FarmConnect – A Farmer-to-Market Supply Chain

Submitted By: Rutuja Nagnath Kamble

ID: 28285

To: SR Umesh Sir

Project Overview

FarmConnect is a Java + MySQL based supply chain application that allows farmers to manage product listings, while enabling buyers to search, place, and track orders. The project integrates object-oriented programming, exception handling, and relational database design to simulate real-world farmer-to-market functionality.

key Features

- **Farmer Panel**: Add, update, delete products (name, price, stock, description)
- **Buyer Panel**: Search/view products, place orders (multiple products)
- **Order Management**: Update order status, view order history
- **Exception Handling**: Invalid data, stock limits, and quantity checks
- **MySQL Integration**: All data stored using JDBC and relational tables
- **Transaction Handling**: Orders and inventory updated atomically

Understanding Requirements

Business Use Case: Digitize and streamline the rural agricultural supply chain.

Entities Identified: Farmers, Buyers, Products, Orders, Order_Products

Tech Highlights

1. Used **PreparedStatement** for SQL operations to prevent injection
2. Used **JDBC** for full database access and transaction control
3. Designed a **modular OOP structure** with model, service, and util layers
4. Exception handling with validations and rollback in failures
5. ER diagram included to explain table relationships clearly.

Database Schema Design

Tables & Constraints:

- farmers: Primary key, optional login data
- buyers: Primary key, optional login data
- products: Foreign key to farmers, price and stock validation
- orders: Foreign key to buyers, status and amount constraints
- order_products: Junction table with foreign keys to orders and products, includes quantity

Data Operations

- Insert product details by farmer
- Place multi-product orders by buyer
- Manage stock automatically after each order
- View order history and update order status

Technologies Used

Language: Java
Database: MySQL
Backend: JDBC
Design: OOP, Exception Handling, Collections
Interface: Console (Scanner)

Additional Features Implemented

- Stored Procedure: GetBuyerOrders – Retrieves full order history for a given buyer, including product details and quantities.
- Trigger (optional): update_stock_after_order – Automatically updates product stock after an order is placed (used cautiously).
- Indexes: Added on product_id, order_id, and buyer_name for optimized performance on common queries.

```
DELIMITER //

CREATE PROCEDURE GetBuyerOrders (IN buyerName VARCHAR(100))

BEGIN

  SELECT o.order_id, o.total_amount, o.status, p.name AS product_name, op.quantity

  FROM orders o

  JOIN order_products op ON o.order_id = op.order_id

  JOIN products p ON op.product_id = p.product_id
```

```
  WHERE o.buyer_name = buyerName;
END //
DELIMITER;
CALL GetBuyerOrders('Rutuja');
```

Triggers

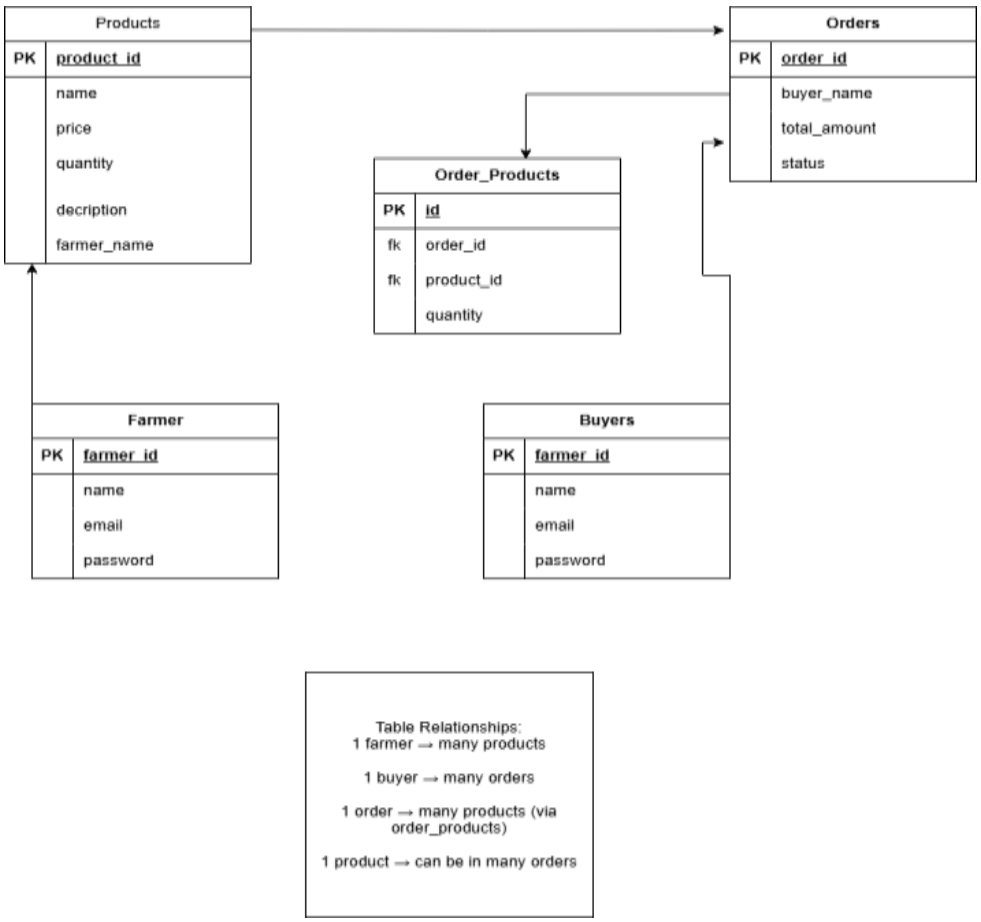
```
DELIMITER //
CREATE TRIGGER update_stock_after_order
AFTER INSERT ON order_products
FOR EACH ROW
BEGIN
  UPDATE products
  SET quantity = quantity - NEW.quantity
  WHERE product_id = NEW.product_id;
END //
DELIMITER;
```

Indexes

```
CREATE INDEX idx_product_id ON products(product_id);
CREATE INDEX idx_order_id ON orders(order_id);
CREATE INDEX idx_buyer_name ON orders(buyer_name);
```

ER Diagram

Entities: farmers, buyers, products, orders, order_products
Relationships: One-to-many (farmers → products, buyers → orders), many-to-many (orders ↔ products via order_products)



Conclusion

FarmConnect is a fully functional Java-MySQL capstone project that mimics a real-world agricultural ordering system. With clean modular code, database integration, and realistic error handling, it demonstrates practical backend development skills essential for placement and real-world applications.