Software Requirement Specification (SRS) for DBMS Project

1. Introduction

1.1 Purpose

The purpose of the *Farmer-to-Market Produce Tracking System* is to design and develop a **database management system** that efficiently manages and tracks the movement of agricultural produce from **farmers** to **retailers** through different stages of the supply chain such as **warehouses** and **distributors**.

This system aims to provide a structured way to store, retrieve, and manage data related to farmers, produce, storage, transportation, and retailers. By maintaining all relevant records in a centralized database, the project seeks to improve **traceability**, **transparency**, and **accountability** across the agricultural distribution process.

The DBMS will help stakeholders such as farmers, warehouse managers, distributors, and retailers to record transactions, monitor produce flow, and ensure that the produce reaches the market efficiently with minimal losses or delays.

In summary, the purpose of this project is to:

- Design the database structure and relationships for an agricultural supply chain.
- Enable accurate tracking of produce from the source (farmer) to the destination (retailer).
- Demonstrate key DBMS concepts such as normalization, keys, relationships, and data integrity constraints.

1.2 Scope

The *Farmer-to-Market Produce Tracking System* is designed to model a simplified agricultural supply chain through a structured database application. It focuses on four major stages of produce flow: farmers supplying harvested crops, warehouses storing and maintaining inventory, distributors handling logistics and transportation, and retailers selling the final produce to consumers. By capturing information across these stages, the system ensures a complete, traceable record of how agricultural produce moves from the point of harvest to the point of sale.

The database will maintain essential records such as farmer details and the types of produce supplied, warehouse capacities and storage logs, stock availability, distribution transactions between warehouses and retailers, and retailer information, along with their purchase activities. Each primary entity—Farmer, Produce, Warehouse, Distributor, and Retailer—will be represented as a dedicated table with well-defined relationships and foreign key constraints to ensure data integrity and consistency throughout the system. Additional transactional tables, such as Storage Records and Distribution Records, will track the movement and status of produce at each stage of the supply chain.

The system offers several key benefits, including streamlined management of agricultural supply chain data, improved traceability of produce movement, and enhanced decision-

making enabled by organized records, queries, and summary reports. It also demonstrates core DBMS concepts such as normalization, relational modelling, entity-relationship design, and constraint enforcement. As an academic project, it effectively integrates theoretical principles with practical implementation, making the system comprehensive yet manageable.

1.3 Definitions, Acronyms, and Abbreviations

Term / Acronym	Definition / Description
DBMS	Database Management System — software used to store, manage, and retrieve structured data efficiently.
ERD	Entity–Relationship Diagram — a visual representation showing entities, attributes, and relationships among data in a database.
RDBMS	Relational Database Management System — a DBMS based on the relational model, where data is organized into related tables.
SQL	Structured Query Language — a standard programming language used for managing and querying data in relational databases.
CRUD	Create, Read, Update, Delete — the four basic operations performed on database records.

Term / Acronym	Definition / Description
Primary Key (PK)	A unique attribute (or set of attributes) used to identify each record in a table.
Foreign Key (FK)	An attribute in one table that refers to the primary key of another table, used to establish relationships.
Normalization	The process of organizing data in a database to reduce redundancy and improve data integrity.
Entity	A real-world object or concept represented in the database (e.g., Farmer, Warehouse, Distributor).
Attribute	A property or characteristic of an entity (e.g., Farmer_Name, Warehouse_Location).
Tuple / Record	A single row in a database table representing one instance of an entity.
Traceability	The ability to track the movement or history of a product or item across different stages of a process.
Transparency	The visibility of data and operations across all stages of the system, ensuring accountability.
Relational Model	A logical structure that organizes data into tables (relations) with defined relationships between them.

2. Overall Description

2.1 Product Perspective

The *Farmer-to-Market Produce Tracking System* is designed as a **standalone database application** that models a simplified agricultural supply chain. It is primarily focused on the **data management and traceability aspect** of the system rather than the front-end user interface or live operations.

The system acts as a **centralized database** that stores and manages all information related to farmers, their produce, warehouses, distributors, and retailers. Each of these entities is represented as a separate table in the relational database, connected using primary and foreign keys.

Although this DBMS can function independently for academic demonstration, it can also serve as the **data layer (backend)** for a larger system in the future, such as:

- A **web-based agricultural logistics portal**, or
- A **mobile app** for real-time produce tracking.

In its current form (for the lab project):

- The system will be **operated through SQL commands** for inserting, updating, querying, and deleting records.
- It is **not connected to external applications** or live data feeds.
- The focus is on **data design, relationships, constraints, and integrity** within a standalone database environment.

2.2 Product Functions

The *Farmer-to-Market Produce Tracking System* provides a structured way to record, manage, and retrieve information related to the agricultural supply chain — from farmers to retailers.
The main functions of the system are categorized based on key entities and operations performed in the database.

Module / Area	**Main Operations**
Farmer Management	Add, update, view, delete farmer details
Produce Management	Track produce data, link to farmers
Warehouse Management	Log storage records, manage stock levels
Distribution Management	Record shipments and delivery details
Retailer Management	Track produce received from distributors
Traceability Reports	End-to-end produce tracking
Admin Controls	Data validation, backup, integrity checks

2.3 User Characteristics

The *Farmer-to-Market Produce Tracking System* is designed for users with basic to intermediate knowledge of database operations. Since this is a DBMS project, the users primarily interact with the system via **SQL queries or a simple interface (optional)**.

User Role	Description	Skill Level / Requirements
Database Administrator (DBA)	Responsible for managing the overall database structure, performing backups, enforcing constraints, and maintaining data integrity.	Knowledge of SQL, database design, normalization, and DBMS concepts.
Farm Manager / Farmer Data Entry	Adds and updates farmer and produce information.	Basic understanding of data entry and produce categorization.
Warehouse Manager	Updates storage records, tracks stock levels, and monitors warehouse capacity.	Ability to interpret inventory data and use SQL queries for record management.
Distributor / Logistics Coordinator	Logs distribution records and tracks deliveries.	Understanding of shipment records and simple query retrievals.
Retailer (Optional)	Views produce received from distributors.	Basic familiarity with database queries or report views.

General User Characteristics:

- Users need **basic computer literacy** (typing, accessing SQL console or DBMS tools).
- Users should understand **produce categories, supply chain stages, and quantities**.
- Users will **not directly manipulate low-level database files**; all operations are through SQL commands or controlled forms.
- Accuracy in data entry is important to maintain **traceability** and **transparency**.

2.4 Constraints

The system has the following constraints that define its operational and design boundaries:

1. **Software Constraints**
    - Must use a **Relational Database Management System (RDBMS)** such as MySQL, PostgreSQL, or SQLite.
    - SQL will be used for data operations (CRUD, queries, reports).
    - No live web or mobile interface is required for the lab version.

2. **Hardware Constraints**
    - Should run on a standard computer or laptop with **minimum 4GB RAM** and **1GB free disk space**.
    - No special hardware (like GPS trackers or IoT devices) is required.
3. **Data Constraints**
    - Produce quantities, warehouse capacities, and shipment details must be **non-negative integers**.
    - Names, addresses, and contact details are **alphanumeric strings with length limits**.
    - Primary key values must be **unique**, and foreign key references must be **valid**.
4. **Performance Constraints**
    - Database should efficiently handle **100–500 records per table** for lab demonstration.
    - Queries should return results in **reasonable time** (seconds) for these volumes.
5. **Operational Constraints**
    - The system is designed for **offline usage** in the lab environment.
    - Users must have access to the **SQL console or DBMS interface** to perform operations.
    - Data integrity must be maintained even if multiple users operate concurrently (basic ACID compliance).
6. **Design Constraints**
    - Database schema must follow **normalization rules** (up to 3NF) to avoid redundancy.
    - Relationships must be clearly defined using **primary and foreign keys**.
    - System should allow **easy expansion** (e.g., adding more entities like Customers or Quality Checks in future).

2.5 Assumptions and Dependencies

The following assumptions and dependencies apply to the *Farmer-to-Market Produce Tracking System*:

Assumptions

1. Farmers, warehouses, distributors, and retailers are **willing to provide accurate data** for the system.
2. The database will be used in a **controlled lab environment**, and users are trained to execute SQL commands correctly.
3. Each produce batch has a **unique identifier** for traceability.
4. Users have access to a **basic RDBMS** (MySQL, PostgreSQL, SQLite) and a computer to perform CRUD operations.
5. The system will **not require real-time updates** from external sensors, IoT devices, or live market feeds.
6. The project focuses on **data management** and **report generation**, not on supply chain optimization or live logistics.

Dependencies

1. The system depends on the availability of an **RDBMS software** (MySQL, PostgreSQL, or SQLite).
2. Users depend on **SQL knowledge** or DBMS tools to interact with the database.
3. Data accuracy depends on **manual entry**; incorrect input may affect traceability and reporting.
4. Hardware must meet **minimum requirements** (computer/laptop with at least 4GB RAM).
5. The database schema and ERD must be **implemented correctly** before functional operations can be performed.

3. Specific Requirements

This section defines the **functional and non-functional** requirements.

3.1 Functional Requirements

ID	Function	Description	Input	Output
FR1	**Add Farmer**	**Add new farmer details to the database**	**Farmer_Name, Contact, Address, Region, Farm_Size**	**Confirmation of record added**

ID	Function	Description	Input	Output
FR2	Update Farmer	Modify existing farmer details	Farmer_ID, Updated Fields	Confirmation of update
FR3	Add Produce	Record new produce batch	Produce_Name, Category, Harvest_Date, Quantity, Farmer_ID	Confirmation of record added
FR4	Update Produce	Update produce details	Produce_ID, Updated Fields	Confirmation of update
FR5	Warehouse Entry	Log produce received in warehouse	Warehouse_ID, Produce_ID, Quantity, Arrival_Date	Updated storage record
FR6	Warehouse Dispatch	Record produce leaving warehouse	Warehouse_ID, Produce_ID, Quantity, Dispatch_Date	Updated storage record
FR7	Distributor Shipment	Record shipment from warehouse to retailer	Distributor_ID, Warehouse_ID, Retailer_ID, Produce_ID, Quantity, Ship_Date	Updated distribution record
FR8	Retailer Reception	Track produce received by retailer	Retailer_ID, Produce_ID, Quantity, Received_Date	Updated retailer record
FR9	Trace Produce	View full produce journey	Produce_ID	Farmer → Warehouse → Distributor → Retailer details
FR10	Generate Reports	Summarize stock, distribution, or farmer contributions	Query parameters (dates, entities)	Tabular/aggregate reports
FR11	Maintain Data Integrity	Enforce PK/FK constraints, validate entries	Database operations	Consistent and accurate database

Notes

- Each function can be implemented via **SQL commands** or **scripts** in the lab version.

- Functions like Trace Produce and Generate Reports demonstrate **traceability and transparency**, key features of your project.

3.2 Non-Functional Requirements

Non-functional requirements describe **how the system performs** rather than what it does. For your DBMS project, these include performance, reliability, usability, and maintainability.

Category	Requirement	Details
Performance	Query Efficiency	Queries on 100–500 records per table should return results within 1–2 seconds.
Reliability	Data Integrity	The database must enforce PK/FK constraints and validation rules to prevent inconsistent or invalid data entries.
Concurrency	Multi-User Access	The system should allow multiple users (e.g., Farmer, Warehouse Manager, Distributor) to access the database simultaneously without conflicts. Basic transaction control (e.g., row-level locking) must ensure that two users do not overwrite or corrupt the same data during updates.
Scalability	Future Expansion	The system should support adding new entities (e.g., Customers, Quality Checks, Alerts) or new relationships without redesigning core tables.
Usability	User-Friendliness	Users should be able to perform CRUD operations and run queries with minimal training or database knowledge.
Security	Access Control	Only authorized users may insert, update, or delete records. Read-only views may be provided for general users (optional in the lab version).
Maintainability	Ease of Updates	The database schema, constraints, and SQL queries should be simple to modify and extend as system requirements evolve.
Backup & Recovery	Data Safety	The system should support export/import or backup scripts to ensure data recovery in case of system failure or accidental deletion.