FarmConnect: Bridging Farmers to Markets

Transforming agricultural supply chains through digital innovation.

Submitted By:- Rutuja-28285

Guided By:- SR-UMESH SIR and Pushpa Ma'am

Rutuja-28285

Presentation Agenda

1 **Introduction & Objectives**

Understanding the challenge and our goals.

2 **System Architecture**

Key components and technology stack.

3 **Core Functionality**

Features for farmers and buyers.

4 **Implementation & Testing**

Detailed features and rigorous validation.

5 **Conclusion & Future Scope**

Summary and potential enhancements.

Rutuja-28285

Addressing the Market Disconnect

The Challenge

- Traditional supply chain inefficiencies
- Limited farmer-market access
- Lack of real-time data

Our Solution: FarmConnect

- Digitizes agricultural supply chain
- Direct farmer-to-buyer interaction
- Real-time inventory management

Bridging the gap with technology for a more efficient agriculture ecosystem.

Core Objectives

Digitize Interactions

Streamline farmer-buyer communication.

Real-time Management

Instant inventory and order updates.

Robust Database Design

Efficient MySQL data handling.

Enterprise Backend Simulation

Using Java and JDBC for scalability.

Rutuja-28285

System Architecture & Requirements

○ Technology Stack

- **Language:** Java JDK 8+
- **Database:** MySQL Server & Workbench
- **DB Access:** JDBC
- **Interface:** Console (Scanner)
- **Design:** OOP, Exception Handling

○ System Requirements

- **RAM:** 4 GB+
- **Processor:** Intel i3+
- **Disk:** 500 MB
- **IDE:** Eclipse / IntelliJ / NetBeans
- **Connector:** MySQL Connector JAR

Database Schema Overview

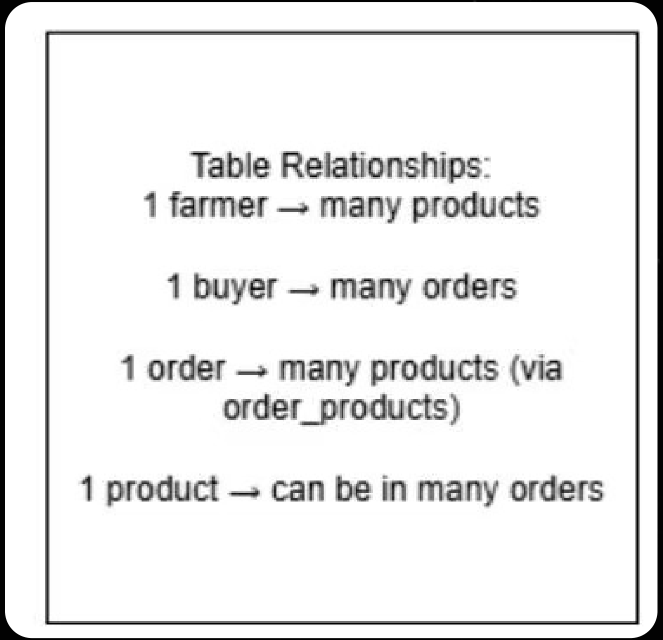

Table Relationships:
1 farmer → many products

1 buyer → many orders

1 order → many products (via order_products)

1 product → can be in many orders

- The ER Diagram illustrates relationships between Farmers, Buyers, Products, and Orders, supported by a clear database structure.

- The ER diagram gives us a high-level overview of the key entities and relationships in our database

- It shows how the core components of our business - Farmers, Buyers, Products, and Orders - are all connected and interact with each other

- The clear database structure underlying this ER diagram ensures we can effectively manage and query the data, which is crucial for gaining insights and making informed decisions

Rutuja-28285

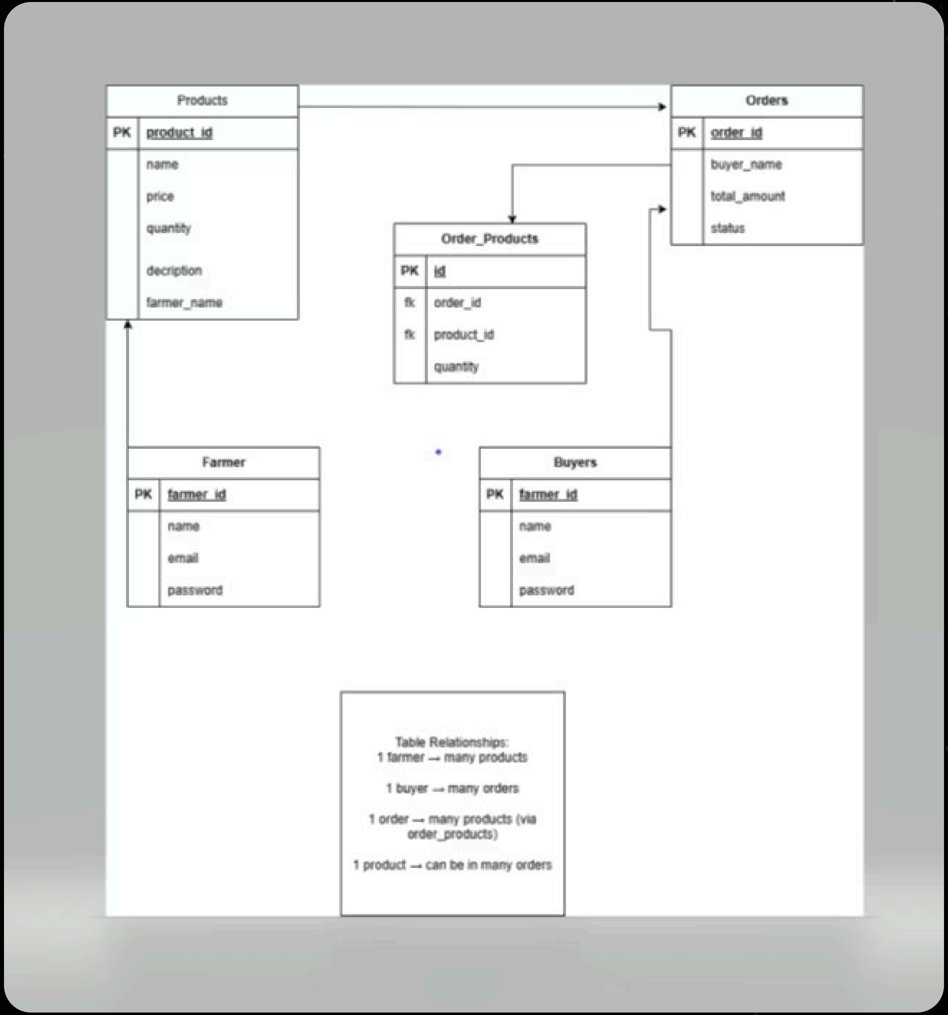

Table Relationships:
1 farmer → many products

1 buyer → many orders

1 order → many products (via order_products)

1 product → can be in many orders

○ In our FarmConnect project, the tables follow clear **‘has-a’ relationships**:

- A farmer has many products.
- A buyer has many orders.
- Each order has many products through the Order_Products table.

  These relationships help maintain proper normalization and ensure data consistency.

There are no ‘is-a’ relationships in this model, as each entity represents a distinct role in the system.

Key Database Tables

farmers	buyers	products	orders	order_products
id	id	product_id	order_id	order_id
name	name	name	buyer_name	product_id
email	email	price	total_amount	quantity
password	password	quantity	status	
		farmer_name		

These tables form the backbone of FarmConnect, ensuring data integrity and efficient operations for product and order management.

Rutuja-28285

Implementation & Core Features

1

Farmer Module

- Add/Update/Delete Products
- View Orders

2

Buyer Module

- View/Search Products
- Place Orders
- View History

3

Order Management

- Auto-update inventory
- Status updates

4

Robust Handling

- Exception Handling
- Database Transactions

Rutuja-28285

Working Demo Highlights

○ **Content:**

- Farmer logs in and adds products to the inventory (insert into DB)
- Buyer searches for products and places an order
- Inventory updates automatically after each purchase
- Real-time MySQL integration using JDBC
- System handles edge cases like stock-out or invalid inputs

○ The farmer logs in and adds products, which are immediately stored in the MySQL database.
On the buyer side, users can search and place orders.
Once an order is placed, the system automatically updates the inventory and reflects the changes in real-time.
This interaction between Java and MySQL is handled via JDBC.
The application also includes input validations and error handling — for example, it gracefully manages stock-out scenarios or invalid product IDs.

Key Learnings / Skills Gained

100

Content:

1. Practical JDBC use
2. Real-world problem solving
3. Exception handling & transactions
4. Database design & ER modeling
5. Modular Java development.

○ This project helped me gain hands-on experience in building a real-world backend system.
I learned how to use JDBC for database connectivity, design normalized ER models, and implement exception handling and transactions to ensure system reliability.
Working on this project also helped me improve my modular coding in Java and understand how to simulate enterprise-level architecture in a console-based environment

Rutuja-28285

Testing & Validation

○ Verification Areas

- Order Tracking Accuracy
- Stock Reduction Post-Order
- Database Consistency (Rollback)

○ Validation Scenarios

- Out-of-Stock Conditions
- Invalid Product ID Inputs
- Empty Cart Submissions

Rigorous testing ensures system reliability and data integrity under various conditions.

Conclusion & Future Directions

○ Project Success

- Real-world backend simulation
- Comprehensive supply chain management
- Modular and scalable design

○ Future Enhancements

- Login/Authentication system
- GUI or Web Front-End
- Payment gateway integration
- Automated report generation

FarmConnect lays a strong foundation for digitizing agriculture, with clear pathways for future growth.

Rutuja-28285

Thank you

I'm open to any questions, suggestions, or feedback you may have.

Feel free to ask anything about the project.