DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

CS4430 Distributed Database Systems

Letsema

1. Problem Statement

Microfinance institutions (MFIs) in Lesotho play a critical role in providing financial services to underserved communities. However, many of these institutions operate in isolation, making it difficult to share borrower credit history, prevent loan stacking, and efficiently manage loans across multiple MFIs. This lack of a shared system results in high loan defaults, inefficiencies in loan processing, and challenges in ensuring fair access to credit.

Letsema, a distributed microfinance loan management system is required to enable multiple MFIs to share borrower data securely, optimize loan approval processes, and ensure transparency in loan disbursement and repayment tracking.

2. Proposed Solution

Letsema, the proposed microfinance loan management system, will be a distributed, containerized platform that allows multiple MFIs to collaborate in a distributed environment. The system will facilitate secure loan application processing, borrower credit history sharing, and real-time loan status updates.

Key High-Level Features:

1. **Borrower Management:** Secure registration and identity verification.
2. **Loan Application & Approval:** Borrowers can apply for loans online; loan officers can assess applications based on borrower credit history.
3. **Credit History Repository:** A distributed database enabling MFIs to check borrower creditworthiness before issuing loans.
4. **Loan Repayment & Tracking:** Automatic payment tracking.
5. **Reporting & Analytics:** Loan performance reports and borrower risk assessments for financial analysts.
6. **System Security & Access Control:** Role-based access to ensure data confidentiality and integrity.
7. **Scalability & Fault Tolerance:** Distributed database architecture to handle system failures and performance bottlenecks.

3. Approach & Technical Constraints

3.1 Development Approach

The project will follow an Agile Development Methodology, with iterative sprints planned for incremental feature delivery. JIRA will be used for sprint planning and task management.

3.2 Technology Stack

- **Backend Framework:** Django (Python) for API development.
- **Frontend Framework:** React.js for user interface development.
- **Database:** Hybrid approach:
    - **PostgreSQL** (Relational) for structured loan records and financial transactions.
    - **MongoDB/Cassandra** (NoSQL) for distributed borrower credit history.
- **Deployment:** Docker for scalability and portability.
- **Authentication:** OAuth2.0 / JWT for secure user authentication.
- **Version Control:** GitHub for source code collaboration and repository management.

3.3 Constraints and Considerations

- **System Availability:** The system must be highly available and fault-tolerant to prevent disruptions in financial operations.
- **Performance Optimization:** Query optimizations for fast borrower lookup and loan approval.
- **Scalability:** The system should handle increasing loads as more MFIs onboard.

4. Project Deliverables

- **Functional Prototype** with borrower registration, loan application, and approval workflows.
- **Distributed Database Schema & Replication Strategy** document.
- **API Documentation** for system integration.
- **Scalability & Performance Testing Report.**

- **Final System Demonstration & Evaluation.**
- **Source Code Repository on GitHub with CI/CD pipelines.**