**FarmConnect – A Farmer-to-Market Supply Chain**

**1. Introduction**

Agriculture is the backbone of India’s economy, especially in rural areas. Farmers often depend on middlemen to sell their produce, which reduces their profit margins. On the other side, buyers face high market prices for produce.

**FarmConnect** is designed to create a digital platform that connects farmers directly with buyers. It allows farmers to register, manage their products, and track orders. Buyers can search, order, and make payments.

**2. Objectives**

* To provide farmers with a direct channel to sell their crops.
* To allow buyers to purchase fresh produce at fair prices.
* To manage inventory, orders, and payments efficiently.
* To handle errors such as invalid entries or out-of-stock cases gracefully.

**3. System Requirements**

**Software Requirements**

* Java
* Eclipse IDE
* MySQL Server & Workbench
* MySQL Connector/J

**Hardware Requirements**

* Minimum 4GB RAM, Intel i3 processor
* 10GB free disk space

**4. Database Design**

**SQL Schema**

CREATE TABLE Farmers (

farmer\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

contact\_number VARCHAR(15),

address VARCHAR(255)

);

CREATE TABLE Buyers (

buyer\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

contact\_number VARCHAR(15),

address VARCHAR(255)

);

CREATE TABLE Products (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

farmer\_id INT,

name VARCHAR(100) NOT NULL,

description VARCHAR(255),

price DECIMAL(10,2) NOT NULL,

quantity INT NOT NULL,

FOREIGN KEY (farmer\_id) REFERENCES Farmers(farmer\_id)

);

CREATE TABLE Orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

buyer\_id INT,

product\_id INT,

quantity INT NOT NULL,

status VARCHAR(50) DEFAULT 'Pending',

FOREIGN KEY (buyer\_id) REFERENCES Buyers(buyer\_id),

FOREIGN KEY (product\_id) REFERENCES Products(product\_id)

);

CREATE TABLE Payments (

payment\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT,

amount DECIMAL(10,2),

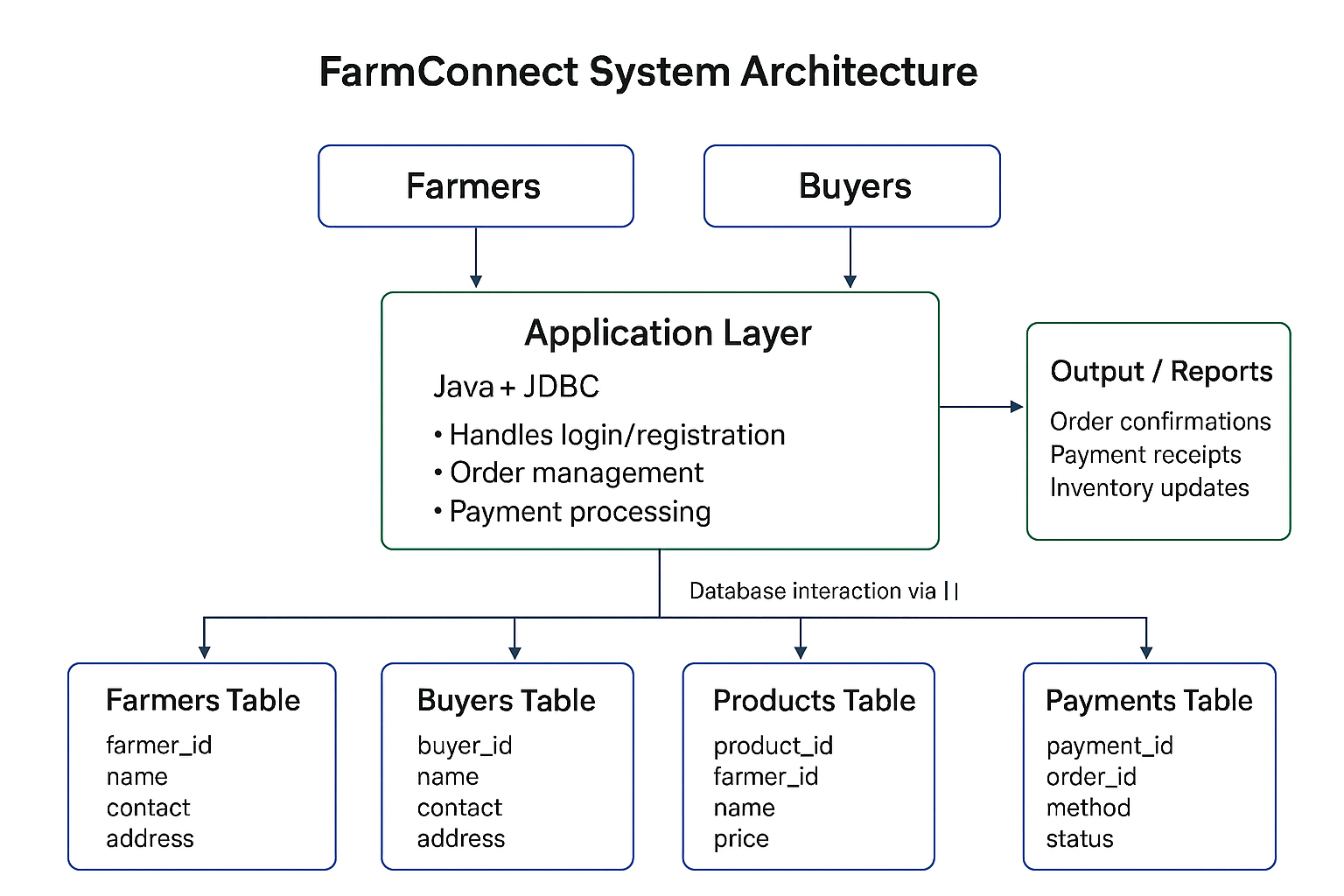
payment\_method ENUM('Cash on Delivery','Card','Online'),

payment\_status VARCHAR(50) DEFAULT 'Unpaid',

FOREIGN KEY (order\_id) REFERENCES Orders(order\_id)

);

**5. System Architecture / Flow**

****

**6. ER Diagram**

**Entities & Relationships:**

* A **Farmer** can add many **Products** (1-to-many).
* A **Buyer** can place many **Orders** (1-to-many).
* An **Order** is linked to one **Product** (many-to-1).
* An **Order** has one **Payment** record (1-to-1).

**7. System Architecture / Structural Diagram**

**Layers:**

* **Presentation Layer:** User interacts via console/GUI (Farmer & Buyer menus).
* **Application Layer:** Core Java classes (FarmerDAO, BuyerDAO, OrderDAO, PaymentDAO).
* **Database Layer:** MySQL storing all farmer, buyer, product, order, and payment data.

**8. Modules**

**Farmer Module**

* Register farmer details.
* Add/update/delete products.
* View/update orders.

**Buyer Module**

* Register buyer details.
* Search/view products.
* Place orders with quantity check.
* Select payment method.

**Order Management**

* Buyers can view order history.
* Farmers can view/update order status.

**Payment Module**

* Payment linked to orders.
* Buyers choose payment method (COD, Card, Online).
* Payment status updated in DB.

**9. Exception Handling**

* Prevents negative or invalid quantities.
* Handles out-of-stock scenarios.
* Ensures valid product details (price > 0).
* Prevents ordering more than available stock.
* Catches database connection failures.

**10. Conclusion**

The project demonstrates how technology can bridge the gap between farmers and buyers. Using Java with MySQL, FarmConnect offers a simple but scalable supply chain solution for agriculture. It reduces dependency on middlemen, increases farmers’ income, and ensures buyers get fresh products.