Data Model Design Document

1. Overview

This document outlines the data model for the Smart Informal Business Credit & Record App. The model is designed to support the functional requirements identified in the Requirement Analysis Document, including transaction logging, credit scoring, report generation, and admin analytics. It uses a relational database schema suitable for PostgreSQL, emphasizing efficiency for offline-first sync, security for financial data, and scalability.

The data model focuses on:

- **Core Entities**: Users, Transactions, Scores, Reports.
- **Relationships**: One-to-many (e.g., User to Transactions), Many-to-Many where needed (e.g., via junction tables for access controls).
- **Offline Sync Considerations**: Timestamps and sync flags for conflict resolution.
- **Security**: Sensitive data (e.g., scores, personal info) encrypted or access-controlled.

We will use PostgreSQL features like UUIDs for IDs, JSONB for flexible data (e.g., insights), and triggers/indexes for performance.

2. Entities and Attributes

Below is a list of key entities (tables) with their attributes, data types, constraints, and descriptions.

2.1 Users Table

- **Purpose**: Stores information about business owners, admins, lenders, and cooperatives.
- **Attributes**:
    - user_id: UUID (Primary Key, auto-generated).
    - username: VARCHAR (50) (Unique, required).
    - email: VARCHAR(100) (Unique, optional).
    - phone_number: VARCHAR(20) (Unique, required for mobile money linkage).
    - role: ENUM('business_owner', 'admin', 'lender', 'cooperative') (Required).

- password_hash: VARCHAR(255) (Required, hashed for security).
- verification_status: ENUM('pending', 'verified', 'rejected') (Default: 'pending').
- created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- updated_at: TIMESTAMP (Default: CURRENT_TIMESTAMP, updated on change).
- last_sync_at: TIMESTAMP (For offline sync tracking).

➢ **Indexes**: Unique on username, email, phone_number; Index on role.

2.2 Transactions Table

➢ **Purpose**: Records daily sales, expenses, and mobile money transactions.

➢ **Attributes**:

- transaction_id: UUID (Primary Key, auto-generated).
- user_id: UUID (Foreign Key to Users.user_id, required).
- type: ENUM('sale', 'expense', 'mobile_money_in', 'mobile_money_out') (Required).
- amount: DECIMAL(15,2) (Required, positive for sales/in, negative for expenses/out).
- category: VARCHAR(50) (Optional, e.g., 'groceries', 'rent').
- description: TEXT (Optional).
- transaction_date: DATE (Required, default: CURRENT_DATE).
- mobile_money_ref: VARCHAR(100) (Optional, for integration refs).
- is_synced: BOOLEAN (Default: FALSE, for offline mode).
- created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- updated_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).

➢ **Indexes**: Index on user_id, transaction_date; Composite index on user_id and type for queries.

2.3 CreditScores Table

- **Purpose**: Stores computed creditworthiness scores and history.
- **Attributes**:
    - score_id: UUID (Primary Key, auto-generated).
    - user_id: UUID (Foreign Key to Users.user_id, required, unique per user for latest score).
    - score_value: INTEGER (Required, range 0-1000 or similar).
    - calculation_date: DATE (Required).
    - factors: JSONB (Required, e.g., {"sales_consistency": 80, "expense_ratio": 70} for transparency).
    - version: INTEGER (For historical tracking, increment on updates).
    - created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- **Indexes**: Unique on user_id and version; Index on calculation_date.

2.4 Reports Table

- **Purpose**: Manages generated financial reports for sharing.
- **Attributes**:
    - report_id: UUID (Primary Key, auto-generated).
    - user_id: UUID (Foreign Key to Users.user_id, required).
    - type: ENUM('weekly_summary', 'monthly_profit', 'full_history') (Required).
    - generated_date: DATE (Required).
    - file_path: VARCHAR(255) (Path to PDF or secure link).
    - shared_with: ARRAY[UUID] (User IDs of lenders/cooperatives who can access).
    - expiry_date: DATE (Optional, for time-limited shares).
    - created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- **Indexes**: Index on user_id, generated_date.

2.5 ScoringRules Table

- **Purpose**: Configurable rules for the credit score engine (admin-managed).
- **Attributes**:
    - rule_id: UUID (Primary Key, auto-generated).
    - rule_name: VARCHAR(100) (Unique, required).
    - weight: DECIMAL(5,2) (Required, e.g., 0.3 for 30%).
    - criteria: JSONB (Required, e.g., {"min_sales": 1000, "consistency_threshold": 0.8}).
    - active: BOOLEAN (Default: TRUE).
    - created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
    - updated_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- **Indexes**: Unique on rule_name.

2.6 Insights Table

- **Purpose**: Stores generated insights and growth suggestions for users.
- **Attributes**:
    - insight_id: UUID (Primary Key, auto-generated).
    - user_id: UUID (Foreign Key to Users.user_id, required).
    - insight_text: TEXT (Required, e.g., "Reduce expenses by 10% to boost score").
    - category: ENUM('growth', 'risk', 'optimization') (Required).
    - generated_date: DATE (Required).
    - created_at: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- **Indexes**: Index on user_id, generated_date.

2.7 AuditLogs Table

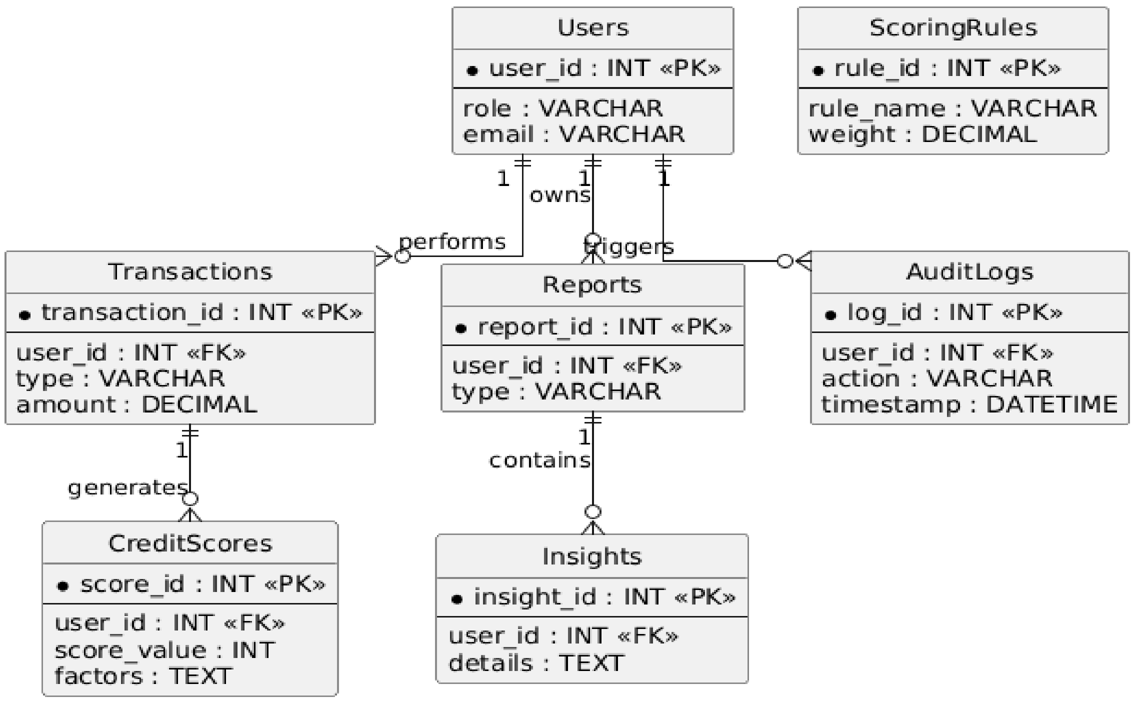
- **Purpose**: Tracks changes for security and analytics (e.g., verifications, accesses).
- **Attributes**:

    - log_id: UUID (Primary Key, auto-generated).
    - user_id: UUID (Foreign Key to Users.user_id, optional).
    - action: VARCHAR(100) (Required, e.g., 'login', 'score_update', 'report_share').
    - details: JSONB (Optional, e.g., {"ip": "192.168.1.1"}).
    - timestamp: TIMESTAMP (Default: CURRENT_TIMESTAMP).
- **Indexes**: Index on timestamp, action.

3. Relationships

- **Users to Transactions**: One-to-Many (A user has many transactions).
    - Enforced via Foreign Key in Transactions.
- **Users to CreditScores**: One-to-Many (Historical scores per user).
    - Foreign Key in CreditScores.
- **Users to Reports**: One-to-Many.
    - Foreign Key in Reports.
- **Users to Insights**: One-to-Many.
    - Foreign Key in Insights.
- **ScoringRules**: Standalone, referenced programmatically in scoring engine.
- **Reports Sharing**: Many-to-Many (via shared with array or separate junction table if needed for complex queries).
- **AuditLogs**: Loose association to Users.

4. ER Diagram



5. Data Integrity and Constraints

- **Primary Keys**: UUIDs for uniqueness and security (avoids sequential ID guessing).
- **Foreign Keys**: Cascade deletes where appropriate (e.g., delete transactions on user delete, but archive for compliance).
- **Unique Constraints**: On usernames, emails, etc.
- **Check Constraints**: E.g., amount > 0 for sales, score_value between 0-1000.
- **Triggers**: Auto-update updated_at; Compute scores on transaction inserts/updates (or via cron job).
- **Indexes**: For frequent queries (e.g., user transactions by date).

6. Offline Sync Strategy

- Local Storage (Mobile): Use Realm or SQLite to mirror key tables (Users, Transactions, partial Scores).
- Sync Mechanism: On connectivity, push unsynced transactions (WHERE is_synced = FALSE), pull updates. Use last_sync_at to fetch deltas.
- Conflict Resolution: Last-write-wins based on timestamps; or merge for transactions.

7. Security Considerations

- Encrypt sensitive columns (e.g., phone_number, password_hash) using PostgreSQL pgcrypto.
- Role-Based Access: Enforce via app logic and database views (e.g., lenders see only shared reports).
- Data Retention: Purge old logs/reports per policy.

8. Deliverables

Based on the project scope, the following deliverables will be produced during development:

8.1 Documentation

- Requirement Analysis Document (already provided).
- Data Model Design Document (this document).
- API Documentation (Swagger/OpenAPI for backend endpoints).
- User Manual (for app usage).
- Admin Guide (for portal management).

8.2 Code and Artifacts

- **Mobile App**: React Native (Expo) source code, including offline sync implementation.
- **Backend**: Node.js server with routes for auth, transactions, scoring, etc.
- **Database Schema**: SQL scripts for PostgreSQL setup (tables, indexes, triggers).
- **Integrations**: Mobile money SDK implementations (e.g., MTN MoMo API stubs).
- **Scoring Engine**: Secure module for rule-based calculations.
- **Admin Portal**: Web interface (possibly React web app) for admin features.

8.3 Testing and Deployment

- Unit/Integration Tests: Coverage for core features (e.g., Jest for Node.js, Detox for React Native).
- MVP Build: Deployable APK/IPA for Android/iOS.
- Cloud Deployment: Scripts for AWS/Heroku setup (e.g., Dockerfiles).
- Security Audit Report: Basic vulnerability scan.

8.4 Timeline for Deliverables

- Data Model Implementation: Week 1 (SQL scripts).
- Backend MVP: Weeks 2-3.
- Mobile App MVP: Weeks 3-4.
- Integrations and Testing: Week 5.
- Full Deployment: Week 6.