**DOCUMENT BY: JOYANN WAIRIMU MWANGI 23/05024**

**supervisor:**

**CHARLES MALUNGU**

**PUBLIC TRANSPORT FARE COLLECTION AND MANAGEMENT SYSTEM**

A project Proposal for the Metro trans Investments Ltd

**System Design for Public Transport Fare Collection and Management System**

**1. Architecture**

The system follows a **client-server architecture** with cloud-based integration for scalability and real-time data processing.

**Hardware Components:**

* **QR/NFC Scanners:** Installed on buses to validate digital tickets.
* **Backend Server:** Hosted on a cloud platform for transaction processing.
* **Database Server:** Centralized database for storing transaction records, user details, and fraud detection logs.
* **Admin Workstations:** Used by system administrators for monitoring and reporting.

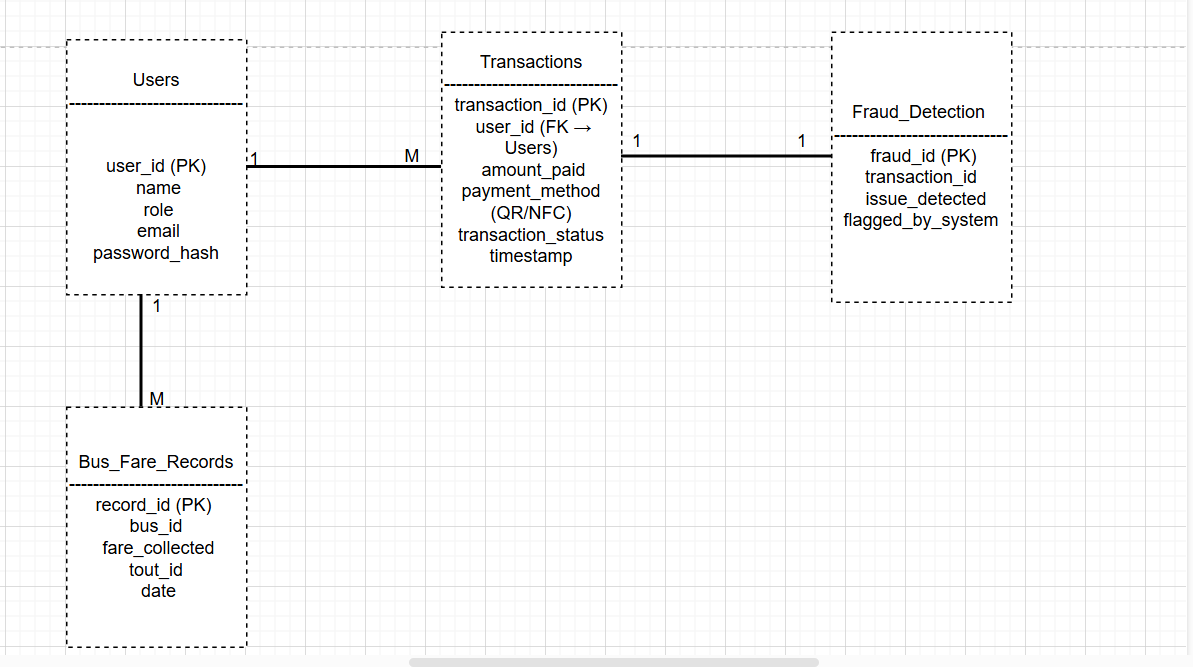
**Software Components:**

* **Frontend:** Web and mobile application (HTML/CSS, React.js for UI)
* **Backend:** Python (Flask)
* **Database:** MySQL for structured transaction records.
* **Authentication:** For secure access control.

**2. Database Design**

The database consists of multiple tables with defined relationships to ensure data integrity and optimized performance.

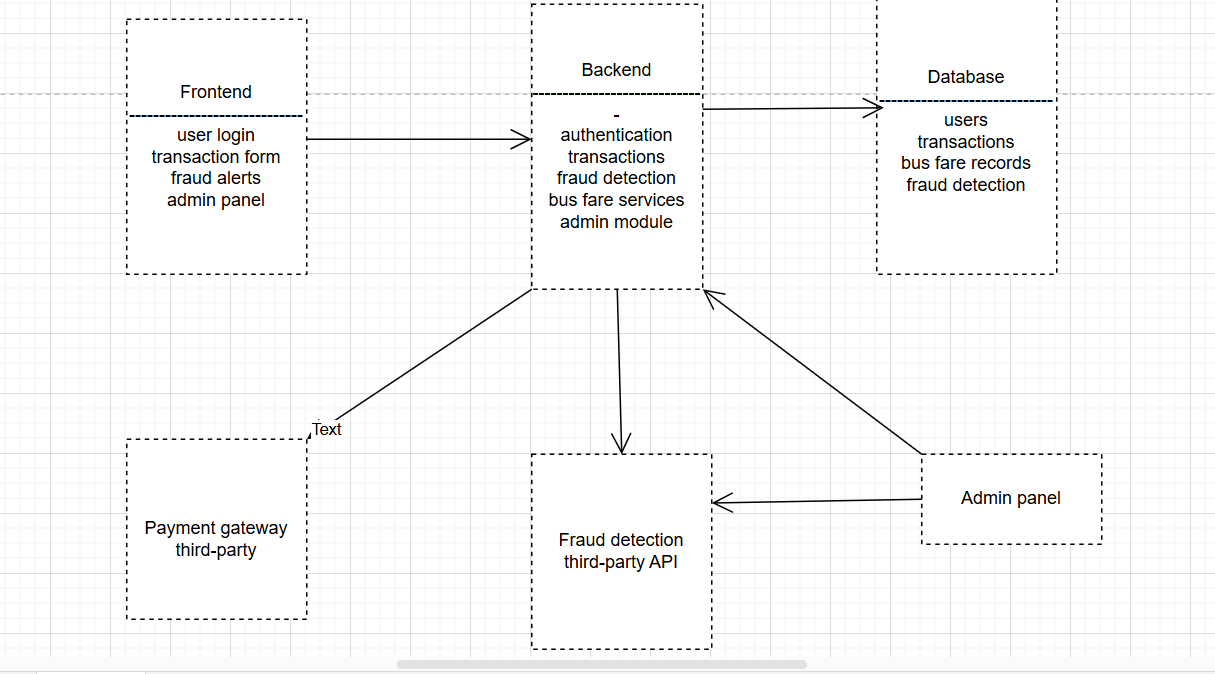
**ENTITY RELATIONSHIP DIAGRAM**



**Key Tables and Relationships:**

1. **Users Table**
   * user\_id (Primary Key)
   * name
   * role (Passenger/Tout/Admin)
   * email
   * password\_hash
2. **Transactions Table**
   * transaction\_id (Primary Key)
   * user\_id (Foreign Key to Users)
   * amount\_paid
   * payment\_method (/QR/NFC)
   * transaction\_status (Success/Failed)
   * timestamp
3. **Bus\_Fare\_Records Table**
   * record\_id (Primary Key)
   * bus\_id
   * fare\_collected
   * tout\_id (Foreign Key to Users)
   * date
4. **Fraud\_Detection Table**
   * fraud\_id (Primary Key)
   * transaction\_id (Foreign Key to Transactions)
   * issue\_detected (Duplicate Payment, Reversal Attempt, etc.)
   * flagged\_by\_system (Yes/No)

**SOFTWARE DESIGN DIAGRAM**

****

**3. Scalability**

The system is designed to handle an increasing number of transactions and users with minimal performance degradation.

* **Horizontal Scaling:** Additional application servers can be added to handle increased demand.
* **Database Sharding:** Divides large datasets across multiple database instances for efficiency.
* **Load Balancing:** Ensures fair distribution of traffic across backend servers.
* **Cloud Hosting:** Deploying on AWS/GCP for auto-scaling and reliability.

**4. Security**

The system employs multiple security layers to protect user data and prevent fraud.

* **Data Encryption**
* **Authentication & Authorization:**
* **Fraud Detection System:** AI-driven anomaly detection for suspicious transactions.
* **Role-Based Access Control:** Different passenger, touts, and administrators access levels.

**5. Performance**

Performance optimization techniques ensure a smooth user experience.

* **Asynchronous Processing:** Reduces wait time by processing payments in the background.
* **Database Indexing:** Speeds up transaction retrieval times.
* **API Rate Limiting:** Prevents excessive traffic requests to maintain system stability.

**6. Usability**

The system is designed for ease of use, ensuring a smooth experience for all users.

* **Mobile-Friendly UI:** Responsive design for seamless interaction across devices.
* **Simple Payment Flow:** Minimal steps for digital fare payment and validation.
* **Language Support:** Available in English and Swahili.
* **Accessibility Features:** High contrast mode, voice commands, and large text options.

**7. Functionality**

Each component of the system provides specific services to ensure smooth operation.

* **Passengers:** Pay fares via NFC, or QR codes.
* **Touts:** Verify payments and track collected fares.
* **Admins:** Monitor transactions, detect fraud, and generate financial reports.
* **System:** Ensures real-time validation, secure transactions, and automated reporting.

**8. Interfaces and APIs**

The system interacts with multiple components through well-defined API endpoints.

* **QR/NFC API:** Validates digital tickets before boarding.
* **Admin API:** Retrieves reports and monitors transactions.
* **Fraud Detection API:** Flags suspicious activity for further review.

**9. Data Handling**

The system efficiently processes and manages transaction data in real-time.

* **Data Validation:** Ensures accurate user and payment data entry.
* **Logging Mechanism:** Stores all transactions and system activities for audit purposes.
* **Archiving Policies:** Older records are archived periodically to improve database performance.

**10. System Design Considerations**

To ensure reliability and fault tolerance, the system implements:

* **Redundancy:** Data is replicated across multiple servers for reliability.
* **Fault Tolerance:** Automatic failover mechanisms in case of server failure.
* **Load Balancing:** Distributes user requests across multiple servers.

**11. System Design Goals**

The primary goal is to build a secure, scalable, and efficient fare collection system that meets functional, technical, and business needs.

* **Efficiency:** Processes transactions in under 2 seconds.
* **Security:** Prevents fraud and unauthorized access.
* **Scalability:** Can support thousands of users with minimal performance loss.
* **User-Friendly:** Easy to use for passengers, touts, and administrators.
* **Reliability:** 99.9% uptime guarantee with automated backup mechanisms.