1–2 to learning Laravel fundamentals (tutorials like Laracasts or Traversy Media)

Build a basic CRUD app to gain familiarity

Focus solely on Laravel; omit Node.js unless required

**9. Optional Enhancements**

Send email notifications on customer status changes

Export repayment and customer info to PDF using DomPDF

Add dynamic search/filter components (Livewire or JS)

Export reports via Laravel Excel integration

**10. Summary**

Streamlined end-to-end workflow: Reception → Officer → Schedule → Print

Well-structured and maintainable architecture: services, policies, migrations

Quality assurance ensured through full test coverage and stakeholder approval

Launch ready with final documentation and go-live criteria met

**Next Steps:**

Review and finalize the plan with stakeholders

Integrate onboarding time into Week 1–2 timelines for framework learning

Set up a Kanban board or sprint tracker for execution